

Tapir ePublish

TDT4290 - Customer Driven Project
Fall 2009



Group 5:

Joakim Bjerkheim • Olav Undheim • Mats Gøran Karlsen • Erik Smistad • Arne Bjørgan

Preface

This project report is one of the deliverables in the course TDT4290 Customer Driven Project. The customer of this project was Tapir Academic Press, a company located in Trondheim. Tapir Academic Press is Norway's largest university press for literature in technology and the natural sciences. The assignment was to create a web site for the Press, emphasizing on the distribution and sale of digital papers online. Another aspect of the assignment was to document how current solutions of distributing electronic papers are done, such as the business domain of eBooks.

The project was carried out during the fall of 2009 at the Norwegian University of Technology and Science, with the help and support of main supervisor Basit A. Kahn and assistant supervisor Bian Wu. We would like to thank the employees at Tapir Academic Press, and especially Yngve Syrtveit, the main representative from Tapir Academic Press, for being of great help during the project.

Mats Gøran Karlsen

Joakim Bjerkheim

Arne Bjørgan

Erik Smistad

Olav Undheim

Trondheim, November 18, 2009

Contents

Contents	iii
List of Figures	vii
List of Tables	x
1 Introduction	2
1.1 Project name	3
1.2 Stakeholders	3
1.3 Customer	3
1.4 Project background	4
1.5 Project scope	4
1.6 Duration	4
1.7 Report outline	5
2 Project Directive	6
2.1 Project mandate	8
2.2 Project plan	9
2.3 Organization	14
2.4 Templates and standards	16
2.5 Project management	18
2.6 Quality assurance	19
3 Preliminary Study	22
3.1 Market investigation	24
3.2 Software development methodology	25
3.3 Version control system	29
3.4 Systems now in use at Tapir	31
3.5 Technologies and programming languages	38
3.6 Content Management System and Frameworks	40
3.7 Piracy and copyright	41
3.8 Paragallo	42
3.9 Third party payment solutions	43
3.10 Customer filtering techniques	46
3.11 Preliminary study conclusions	49
4 Requirements Specification	52
4.1 Functional requirements (product backlog)	54
4.1.1 Use cases	57
4.1.2 Detailed description of product backlog	69

4.2	Non-functional requirements	74
5	Sprint 1	76
5.1	Sprint plan	78
5.2	Sprint backlog	79
5.3	Design - Software architecture	80
5.3.1	Introduction	80
5.3.2	Model View Controller Pattern	80
5.3.3	Modules	81
5.3.4	UML Model Diagram	82
5.3.5	Database model	83
5.4	Implementation - Graphical User Interface	84
5.5	Tests and results	89
5.6	Sprint evaluation	90
6	Sprint 2	92
6.1	Sprint plan	94
6.2	Sprint backlog	95
6.3	Design	96
6.4	Implementation	97
6.4.1	Products and groups	97
6.4.2	Product search and browsing	99
6.4.3	Admin interface	102
6.4.4	IP range for institution	105
6.5	Tests and results	106
6.6	Sprint evaluation	107
7	Sprint 3	110
7.1	Sprint plan	112
7.2	Sprint backlog	113
7.3	Design	114
7.4	Implementation	115
7.4.1	My Account Interface	115
7.4.2	Journals	118
7.4.3	Discounts	119
7.4.4	Watermarking of PDF files	120
7.4.5	Checkout process and PayEx integration	120
7.4.6	Statistics	122
7.5	Tests and results	124
7.6	Sprint evaluation	125
8	Sprint 4	128
8.1	Sprint plan	130
8.2	Sprint backlog	131
8.3	Design	132
8.4	Implementation	133
8.4.1	The front page	133
8.4.2	Journals	134

8.4.3	Managing subscriptions	135
8.4.4	Shopping cart	136
8.4.5	Discounts	139
8.5	Acceptance testing	140
8.6	Results	145
8.7	Sprint evaluation	146
9	Overview of system structure	148
9.1	System structure	150
9.2	Modules	153
9.2.1	Account	153
9.2.2	Product	155
9.2.3	Order	158
9.2.4	Discount	160
9.2.5	Statistics	161
9.2.6	Journal	161
9.3	Database	162
9.4	PayEx integration	164
9.5	Security	165
9.5.1	File security	165
9.5.2	Account security	165
9.5.3	Payment security	165
9.5.4	Access control	165
10	Evaluation	166
10.1	Work process	168
10.2	Results	174
10.3	The customer and the project	175
10.4	The supervisors	175
10.5	Further work	176
10.6	Suggestions for improvements	177
10.7	Concluding remarks	178
	Glossary	192
	References	193
	Appendices	1
A	Project directive	2
A.1	Contact information	2
A.2	Meeting notice	3
A.3	Meeting notice - supervisors	4
A.4	Meeting minutes	5
A.5	Weekly status report	6

B	Sprint 1	4
B.1	GUI sketches	4
C	User manual	7
C.1	Log in	9
C.2	Account	10
C.3	Product	12
C.4	Groups	13
C.5	Journal	14
C.6	Subscription	14
C.7	Statistics and log	15
C.8	Discount	16

List of Figures

2.1	Organization chart	14
3.1	Scrum procedure[1]	25
3.2	Waterfall model[2]	28
3.3	Version control system[3]	29
3.4	Level 1 DFD	31
3.5	Customer using the net store	34
3.6	Process order and send books	35
3.7	Customer buys subscription	36
3.8	Changing the IP range of a customer	37
5.1	MVC pattern [4]	80
5.2	Class diagram of important classes	82
5.3	ER diagram of our database	83
5.4	An early sketch of how the frontpage should look like	84
5.5	The front page of the web site	86
5.6	Website registration form	86
5.7	Registration complete	87
5.8	Logged in at the web page	87
5.9	Web site when you forget the password	88
5.10	Administrator uploading new file	88
5.11	Sprint 1 burndown graph	90
6.1	The updated database	96
6.2	Product overview	97
6.3	Products details	98
6.4	Group overview	98
6.5	Group details	98
6.6	Sprint 2 backlog	99
6.7	View of an article	100
6.8	Advanced search	101
6.9	Overview of users	102
6.10	Adding a new user	103
6.11	Editing an existing user	104
6.12	IP filtering overview for instituion	105
6.13	IP Subnet for instituion	105
6.14	The burndown chart of this sprint	108
7.1	Updated database model	114
7.2	Personal details	115
7.3	User edit page	116

7.4	Order history	117
7.5	Order details	117
7.6	The user interface for a given journal	118
7.7	The form for subscribing to a journal	118
7.8	Watermarked PDF file	120
7.9	Dialog for summary when not logged on	120
7.10	PayEx payment form	121
7.11	Download section	121
7.12	Statistics main page	122
7.13	File visits and download counter	122
7.14	Administration log	122
7.15	User interface for adding a new customer discount	123
7.16	The burndown chart of this sprint	126
8.1	Database model	132
8.2	Front page part 1	133
8.3	Front page part 2	134
8.4	Personal details	135
8.5	Adding a subscription	135
8.6	Shopping cart summary	136
8.7	The shopping cart index	137
8.8	The user can enter saved shopping carts from the user menu	137
8.9	Overview over saved shopping carts	138
8.10	Product summary with buy button	138
8.11	Shopping cart showing an order with multiple discounts	139
8.12	The order history showing order with multiple discounts	139
8.13	The burndown chart of this sprint	147
9.1	Diagram of the entire system	150
9.2	Filestructure	151
9.3	Attribute type class diagram	156
9.4	ER diagram of the attribute system and product module	157
9.5	Group	158
9.6	Final ER diagram of the database	163
9.7	Sequence diagram of payment	164
10.1	Bar chart of the phases	169
10.2	The total hours used for each group member	170
10.3	Hours worked per week	170
10.4	Activity on design and implementation	171
10.5	Activity on the report	171
10.6	Activity per hour on the implementation	172
10.7	Lines of code on the web page	173
A.1	Contact list	2
A.2	Notice of meeting	3
A.3	Notice of meeting, supervisor	4
A.4	Meeting Minutes	5

A.5	Weekly status report	6
A.6	Status report, part 2	1
A.7	Status report, part 3	2
A.8	Status report, part 4	2
B.1	Page listing scientific publications	4
B.2	Page for a specific journal	5
B.3	Page of a specific article	5
B.4	Customer's personal information page	6
B.5	Shopping cart checkout	6
C.1	Admin menu	9
C.2	Admin menu	10
C.3	Choosing edit user	10
C.4	Choosing admin and save	11
C.5	Ip range page	11
C.6	Admin menu	12
C.7	Journal admin view	14
C.8	Google Analytics	15

List of Tables

2.1	Workload	11
2.2	Gant Diagram	11
2.3	Milestones	12
2.4	Risk-table	13
2.5	Risk-matrix	13
3.1	Group competence	39
3.2	IPv4 class network structure[5]	47
4.1	Customer browsing products	57
4.2	Customer user account registration	58
4.3	Customer searches for product	59
4.4	Admin uploads new product	60
4.5	Admin change product attributes	61
4.6	Admin changes IP range	62
4.7	System admin controls access of users	62
4.8	Admin manages groups of products	63
4.9	Admin views statistics	63
4.10	Customer purchase product	64
4.11	Customer subscribes to journal	65
4.12	Administrator watermarks file	66
4.13	Administrator sets a discount	66
4.14	Customer using shopping cart	67
4.15	Customer puts together a compendium	68
4.16	Customer order history	68
5.1	Sprint 1 backlog	79
5.2	Sprint 1 backlog with used time	90
5.3	Sprint 1 burndown table	91
6.1	Sprint 2 backlog	95
6.2	The sprint backlog with hours used	107
7.1	Sprint 3 backlog	113
7.2	Discounts for an order	119
7.3	Customer and group discounts	123
7.4	The sprint backlog with hours used	125
8.1	Sprint 4 backlog	131
8.2	Correspondence between tests and backlog items	141
8.3	The sprint backlog with hours used	146

10.1 Hours used on each phase	169
10.2 Lines of code for each group member	173
10.3 Items in the product backlog	174

Abstract

Tapir Academic Press is Norway's largest university press for literature in technology and the natural sciences. Today Tapir distributes scientific papers on hard format, sold in bookstores and through a net store. The press would also like to distribute digital versions of such papers. The assignment is to make a prototype of a new net store that can distribute digital articles and books.

A fully functional web site was created. The web site is made with the Zend Framework, an open source, object-oriented web application framework implemented in PHP. The resulting net store has support for different kind of products and file types. Journal and paper grouping system was made to ease the uploading of a new file. Files can inherit properties, such as author and price, from the group system, which makes it easier and faster to upload a new product.

The net store has functionality for IP filtering. Institutions, for example NTNU, can buy or get discount on certain products. If a user sits on an IP belonging to that institution, the user will get access to products already bought by the institution. Another important aspect of distributing digital products is the copyright issues. A soft copyright measure was used, where text documents can be watermarked with customer information. The payment system was connected to PayEx, a company that takes care of the actual transferring of money for a product.

CHAPTER 1

Introduction

The purpose of this chapter is to give a brief introduction of the project. This includes the background for the project, what kind of product the project group is to make, general information about the the course and an outline of the context of this report.

1.1 Project name

The title of the assignment is "Tapir Academic Press: IT system for distributing scientific papers online." The title of the project and planned prototype will be Tapir ePublish.

1.2 Stakeholders

This project is a part of the TDT4290 Customer Driven Project course. Tapir Academic Press' main representative is Yngve Syrtveit, and Lasse Postmyr will attend some meetings. The main supervisor of the group is Basit A. Kahn and the assistant supervisor is Bian Wu. The project group consists of Joakim Bjerkheim, Mats Gøran Karlsen, Olav Undheim, Erik Smistad and Arne Bjørgan.

1.3 Customer

The project sponsor, often referred to as the customer or the client of the project, is Tapir Akademisk Forlag, translated to Tapir Academic Press in English. Tapir Academic Press is a part of SiT Tapir AS, owned by "Studentsamskipnaden i Trondheim" (SiT). The customer is represented by Yngve Syrtveit and Lasse Postmyr. Yngve Syrtveit is Editor and developer of digital learning facilities, has a master's degree in computer science from NTNU and is the main contact for the project group. Lasse Postmyr is Editor for technology and the natural sciences and has a Ph.D in chemistry from NTH.

Tapir Academic Press is Norway's largest university press for literature in technology and the natural sciences. The Press is based in Trondheim. The publications include textbooks and academic literature for universities and university colleges, as well as for vocational and professional education. They also publish high quality literature of a more general nature. All in all, the compay publish about 120 new titles annually.[6]

The product lines are

- Textbooks for university and university college education
- Textbooks for vocational and professional education
- Research literature
- Journals

1.4 Project background

During the last decades, digital information has had an enormous development. This information, which could be electronic books and articles, is generally fast, easy and cheap to duplicate. Tapir Academic Press wants to advance in the field of digital education and eLearning. A part of this is trying to distribute scientific papers online in a digital version, such as pdf files.

The customer of the project is Tapir Academic Press. The Press currently has IT systems and a net store for sale of regular books, but is looking into ways of distributing electronic articles. The objective of this project is to create a web site for selling electronic papers. The assignment requires the project group to gain insight into some technological aspects of distribution of scientific papers, e.g. different solutions for payment, IP-filtering and access control based on IP range, DRM and issues regarding distribution of content protected by copyrights. The research into different technologies will be an important part of the final delivery, together with the web site prototype. Through this project, Tapir Academic Press wants to create a prototype of a web site for electronic articles, with a possible addition for other formats such as regular eBooks and multimedia files. By having such a web site, the Press will also get an insight into the customer usage and the possible market for electronic articles.

1.5 Project scope

The project scope is defined as the work that needs to be accomplished to deliver a product service or result with the specified features and functions. This is "the hows" of the project, whereas product scope is "the what". The goals are listed in the next section.

The main delivery is a net store which is tailored towards the distribution of electronic articles and eBooks. Tapir Academic Press will continue to have the current net store for paper books, and the existing IT systems. The new net store is to have its own database and not tamper with the current customer data. If the new site works well and there exists a demand for electronic papers, then the current net store and the new store might be "forged" together, but this is specified by Tapir to be outside the scope of this project.

The second part of the delivery is the study of available technologies for electronic books and articles. Tapir Academic Press is continuously considering new ways to improve customer satisfaction, and a part of the product scope is thus to do some study into ways of offering new services to the customer.

1.6 Duration

The project started on the 25th of August 2009. A pre-delivery of the report is to be handed in to IDI department at NTNU before 12:00 on the 28th of September 2009. Project demonstration and final delivery is on November 19th 2009.

1.7 Report outline

The first chapter, i.e. the current chapter, contains a brief introduction about the project and how the report is organized. The second chapter is the project directive, which contains information about the background of the project and general project management. Chapter three is a preliminary study of the topics that are relevant for this task. Chapter four is requirement specification, where it is stated in detail what kind of functional and non-functional requirements the solution has to have. Then each sprint has a chapter for itself. The sprint chapters are made in the same format, with a general introduction of what is to be achieved that sprint and a conclusion of what was actually achieved. After the sprints, an evaluation of the course and the project is given. The project ends with a conclusion of what was made, the limits of it and potential improvements that can be made in later work with the prototype.

CHAPTER 2

Project Directive

The project directive describes the administrative aspects of the project, and describes the foundations on which this project is based on. This section will give an overview of the purpose, the scope and a chapter overview of the project directive.

Purpose

The purpose of the project directive is to establish routines and set guidelines for the project management and execution. It provides a full and firm foundation for the initiation of the project.

Scope

The project directive contains administrative information about the project, including information about the task, the team, the customer and the project plan. It is a dynamic document that may be changed if changes in the project or its premises occur. Throughout the project it is likely that the team will figure out better and more efficient routines and standards, and then the project directive will be changed accordingly.

Chapter overview

The project directive contains the following chapters:

- Chapter 2.1 Project mandate
This section contains the stakeholders, objectives, resources and economy of the project. In addition we have the general terms.
- Chapter 2.2 Project plan
This section describes the different phases of the project. It also contains the project workload, milestones and risks that can occur during the project.
- Chapter 2.3 Organization
This chapter gives an overview of how the group is organized, which roles each participant has and the responsibilities connected to a specific role.
- Chapter 2.4 Templates and standards
This chapter states the specific templates and standards used by the group during all phases of the project.
- Chapter 2.5 Project management
This chapter contains the meeting routines and the timekeeping for the project.
- Chapter 2.6 Quality assurance
This chapter describes how the development process can be tailored to meet the requirements of the customer, such as project routines, response times for notices and minutes of meetings.

2.1 Project mandate

The project mandate contains information about the project's background, objectives and constraints, which was not covered in the first chapter.

Stakeholders

Interested parties are Tapir Academic Press, the project team and the project team's supervisors. Contact information for the stakeholders is enclosed in Appendix A.

Project objectives

The following list summarizes the overall project objectives:

- Make an overview of the current information system in Tapir Academic Press
- Do a study into subjects related to distributing electronic articles, such as DRM, IP-range, security, payment etc.
- Create a standalone web site for distributing electronic articles and periodical papers.

General terms

Tapir Academic Press will provide a server for the application to run on. The implementation should preferably be written in PHP and the web site should preferably be in both Norwegian and English. There are few demands from the customer, except that the prototype should be a standalone product and the existing web shop and IT system should continue in the same manner as before. An agreement on "Intellectual property rights for master's theses and student project reports at the IME faculty" have to be signed by the parties involved in the project, meaning the project group, main supervisor, the customer and the faculty. The project group is not allowed to publish material received from Tapir Academic Press, such as articles, transaction orders and the Press' customer data.

Resources

- Computers: There are available PCs at the P15 building at NTNU for the members of the project group. Everyone in the project group have their own laptops as well.
- Printouts: The group has a quota of 500 pages each.
- E-mail: The project group is assigned an electronic mailing list in order to simplify communication, kpro5@idi.ntnu.no.
- Photo copying, telephone and telefax may be provided by IDI by request.

Economy

Estimated workload of the TDT4290 course is 310 hours per student for the whole semester. For the group as a whole, counting five members, this equals 1550 hours workload. From the start to the end of the project, this means an estimated 25 hours per week for each student in the project group.

2.2 Project plan

The project plan defines the concrete work plan that is to be used during the lifespan of the project. This includes the different phases of the project, the milestones and the workload distributed into each phase.

The group, together with the customer, has decided that Scrum will be the development model to be used in the project. The main reason for choosing Scrum is because of its incremental nature. The main delivery of the project is a web site with net sales of electronic articles. Two of the most important aspects of the web site are modularity and usability. By using Scrum, it is possible to show the customer what the product will look like early in the project. Although the contact persons (listed in Appendix A) have experience with information technology, the main users of the system will be people without programming experience. As workable deliveries are to be delivered in the end of each sprint, it will be easier to keep the system modular.

We have divided the project work into three sprints after the initial preliminary study and the requirement specification. Each sprint will start with a sprint planning meeting with the customer and last for two weeks. At the end of each sprint there is a demo with the customer where everyone can attend. On the demo the product of the sprint will be showed. After the demo there will be a review meeting of reflections from the last sprint; what was good and what could be done better. A discussion of Scrum compared to other development models is given in chapter 2 (the preliminary study).

Phases

The group has divided the project into eight phases, excluding project management and self study which run over all phases. There is a strong correlation between the phases of the project and the actual chapters in the report.

- **Planning** is the first phase and consists of the initial start up of the project and the subjects described in the project directive chapter.
- **Preliminary study** is the second phase and consists of a research into subjects for the project, for example what programming languages that can be used for the prototype, and a discussion of what is best of the available options. This phase also requires a look into the existing systems in use by Tapir Academic Press.

- **Requirement specification** is the third phase and consists of mapping out the requirements for the prototype to be made, both functional and non-functional. In addition to labeling the requirements, it will also be necessary to make use case and diagrams of the required system.
- **Sprint 1** starts after the first three phases. Before the first sprint, the requirements will be translated into a backlog which is to be used for the sprints. The first sprint will start with the highest priorities from the backlog, which will be a working prototype with the most important functionality.
- **Sprint 2** will be similar to sprint 1, except that now we already will have something to build on. This sprint will typically be to add functionality to the product.
- **Sprint 3** is the last sprint and at this point it will be important to produce a final product to the customers satisfaction. If needed we have the possibility to add a fourth sprint at the end of the project.
- **Project evaluation** is a phase where we will evaluate the project as a whole and look back at the different phases.
- **Presentation and demonstration** is the last phase. In this stage we prepare the final presentation, finish the report and deliver the final product.

Workload

The estimated workload is given in Table 2.1. The estimate was made by looking at the recommended work distribution in the information booklet for the course and by looking at old project reports. The most significant difference from other reports is a shorter pre study but a bit longer sprints. This is as we plan to delve deeper into subjects in the start of each sprint, depending what we choose to focus on.

Phase #	Phases	Planned hours	Planned %
1	Project	140	8,97 %
2	Lectures and self	120	7,69 %
3	Project planning	120	7,69 %
4	Pre study	120	7,69 %
5	Requirements	215	13,78 %
6	Sprint 1	240	15,38 %
7	Sprint 2	240	15,38 %
8	Sprint 3	240	15,38 %
9	Project evaluation	75	4,81 %
10	Presentation and	50	3,21 %
	Total:	1560	100 %

Table 2.1: Workload

The gantt-diagram in table 2.2 shows the planned work for the project period. More information about milestones will be written in the next section.

ID	Task Name	Start	Finish	Duration	sep 2009			oct 2009			nov 2009					
					23.9	30.9	6.9	13.9	20.9	27.9	4.10	11.10	18.10	25.10	1.11	8.11
1	Introduction/Project Directive	24.08.2009	11.09.2009	15d												
2	Pre-study	31.08.2009	18.09.2009	15d												
3	Requirements specification	07.09.2009	18.09.2009	10d												
4	Sprint 1	21.09.2009	06.10.2009	11d												
5	Sprint 2	06.10.2009	19.10.2009	11d												
6	Sprint 3	19.10.2009	02.11.2009	11d												
7	Evaluation/Presentation	02.11.2009	19.11.2009	14d												

Table 2.2: Gant Diagram

Milestones

The milestones for the project are given in Table 2.3. Most of the milestones are made internally by the project group in order to have something to work towards, to plan progress. The most important date internally is the 20th September, which is the date we want to be finished with the three first chapters; project directive, preliminary study and requirement specification. Considering the workload distribution, this means a lot of work early on in the project. This is a choice made by the group in order to avoid running out of time later before the final deadline, and to reduce the risk of project work clashing with exercises in the other courses.

Week	Week start	Week stop	Milestone date	Milestone
35	8/24/2009	8/30/2009		
36	8/31/2009	9/6/2009		
37	9/7/2009	9/13/2009		
38	9/14/2009	9/20/2009	9/14/2009	Requirements confirmation
			9/20/2009	End of prestudy, requirements and project directive.
39	9/21/2009	9/27/2009	9/21/2009	Start of sprint 1
40	9/28/2009	10/4/2009	9/28/2009	Pre delivery of project report. End of sprint 1, Start of
41	10/5/2009	10/11/2009	10/5/2009	
42	10/12/2009	10/18/2009		
43	10/19/2009	10/25/2009	10/19/2009	End of Sprint 2, Start of sprint 3
44	10/26/2009	11/1/2009		
45	11/2/2009	11/8/2009		
46	11/9/2009	11/15/2009	11/9/2009	End of Sprint 3
47	11/16/2009	11/22/2009	11/19/2009	Project demo, delivery

Table 2.3: Milestones

Project risks

To keep track of risks in the project we have classified and written them down. The risk table with some of the initial risks we had is found in table 2.4. The risks are changing continuously through the project and so will the risk table be. We have classified them as either High, Medium or Low. Every risk has a strategy or action to deal with it. The four types of actions are:

- Avoid: Take action to make sure the risk will never occur
- Reduce: Reduce the probability that the risk will occur by taking actions, or prepare for the risk so the damage when it occurs is lower
- Transfer: Let a third party handle the risk
- Accept: Do nothing with the risk but realize that the risk will be there, and handle it if it arises

The total risk of a record is achieved by combining the consequence with the probability, see the matrix in table 2.5

Nr	Activity	Risk factor	Consequents	Probability	Strategy and actions
	Which of the activities of the project affected	Describing name of the risk factor	Start with H, M or L at a single line before describing the consequences	H, M or L	Select strategy: avoid, transfer, reduce or accept. Then on the next lines describe the measures
1	All	Sickness	H - hard to get anything done when being ill. Swine flue might affect multiple members of the	H	Transfer - Rediligate work to other members of the group, plan work so there's extra time before deadlines in case of multiple people becoming sick
2		Use more time than scheduled	M - might have less time for other phases or the quality will suffer before deadlines	H	a) work more in the beginning to create a time buffer in case of problems or b) work extra much before deadlines to make up for it
3		Tasks and tests in other courses	L - can affect how much time people in the group can work on the project	M	Avoid - we can try to plan according to the other courses to avoid conflicts
4		Customer changes requirements	H - this might mean that we have worked on something that the customer didn't want	L	Avoid - by writing things that are decided in the minutes of meetings and showing it to the customer
5		The group "miss" the task	H - this might mean that we have worked on something that the customer didn't want	L	Reduce - by being in close contact with the customer through weekly meetings
6		"Fighting" in the group	M - discussions are ok, but if the group doesn't function socially it can severely decrease the	L	Attend seminars held by the course in group dynamics, or get help from the supervisors if it has already happened
7		No way to use the new product with the existing	M - it would mean the final product couldn't be used, but it would still be able to finish the	L	Accept: The product of this project is supposed to be a stand-alone system, and thus there's not that much need to "work with" schilling (just data types)
8		Some things can't be implemented (e.g. Drupal)	M - it means that the original plan should be changed and something else used instead	M	Avoid: Do as good research as possible before starting to implement (what is possible and what is not?)
9		Files and books from Tapir are leaked	H - leaking books and articles to the public would be a violation of the contract with Tapir	M	Avoid: Be sure to check that you put the right privileges on files and folders
10		The product (the system) is too hard for the	H - if the customer can't use it, then there's no need for it	M	Avoid: Use prototypes and test with the people at Tapir who are supposed to use the system
11		Learning php and zend takes too much time	M - this could lead to less time for tasks in the sprint	M	Let the experienced users of zend make the most complex code, leave the trivial tasks to those who use most time learning the basics

Table 2.4: Risk-table

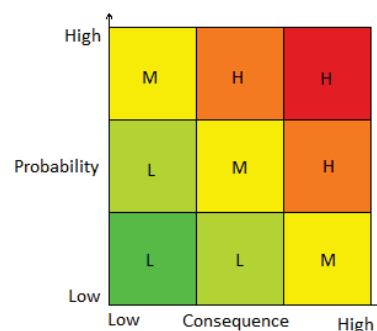


Table 2.5: Risk-matrix

2.3 Organization

In this section we will outline how the project group is organized, the role assignments and the responsibility of each group member.

Organization chart

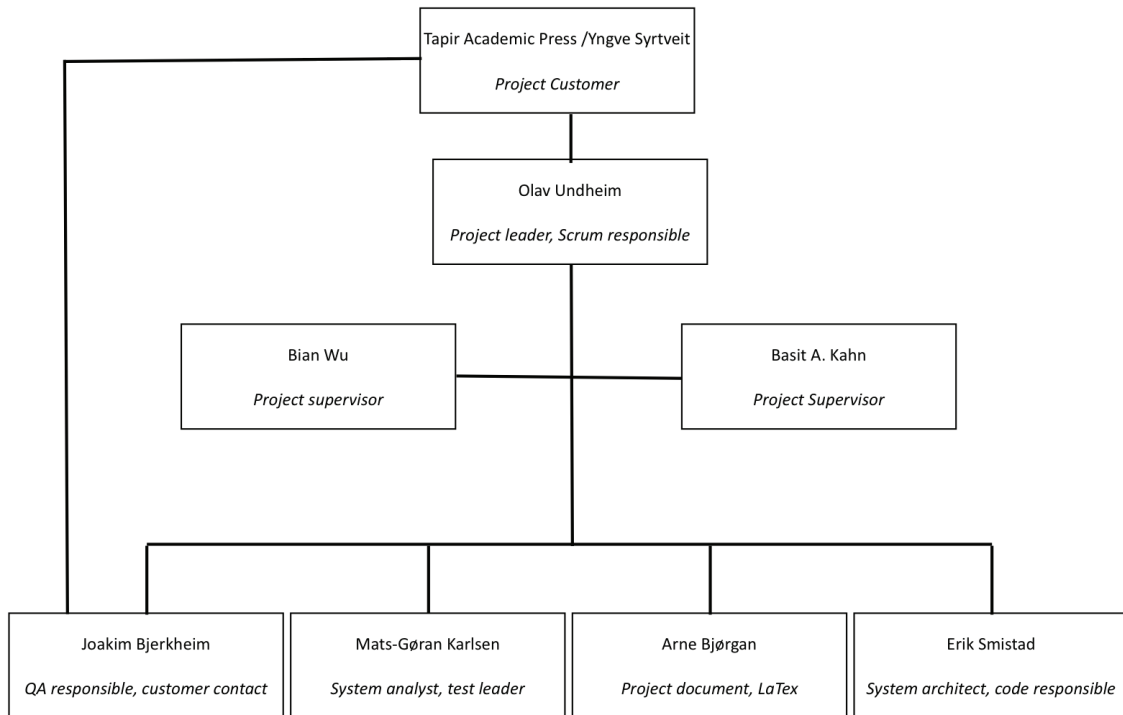


Figure 2.1: Organization chart

Although the organization model is organized as an ordered hierarchy scrum suggests that the group works more like a flat structure with no specific roles where instead the whole team collectively focus on completing the tasks within the sprint. The reason for using the model depicted in figure 2.1 is to make sure that all the different aspects of the project are taken care of; i.e the final delivery has a high degree of quality.

Scrum Roles

There are three roles in Scrum:

- Product owner - who has the responsibility of prioritizing the tasks in the backlog and evaluating the sprint results. The product owner is Tapir Academic Press, represented mainly by Yngve Syrtveit.
- Scrum master - is the facilitator of the sprints and have the responsibility of enforcing Scrum processes and act as a buffer between the scrum team and outer distractions. The scrum master for this project is Olav Undheim.

- Team members - are responsible for the actual work. The scrum team is self regulating and responsibility is divided dynamically.

Responsibilities

Although a scrum team does not have fixed roles, we have divided each chapter so that one person has the main responsibility of its delivery and quality. This person also has the responsibility of writing the introduction and conclusion if needed.

- Project directive - Olav Undheim.
- Preliminary study - Joakim Bjerkheim.
- Requirement specification - Mats Gøran Karlsen.
- Sprint 1 - Erik Smistad.
- Sprint 2 - Arne Bjørgan.
- Sprint 3 - Mats Gøran Karlsen.
- User and system documentation - Joakim Bjerkheim.
- Project evaluation - Olav Undheim.
- Conclusion - Olav Undheim.

2.4 Templates and standards

This chapter describes all templates and standards used in the project.

Templates

Templates are to be made for documents that should be in the same format. This includes:

- Notice of meeting, found in Appendix A.2, A.3
- Minutes of meeting, found in Appendix A.4
- Status reports, found in Appendix A.5

Folder structure and file naming

The folder structure is made in the same manner as the project report. LaTeX files should have the same filename as the headline of the document in order to keep the folders structured.

Version control procedures

- Commits should always have a comment written in English and should explain short and direct what has been changed
- Untested code or code with errors should never be committed
- Conflicts should always be resolved properly with the one who first discovers it

Code conventions and standards

Because we are using Zend Framework for PHP, the Zend coding standards should be used. These are defined in <http://framework.zend.com/manual/en/coding-standard.html>, but a summary and some additions/differences are mentioned below.

Encoding

Code should always be encoded with UTF-8

Commenting and language

The language in the code should be strictly English. This counts for both comments and naming of variables, functions and classes. Comments in the code should be written docblocks which follow the PHP Doc standard and should be parseable by the php documentor <http://phpdoc.org/>. Inline comments should start with `//`. All methods and classes should be documented with docblocks.

Naming conventions

Naming of variables, functions and classes should be as verbose as possible. Functions and variables names should be written in the "camelCase" format. No underscores should be used in variable names. Except for members that are declared as private or protected which should start with one single underscore. Constant should be written in upper case and can use underscore to separate words.

For example LOOP_LIMIT and DEBUG_MODE.

Coding style

Strings should be quoted by a single quote, like: \$a = 'this is a string';

A comma in an argumentlist or in array declaration should always have a trailing space for increased readability:

```
$array = array(1, 2, 3, 'foo', 'bar');  
function a($foo, $bar, $deluxe) ..
```

Curly brackets on control structures such as if/else, loops and exceptions should start on the same line the structure begins on and should have a space between the structure start and the curly bracket.

```
if($foo == $bar) {  
    // Do something  
}
```

```
while(true) {  
    // Do something forever  
}
```

But curly brackets defining the scope of functions and classes should start on a new line.

```
function fooBar()  
{  
    // Do some stuff  
}
```

```
class Cool_Class  
{  
    public function coolMethod()  
    {  
        // Do stuff  
    }  
}
```


2.5 Project management

This section outlines the way the project management will be done in the project.

Meetings

There are three different types of meetings. Customer meetings are scheduled every Monday at 12:30 in room R92. We use a shared Google Docs document for the agenda where everyone, including the customer, can add issues that they want to have addressed at the next meeting. Before the meeting, meeting minutes from last week meetings will be sent to the customer. Supervisor meetings are scheduled each Wednesday from 14:15 to 15:00 at ITS-236. A status report of what tasks has been performed since the last meeting is to be handed over to the supervisors before each meeting. Internal group meetings, often just referred to as "Group meetings", are set to be held at least once per week. This meeting is scheduled on Thursdays from 10:15. Team work sessions are working hours where the whole group, or those available in the group, sit together and work with the project. This is done to encourage better communication when working with tasks and better cooperation. We also believe it will be more motivational to have such team work hours, and in the early phase it will be important in order not to procrastinate tasks which doesn't have an immediate deadline in the near future. The hours given here does not account for exercise lessons in other courses:

- Monday: 12:30 to 16:00, after the customer meeting
- Tuesday: 16:00 and onwards
- Wednesday: 15:00 and onwards
- Thursday: 10:15 to 14:00, after the group meeting
- Friday: 14:00 and onwards

Timekeeping

A spreadsheet has been created in Google Docs that all group members can edit concurrently. Every group member is responsible for documenting how many hours he has used for each phase at each week.

2.6 Quality assurance

This chapter aims to guarantee the quality of delivered work.

Routines for producing quality

This section outlines the routines concerning the written material.

All documents that are written and delivered is to follow a template. To avoid misspellings and such, one other group member has to read censorship before the text is published.

New code shall be reviewed by one other team member before submitted to the system. All written code should be tested, either manual or automatic. The code should be well documented so that external people can understand the code. The code is to follow conventions.

We have got a dedicated person with main responsibility on the written report, Arne Bjørgan. When a chapter in the report is finished, the group reviews the material and discusses the result. Joakim Bjerkheim are appointed the Quality assurance responsible.

Routines regarding customer

This section takes care of the aspects of quality assurance concerning the customer.

Confirmation of meeting requests should be made within 24 hours. Answers to questions should be made within 24 hours. The other group members should approve the meeting minutes. Meeting minutes from customer meeting should be sent to the customer within 24 hours after the meeting.

Routines regarding the supervisors

This section takes care of the aspects of quality assurance concerning the supervisors.

All sent e-mails are to be sent to both Basit and Bian at the same time. We shall have at least one meeting every week. Room is to be booked by the supervisors. The group sends the agenda, report, changelog on the report, meeting summary from last time, meeting summary from customer meeting and status report. This is preferably sent in one mail, to both Basit and Bian, and the subject should contain "Kpro5" These documents are to be sent at least 8 working hours before the meeting.

Internal routines

This section takes care of routines within the group.

Before any meeting starts, we assign one person as responsible for writing meeting minutes. We will have a group meeting every thursday between 1015 and 1400. Meeting minutes will be sent to all project participants within 24 hours. Every group member should check it's learning and email at least twice a day. In weekends one check is enough. Contact via cell phones will be used in emergency situations. If unexpected problems come by, this should be adressed in an extraordinary meeting.

Testing

All written code should be tested, either manual or automatic. The sequence of the testing phases:

- Unit test: Testing of single methods, done while programming from early in all sprints
- Module test: Communication between parts of program, more comprehensive than unit testing but done simultaneously
- System test: Testing of the system as a whole, done in the end of the sprints
- Integration test: Testing of the system together with external participants, done in the end of sprint 3

Non functional tests including end user:

- Usability tests: Secure that the communication between the user and the system is as expected, these tests takes place from sprint 2 and outwards
- Acceptance tests: End user tests the system, based on these tests the customer decides whether the product should be used, after sprint 3.

CHAPTER 3

Preliminary Study

Purpose

The purpose of this chapter is to give an overview of the relevant knowledge in connection with the project. While we are in the preliminary phase we will try to get as much information about useful technologies as possible.

We will also look into existing solutions to similar tasks to see if we can apply any of these to our project.

Scope

Based on the lack of time we will only scratch the surface in this phase. Some aspects will not be covered. We have not written about equivalent web services, although they are listed here.

Because this is only a superficial overview, we will need to look more deeply into the themes while working on the project. Such studies will then be included in the respective sprint chapter.

Chapter overview

The pre-study contains the following chapters:

- Chapter 3.1 Market investigation
This section tells us what solutions the competitors use.
- Chapter 3.2 Software development methodology
This section contains an introduction to waterfall and scrum, and a discussion about which we should use in the project.
- Chapter 3.3 Version control system
This section discusses which VCS we should use for collaboration and version control.
- Chapter 3.4 Systems now in use at Tapir
This section gives an overview of the systems that Tapir uses today, and how the buying process is done in today's systems.
- Chapter 3.5 Technology and programming languages
This section discusses what technology we should use when implementing the solution.
- Chapter 3.6 Content management systems and frameworks
This section discusses what CMS and framework we should use for the project.
- Chapter 3.7 Piracy and copyright
This section discusses the use of DRM and digital watermarking
- Chapter 3.8 Paragallo
This section gives an insight into Paragallo and the solutions they can offer to us and Tapir

- Chapter 3.9 Third party payment solutions
This section compares the different payment solution providers that exists on the market.
- Chapter 3.10 Customer filtering techniques
This section discusses the possibilities and solutions for filtering customers from their IP-address.
- Chapter 3.11 Preliminary study conclusions
Summary of the conclusions made in the preliminary study chapters.

3.1 Market investigation

Websites selling ebooks, scientific articles and journal already exists. And there is a lot of them. From large ones such as books.amazon.com and www.springerlink.com to smaller ones such as pubs.acs.org. There is also a very high competition in this industry. But the content that Tapir wishes to publish(at least in the beginning) is unique. The five journals that they wish to sell on the site are only sold by Tapir and will not exist on any of the other websites. They also don't want to sell other publishers material on their website, and therefor the content on the website will be unique. And it is Tapirs goal that the scholars who are interested in this content will be able to find their way to website we will make. Search engine optimization (SEO) should therefore be one the goals of this project. When someone searches for Tapirs articles and journals on the web, this projects website should rank high.

Functionality on the competitors websites

We have been looking at what functionality the existing websites offer. Here is a list of the websites we checked out:

- <http://butikk.tapirforlag.no>
- <http://scholar.google.com>
- <http://books.google.com>
- <http://www.sciencedirect.com>
- <http://pubs.acs.org>
- <http://www.springerlink.com>
- <http://cat.inist.fr>
- <http://adsabs.harvard.edu>
- <http://www.sciencedirect.com>

We noticed when sitting on NTNU's network that we gained access to a lot of the articles and journals on these websites, so the use of IP range for detecting their customers is already in widespread use in this market. Most websites also support payment online using a third party payment solution, but none offer payment by SMS. Websites with a large amount of products usually offer a way to browse easily through the products by dividing it into different categories and using tagging. Most of them also have search box on their website. Some of the websites provide more fancy functionality like the possibility to preview the PDFs and to view the entire content of the PDF as a HTML document.

3.2 Software development methodology

Introduction

This section gives a brief discussion around the different development methodologies that could have been used for this project. Even though there are a lot of different methodologies, we have chosen just to discuss the two methods we considered the most, waterfall and scrum. Scrum is an agile software development method that put focus on small iterative "Sprints". Waterfall is a sequential development methodology where the focus is to completely finish one phase at a time.

Scrum

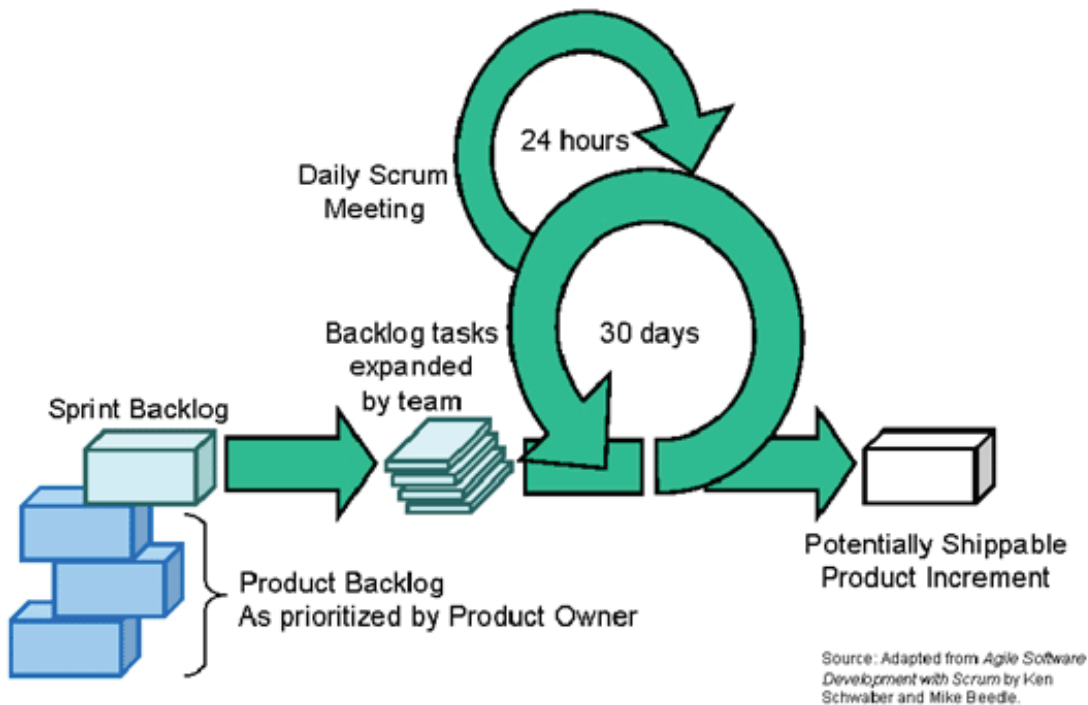


Figure 3.1: Scrum procedure[1]

Scrum is an incremental framework, commonly used with agile software development. Scrum was intended for management of software development projects, and has gained a lot of popularity in recent years. Many companies have now started using scrum instead of the waterfall method, because of the iterative processes. These processes gives the software developer a chance of getting one part of the program tested and accepted before moving on to the next part. A project group will also have the chance to get more feedback from the customer during the development process. Scrum reduces the risk of ending up with a product that the customer doesn't recognize as what they wanted. Scrum contains sets of practices and predefined roles [7]. The main roles in scrum are:

- Scrum Master, who maintains the processes
- Product Owner, who represents the stakeholders
- The Team, a cross-functional group who do the actual analysis, design, implementation, testing, and so on.

When working with scrum, we have sprints, typically a two to four week period. In this period the team produces a product increment. This would typically be a working and tested software module. The set of features that go into a sprint come from the product backlog, which is a prioritized set of high level requirements of work to be done. These sets of requirements are put in sprints during the sprint planning meeting. During a sprint, no one is allowed to change the sprint backlog, which means that the requirements are frozen for that sprint. After a sprint is completed, the team demonstrates the use of the software.

Daily scrum meeting

When using scrum we have different kind of meetings that should be executed. The daily scrum meeting is a time boxed meeting, typically with punishments for the participants being late. The daily scrum meeting should also have a fixed place and time every day. During the meeting, each team member answers three questions:

- What have you done since yesterday?
- What are you planning to do by today?
- Do you have any problems preventing you from accomplishing your goals?

The daily scrum meeting should be an efficient way of managing the progress in the group.

Sprint planning meeting

At the beginning of the sprint cycle this meeting is held to:

- Select what work is to be done.
- Prepare the sprint backlog that detail the time it will take to do that work, with the entire team.

- Identify and communicate how much of the work is likely to be done during the current sprint.

At the end of a sprint cycle, two meetings are held: the "Sprint Review Meeting" and the "Sprint Retrospective", which is presentation and reflections about the sprint just finished.[7]

The product backlog

The product backlog is a high-level document for the entire project. It contains what will be built, backlog items. It is open and editable by anyone and contains rough estimates of both business value and development effort. The backlog tells us what should be done, and how the prioritizing is among the items.[7]

The Sprint backlog

The sprint backlog is a document containing information about how the team is going to implement the features for the upcoming sprint. Features are broken down into tasks which are easy to understand for the team members.[7]

The Burn Down

The sprint burn down chart is a publicly displayed chart showing remaining work in the sprint backlog. Updated every day, it gives a simple view of the sprint progress.[7]

Waterfall

Waterfall has been a standard development methodology for a long time. It originated from manufacturing and construction industry, and was adapted into the software industry as well. It is sequential; starting and finishing one phase before the next. Some claim it is called the waterfall methodology because each phase flows naturally into the next phase like water over a series of falls. The phases are in sequential order; requirements specification, design, construction, integration, testing, installation and maintenance.[8]

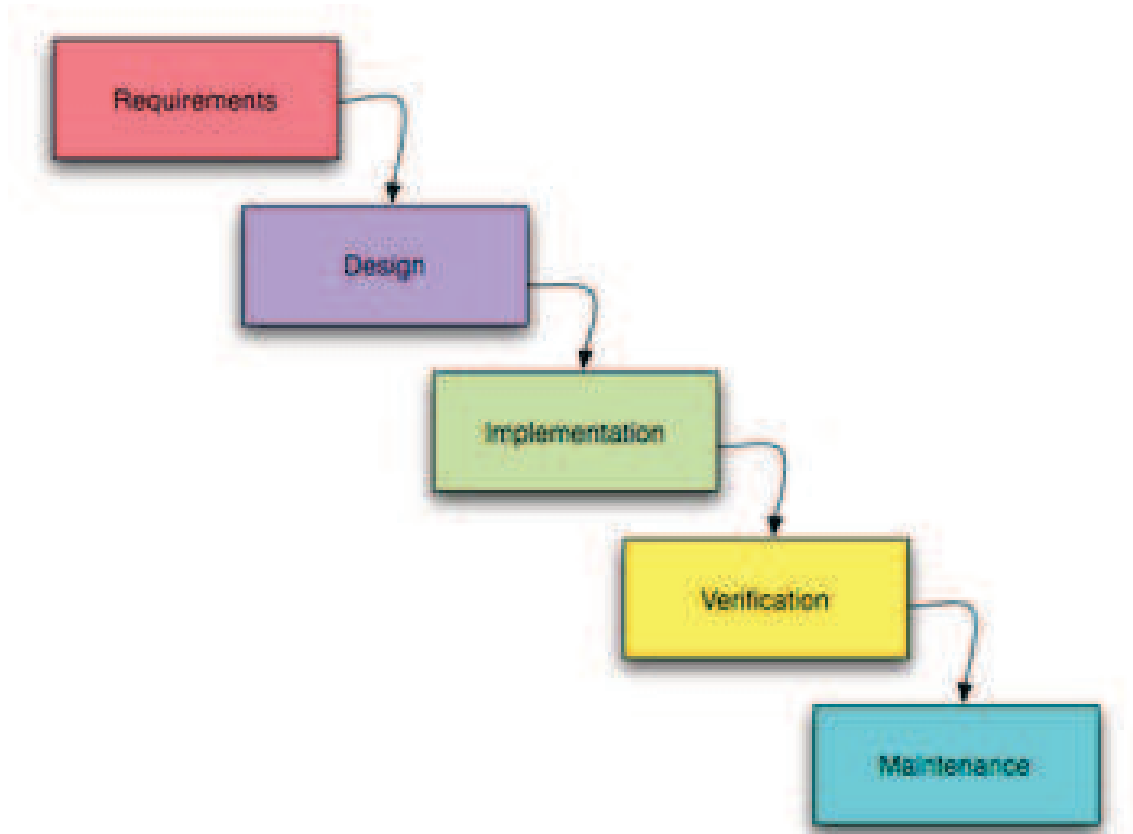


Figure 3.2: Waterfall model[2]

The advantage of the waterfall method is that it is easy to understand and it encourages a disciplined process. First you think about what should be built, then you make a plan to build it and finally you build it. It allows for easy managerial control. Waterfall is a well suited method when the requirements are fixed and won't change during the execution of the project. Another possible advantage is that you have to completely finish and document one phase before moving to the next, which force you put a proper effort into each phase, and not make any shortcuts in effort or documentation. The disadvantage of the waterfall model is mainly the lack of flexibility. It does not allow for much reflection or revision while the project is running. It becomes harder and harder to do changes in the early stages the further into the project you come. This means if the customer decide to change their initial requirements, which they often do with software products, then it will be a costly change with this development model. Another disadvantage is that all the testing is done when the system is supposed to be almost finished, i.e. at the later stages of the project. At this point it may already be too late to implement the possible changes that is suggested based on the test results.

Methodology conclusion

Scrum was chosen to be the software development process for this project, and there are several reasons for this choice. From earlier experiences the team members have learned that requirements tend to change during the course of a project. In addition, the project group does not have a lot of experience making the web system described in the project objective. This means that having multiple iterations with focus on making a system that is up and running as quickly as possible will make it easier to get new ideas which wasn't included at the original requirement specification. Also since scrum in contrary to the waterfall model does not rely on the overall system design being finished before implementation changing requirements would be easier to implement into the system.

Team member roles are dynamic in Scrum, while the waterfall method has a more traditional separation of different responsibilities. For a student group a dynamic team role seems more appropriate. Scrum has also been quite popular the recent years and as such the project group and the customer felt it would be beneficial to learn more about working with Scrum. By using Scrum we get working software at an early stage and will be able to make good use of the continuous customer feedback that is offered to us.

3.3 Version control system

Version control systems manages the changes of documents. These systems are very useful when several people are changing the same files, like we will do in this software development project. The files that are committed are stored at a repository, which usually reside on a server. When you perform what is called a checkout at your local machine you get a copy of this repository as it was when you performed the checkout. You can also do an update on your local machine to get the latest updates from the repository. When someone are finished changing some files and want to share them with the others, they can commit the changes to the repository. Most modern version control systems support non-linear development. What this means is that when different users start making changes they create branches from the main development line (trunk). These branches can later be merged with the trunk or discontinued. This is illustrated in the picture below.

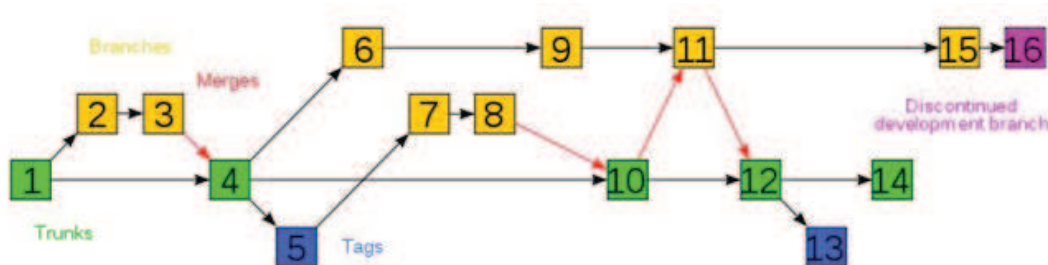


Figure 3.3: Version control system[3]

Different version control systems

Different version control systems exists, here are the three most popular.

- **CVS (Concurrent Versions System)**

One of the first version control systems out there. It was developed in the 1980s and is know seen to be outdated, but is still in widespread use.[9]

- **SVN (Subversion)**

Was developed to be the successor of CVS. The plan was to remove the bugs that CVS had and add the features it was missing. SVN is wide use today and is know to be faster and more reliable than CVS.[10]

- **GIT**

Developed by Linus Torvald as a tool for developing his Linux kernels. Has a strong support for non-linear development by using branching and merging. Both CVS and SVN has the support of non-linear development, but Git is know to be very fast and scalable when it comes to non-linear development and therefor being ideal for very large projects.[11]

Conclusion of which version control system to use

All the VCS mentioned above are installed on NTNUs servers. But none of the group members have experience with GIT. And since SVN is more recommended than CVS and most of the groups members have previous experience with it we decided to use SVN as our VCS.

3.4 Systems now in use at Tapir

Figure 3.4 is a data-flow diagram (DFD) showing the data flow when a customer order of a book is processed. Not all the systems in Tapir Academic Press are pictured in this DFD, but it gives an insight into the most important system, Schilling.

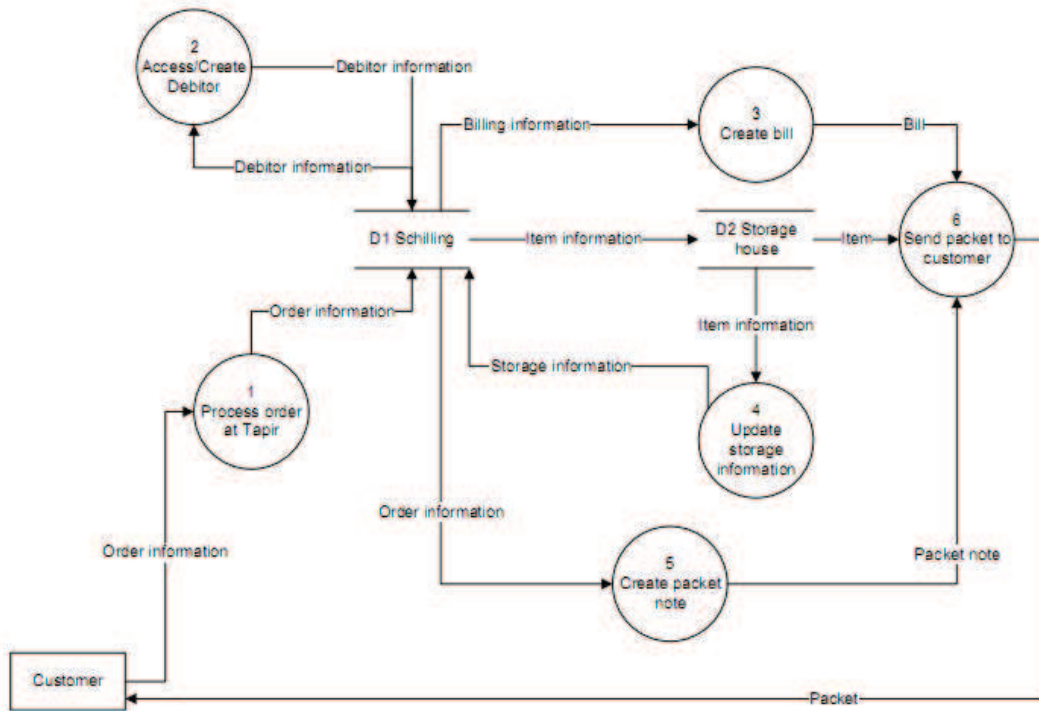


Figure 3.4: Level 1 DFD

The processes involved in this DFD are:

- Process 1: The staff at Tapir receives an email, fax or telephone call from a customer with order information
- Process 2: If the debtor exists in Schilling we access that account, else we have to create a new debtor
- Process 3: The bill is created at Tapir with information from the Schilling-system
- Process 4: The storage information must be updated according to the item(s) that are picked for delivery
- Process 5: The packet note with order information is created as a sticker that's put on the packet that will be sent to the customer
- Process 6: The packet is sent to the customer via mail delivery. Contains a bill, a packet note and the item(s) that was ordered

Schilling

Schilling is an enterprise resource planning system especially designed for publishing companies. The system takes care of tax, percentage of sales price that the writer collects (royalty), storage management, invoice sending and accounting.

Moses

Moses is a system that among other things converts files into PDF-files. Files is to be delivered to the system, for instance .docx-files, and you get out a much nicer looking file in PDF-format. Moses takes also care of the e-subscriptions.

Mentor

Mentor is Forleggerforeningen's information portal that is used by bookstores. The publishers gives information about their books, such as title, theme, a description of the book, number of pages, and where to order them. Employees in the bookstores use Mentor to search for books when asked by customers, and gets ordering information.

Aries

Aries administrates the process a text have to go through before it can be published. This embraces both the editorial work, such as evaluation and academic common assessment, and the work to be done before pressing the papers (proofreading etc). For these purposes Aries is divided into two subsystems called Editorial Manager and PrePrint Manager. Aries includes a function for version control for documents who are edited online by more than one person at time.

euroConnex

This system delivers an integrated payments processing service for credit and debit cards to banks, payment processors, and merchants.

Use of existing systems

We are not going to change any of these programs. What we might want to do is to code up against Schilling. We have to pay for doing this, and it could be complicated, but it would decrease the amount of work to be done for Anne.

Use cases

To better understand the current information system at the company, we made four use cases of typical tasks performed by the customers and the employees at Tapir. The use cases are given one by one at the next 4 pages.

Each use case has a unique name which identifies that use case. Participating actors are the actors involved for this specific use case. Flow of events is the major part of the use case, which states the typical sequential line of events, and the "extensions" field depicts alternatives for a given event. Entry and exit conditions give the starting and finishing states, respectively, while the quality requirement field states possible quality requirements involved in the specific use case.

Use case name	Customer at net store
Participating actors	Initiator customer Communicates with Tapir net store
Flow of events	<ol style="list-style-type: none"> 1. Customer enters the net store http://butikk.tapirforlag.no/no 2. Customer search for a book by title, author, ISBN or keyword 3. Customer adds a book to his shopping chart 4. Customer logs in with his username and password 5. Customer enters name and address for delivery (if needed) 6. Customer decides payment method 7. Customer pays with card and enters card number, expiry date and cvc2 number 8. Order is finished and a confirmation email is sent to the customer's email address
Extensions	<ol style="list-style-type: none"> 3a. If multiple books are to be ordered at the same time, the customer repeats the process from step 2 4a.1: The customer has not registered before, and will have to register with a username and password <ol style="list-style-type: none"> .2: Tapir sends a confirmation email, which the customer has to confirm .3: The customer can log in with his newly created username and password 4b. The customer has forgotten his password and have to use "get new password" 7a. The customer can decide to pay with invoice (not for foreigners)
Entry conditions	Net store available online
Exit conditions	<ul style="list-style-type: none"> •Net store confirms purchase •Net store indicates to customer why order failed
Quality requirements	Net store responds within 10 seconds

Figure 3.5: Customer using the net store

Use case name	Process order and send books
Participating actors	Initiator net store Communicates with administration at Tapir Administration communicates with Schilling, Storehouse
Flow of events	<ol style="list-style-type: none"> 1. The net store sends an email to Tapir with information about the order 2. The order information is manually entered in the Schilling system 3. The customer is added as a debtor in the Schilling system 4. If the customer decided to pay with invoice, and invoice is created 5. A message is sent to the storehouse about which book(s) are ordered 6. The books and the invoice are put together and sent to the customer
Extensions	<ol style="list-style-type: none"> 2a. An order can also be made by phone directly or by fax 4a. The customer might be added already, and is then just selected using name or number 5a. This step is skipped if the customer paid by card
Entry conditions	Customer has made an order Administrator at Tapir is logged into Schilling
Exit conditions	<ul style="list-style-type: none"> • Order is shipped • Administration at Tapir has received indication why adding debtor in Schilling failed
Quality requirements	Customer receives order within estimated time

Figure 3.6: Process order and send books

Use case name	A customer gets subscribed to a periodical
Participating actors	Initiator customer Communicates with administration at Tapir. Administration communicates with Schilling and Moses
Flow of events	<ol style="list-style-type: none"> 1. Customer sends email to Tapir and requests to subscribe a periodical. Either paper, e-format or both 2. Anne registers order in Schilling 3. Anne sends payment information to customer 4. Tina Skjærvik gives customer admission to the periodical 5. Tapir receives payment
Extensions	<p>3a. Customer is an institution and no information about IP-range is given</p> <p>.1 Tina Skjærvik also requests for IP-range</p> <p>4a. The order contains e-format</p> <p>4b. Customer is an institution:</p> <p>.1 Tina Skjærvik registers the IP-range in Moses</p> <p>.2 Tina Skjærvik notifies the customer that they have access</p> <p>4c. Customer is a private customer:</p> <p>.1 Tina Skjærvik registers the customer in Moses.</p> <p>.2 Tina Skjærvik sends customer username and password needed to log in</p>
Entry conditions	Anne is logged into Schilling Tina Skjærvik is logged into Moses
Exit conditions	<ul style="list-style-type: none"> •Schilling acknowledges new customer registration •Tina Skjærvik has received indication why customer could not be registered •Moses acknowledges request to give user admission •Moses indicates why admission request could not be registered.
Quality requirements	<ul style="list-style-type: none"> •Schilling acknowledges request within system requirements •Moses acknowledges changes within system requirements

Figure 3.7: Customer buys subscription

Use case name	Change IP range
Participating actors	Initiated by network administrator Communicates with adminSystem
Flow of events	<ol style="list-style-type: none"> 1. Network administrator is informed about change in customers network 2. Network administrator logs in to network administrator system 3. Network administrator locates customer listing 4. Network administrator changes the IP listing 5. System confirms the change
Extensions	<p>3a: Customer has no preexisting IP listing .1 Network administrator registers customer and fills in the given IP range, returns to step 5</p> <p>4a: Customer wants to add another IP range to the preexisting .1 Network administrator adds IP range to the customer listing, returns to step 5</p>
Entry conditions	Network administrator is logged into adminSystem
Exit conditions	<ul style="list-style-type: none"> • Network administrator has received acknowledgement from adminSystem • Network administrator has received explanation indicating why changes could not be processed.
Quality requirements	<ul style="list-style-type: none"> • The network administrator's changes are acknowledged within 10 seconds. • The changes take effect 10 seconds after the changes are accepted.

Figure 3.8: Changing the IP range of a customer

3.5 Technologies and programming languages

Different types of languages

PHP is the most common scripting language for web development. It's an open-source interpretative language. In an interpretative language the code is not compiled, but interpreted at run time. The language is dynamically typed like python, which means that you can't initialize variables with types like integer or string. The syntax is similar to that of C++ and Java with some exceptions. As of PHP version 5 the language got a descent object model, resembling the Java object model, making PHP a good language for writing object oriented code. The PHP language is implemented as a module for the popular Apache web server.[12]

ASP(Active Server Pages) is a technology for making dynamic web pages and not a language. It is developed by Microsoft and runs on their IIS(Internet Information Service) server. The language usually used in ASP is VBScript, but many other languages like Perl and Javascript can be used.[13]

Python is a language originally not made for web development, but with the mod_wsgi or mod_python Apache modules and the Django framework, this language has been enabled for web development. The python language itself is an interpretative language and dynamically typed often used for algorithm construction.[14]

Perl is like PHP a very well know programming language used for web development. It runs on the Apache web server and is famous for its text manipulation capabilities. The syntax of perl resembles PHP and is also dynamically typed.[15]

HTML/CSS is used for describing the content and the layout of a web page. HTML documents are interpreted by the web browser and comes in many different dialects where HTML 4 and XHTML 1 are the two main dialects. They are called dialects of HTML because they don't differ very much in syntax. The syntax is very similar to XML syntax like <tag>text</tag>.[16]

CSS (Cascading Style Sheets) is a language used for defining the looks of web pages mostly together with HTML.[16]

Javascript is a client side script language used to manipulate the content of web pages without the need of sending a new HTTP request to the server. Having javascript enabled in your browser poses a real security threat and therefor a website should never rely 100 percent that the client has javascript enabled.[17]

Flash allows for more graphical websites that are not limited by the simplicity of HTML and images. By using Flash you can make animations and integrate videos and a lot of other applications in your website. Some of the drawbacks of Flash is that the clients has to have a Adobe Flash Player plugin installed in their web browser and search engines can't see into flash websites and therefor can't index the content of flash websites.[18]

Team programming competence

Table 3.1 gives an overview of the experience each team member has with the different programming languages that has been described. We used a classification with "high", "medium", "low" or "none", to indicate how much knowledge and experience each group member has for a given technology.. The higher the group total ranking is for a given language, the better it is to use it.

Group member/Tech	Erik	Mats	Olav	Arne	Joakim	Groups total
PHP	High	Medium	None	Low	Low	Medium
ASP	Low	None	None	None	None	None
Python	Medium	Medium	Low	Low	Low	Low
HTML / CSS	High	Medium	Medium	Medium	Medium	Medium
Perl	None	Low	None	None	None	None
Javascript	Medium	None	Low	None	None	Low
Flash	Low	None	None	None	Low	None

Table 3.1: Group competence

Final choices of technologies and programming languages

As for which server-side programming language to use our group has the most experience with PHP and Python, but those who have some experience with Python does not have any experience with Django which you have to use for developing websites in Python. Because of this, the popularity of PHP and the fact that the customer requested PHP to be used we decided to use PHP as our server-side programming language.

3.6 Content Management System and Frameworks

Different types of Content Management Systems and Frameworks

Drupal

Drupal is an open-source PHP-based CMS(Content Management System) and sometimes referred to as a CMF(Content Management Framework) because Drupal is made to be configurable and customizable. The development of Drupal started as early as in 2000 and has a lot of backward compatibility. It is not an object oriented system, instead Drupal is a layer-based system with 5 layers. Here is an overview of the layers:

- The bottom layer is the data layer
- A layer of modules; modules are parts that can be added and removed from the system with a loose coupling as possible to the other modules
- The Blocks and Menus layer get output from a module and can be configured to output in different ways.
- The user permission layer controls who has access to what in the system.
- On the surface we have the presentation/template layer made primarily up of HTML and CSS.

On the Drupal website you will find a lot of plugins and modules that can be downloaded and installed. They have both an e-commerce and a journal module.

Zend framework

Zend Framework is a PHP based framework and not a CMS. The framework is developed by Zend company the same company that is involved with the development of the PHP language itself. New versions of the framework is added every month as a result of constant testing and development of new features. It requires PHP5 and uses best practices within object oriented programming and is based on the design pattern MVC (Model View Controller). It is highly customizable and all of its internal code has been made so that it can be overloaded and get added functionality. It also supports the PHP unit testing framework, which allows for test driven development.[19]

Cake PHP

Cake PHP is also a PHP based framework. It works on both PHP4 and PHP5, which means that this framework does not take the full advantage of PHP5 object model which means that it has among other thing no support for interfaces and encapsulation of members and methods. This framework is also based on the MVC(Model View Controller) design pattern.[?]

Django

The Django framework for Python is a new framework which has gained rapid

success in web development because of its use of best practices and encouragement to rapid development and clean, pragmatic design.[20]

Final choices of CMS/CMF and framework

The customer initially suggested that we use Drupal since Tapirs other e-store (butikk.tapirforlag.no) uses Drupal. After the preliminary study the group concluded together with the customer that we will not use Drupal. One of the reason for this is that Drupal has a reputation for being messy to configure and to program new modules for, especially if you have no previous experience with it. It is also not object oriented which is a major drawback for our group because our programming/system developing education at school has been very object oriented. So because of this and since we are going to make system that requires a lot of customized solutions we decided not to use a CMS like Drupal for development, but instead use a framework that allows us to develop the website from scratch rapidly by using the tools in the frameworks. Since we decided on using PHP, there is no reason to discuss using the Django framework as it is in the Python language. One in our group has a lot of experience using the Zend Framework for PHP and recommends it very much. And since this framework employs the best practices in object oriented programming we decided together with the customer to use the Zend Framework.

3.7 Piracy and copyright

Since we are going to sell digital material like PDF's in the online store we have to take into consideration using methods to prevent piracy of the material. This will be a security for Tapir Academic Press that the articles won't be spread among institutions and private customers.

Digital Rights Management(DRM)

Digital rights management (DRM) is a generic term that refers to access control technologies that can be used by hardware manufacturers, publishers, copyright holders and individuals to try to impose limitations on the usage of digital content and devices. The term is used to describe any technology which inhibits uses (legitimate or otherwise) of digital content that were not desired or foreseen by the content provider.[21]

DRM is most known for forcing the user to follow one or more restrictions that follow the copyrighted material, by implementing technical solutions on how the material can be used. Typical examples on these solutions are:

- copy protection, that the material is locked for printing or limitations on how many times you can open/run the material.
- contain a license that describes the rights that followed the material, and also a protection of this license.

The adversaries of DRM think that this type of technology charges the consumers with too many restrictions and enable misuse. They also think that DRM

unreasonably limits the user for the usage of a product that is bought legally. An example of this is that you may not copy the material to another computer for your own use. DRM has also gained criticism because it's more build upon restrictions than rights. The supporters claims that DRM is necessary to prevent extended piracy of digital material.[21]

Digital Watermarking

Digital watermarking is the process of possibly irreversibly embedding information into a digital signal. The signal may be audio, pictures or video. If the signal is copied, the information is also carried in the copy.

In visible watermarking, the information is visible in the picture or video. Typically, the information is text or a logo which identifies the owner of the media.

In invisible watermarking, information is added as digital data to audio, picture or video, but it cannot be perceived as such (although it is possible to detect the hidden information). An important application of invisible watermarking is to copyright protection systems, which are intended to prevent or deter unauthorized copying of digital media. While some file formats for digital media can contain additional information called metadata, digital watermarking is distinct in that the data is carried in the signal itself.[22]

Which way do we go?

Just as important for us as the prevention of piracy are satisfied customers. Putting a rigid DRM into the articles may lead to customers that are not satisfied and chooses to end their subscription or refuse from buying the articles online. Many have been used to buying a physical copy of the articles, so we can't risk losing customers due to strict DRM solutions when going towards digital distribution. In addition the articles are not that valuable compared to books, so the market for piracy products is not that big. Watermarking is a method for preventing piracy that we find interesting because the customer is not restricted in usage of their legally bought article. As described above, watermarking consists of either a visible or invisible mark that connects the product to a customer or distributor. Worst-case scenario for the customer is that the article contains for example a email or a name. This will not decrease usability in any extent mentionable. We have decided to make use of watermarking. Since we are in a pilot project, the distribution will be limited in volume so the risk is not that high. The system should be made so DRM can be added in case of revelation of extended piracy.

3.8 Paragallo

[23]

The web site that is to be built in this project needs support for different kind of file types and content. Paragallo is a Norwegian company that specializes in delivering solutions and services for management of multimedia content, which may also be copyrighted. The reason for doing a brief study into Paragallo is because they have already set a date for a demonstration with Tapir Academic Press, and as such

their solution might be used for this project's web site. Paragallo delivers "solutions and services that manage multimedia content, copyrights, digital distribution, and interactivity". Paragallo delivers plug and play based multimedia solutions for companies that have, or want to have, online digital distribution of media content as a part of the business model. Both services and solutions can be supplied by Paragallo. Services will typically be business development such as consulting online distribution of multimedia content. Solutions are typically modules or systems that take care of certain aspects of an online shop. The provided solutions are described in the following paragraphs.

- **Media Bank** is a solution for importing, encoding, storing and managing multimedia files. The media bank supports all major multimedia technologies and is thus a good way of handling the data files, which in the early phases of this project will be electronic articles in pdf format.
- **Online shop** is currently just for music and audio books, but Paragallo is working on supporting eBooks as well. What this solution currently offers is licensed music content that can be sold on the web site, typically saved in the media bank.
- **On Demand** is Web- and MobileTV support, which can be run directly from the web site. This includes technologies such as Video On Demand, Live Streaming and Radio Streaming. Although Tapir Academic Press is mainly focusing on written material, it is a possibility to have video material on the web site as well.
- **Interactive** is a solution for marketing and attracting new customers. There are three sub categories under this; SMS messaging, social networking and mobile marketing.
- **Admin Portal** is an administration solution with live sales statistics, content management system, partner profile, sales reporting templates etc. Creating a well made administration interface for the web site is an important part of the project, and the "Admin Portal" is the solution that Paragallo has come up with.
- **Billing** is the set up and cooperation with 3rd party payment methods. This includes SMS and Wap (CPA) payment, Visa / Mastercard and micropayments. This solution is also highly relevant for what Tapir Academic Press wants to have on the web site.

3.9 Third party payment solutions

Payment methods

The website should be integrated with a third party payment system. Such a system is needed for receiving payment from customers on the Internet. The way these systems usually work, which is called the re-direct method, is explained below.

- The customer select the products he wishes to purchases and goes to checkout.
- The customer is then redirected to a third party payment system
- The third party system receives the payment details from the merchants website. Details such as price, amount and product name
- The customer supplies his payment details (ex. Credit card type and details) to the third party website
- When payment is complete or canceled the customer is redirected back to the merchants website.
- When payment has been processed, and has completed or failed, the third party system sends a message, usually by means of a HTTP packet, to the merchants website. The transaction can then be processed on the merchants website and the customer can get his product.

Another popular method is the direct method, where the customer always stays on the merchants website. This can be less confusing for the customer, but it requires that the merchant has a website with a proper SSL certificate, which can be expensive and complex to implement.

SMS payment

The customer also wanted us to look into payment solutions by SMS(Short Message Service). SMS is a service you can use with most mobile telephones today. One SMS can contain up to 160 characters and you can send to any telephone that support the SMS service. With SMS payments we mean that you send an SMS to specific number and then your mobile subscription is overcharged for that one SMS. And then the money gets transferred to the merchant. There are restrictions of how much you are allowed to receive from SMS payments. In Norway its 1NOK to 100NOK, and it can be different for each country. For instance in Denmark the maximum price is 75 DKK.

Different payment solution providers

Here is a description of a few third party Internet payment solutions.

DIBS

DIBS Payment services is one of Scandinavias largest supplier of payment systems over Internett. They offer features like payment by all the major credit cards. But they don't have any functionality for payment by SMS or invoice. DIBS has a lot of great technical documentation on their tech website[<http://tech.dibs.dk>].[24]

PayEx

PayEx is also one of the largest supplier of payment solutions in the Nordic countries. They also have documentation on how to integrate your own system with theirs. But they have a lot more features than DIBS like payment by SMS and invoice. For SMS payments the customer needs to have a Norwegian, Swedish or Danish mobile subscription. Supports both the direct and re-direct methods explained in the previous section regarding SMS payment. PayEx is implemented as a web service using WSDL and SOAP to exchange information between the merchants website and PayEx.[25]

PayPal

PayPal is a large international payment solution which have no monthly or establishing fees, only transactional fees. It has a great API which is easy to use. It uses the re-direct method, with a simple HTTP post as the callback mechanism. But it does not have SMS or invoice payment methods.[26]

Sendega

Sendega is an SMS gateway company. They have features for sending and receiving SMS and receiving payment by SMS.[27]

Final choice of payment solution

We have decided to use the re-direct method for payment as it is fairly easy to implement, is well support and in common use. Together with the customer we decided to use PayEx as the best third party payment solution since they are the online nordic payment solution company that support the methods we wish to use: payment by card, invoice and SMS.

3.10 Customer filtering techniques

The project customer wants the store to use automated filtering techniques to simplify the authentication process for the user. Looking at sites offering similar products advanced IP range filtering is commonly used. The same kind of functionality is wanted by the user.

The IP protocol

An IP address is a numbering identification and logical address that is assigned to devices participating in a computer network utilizing the internet protocol for communication between the nodes. The original design of TCP/IP stack, which is mostly used today in public networks, consists of an address space of 32 bits. This is called the IPv4. Due to the increased number of computers in the internet today a new standard for addressing has been proposed IPv6 which is a system that uses 128 bits to define an address space.

Subnetworks

The IP protocol also has the task of routing data packets between networks. IP addresses used in TCP/IP specifies source and destination nodes in the topology of the routing system. For this purpose some of the bits in the IP range are designated to subnetworks (subnet). A subnet describes network computers and devices that have a common, designated routing prefix. This is done by dividing the IP address into two parts: network identifier and the host identifier. Subnetting is supported by both the IPv4 and IPv6. Subnet masking is only supported by IPv4; this is done by specifying which bits of the address that designate subnetworks. The other method used is CIDR (Classless Inter-Domain Routing). This is supported by both IPv4 and IPv6. Here the IP address is followed by a slash and the number of bits (decimal format) used for the network part, called the network prefix. An example of IPv4 CIDR notation: 129.241.111.111/24. Here the 24 first bits are used to define the network and subnet. By utilizing subnets one gets a hierarchy with the organization network address space partitioned like a tree structure. The borders between the networks are often routers. By organizing the network in such a way excessive rates of packet collision are minimized.

Internet Protocol version 4 Classes

Classful network is a term used to describe the network architecture of internet until 1993. It divided the address space of IPv4 into the five address classes as shown in figure 3.2. As seen from the figure the first three bits defines the class.[28]

By todays standard dividing networks into classes is obsolete. This is mainly because too many sites were too big for a C class network and therefore received a B class block. This caused the problem of the available addresses in the B pool became too small for the rapid growing internet. Therefore a new way to divide networks were proposed using classless domains; Classless Interdomain Routing (CIDR).[28]

The term network class is mainly today used in general discussions about networks to describe. But for the sake of completeness we have included here.

Class	CIDR	Subnet mask	Start	End
A	/8	255.0.0.0	0.0.0.0	127.255.255.255
B	/16	255.255.0.0	128.0.0.0	191.255.255.255
C	/24	255.255.255.0	192.0.0.0	223.255.255.255
D	-		224.0.0.0	239.255.255.255
E	-		240.0.0.0	255.255.255.255

Table 3.2: IPv4 class network structure[5]

While the 127.0.0.0/8 network is in the Class A area, it is designated for loopback and cannot be assigned to a network. Class D is reserved for multicasting. Class E reserved for future use

Network address translation (NAT)

NAT is the process of modifying the network address information in datagram packet headers while in transit from source to destination. Normally this is done in routers or computers connecting the rest of the local network to the internet. By using equipment that does NAT multiple client devices can appear to share IP addresses.[29]

IP filtering

IP filtering is a network layer facility used to decide which types of datagrams which will be processed normally and which will be discarded. Discarded simply means wheter the datagram is to be deleted or ignored. Different sorts of criteria for datagram filtering is possible. Commonly used criterias are:

- Protocol type
- Socket number
- Datagram type
- Datagram source address
- Datagram destination address

IP filtering is done in the network layer. This means that the filtering does not know about the semantics of the package content or the origin of the package. This means that filtering of this type alone may not be enough. This is where setting up proxy servers may come in handy[30]

Proxy servers

To prevent behavior mentioned above one can use proxy servers for each service that one wishes to pass through. Proxy servers understand the application they were designed to proxy and thereby prevent abuses, such as another service using the same port as the web server. This is handled such that the proxy will answer requests to the assigned ports and only allow datagrams designed for the service to pass through. A number of proxy server programs exist both free and commercial.[31]

Access control

Access control systems are designed to enable control access authority to areas and resources in a given computer based information based system. ACL includes authentication, authorization and audit. It also includes measures such as encryption, digital signatures, monitoring by users/automated. [32] There are two classes of access control models:

- Capabilities based
- Access control lists (ACL)

Capability based models are based on entities holding capabilities granting them access to the object. Access is conveyed by transmitting capabilities over a secure channel.

ACL models are based on granting subjects access by validating them against a list associated with the object in question. Access is conveyed by editing the list.

Both models have mechanisms to allow access rights to be granted to members of groups.

ACL systems provide services for identification and authentication, authorization and accountability where:

- Identification and authentication determines who can log on to a system and the association with the software subjects they are able to control as a result of logging in.
- Authorization determines what a subject can do
- Accountability identifies what a subject has done

FEIDE

FEIDE stands for "Felles elektronisk ID" which in English can be understood as an electronic ID which you can be used from any location. And that is basically what Feide is. Feide is an identity management system on a national level for the

educational sector in Norway. Almost every educational institution such as NTNU, UiO, UiS, HiST and all the other universities and schools in Norway is using or is starting to use Feide..[33]

Therefore we suggested to the customer that this might be a good alternative to IP filtering, as that solution limits the user in the way that the user has to be on campus to get the products. With Feide the users can get the products from anywhere in the world.

The way Feide work is by the method of re-directing. When a website wants a user to identify himself with Feide. The website redirects the user to the Feide login page. When the user has successfully logged in at the Feide login site, feide redirects the user back to the main website and sends a set of attributes to the website with information about the user. This information can be like name, address, phone number, institution, faculty etc

It is very easy to implement, but you should have an SSL certificate so that the traffic can run on https and hence encrypt the information about the user, but it's not required.

Final choices of customer filtering techniques

We want to use a combination of ACL and capabilities based access control. By letting the system administrator edit a list of the customers IP ranges we can check against the ACL over what services are to be provided.

From the users standpoint when he/she connects from a network in the ACL their client is recognized and hereby given access to all the documents that the account in the ACL have subscribed to.

For users belonging to a institution but connecting from an outside network than the ones registered in the ACL we propose using Feide. The problem here are that this solution are constrained to public institutions only in collaborating countries. This can be solved by connecting user accounts in the system to companies/institutions.

3.11 Preliminary study conclusions

The group picked scrum for this project. The two major reasons for this was getting the system up and running early in the development process and a process that better supported change in system requirements due to continuous customer feedback.

PHP was the programming language of choice. We decided to build the store from scratch instead of relying on an existing solution that needed customization. The reason for this was that a lot of the team members weren't that familiar with any website scripting language but PHP seemed the one that seemed closest to our competence (See table 3.1). To prevent too much coding a framework was chosen. Here we picked one that was familiar to one of our group members and that supported MVC: Zend Framework.

SVN was chosen as revision control for the project code. We chose this solution based on group experience; by using MVC, Zend uses nested folders, we knew that we needed something that supported recursive adding (CVS out of the picture) but weren't too advanced (git out of the picture).

To prevent piracy but to maintain fair use and offering a user friendly solution we decided on a lightweight DRM: Digital Watermarking

The payment solution chosen was PayEx. We chose this solution for their SMS support and because the customer already used them in their online bookstore.

One of the requirements early on from the customer was content filtering using IP addresses. We chose a solution based upon ACL: customers linked with IP addresses, IP filtering. For users belonging to an ACL but connected from outside IP addresses of the ones given in the ACL, ex. Professor from NTNU on business trip to Haag, Feide was proposed as a solution.

CHAPTER 4

Requirements Specification

Purpose

The requirement specification describes the wanted effects of the new system. The purpose of the requirement specification is to figure out how the new system should be and what the system should be able to do. This chapter starts with user stories and use cases, which is how the system will look like from the point of view of the potential users of the system. The product backlog and the requirements work as a contract of what the customer wants and what the team should focus on delivering.

Scope

This chapter covers what the new system should be like. Everything is looked upon from the users point of view, the details of implementation will be dealt with in later chapters. In a waterfall development model the requirement chapter will contain more information about the actual implementation of the system, working as a blueprint before the implementation phase. This could for example be a data flow diagram of the information system, class diagrams etc. As this project is carried out using scrum, we only needed the product backlog, which is the basis of the sprints. Architectural decisions, documented as DFDs and class diagrams, will be added in the sprint chapters. We decided to add the requirement list in this chapter, as the customer felt it could be beneficial to give the project team a clear understanding of what was wanted. The requirement list will not be as static as in a waterfall project, and possible changes will be discussed in the upcoming chapters for each sprint.

Chapter overview

This chapter contains the following sections:

- Chapter 4.1 Functional requirements (product backlog)
This section lists the items in the product backlog, which is the functional requirements as seen from the users' perspective.
- Chapter 4.1.1 Use cases
This section lists the use cases based on the items in the product backlog.
- Chapter 4.1.2 Detailed product backlog
This section lists the decided functional requirements in detail.
- Chapter 4.2 Non functional requirements
This section lists the decided non functional requirements in detail.

4.1 Functional requirements (product backlog)

The product backlog is the main list of all wanted functionality in the system, as seen from the user perspective. The product backlog is a dynamic list, which is allowed to grow or shrink during the sprints. If for example in the first sprint, a new wanted functionality is discovered, it will be added to the product backlog and can be added into the next sprint backlog. In other words, the product backlog is not a fixed document, but may change as the project progress and new ideas and needs are discovered.

We decided to make user stories as the backbone of our product backlog. The user stories states what kind of goals a user has for the web system - what functionality the user wants to have. User stories are used early in the project. The idea is that you write short notes about what the different users would like to do. They differ from typical requirement specifications in that they focus on user needs and not how the system is implemented. By looking at a user story the developer should make an estimate about how much time it will take to implement, that is the estimated complexity of the implementation.

Our whole group sat together and wrote the user stories on post-it notes. Together with the customer we estimated the time needed to implement the functionality of a user story, and gave a priority for each of the stories. The priorities we used were:

- High: These stories are critical for the projects success and must be implemented.
- Medium: These stories determines the success of the project and should be implemented.
- Low: These stories contain the features that we hope to have time to implement.

System users

- Administrator - The administrator can do typical tasks as seen from the employees point of view, such as uploading new files, managing groups of products, get sale information etc.
- Customer - This user is the one who browses through the content and purchases products. People who are just visiting the site, meaning potential customers, are also a part of this category.

On the next page are the items of the product backlog, written as a user story with a time estimate and a priority. The user stories are written: As a 'user' I want to 'goal' so that 'reason'. The time is estimated based on complexity; high, medium or low. In the sprints we will break down a product backlog item into smaller tasks in order to give a real time estimate of how long it takes to complete the task.

1. **Customer browsing products**

As a customer I want to browse through all the products so that I can easily select the products that I want.

Time complexity: Medium

Priority: High

2. **Customer registers account**

As a customer I want to be able to register and use an account so that I don't have to write my details many times.

Time complexity: Medium

Priority: High

3. **Customer searches for products**

As a customer I want to search for products so that I can easily find the products that I want.

Time complexity: Low

Priority: High

4. **Admin uploads products**

As an admin I want to upload new products so that I can sell new products on the site.

Time complexity: Medium

Priority: High

5. **Admin changes product attributes**

As an admin I want to administrate the attributes of the products so that the customer can see all the attributes they need.

Time complexity: High

Priority: High

6. **Admin changes IP range**

As an admin I want to administrate IP ranges for institutions so that we can give privileged access to institutions.

Time complexity: High

Priority: High

7. **System admin controls access of users**

As a system admin I want to control the access of all the users of the system so that we can give privileged access to those who wish to pay for it.

Time complexity: High

Priority: High

8. **Admin manages groups of products**

As a system admin I want to manage groups of products so that I can control large groups of products.

Time complexity: High

Priority: High

9. **Admin views statistics**
As an administrator I want to see statistics for the website so that I can judge if the project is worth the effort and optimize sales on the site.
Time complexity: High
Priority: Medium
10. **Customer purchases product**
As a customer I want to be able to purchase products.
Time complexity: High
Priority: High
11. **Customer subscribes to journals**
As a registered customer I want to be able to subscribe to journals so they I can read every issue.
Time complexity: Medium
Priority: High
12. **Admin watermarks file**
As an admin I want the products to be watermarked to prevent illegal copying so that I don't lose money.
Time complexity: Low
Priority: Medium
13. **Customer shopping cart**
As a customer I want to be able to collect products in a shopping cart so that I can pay for many product at once.
Time complexity: Low
Priority: Medium
14. **Admin gives discounts**
As an administrator I want to give discounts so that some customers buy products more frequently.
Time complexity: Low
Priority: Low
15. **Customer makes compendiums**
As a customer I want to be able to make compendiums so that my student can buy a collection of articles that I want as curriculum for a course.
Time complexity: High/Medium
Priority: Low
16. **Customer order history**
As a registered customer I want to se my order history so that I can keep track of how much money I have spent on the site.
Time complexity: Medium
Priority: Low

4.1.1 Use cases

From the user stories we made three use cases, that try to give a feel of how the new net store will work. The following use cases are made the same way, and the fields have the same meaning, as the use cases in Chapter 3.2. The three scenarios given here are the ones we feel are the most important and cover "user registration", "product upload" and "product purchase". Use cases for each item in the product backlog will be given at a later point in the project.

Use case name	Customer browsing products
Backlog ID	1
Participating	Customer Net store
Flow of events	<ol style="list-style-type: none"> 1. Customer enters the net store 2. Customer selects "Browse all documents" 3. Customer clicks on interesting article 4. Customer reads about article 5. Customer clicks buy article 6. Websites gives feedback that article is added to shopping cart 7. Customer proceeds to checkout
Extensions	<ol style="list-style-type: none"> 2a. Customer selects subcategory. Returns to step 3 5a. Customer finds article useless and clicks back to continue browsing. Returns to step 3 7a. Customer hasn't selected any documents and exits the website
Entry conditions	Customer wants to browse products to find articles with specific content
Exit conditions	<p>Customer finds documents and proceeds to checkout</p> <p>Customer can't find any documents and exits website</p>
Quality requirements	None

Table 4.1: Customer browsing products

Use case name	Customer user account registration
Backlog ID	2
Participating	Customer Net store
Flow of events	<ol style="list-style-type: none">1. Customer enters the net store2. Customer use the "register new account" function on the net store3. Customer enters his wanted username4. Customer enters his wanted password twice5. Customer enters personal details6. Customer confirms the registration7. The customer is automatically logged in with his new username8. The net store sends a confirmation email to the email address specified
Extensions	<ol style="list-style-type: none">2a. If customer is already logged in, he has to log out before creating a new account5a. Personal details may be optional, then proceed to event 68a. Only done if an email was registered
Entry conditions	Customer wanting a user account on the website
Exit conditions	Customer has a user account
Quality requirements	None

Table 4.2: Customer user account registration

Use case name	Customer searches for product
Backlog ID	3
Participating	Customer Net store
Flow of events	<ol style="list-style-type: none"> 1. Customer enters the net store 2. Customer clicks on search field 3. Customer enters search criteria 4. Customer presses enter or clicks search 5. Website displays search results 6. Customer clicks on article 7. Customer finds article useful and clicks buy 8. Website confirms purchase added to shopping cart 9. Customer proceeds to checkout
Extensions	<p>2.a Customer decides to browse for document. Customer exits this use case and enters use case "Customer browsing for product".</p> <p>5a. Website returns 0 results or none of the results are relevant.</p> <p>5a1. Customer decides to change search criteria, returns to 3</p> <p>5a2. Customer exits website</p> <p>5a3. Customer decides to browse for articles. Customer exits this use case and enters use case "Customer browsing for products"</p> <p>9a. Customer wants to continue browsing through search results and clicks back, returns to 6.</p>
Entry conditions	Customer wants to search for articles containing specific criteria
Exit conditions	<p>Customer finds document(s) and proceed to checkout</p> <p>Customer can't find any documents and exits website</p>
Quality requirements	None

Table 4.3: Customer searches for product

Use case name	Admin uploads product
Backlog ID	4
Participating	Administrator Net store
Flow of events	<ol style="list-style-type: none"> 1. The administrator logs onto the administrator interface with his/her username and password 2. The administrator selects the product to upload from local machine or intranet 3. The administrator fills in all the needed details about the product 4. The administrator then chooses "Upload and publish" in the interface 5. The administrator confirms all the details, products and settings for this product 6. The upload is finished, and the log contains information about what have been done
Extensions	<ol style="list-style-type: none"> 1a. The administrator has forgotten his/her password, and needs to reset it and get it sent by email 4a. The administrator has put in some illegal details that needs to be corrected, repeat step 4-5 4b. The administrator doesn't want to publish yet, and chooses "just upload" 5a. If multiple are to be uploaded, repeat step 3-6
Entry conditions	Net store available online
Exit conditions	Net store confirms the upload to the server Net store publishes the upload
Quality requirements	The upload takes reasonable time to finish

Table 4.4: Admin uploads new product

Use case name	Admin change product attributes
Backlog ID	5
Participating	Administrator Net store
Flow of events	<ol style="list-style-type: none"> 1. The administrator logs onto the administrator interface with his/her username and password 2. The administrator selects the product that needs change 3. The administrator changes the attributes in question 4. The administrator clicks the apply changes button 5. Net store shows the changes to the administrator and asks for confirmation on the changes 6. Administrator clicks the confirmation button 7. Net store confirms changes
Extensions	<ol style="list-style-type: none"> 1a. The administrator has forgotten his/her password, and needs to reset it and get new password sent by email 4a. The administrator has entered illegal attribute values, returns to 3 5a. The administrator has entered wrong values clicks back and returns to 3
Entry conditions	Net store available online Administrator wants to change attributes of product
Exit conditions	Product has new attributes
Quality requirements	None

Table 4.5: Admin change product attributes

Use case name	Admin changes IP range
Backlog ID	6
Participating	Admin
Flow of events	<ol style="list-style-type: none">1. Admin goes to the page of the web site where the IP ranges are2. Admin selects the institute he wants to administer3. The IP ranges of the chosen institute is displayed4. Admin does the changes he/she would like. This is done by adding or removing intervals of IP address5. Admin presses the save button6. The web page in 1 is displayed
Entry conditions	Admin is logged in and wants to administer the IP ranges
Exit conditions	The IP ranges of the institutions are updated
Quality requirements	The forms to edit IP ranges should be user friendly

Table 4.6: Admin changes IP range

Use case name	System admin controls access of users
Backlog ID	7
Participating	System admin, admin
Flow of events	<ol style="list-style-type: none">1. System admin enters the page where user accounts are displayed2. System selects a user account to edit3. System admin sets the user accounts rights to be admin4. System admin clicks on the save button5. The page in 1 is displayed
Extensions	3a. System admin is editing an admin account and sets this accounts rights to normal user
Entry conditions	System admin is logged in A user account that should become an admin is created
Exit conditions	Admin is granted the accesses he should have

Table 4.7: System admin controls access of users

Use case name	Admin manages groups of products
Backlog ID	8
Participating	Admin
Flow of events	<ol style="list-style-type: none"> 1. Admin finds a product he wants to add to a group 2. Admin finds the group he would like the product to be in 3. Admin adds the selected product to the wanted group
Extensions	<ol style="list-style-type: none"> 1a. He wants to remove the product from a group 2a. If the product already has a group the admin is noticed about that
Entry conditions	Admin is logged in and wants to manage groups of products
Exit conditions	A group of products are updated
Quality requirements	None

Table 4.8: Admin manages groups of products

Use case name	Admin views statistics
Backlog ID	9
Participating	Administrator
Flow of events	<ol style="list-style-type: none"> 1. Administrator logs in with his username and password 2. Administrator chooses the statistics link in his administrator interface 3. Administrator chooses between the available statistics 4. Administrator finds the statistics in interest and can see it in the interface
Extensions	<ol style="list-style-type: none"> 1a. If administrator is logged in, go to step 2 1b. Administrator has forgot his password and needs to get it sent, then go to step 1
Entry conditions	Administrator wants to see statistics for the website
Exit conditions	Administrator has seen the statistics
Quality requirements	None

Table 4.9: Admin views statistics

Use case name	Customer purchase product
Backlog ID	10
Participating	Customer, Net store, 3rd party payment solution
Flow of events	<ol style="list-style-type: none"> 1. Customer searches for the article he wants to purchase 2. Customer open the detailed page of the article 3. Customer presses the "add to shopping cart" button 4. The website gives feedback to the customer that the article has been put in the shopping cart and presents the customer with an option to continue shopping or proceed to checkout 5. The customer select the go immediately to checkout 6. The customer looks over the price and personal info 7. The website redirects the customer to the 3rd party payment solutions payment site 8. The customer provide his payment details to the 3rd party payment solution 9. The 3rd party payment site redirect the user back to the website 10. The 3rd party payment solution sends data to the website with information about the payment 11. The website gives the user confirmation of the purchase and presents the customer with file on the website and sends an email with URL to where the user can download the file 12. The customer downloads the file
Extensions	<ol style="list-style-type: none"> 2a. The customer does not find the article 11a. The data from the 3rd party payment solution indicates that the transaction was not completed. The customer gets a message from the website about this and is offered to try again 13a. The customer tries to download the file after the 3 hours limit
Entry conditions	Customer at the net store, wants to buy an article
Exit conditions	Customer has a watermarked PDF copy of the article
Quality requirements	<p>The search should only find relevant results</p> <p>The customer should never have to wait more than 2 seconds for response on the website or the 3rd party payment website</p>

Table 4.10: Customer purchase product

4.1. FUNCTIONAL REQUIREMENTS (PRODUCT BACKLOG)

Use case name	Customer subscribes to journal
Backlog ID	11
Participating	Customer
Flow of events	<ol style="list-style-type: none"> 1. Customer logs in with username and password 2. Customer searches for product 3. Customer puts product in shopping cart 4. Customer chooses to check out 5. Customer chooses the subscribe button 6. Customer chooses to get notifications on email when a new article is published 7. Customer enters payment details 8. Customer is sent through the third-party payment solution provider 9. Customer gets confirmation of the purchase 10. Customer gets email when a new article is available for download
Extensions	<ol style="list-style-type: none"> 1a. Customer chooses to skip logging in, go to step 2 2b. Customer has forgot password, gets it sent by email, go to step 1 3a. If customer wants to buy more products, repeat step 2 and 3 4a. If customer is not logged in, log in 7a. Customer can choose a saved payment card in his account 10a. If step 6 is performed
Entry conditions	Customer wants to subscribe to product(s)
Exit conditions	Customer is subscribing to product(s)
Quality requirements	None

Table 4.11: Customer subscribes to journal

Use case name	Admin watermarks file
Backlog ID	12
Participating	Administrator
Flow of events	<ol style="list-style-type: none"> 1. Administrator logs in 2. Administrator chooses product upload 3. Administrator chooses a file to upload 4. Administrator fills in the attributes for this file 5. Administrator chooses the Watermark option in the admin interface 6. Administrator chooses upload file(s) 7. The system adds a digital watermark, visible or non-visible to the file, and uploads it
Extensions	<ol style="list-style-type: none"> 1a. Administrator forgot his password, gets it sent by email 6a. Administrator wants to add more products before uploading, repeat step 3 to 5
Entry conditions	Administrator wants to add watermarked file(s)
Exit conditions	Digitally watermarked file(s) is uploaded

Table 4.12: Administrator watermarks file

Use case name	Administrator gives discounts
Backlog ID	14
Participating	Administrator, Net store
Flow of events	<ol style="list-style-type: none"> 1. Admin logs in on the web page 2. Admin finds a product or customer that he wants to put a discount on 3. Admin goes to the information page for a given customer or product group 4. Admin edits the discount for the customer or product group 5. The new discount is updated on the web page
Extensions	2a. The discount can also be set to a specific amount, for example you could get 10% discount if you buy for 1000 NOK
Entry conditions	Admin wants to give a discount to customers or on products
Exit conditions	The customer or products has been updated with a discount
Quality req.	The web site should be updated withing 60 seconds

Table 4.13: Administrator sets a discount

Use case name	Customer using shopping cart
Backlog ID	13
Participating	Customer, Net store
Flow of events	<ol style="list-style-type: none"> 1. Customer enters the net store 2. Customer finds a product he wants to buy 3. Customer press the "buy" button or "add to shopping cart" button 4. The product is added to the shopping cart 5. The customer is asked if he wants to continue shopping or proceed to checkout 6. The customer proceeds to checkout 7. All the items in the shopping cart are listed 8. The customer proceeds to payment solution 9. The customer pays and may download the products.
Extensions	<ol style="list-style-type: none"> 5a. Step 2-5 are repeated if more products are added 8a. The customer may chose just to store his shopping cart 8b. The customer can decide to share his cart with other people, by getting an url that leads to the cart and other users may buy them themselves.
Entry conditions	Customer wants to put a product in the shopping cart
Exit conditions	Customer has one or many products in the shopping cart

Table 4.14: Customer using shopping cart

Use case name	Customer makes compendium
Backlog ID	15
Participating	Customer Net store
Flow of events	<ol style="list-style-type: none"> 1. Customer enters the net store 2. Customer clicks on "make a new compendium" 3. Customer finds articles that he wants to have in the compendium 4. The articles found are merged together 5. The customer can share his compendium to other people
Extensions	4a. A possibility is that articles are just combined to a zip, or that multiple pdfs can be concatenated
Entry conditions	Customer wants to make a compendium of multiple articles
Exit conditions	Compendium is made, and can possibly be shared with others

Table 4.15: Customer puts together a compendium

Use case name	Customer order history
Backlog ID	16
Participating	Customer Net store
Flow of events	<ol style="list-style-type: none"> 1. Customer enters the net store 2. Customer logs into his account 3. Customer enters his "personal information" page 4. Customer clicks on "order history" 5. Former order history is provided 6. The customer can get detailed information for each order
Entry conditions	Customer is already registered
Exit conditions	Customer has received information about his order history
Quality requirements	None

Table 4.16: Customer order history

4.1.2 Detailed description of product backlog

Our customer wanted a formal requirement specification early in the project. Typically in a project using scrum a formal list of the requirement specification would not have been made so early in the project, but as we created this requirement specification we decided to add it as a detailed description of the product backlog.

Below are the functional requirements categorized after feature. The non-functional requirements are given in the last part of this section.

Product administration

1. It should be possible to publish documents in the form of journals and articles on the website The website should primarily support the PDF file format
 - (a) The website should be constructed so that it can easily be extended to support other types of products like mp3-, flash-, php-/html- and video files.
 - (b) It should be possible to add, edit and delete product-parameters like abstract, writer, price etc. through an administrator web interface.
 - (c) The files should be uploaded via the admin web interface (not ftp, ssh etc.)
2. The files should be visible on the website when choosing to publish the product. This should be a checkbox option in the admin interface.
3. It should be possible to tag products with keyword to increase search engine optimization
4. Each file should have an unique file-nr (file-id).
5. Administration of attribute/parameters
 - (a) It should be a default set of attributes for each file type e.g. pdf.
 - (b) This set of attributes should me able to alter for each file.
 - (c) It should be possible to add, edit and delete attributes for each file using box.
 - (d) There should be a possibility to group files where some attributes would be the same for the entire group, e.g. price, who has access etc since most subscribers will have access to a multitude of files through the periodicals.
 - (e) Which attributes are common for the group should be possible to specify when creating the group and alter later.
 - (f) It should be possible to add and remove files from groups.
 - (g) There should be an option to place an entire group as part of a group, resulting in the parent group controlling the attribute specified, and the child group controlling the attributes specified in the child group and not in the parent group.

- (h) It should be possible to override the attributes for a file, but if this is done there should be an alert box visible in this files admin page.
 - (i) Each group should have an admin page.
 - (j) Each file should have an admin page.
 - (k) Visible at each admin page for groups and files should be an info box listing the files attributes, dependencies of groups, date uploaded, and user who uploaded it.
6. Administration of categories
- (a) It should be possible to add, edit and delete categories
 - (b) There are two types of categories
 - i. Content type e.g. chemistry, mathematics, computer science etc.
 - ii. Product type e.g. articles, journal articles, journals etc.
7. Journals
- (a) It should be possible to add and edit each issue of a journal
 - (b) It should be possible to add, edit and delete all the articles in each issue in a journal
 - (c) Administrators should be able to view, add, edit and delete all the subscribers for each journal
8. The system should be able to share a shopping cart on facebook, twitter, direct link, email etc. to accommodate the use of several articles as part of a course.
9. There should be a concatenation option making several pdf's into one pdf.
- (a) There should be taken into account that we might want to sell printed concatenated pdf's at a later stage, and the software should therefor be customizable to this. The specifics of this should be investigated and documented.
 - (b) Provide a direct link to the compendium

Product browsing

1. A paging system should be used when browsing products so that only a limited amount of products are shown at the same time
2. It should be possible to search for products on the website
 - (a) The search field should be a pure text search in all parameters by default.
 - (b) There should be a advanced option for specifying parameters to search within.
3. It should be possible to view products in a specific category

4. There should be a public page per file that have been checked "published" in the admin interface.
 - (a) Each file type should have a default set of parameters being displayed at the files public page.
 - i. PDF: Title, ISSN/ISBN, Authors, Date published (in periodical or book), Abstract, keywords, bibtex-link, endnote-link
 - (b) Each public file page should have a set of links making it possible to share the link on a set of social websites e.g. facebook, twitter.
 - (c) Each public file page should have a small form to "Tip a friend" where one inputs the readers email, the friends email, and a short message.

Access control

1. The website should support IP Filtering for giving privileged access to products to specific IP ranges
 - (a) Add, edit and delete IP ranges
 - (b) Assign IP ranges to specific products, categories and all products
 - (c) IP range filtering down to 10 clients must be supported
2. Third party identity providers like Feide
 - (a) Add, edit and delete institutions
 - (b) Assign institutions to specific products, categories and all product

Customer accounts

1. Customers should be able to register
2. Login / Logout
3. The customer should be able to recover access to their account if they have lost their password
4. The website should store customers history
5. Customer should be able to view their purchases and order history
6. Customers should be able to receive notifications for new products
7. Customers should be able to administer their subscriptions on journals when logged on the website
8. It should not be necessary to create an account if just want to by a few articles
9. If the user has a IP which is in the range of an institution with privileged access, the user should be notified

Purchase process

1. Customers should be able to add products in a shopping cart
2. There should be a concatenation option regarding pdf's where the customer can add several files to a single pdf.
3. There should be a possibility of making a rule-base for pricing files, where one e.g. can get a discount of 10% when purchasing 10 articles, or that there is a price roof on the periodicals as one can subscribe to the entire set within a title for ca. 300NOK. This could be connected to the file groups in the admin interface.
4. The website should have a checkout section where the user can view all the products in the cart and the total price
5. The website should be integrated with a 3rd party payment solution
6. A watermark should be added to PDF documents before they are downloaded
 - (a) This watermark should consist of a brief text explaining that Tapir have the copy right for the material, the name/email of the person or institution
7. Accounting should receive an email with transaction details for each order
 - (a) Only one summary email per day should be sent at maximum
8. The customer must receive a digital receipt after payment
9. The customer can check a checkbox to get a receipt by email.
10. The customer should be able to download the product infinite many times 3 hours after the purchase.
11. The customer should get a link to where he can download the product(s) in an email after purchase as well as been offered to download the files right after the payment is complete

Site administration

1. counting and other administrators should be able to view transaction and order details
2. User administration
 - (a) Existing administrators should be able to add, edit and delete users
 - (b) Administrators should be able to give certain users access to parts of the site administration interface
 - (c) Administrators should be to give certain users free access to specific product or group of products

- (d) There should be two different admin-permissions read and read-write, where only the read-write permission will be able to alter the parameters in the admin section.
- (e) The read-users should be able to view statistics and alter the views in the statistics page.

3. Statistics

- (a) Administrators should view site traffic statistics
- (b) Administrators should be able to view detailed sales statistics for all product
- (c) There should be available statistics and analysis in the admin interface.
- (d) There should be an admin page displaying interesting parameters on the successfulness of the service:
 - i. Nr of downloads
 - ii. Visits (pageloads)
 - iii. Sales and left shopping carts sorted by date group and by file
- (e) It should be possible to navigate the statistics parameters to display different granularity regarding files, dates, and groups.
- (f) There should be a finite set of predefined statistics views easily accessible:
 - i. Downloads per month
 - ii. Downloads per week
 - iii. Visits per download per week
 - iv. Visits per file (Top 10)
 - v. Downloads per file (Top 10)
 - vi. Sales per file (Top 10)
- (g) There should be a detailed log of what is done in the admin interface writing out one line of textual description per action with time stamp and user.
 - i. Example: "19.10.2009 - 14.15: Testuser changed the price of filenr. 14, "TestdocumentTitle", from 14kr to 19kr.

4. Administrators should be able to set the current VAT to be used

Security

1. Password quality conform to these minimum requirements
 - (a) At least 8 characters
 - (b) Both upper and lower case letter
2. The password should be encrypted with the SHA algorithm when stored in the database
3. Documents must include protection from software piracy

- (a) The system must support DRM and Watermarking.
- 4. User information is not to be shared with third parts
- 5. The products that are not free should only be accessible for those who have paid for them
- 6. All the files stored on the web server must have proper file permissions so that none of the files are readable for others than apache and the development team

4.2 Non-functional requirements

- 1. The site should be available in both English and Norwegian
- 2. The website should be written in PHP, HTML and CSS.
- 3. The website should be optimized for search engine crawlers, and the process documented.
- 4. The code should be documented in line properly
- 5. The system must be highly modular so it will be easy to modify it later
 - (a) The parts that should be possible to exchange with relative ease is:
 - i. Public GUI
 - ii. Admin GUI
 - iii. Admin logic
 - iv. File handling (CMS, Mediabank)
 - v. Payment module
- 6. The online store must be available to international markets
- 7. The service must offer easy administration for the customer
- 8. The web site must look similar to the other Tapir pages <http://www.tapirforlag.no/>
- 9. The website should work properly in all the major web browsers
- 10. The GUI and interaction design should be as effective as possible, with as few clicks as possible to achieve goals, and screens should not contain more information than necessary.
- 11. It should be as easy as possible to add new IP ranges

CHAPTER 5

Sprint 1

Sprint duration: 21st September - 4th October (14 days)

This chapter documents the work done and the artifacts created throughout this sprint. This is the first of three sprints, and its purpose is to get started on the design and implementation of the system. The main objective for the project group in this period is to get a clear understanding of the architecture of the system, and implement some simple features of the system.

All the sprint chapters have the same generic structure. First of all the goals are given, as described in the sprint backlog. The main section of the chapter is the documentation of the work that was performed. As for sprint one, we have an own section for the design of the software architecture, and a section describing what was implemented. The last part of each sprint chapter gives a summary of what was done, including some comments about how we can improve our work process.

Chapter overview

The chapter of the first sprint contains the following sections:

- Chapter 5.1 Sprint plan
This section describes the planned work for the sprint.
- Chapter 5.2 Sprint backlog
This section gives the goals for this sprint that was decided at the planning meeting.
- Chapter 5.3 Design - Software architecture
This section has detailed descriptions of the design documents of the software architecture.
- Chapter 5.4 Implementation - Graphical User Interface
This section contains the paper prototype and some pictures of what was implemented.
- Chapter 5.5 Tests and results
A summary of the sprint; what tasks were completed and what was postponed.
- Chapter 5.6 Sprint evaluation
A retrospective of what went good and bad in the sprint and what can be done to improve the process.

5.1 Sprint plan

Sprint 1 starts after the customer meeting the 21. of September and ends the 4. of October, after a duration of two weeks. In this sprint we have put 248 hours. We will have a meeting with the customer in the middle of the sprint, in addition he also want to come to campus to see how we are working.

In this first sprint we aim to get more familiar with the programming language and the development framework we are going to use. This will make us more effective in later sprints even though we have to use some time on this now. Creating some early models of the system will also take place in this sprint, which we will do in communication with the customer to assure that we have the right understanding of the system. We need to define what modules to include in the system, as it will be implemented in parts.

Because we are not experienced with the development tools we are going to use, we have not put much implementing tasks in this sprint. Still we will create a graphical user interface to have something visual to show the customers at the end of the sprint. With this we will include some minor basic functionality. This way those at Tapir he do not read code can see we are getting somewhere too.

5.2 Sprint backlog

The sprint planning meeting was held with the customer present on the first day of the sprint, September the 21st. The product backlog, made at an earlier point and described in chapter 4, was the basis of this meeting.

At first we discussed how much time we had available for the sprint as a whole. An agreement was made that this sprint should not have too many tasks and items from the product backlog, as the majority of the group members are neither experienced with the PHP language nor the Zend framework. The time it will take to learn the programming languages and tools is a big uncertainty in this project, and this makes it difficult to set a time estimate for how much time it takes to implement something.

It was decided that user story 2 and 4 from the product backlog was the most important ones to implement first. In addition, we added item "0", which is to do the initial setup, plan the architecture and plan how the graphical user interface(GUI) should look like. In total we had three tasks. These tasks were divided into smaller subtasks, which we then could estimate an implementation time for. These tasks are listed in table 5.1.

The estimated time in total is 248 hours. In addition to attending lectures, supervisor meetings, customer meetings and report writing, this estimate is quite high. The reason for it being so high is that we have given each task a relatively big time estimate, as we are unsure about how much time it will take to learn things in the first place. The time it takes to implement the first backlog item will be high, but then we estimate each task after that to take less time when everything is already up and running. The majority of the items in the product backlog will be implemented in sprint two and three.

Story ID	Description	Estimated implementation time	
0	Setup and planning		168
	Task ID	Task description	Responsible
	0,1	Set up framework on webserver	Erik
	0,2	Plan the architecture and create UML diagrams	Mats
	0,3	Create a database model and SQL queries for the DB	Arne
	0,4	Plan the layout of the GUI and create style frame	Olav
2	As a customer I want to be able to register and use an account		50
	Task ID	Task description	Responsible
	2,1	Create a registration form	Joakim
	2,2	Implement login and logout	Erik
	2,3	Implement forgot password functionality	Mats
4	As an system admin I want to upload new products		30
	Task ID	Task description	Responsible
	4,1	Implement login and logout for admins	Arne
	4,2	Create an upload form for new products	Olav
	Total:		248

Table 5.1: Sprint 1 backlog

5.3 Design - Software architecture

5.3.1 Introduction

In the pre-study chapter we decided to use Zend Framework for developing the website and in this section we will explain how we have planned the architecture of the website.

5.3.2 Model View Controller Pattern

As explained in the pre-study chapter, Zend Framework is based on the MVC(Model View Controller) design pattern. With this pattern we get a clean separation of the code in three different parts: models, views and controllers. Here is a short explanation of the different parts:

- Model: Represents data.
- View: Presents the data. On a website the views are usually written in HTML and CSS.
- Controller: The controllers handles events and their effect on the data in the models and the way they are represented in the views.

The figure below displays how the MVC pattern works.

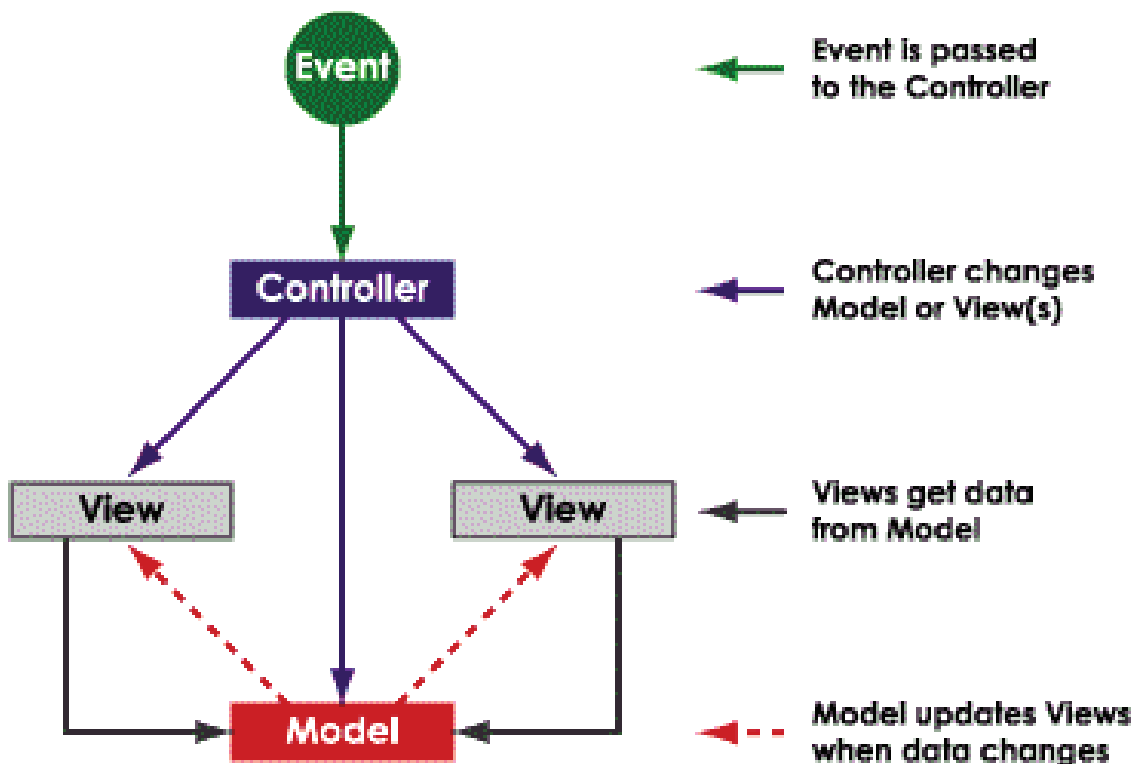


Figure 5.1: MVC pattern [4]

5.3.3 Modules

We decided to split up our website into different modules. Modules are different parts of a system that provide different functionality and could easily be added, modified and removed from the system. The reason why we decided to use this architecture is because the customer has ideas about expanding and changing parts of the system after we have made it. By using modules we insure that we have a better separation of the code with respect to what functionality the code actually gives. So if the customer wants to change just the payment system after we are done, he just needs to change/replace one module. The following are the modules in our system:

Product

The product module handles the management of products for the administrators, the browsing and viewing functionality of the products for the customers as well as the group hierarchy and attribute system.

This is the largest and probably the most important module. The group and attribute system could have been put in their own separate modules, but we think that this would have resulted in a very high coupling between these modules and therefore we kept everything in this product module.

Account

This module basically contains the account system. It provides basic functionality as login, logout, forgot password and registration as well as more advanced features like access control using account roles and IP ranges.

Order

The order module contains all the functionality tied to purchasing products i.e. creating an order.

Journal

The journal module contains functionality for creating journals, and new issues in existing journals.

Compendium

The compendium module contains the functionality of creating and sharing a collection of articles which we call a compendium.

Statistics

The statistics module contains the functionality of viewing detailed statistics of sales and the usage of the website.

5.3.4 UML Model Diagram

A class diagram of the most important classes in our system is given in Figure 5.2. Standard UML notation is used, where each class has a name, some attributes and some methods. All classes follow the same naming convention. First the module name is given, for example "product" or "journal". Then every class in this figure is named "Model", related to the MVC pattern described in the start of this section. The third part of the class name is the more typical identification of a class, for example "Account" and "AccountRole".

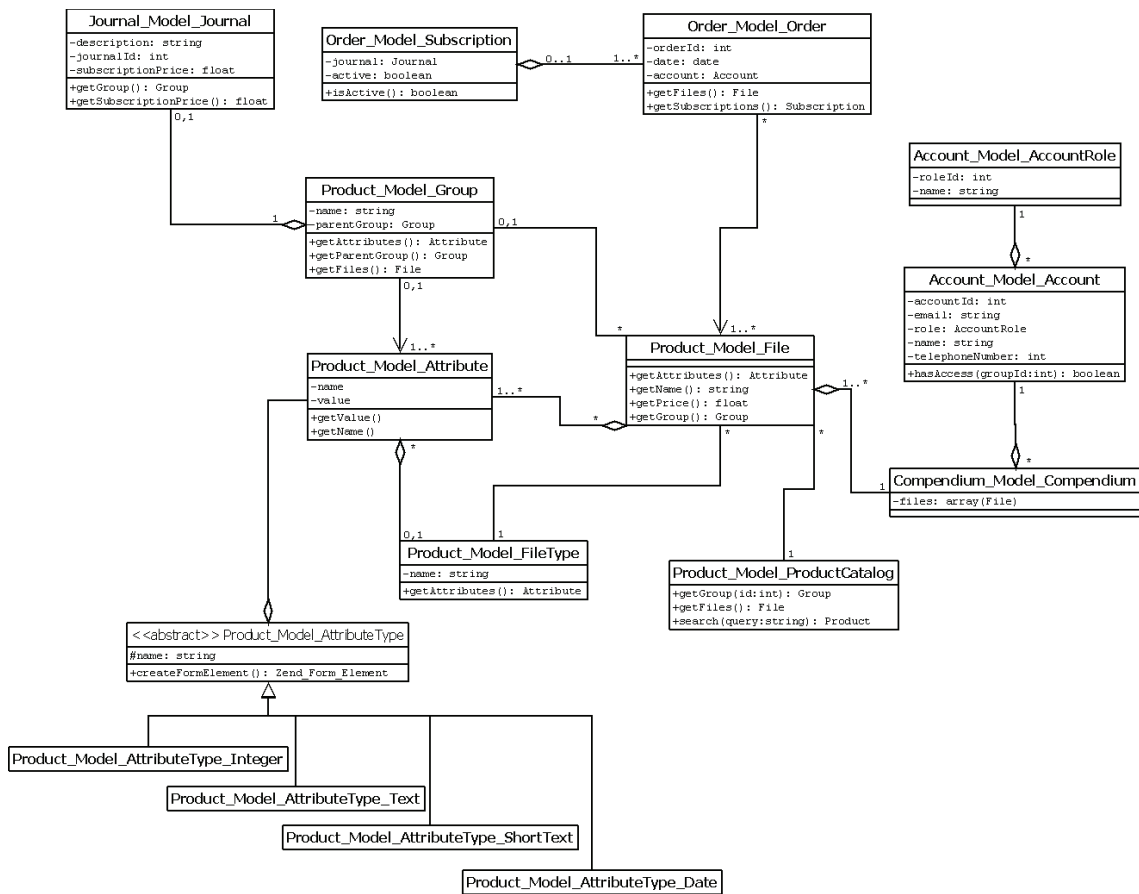


Figure 5.2: Class diagram of important classes

5.3.5 Database model

From all of the user stories and requirements we have collected, we made a complete database structure. An ER(entity relation) diagram of this structure is given below 5.3. The diagram is using the Crows foot notation, defined in [34]. Fields with a yellow key to the left are primary keys and those with a red star are foreign keys.

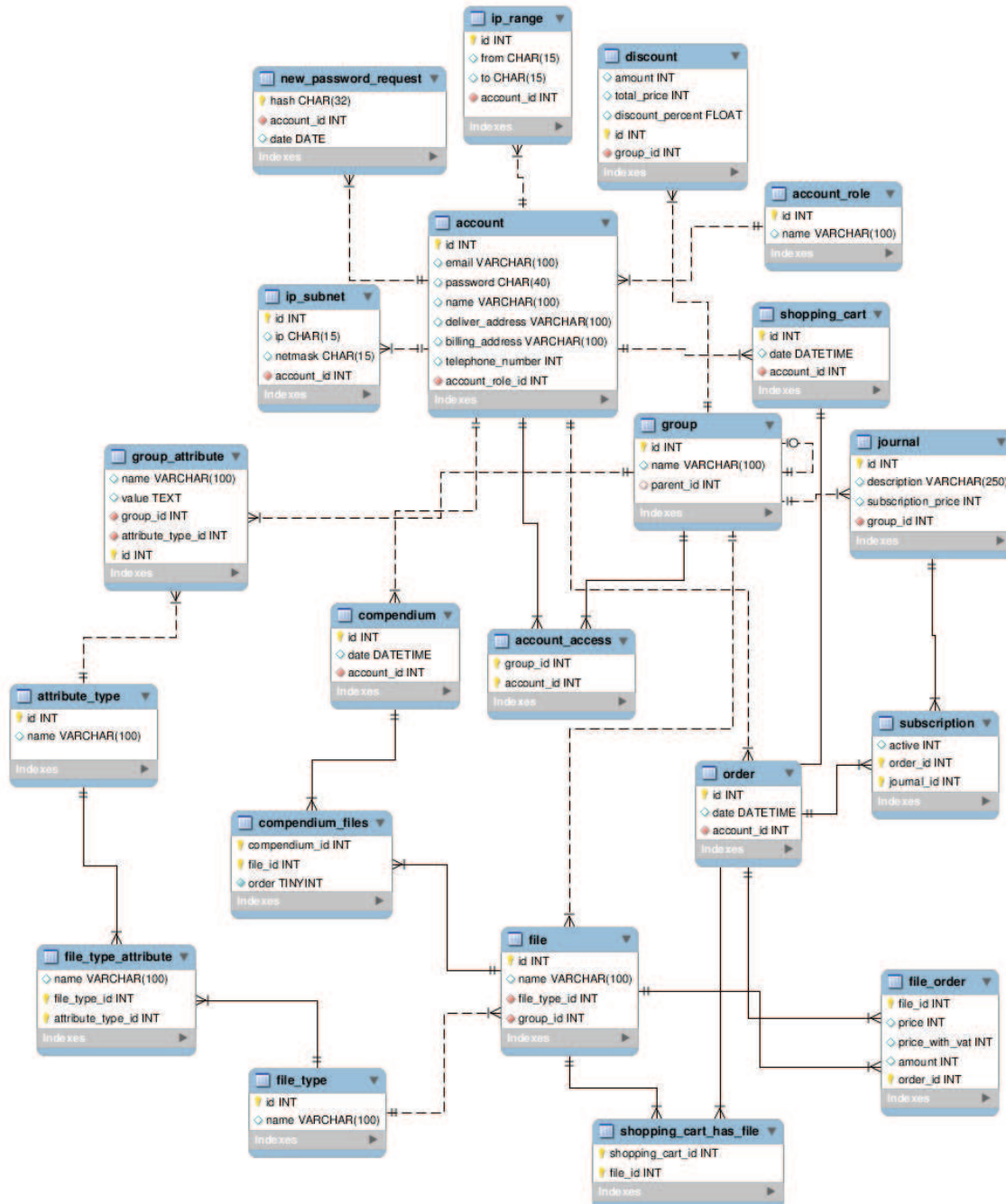


Figure 5.3: ER diagram of our database

5.4 Implementation - Graphical User Interface

The first thing we did in the sprint was to make a paper prototype of how the web site should look like. There were several reasons for making a prototype of the system before starting to implement. First of all we used the paper prototype to discuss with the customer, both during the process of making the paper prototype, but also afterwards. Secondly the prototype was a way for everyone to get the same "mental model" of the website. Just reading text about how the system should look like can easily lead to people having different ideas of how it looks like. A third reason was to have the prototype to look at while implementing, saving time when actually implementing.

All the scenarios made were related to the customer's use cases. We plan on designing the web page for the administrators in sprint 2. The paper prototype of the front page is given in figure 5.4. It is in Norwegian as we prioritized the Norwegian version of the web site in this sprint. The important part of the figure is the structure. On the top is the Tapir logo, a "contact us" button and a button to change to English. In the top left margin is the customer log in. Below that is shortcuts to journals, eBooks etc. In the top right margin is the shopping cart, and below that is room for advertisements. The middle of the has a search field, and then a list of the newest articles and journals. The yellow note in the corner is just used for comments on the prototype, and will not be implemented.

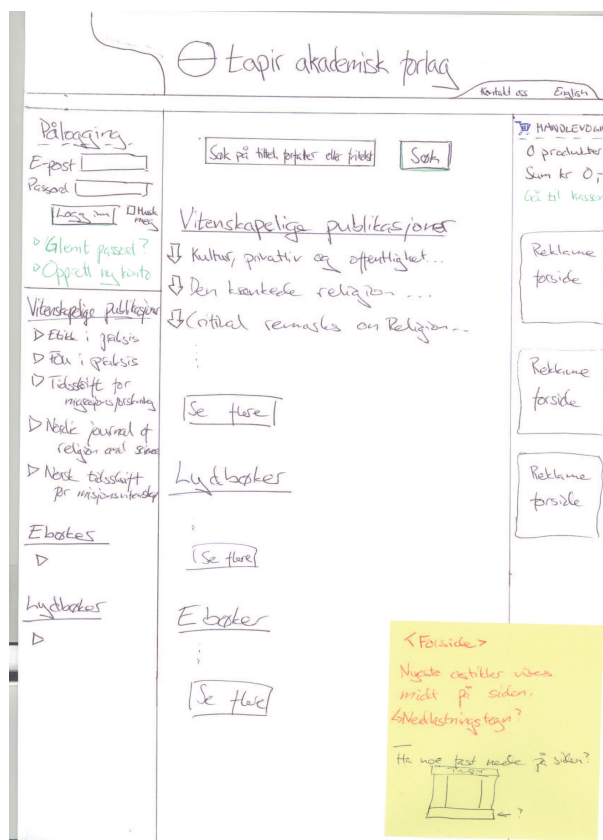


Figure 5.4: An early sketch of how the frontpage should look like

In total we made 6 figures of typical pages that the user might encounter. These sketches can be found in the appendix. The plan is that the top header, and the margins on the left and right side will be similar on all pages, and that only the center section of the page will change. All pages will have the search field except the page for a specific article, given in figure 5.7.

Figure 5.5 shows the site you will come to if you click on "more...scientific publications". The articles will be listed, with fields for title, date, journal, author and price. Initially the articles will be sorted on date, but the user can sort the articles on any field.

GUI implementation

Figure 5.10 up until 5.15 are screenshots of how the web site looks after sprint 1. As can be seen, only a small part of the web site is implemented. Still, the tasks in the backlog for this sprint are successfully implemented. One example is the user registration. When you register, the information is saved in the database and you can use the new account to log in on the page with the email and password that you decided. The registration form has a check that you provide a proper email, if the password has a length of at least 6 letters and that you provide all the mandatory information.



Figure 5.5: The front page of the web site



Figure 5.6: Website registration form



Figure 5.7: Registration complete



Figure 5.8: Logged in at the web page



Figure 5.9: Web site when you forget the password

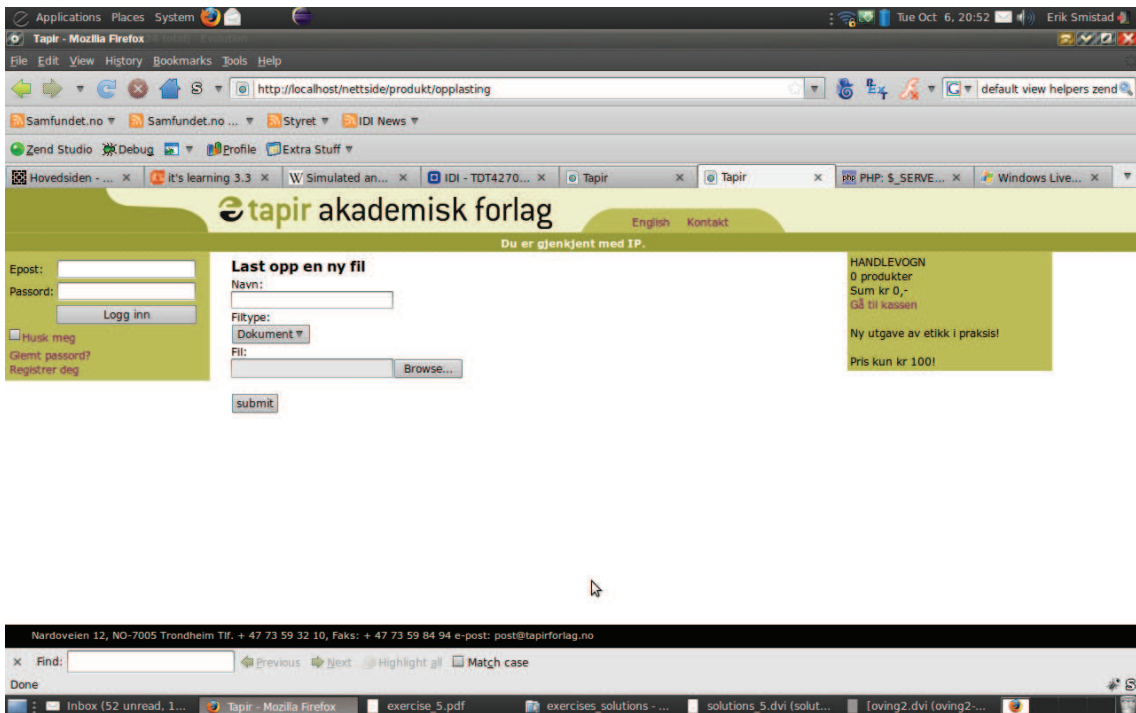


Figure 5.10: Administrator uploading new file

5.5 Tests and results

Due to the kind of implementing we have not done much testing in this sprint. Testing was done on the form that while we were designing the web site we checked what it looked like in the most common browsers. We also registered users and tested to log in and out. On the product uploading area we tested by uploading different types of files.

The sprint ended on October the 4th. The demonstration of the product was performed at the customer meeting on Monday the 5th. The three tasks we had in the sprint backlog were completed and we will not have to transfer any tasks to the next sprint. The materials presented to the customer were the database model, the class diagram, the paper prototype and the demonstration of the web site.

5.6 Sprint evaluation

We completed the tasks in the sprint backlog, using 10 hours less than planned. Table 5.2 is the same table as in the Sprint Backlog chapter, but not with the actual time used for each task. The most noticeable was the framework that took a lot more time than planned. Because of different hardware and operating systems, we had to make some changes for the web site to work on all the computers.

Story ID	Description	Estimated implementation time	Actual implementation time
0	Setup and planning	168	178
	Task ID	Task description	Responsible
	0,1	Set up framework on webserver	Erik
	0,2	Plan the architecture and create UML diagrams	Mats
	0,3	Create a database model and SQL queries for the DB	Arne
	0,4	Plan the layout of the GUI and create style frame	Olav
2	As a customer I want to be able to register and use an account	50	40
	Task ID	Task description	Responsible
	2,1	Create a registration form	Joakim
	2,2	Implement login and logout	Erik
	2,3	Implement forgot password functionality	Mats
4	As a system admin I want to upload new products	30	20
	Task ID	Task description	Responsible
	4,1	Implement login and logout for admins	Arne
	4,2	Create an upload form for new products	Olav
Total:		248	238

Table 5.2: Sprint 1 backlog with used time

The burndown chart is given in figure 5.11. As can be seen, we managed to work a little bit each day, except day 6 and 7, which was the first weekend. Day 10 and 11, before the second weekend, were very productive, leaving us with just some hours of work left before the weekend. Table 5.3 on the next page is the table used to make the burndown chart, where you can see the remaining time for each task in the sprint backlog.

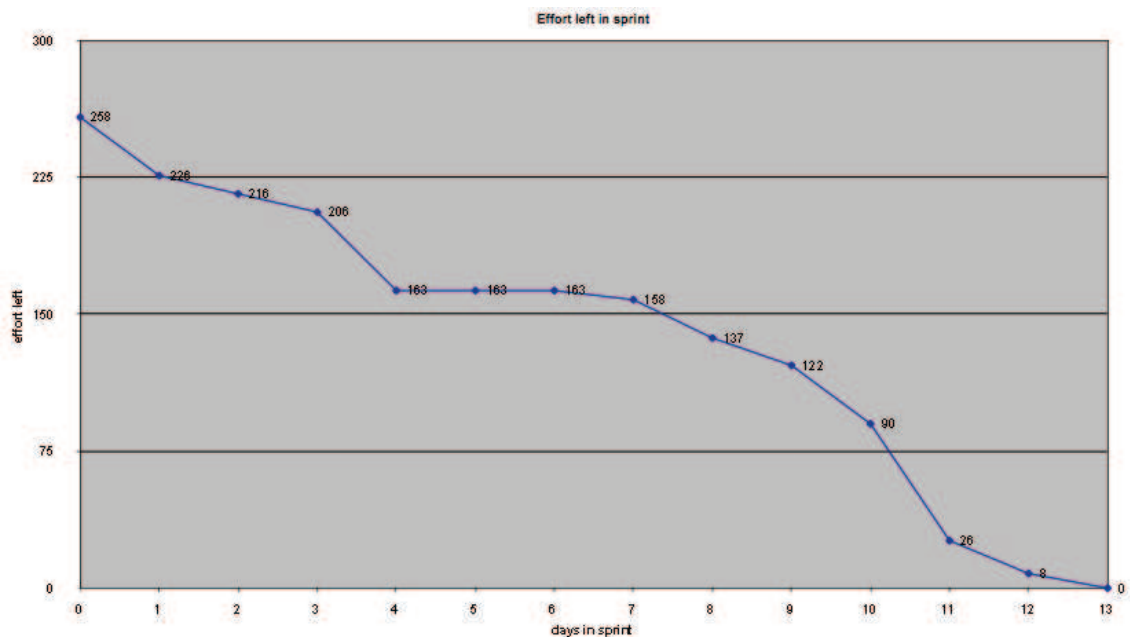


Figure 5.11: Sprint 1 burndown graph

Story ID	Story/task	days in sprint / effort left													
		0	1	2	3	4	5	6	7	8	9	10	11	12	13
0	Setup and planning	258	226	216	206	163	163	163	158	137	122	90	26	8	0
0,1	Set up framework on webserver	10	10	10	10	10	10	10	7	7	5	5	3	2	0
0,2	Plan the architecture and create UML diagrams	70	60	50	40	30	30	30	30	13	7	5	5	2	0
0,3	Create a database model and SQL queries for the DB	46	40	40	40	7	7	7	5	1	0	0	0	0	0
0,4	Plan the layout of the GUI and create style frame	52	36	36	36	36	36	36	36	36	30	20	0	0	0
2	As a customer I want to be able to register and use an account														
2,1	Create a registration form	10	10	10	10	10	10	10	10	10	10	10	10	0	0
2,2	Implement login and logout	10	10	10	10	10	10	10	10	10	10	10	2	0	0
2,3	Implement forgot password functionality	30	30	30	30	30	30	30	30	30	30	15	0	0	0
4	As a system admin I want to upload new products														
4,1	Implement login and logout for admins	10	10	10	10	10	10	10	10	10	10	10	2	0	0
4,2	Create an upload form for new products	20	20	20	20	20	20	20	20	20	20	15	4	4	0

Table 5.3: Sprint 1 burndown table

Positive experiences

- We have had a good dialogue with the customer
- The daily meetings eases the implementing

Negative experiences

- We had to go back in the report and add references and glossary

Ideas for improvement

- Put more hours into the project early in the sprint to avoid accumulation before the sprint deadline
- Get better at updating the hours used (for the burndown chart)

CHAPTER 6

Sprint 2

Sprint duration: 5th October - 18th October(14 days)

This chapter documents the work done and the artifacts created throughout this sprint. This is the second of three sprints, and its purpose is to further implement functionality in the system.

Chapter overview

The chapter of the second sprint contains the following sections:

- Section 6.1 Sprint plan
This section describes the planned work process of this sprint.
- Section 6.2 Sprint backlog
This section gives the goals for this sprint that was decided at the planning meeting.
- Section 6.3 Design
This section contains the updated database model.
- Section 6.4 Implementation
This section contains all the implemented functionality of the system.
 - Section 6.4.1 Products and groups
This section describes the group system used for products.
 - Section 6.4.2 Product search and browsing
This section describes how we implemented product browsing and searching.
 - Section 6.4.3 Admin interface
This section describes how we implemented the administration interface and setting account access.
 - Section 6.4.4 IP range for institutions
This section describes the use of IP filtering to recognize users that are using computers on an institution with specific IP ranges/subnets.
- Section 6.5 Tests and results
A summary of the sprint; what tasks were completed and what was postponed.
- Section 6.6 Sprint evaluation
A retrospective of what went good and bad in the sprint and what can be done to improve the process.

6.1 Sprint plan

Sprint 2 starts with the planning meeting on October 5th and end with the sprint review on October 19th, a total of 14 days. The number of days in the sprint is the same as the other sprints, which equals a budget of 240 hours for the group in total. During this period a regular customer meeting is scheduled in the middle of the sprint, and email will be used for minor questions and issues.

What is different from the first sprint is that group members have more obligations in other classes and activities in the weekends. We will take this into account while planning, in addition to the usual risk factor of people getting sick. Nevertheless, we still plan on working at least the same amount of hours as the first sprint. If worst case scenario happens, we will have to transfer the tasks we can't finish to the next sprint.

This sprint will have a lot more implementation tasks than the first. We plan to sit together as a group and work as much as possible, as it will ease communication and motivation while implementing. The plan is to have one group member responsible for each task in the sprint backlog. That person will do most of the implementation of that specific task and will have the responsibility of writing about it in the report afterwards. We plan to get everyone in the group involved in the programming, instead of just having a few persons implementing and the others writing the report.

6.2 Sprint backlog

In sprint 1 we completed two out of sixteen items in the product backlog. A lot of time was used on just getting things started; learning the Zend framework and getting the programming environment up and running on each group member's PC. This sprint we plan on completing seven more items, as can be seen in table 6.1, with time estimate for each item. These items were picked from the product backlog according to the priorities given by the customer. Thought was given to which items are smarter to implement before others, as some product items have a strong connection to others.

We estimate that the total time needed for all the items in this sprint backlog is 236 hours. This is quite a lot, considering we will have to attend meetings and write in the report as well. We do, however, feel that the estimate reflects the amount of work we need to put into this sprint, which is a lot. We predict that the time estimate can vary a lot, as most group members still have little experience with the programming language and this making it harder to get good estimates.

Story ID	Description	Estimated implementation time
1	As a customer I want to browse through all the products	31
	Task ID Task description Responsible	
	1.1 Implement product catalog class	7
	1.2 Create product browsing controller	12
	1.3 Create views for browsing	10
	1.4 Testing	2
3	As a customer I want to search for products	19
	Task ID Task description Responsible	
	3.1 Create search sql	12
	3.2 Create view for displaying results	5
	3.3 Testing	2
5	As a system admin I want to administrate the attributes of the products	55
	Task ID Task description Responsible	
	5.1 Create forms for controlling attributes of products	30
	5.2 Create methods for retrieving group and file type attributes	25
	5.3 Testing	2
6	As a system admin I want to administrate IP ranges for institutions	31
	Task ID Task description Responsible	
	6.1 Create forms for controlling IP ranges	7
	6.2 Check users IP address against IP range	14
	6.3 Connect IP range to accounts and groups	8
	6.4 Testing	2
7	As a system admin I want to control the access of all the users of the system	36
	Task ID Task description Responsible	
	7.1 Create an overview of all users in admin interface	12
	7.2 Implement account roles	10
	7.3 Create forms for giving users access to certain groups	7
	7.4 Create form for adding new users and giving them roles	7
	7.5 Testing	2
8	As a system admin I want to manage groups of products	36
	Task ID Task description Responsible	
	8.1 Create functionality for adding products to groups	18
	8.2 Create functionality for creating new product groups	18
	8.3 Testing	2
11	As a registered customer I want to be able to subscribe to journals	28
	Task ID Task description Responsible	
	11.1 Let admin create journals and issues	10
	11.2 Implement functionality for subscribing to journals	8
	11.3 Implement functionality for unsubscribing to journals	8
	11.4 Testing	2
	Total:	236

Table 6.1: Sprint 2 backlog

6.3 Design

The database model was changed during sprint 2. The major change was the added tables to support attribute types, namely short text, text, int and date. Another change was the added table of "IP subnet", which we needed when implementing the IP filtering. In addition to these two major changes, some fields in the tables were added.

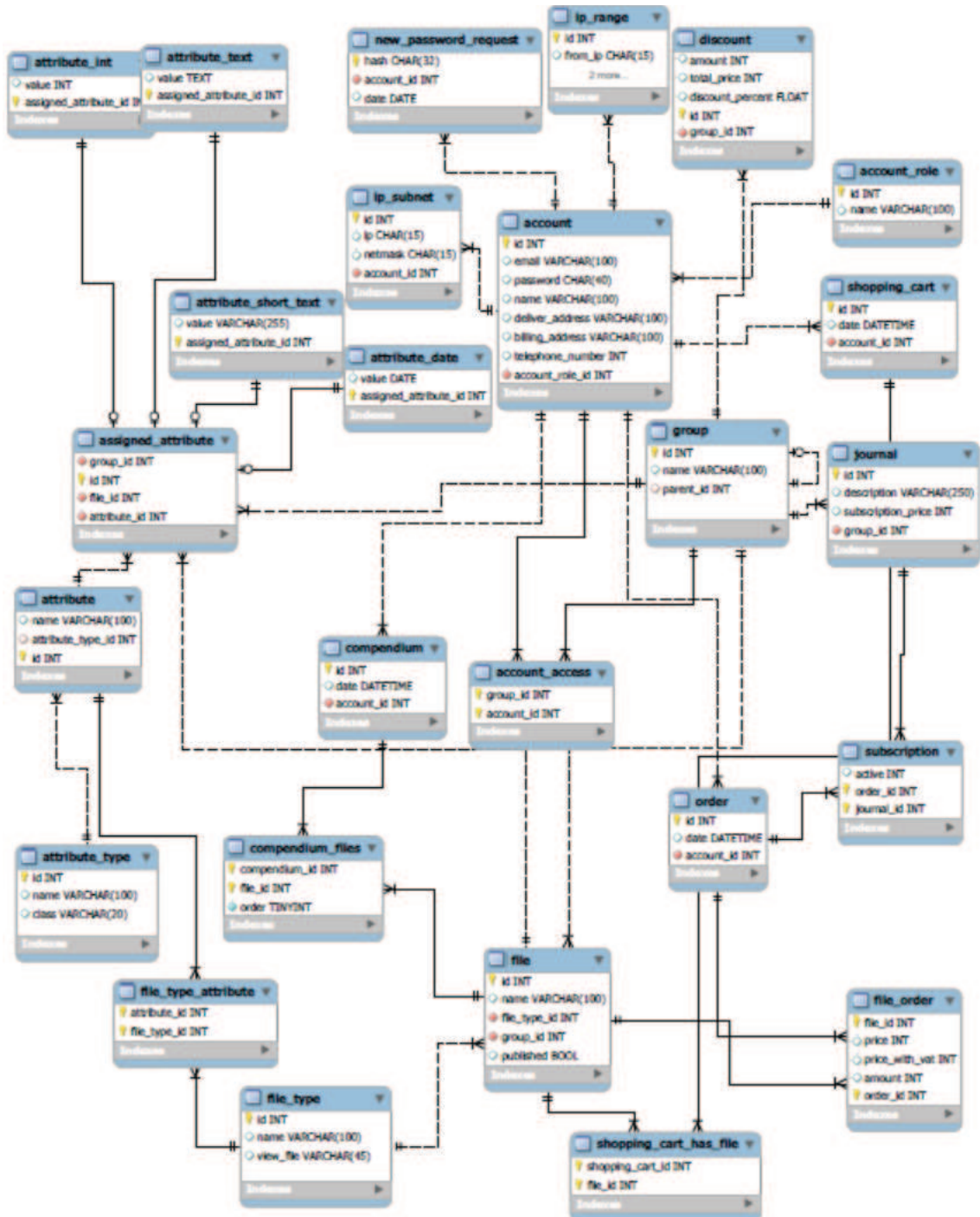


Figure 6.1: The updated database

6.4 Implementation

6.4.1 Products and groups

This was one of the major tasks in this sprint. We needed to implement the slightly complex attribute system together with the group system. The reason why this was such a complex task was because products and groups should be able to inherit attributes from parent groups, and on top of that a child should be able to override a parent group's attribute.

The way this system is built up is that there is one attribute table which is a list of all the available attributes and an `assigned_attribute` table which is a list of all the attributes assigned to different files and groups with a specific value. Since we have different types of values, currently ShortText, Text, Integer and Date, we store the values themselves in separate tables, but they have the same primary key as the entry in `assigned_attribute`. This solution was a little bit different from what we had planned to implement in the start. The major difference is that now the website defines the attributes instead of the user. This made the implementation much simpler and the usability of the system much higher as it became simpler to use. Below is a screenshot of the product overview. This overview gives the admin a list of the published and unpublished products as well a search box to find products.



Filer
Legg til en ny fil
Søk etter en fil:

Upubliserte filer

Navn	Type	Publisjer	Rediger	Slett
FoU-leder 1-08	Vitenskaplige artikler	Publisjer	Rediger	Slett
test for tapirforlag.no	Vitenskaplige artikler	Publisjer	Rediger	Slett
Teknisk Utvikling	Vitenskaplige artikler	Publisjer	Rediger	Slett
Vitenskap	Vitenskaplige artikler	Publisjer	Rediger	Slett

Publiserte filer

Navn	Type	Depublisjer	Rediger	Slett
Lær middel å kjenne	Læremidler	Depublisjer	Rediger	Slett
NavnEksemple	E-bøker	Depublisjer	Rediger	Slett
Prosjekt i tidligfasen – valg av konsept	Vitenskaplige artikler	Depublisjer	Rediger	Slett
Student som læremiddel	Læremidler	Depublisjer	Rediger	Slett

Figure 6.2: Product overview

Figure 6.3 on the next page is a screenshot of a products details. This page shows all the attributes that a file has, which group it is part of. It also presents options for publishing/depublishing files as well as adding, deleting and editing attributes. Figure 6.4 is a screenshot of the group details page. It shows which files and attributes the group has and presents the administrator with options for adding, editing and deleting both attributes and files. Figure 6.5 is a screenshot of the group overview which shows the group hierarchy and options to add new groups and subgroups.

tapir akademisk forlag English Kontakt
Din Ip adresse er 127.0.0.1

Filen: TestNavn
Medlem av gruppen: [Undergruppe](#)
Trykk her for å endre gruppe
Publiser denne filen
Rediger denne filen
Slett denne filen

Attributter
Legg til attributter til denne filen

Navn	Type	Verdi	Arvet fra gruppen	Rediger	Slett
Annet	Tekst	Sidekommentar til alle produkter i denne...	Supergruppe - Overstyr	Rediger	Slett
Pris	Tall	200	Ikke arvet	Rediger	Slett
Forfattere	Kort tekst	Terje og Anne	Ikke arvet	Rediger	Slett

Figure 6.3: Products details

tapir akademisk forlag English Kontakt
Din Ip a

Grupper
Lag en ny gruppe

Navn	Legg til undergruppe	Slett
Supergruppe	Legg til undergruppe	Slett
- Undergruppe	Legg til undergruppe	Slett
- - Test	Legg til undergruppe	Slett
- Undergruppe	Legg til undergruppe	Slett
- - Testgruppe	Legg til undergruppe	Slett

Figure 6.4: Group overview

tapir akademisk forlag English Kontakt
Din Ip adresse er 127.0.0.1

Gruppe: Undergruppe
Foreldre gruppe: [Supergruppe](#)
Rediger denne gruppen
Slett denne gruppen

Attributter
Legg til attributter til denne gruppen

Navn	Type	Verdi	Arvet fra gruppen	Rediger	Slett
Annet	Tekst	Sidekommentar til alle produkter i denne...	Supergruppe - Overstyr	Rediger	Slett

Filer i denne gruppen
Legg til en ny fil i denne gruppen
Legg til en eksisterende fil i denne gruppen

Navn	Type	Fjern fra gruppe	Rediger	Slett
TestNavn	Vitenskapelig artikkel	Fjern fra gruppen	Rediger	Slett

Figure 6.5: Group details

6.4.2 Product search and browsing

This task was to implement functionality for browsing through the products on the web site. The user clicks on a category in the left menu and in the content column in the middle the products should appear in a reasonable way. First a list should appear of products. Then the user can click on a product to get a page with more information of the clicked product.

When the products is showed to the user, which file type the product is must be taken into consideration. Therefore different views for each file type are used for the products. The customer gave requirements on exactly which attributes was relevant for which file types. The sound files should present their length in hours, minutes and seconds, while an e-book chapter(s) should display its page interval. This way only relevant attributes are displayed.



Figure 6.6: Sprint 2 backlog

Attributes are typically set by administrators when the product is uploaded. He then writes in information in a form of attributes where he finds it necessary. To prevent empty attributes from being brought to screen, a check that the attribute is not empty was done. If the attribute was empty, it is skipped and the next is checked.

Standard functionality for browsing was implemented. Some of these things are pagination, link for downloading/buying the product

Later we will add pictures to the product for a more interesting appearance. Functionality regarding the buying of the products will be implemented in sprint 3.

The screenshot shows a web interface with a green header bar. The header contains the text "Din ip adresse er 2001:700:300:2121:c17" and a small "En" icon. Below the header is a sidebar with a "eg" button. The main content area displays the title "Den store Vitenskapsartikkel" and a list of attributes in a table format. Below the table is a "Kjøp" button and an "Abstract" section with a paragraph of text.

Den store Vitenskapsartikkel	
Forfattere	Jon og Kari Aasen
Pris	145 kr
Kategori	Kategoritest
Utgiver	Tapir
Utgave	2
Publiseringsdato	2007-10-13
ISSN/ISSN2	4353452334535/234324324
ISBN/ISBN2	3212345345324/3453425345345
DOI	234324324324

Kjøp

Abstract
 Følgende er abstract. Følg lenken «Rediger» fra ditt dagnadsforslag og tilf av dagnaden. (ikke nødvendig å signere, da det skjer automatisk. Når du g en f for fullført krevet ut som en sekundær parameter slik {{Dagnadsforsl dagnaden viser seg å fungere mot sin hensikt, kan dagnaden avbrytes. D sekundær parameter slik {{Dagnadsforslag|Navn på siden|a}}).

Figure 6.7: View of an article

Searching is executed by doing SELECT statements in the database. A plain, simple search field will be in the left margin, and an advanced search will be on its own page. In the advanced search it will be possible to search on a specific file type or a specific parameter, like author, or both at the same time. Normally the search will return the most relevant hits first, but it will also be possible to sort alphabetically. We use "pattern matching" which counts the number of hits a word gets in a text. The pattern matching does not search for words that are only three characters or less.

tapir akademisk forlag
Din ip adresse er 80.213.

Søk:
Søk

Epost:
Passord:
Logg inn
 Husk meg
Glemt passord?
Registrer deg

Avansert søk
Søk på ordene:
test
Filtype:
Alle
Parameter:
Alle
Sorter resultatene etter:
 Relevance
 Alfabetisk stigende
 Alfabetisk synkende
Resultater per side:
10

Utfør søk

Vitenskapelig artikkel

Test
Ebok

Test i ebok
Den store bok
Johan Jensen, Silje Hagen og Samuel Henkartier
1999

Nardoveien 12, NO-7005 Trondheim Tlf. + 47 73 59 32 10, Faks: + 47 73 59 84 94 e-post: post@tapirforlag.no

Figure 6.8: Advanced search

6.4.3 Admin interface

This section documents how the administrator interface was implemented and how it is connected to the account module. It belongs to User Story 7: "As a system admin I want to control the access of all the users of the system" from the sprint 2 backlog.

Access to the administrator sites

The account object that is fetched from the database when a user logs on to the site, contains an attribute that is called `account_role_id`. This tells us whether the person logging on is a regular customer, an admin or an institution. By having this attribute we can separate between the different roles, and adjust the menu when users with different roles logs on. When an admin logs in he gets an additional menu in the left bar. There the admin can choose to get an overview of the users in the system, or add users to the system.

The user overview

Oversikt over brukere

Brukernavn	Navn	Rolle	Rediger bruker	Slett bruker
sodd@sodd.eu	sodd	Admin	Rediger	Slett
testbruker@testbruker.com	testbruker	Admin	Rediger	Slett
joakim@joakim.no	Joakim	Superadmin	Rediger	Slett
sodd@snus.com	testbruker	Kunde	Rediger	Slett
sodd@lolz.com	daad	Superadmin	Rediger	Slett
admin@sodd.com	soddadmin	Superadmin	Rediger	Slett
lol@lol.com	ffsdfs	Superadmin	Rediger	Slett
joa@joakim.no	Joki	Superadmin	Rediger	Slett
ntnu@ntnu.no	NTNU	Institusjon	Rediger	Slett
nytest@nytest.com	nytest	Kunde	Rediger	Slett
nyadmin@nyadmin.com	nyadmin	Admin	Rediger	Slett
sodd@sodd.no	dsfsf	Institusjon	Rediger	Slett
hei@hei.no	dsfsf	Institusjon	Rediger	Slett
hei@hallo.no	dsfsf	Institusjon	Rediger	Slett
hei@hallo.com	dsfsf	Institusjon	Rediger	Slett
hei@hallo.eu	dsfsf	Institusjon	Rediger	Slett
hei@hallo.se	dsfsf	Institusjon	Rediger	Slett
ntnu@idi.no	NTNU Nytest	Institusjon	Rediger	Slett
kunde@kunde.no	Vanlig Kunde	Kunde	Rediger	Slett
UO@UO.no	UO	Institusjon	Rediger	Slett
yngre.syrteit@tapir.no	Yngve Syrteit		Rediger	Slett

Legg til ny bruker
Hjem

Figure 6.9: Overview of users

The site with overview of the users contains a table of the users, with the most important attributes listed. For every user in the table two links are created per row, one for editing the users information, and one for deleting the user. So the admin has control over the most important aspects of managing the users directly from this page. Typical tasks for the admin can be to set the roles for a user. Since only the admin can change roles for other users, new admins must be "approved" by an existing admin to get access to admin sites. This will also work as extra security for the system.

Adding a new user

Legg til ny bruker

Fullt navn: *

Epost: *

Passord: *

Må bestå av minst 6 tegn

Gjenta passord: *

Telefon:

Faktura-adresse

Gate: *

Postnr: *

Poststed: *

Leverings-adresse

Gate:

Postnr:

Poststed:

Rolle: *
Kunde ▾

[Avbryt](#)

Figure 6.10: Adding a new user

When the admin follows the "add user" link from the overview page, he is directed to the page seen in the picture. Here he can fill in information for the new user, and choose which role the user should have. Equal to the user registration page, some fields are required and some is not. We need a basic information pool about the customer, but then again the customer should not feel that he or she has to fill out an endless form of information. When saving the user into the database, the admin is redirected to the user overview page. If the role chosen is "Institution", he will be redirected to a page for specifying the IP range. More information about this will come in the following section called "IP range for institutions".

Editing existing users

If an admin clicks on the edit user link in the table row for a user, he is redirected to the edit user page, which gives the admin the possibility to change personal information, role and IP ranges of users. Here we have the same form as for adding new users, but the form is filled out with the data currently existing for this user, by sending the `account_id` attribute with the redirect. So if the admin only wants to edit one attribute for the user, this should be a fairly quick task when not having to specify everything all over again. When it comes to redirecting here, the same goes as for the add user page.

Rediger bruker

Fullt navn: *

Epost: *

Passord:

Må bestå av minst 6 tegn

Gjenta passord:

Telefon:

Faktura-adresse

Gate: *

Postnr: *

Poststed: *

Leverings-adresse

Gate:

Postnr:

Poststed:

Rolle: *

[Avbryt](#)

Figure 6.11: Editing an existing user

6.4.4 IP range for institution

This part implements user story 6: As a system admin I want to administrate IP ranges for institutions. When a new user or existing user is added/edited with role as institution, we have a special redirecting for these accounts. The admin will, when saving personal information, be redirected to the IP range page for institutions, by redirecting with the account_id attribute. The filtering supports 2 formats. One can

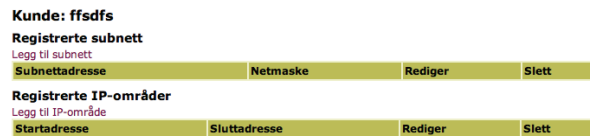


Figure 6.12: IP filtering overview for institution

either specify the starting subnet address accompanied by the subnets netmask or the administrator can specify the network range by entering the starting and ending IP address. The functionality supports multiple subnets and ranges per institution.



Figure 6.13: IP Subnet for institution

Customer access control

Each time a new client accesses the site their IP is checked against entries in the IP range and IP subnet table. If the address is located in one of those the client is recognized as coming from this specific institution. The system sets a session variable that follows the client. The session variable is used when checking against product pricing (ex free for all customers connected from an institution) and checkout. This is done because the customer wants to keep track of the transaction history for both user (if logged in) and institution.

6.5 Tests and results

The testing of sprint 2 was mainly unit testing that was performed while implementing. This is called white box testing, where the tester has access to the internal data structures and the code that implement it. The unit testing is testing specific small parts of the system. An example is the search function. The testing of it was performed with different search words and the output was compared to what was expected.

The demonstration of the sprint results was held on Monday the 19th of October. The group member responsible for each backlog item showed the customer how it was implemented. The delivery was everything put together and uploaded on a site that the customer could access. By doing this, the employees at Tapir had a chance to look at the site, and can give feedback which in turn can be used to tweak and improve the site at the next sprint.

The tasks that were finished in the sprint were:

- 1: As a customer I want to browse through all the products
- 3: As a customer I want to search for products
- 5: As a system admin I want to administrate the attributes of the products
- 6: As a system admin I want to administrate IP ranges for institutions
- 7: As a system admin I want to control the access of all the users of the system
- 8: As a system admin I want to manage groups of products

The task not started was:

- 11: As a registered customer I want to be able to subscribe to journals

6.6 Sprint evaluation

We had a lot of work planned for this sprint, and we ran into some time trouble. In the end we finished 6 out of 7 items from the sprint backlog, and we are satisfied with that. As can be seen in the table 6.2 we managed to do almost as much as planned, with some extra hours used on the report and meetings that are not included in the table. We used a little more time in general on each task, so we ended up running out of time in the end of the sprint.

Sprint 2				
Story ID	Description		Estimated implementation time	Actual implementation time
1	As a customer I want to browse through all the products		31	33
	Task ID	Task description	Responsible	
	1.1	Implement product catalog class	Joakim	9
	1.2	Create product browsing controller	Joakim	10
	1.3	Create views for browsing	Joakim	12
	1.4	Testing	Joakim	2
3	As a customer I want to search for products		19	25
	Task ID	Task description	Responsible	
	3.1	Create search sql	Olav	14
	3.2	Create view for displaying results	Olav	8
	3.3	Testing	Olav	3
5	As a system admin I want to administrate the attributes of the products		55	53
	Task ID	Task description	Responsible	
	5.1	Create forms for controlling attributes of products	Erik	27
	5.2	Create methods for retrieving group and file type attributes	Erik	23
	5.3	Testing	Erik	3
6	As a system admin I want to administrate IP ranges for institutions		31	37
	Task ID	Task description	Responsible	
	6.1	Create forms for controlling IP ranges	Mats	12
	6.2	Check users IP address against IP range	Mats	16
	6.3	Connect IP range to accounts and groups	Mats	7
	6.4	Testing	Mats	2
7	As a system admin I want to control the access of all the users of the system		36	33
	Task ID	Task description	Responsible	
	7.1	Create an overview of all users in admin interface	Arne	11
	7.2	Implement account roles	Arne	12
	7.3	Create forms for giving users access to certain groups	Arne	0
	7.4	Create form for adding new users and giving them roles	Arne	8
	7.5	Testing	Arne	2
8	As a system admin I want to manage groups of products		36	43
	Task ID	Task description	Responsible	
	8.1	Create functionality for adding products to groups	Erik	22
	8.2	Create functionality for creating new product groups	Erik	19
	8.3	Testing	Erik	2
11	As a registered customer I want to be able to subscribe to journals		28	0
	Task ID	Task description	Responsible	
	11.1	Let admin create journals and issues	All	0
	11.2	Implement functionality for subscribing to journals	All	0
	11.3	Implement functionality for unsubscribing to journals	All	0
	11.4	Testing	All	0
	Total:		236	224

Table 6.2: The sprint backlog with hours used

As can be seen on the burndown chart on the next page, we managed to work steady throughout the sprint, putting in some hours each day. We could have worked some more the first week, but it is expected to have some variation in the time each member can work. This is due to other classes and arrangements that need an extra effort some weeks. As the burndown chart shows, we had about 35 hours left of working, which is the last item in the backlog and some fine tuning on the other tasks.

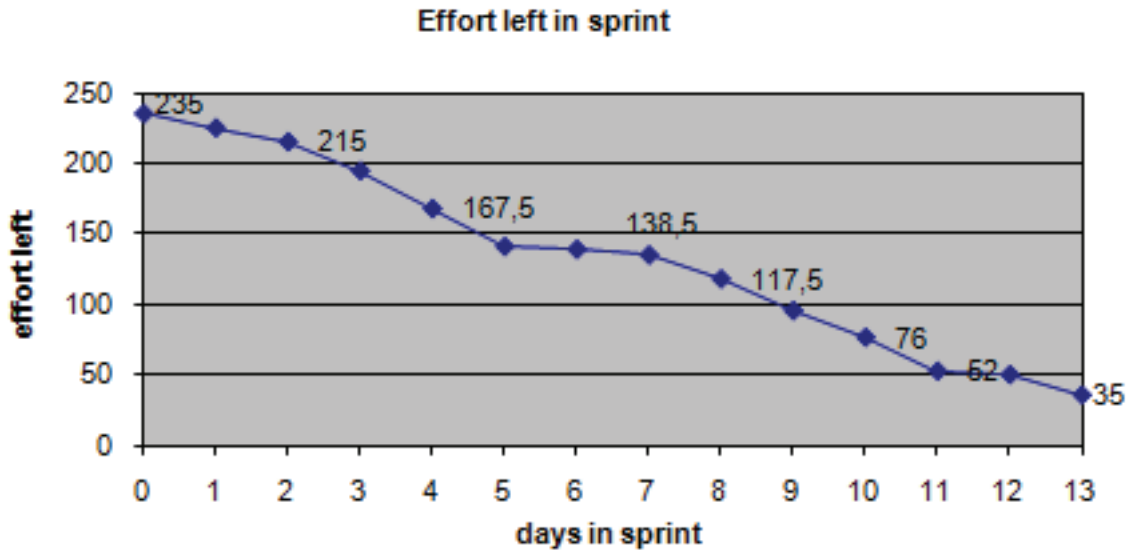


Figure 6.14: The burndown chart of this sprint

Positive experiences

- Managed to do most of the tasks, it felt like we had good progress with the project
- Still have a very good group chemistry, everyone is contributing
- The contact with the customer feels very good

Negative experiences

- Could have worked more in the start of the sprint, 4 out of 5 group members had under the 24 hours which should be the aim for a week
- We could improve the documentation done while implementing to ease the writing of the report when the sprint is finished

Ideas for improvement

- Use more hours on the sprint
- Document as we implement

CHAPTER 7

Sprint 3

Sprint duration: 19th October - 1th November(14 days)

This chapter documents the work done and the artifacts created throughout this sprint. This is the third sprint, and its purpose is to implement the remaining functionality.

Chapter overview

The chapter of the third sprint contains the following sections:

- Chapter 7.1 Sprint plan
This section describes the planned work process of this sprint.
- Chapter 7.2 Sprint backlog
This section gives the goals for this sprint that was decided at the planning meeting.
- Chapter 7.3 Design
This section contains the updated database model.
- Chapter 7.4 Implementation
This chapter contains all the implemented functionality of the system.
 - Chapter 7.4.1 My account interface
This section describes the customers account page with personal information and order history.
 - Chapter 7.4.2 Journals
This section describes the use of journals and how customers can subscribe to journals.
 - Chapter 7.4.3 Discounts
This section describes how discounts can be set on files or on total costs of a shopping cart.
 - Chapter 7.4.4 Watermarking of PDF files
This section describes how watermarking of PDF files was implemented.
 - Chapter 7.4.5 Checkout process and PayEx integration
This section describes the checkout process and the payment process with PayEx.
 - Chapter 7.4.6 Statistics
This section describes implementation of traffic and sales statistics.
- Chapter 7.5 Tests and results
A summary of the sprint; what tasks were completed and what was postponed.
- Chapter 7.6 Sprint evaluation
A retrospective of what went good and bad in the sprint and what can be done to improve the process.

7.1 Sprint plan

Sprint 3 started with the planning meeting on Monday the 19th of October and will end with the sprint review on November the 2nd. The duration is, as before, 2 weeks with a budget of roughly 240 hours.

This was supposed to be the last sprint, but we decided to add a fourth sprint lasting one week after this one. At the start of the project, we thought the project report would have to be handed in some time before the final demonstration, for example one week before. This turned out to not be the case, and that's why we feel we will have some extra time after sprint 3 is finished. This will increase the value of the product for the customer, which is one of the major goals of this project.

The main purpose of this sprint is to finalize the web system we have been working on. This means that whatever functionality not implemented in the end of this sprint, will not be delivered in the final product. The purpose of the fourth sprint will be to "fine tune" the web site, and especially the appearance of it. There is a lot to do this sprint, but we feel we are working faster now as well, primarily because everyone in the group has learned more about the Zend framework we are using.

7.2 Sprint backlog

From sprint 2 we had one backlog item that we did not have time to implement - customer subscribing to journals. In sum there are 8 items remaining in the product backlog. We have decided to include all these 8 items in the sprint 3 backlog. As can be seen in table 7.1, the estimated time to implement everything is about 290 hours. It is very likely that we will not be able to finish all the sprint items, but in agreement with the customer we decided to include all these items for this sprint and see how much we get done. Table 7.1 has the backlog items ranked by priority, with the highest priority listed first. We are at this point unsure about how accurate the time estimates are, and that is why we think it is best to include all the remaining to the sprint backlog, just in case some parts go faster than expected. The current situation is that if we work about 290 hours in this sprint we should be able to finish all the items in the sprint backlog.

Sprint 3		19. October - 1. November		
Story ID	Description			Estimated implementation time
10	As a customer I want to be able to purchase products (H)			48
	Task ID	Task description	Responsible	
	10.1	Create a checkout qui	Erik	20
	10.2	Integrate Payex payments solutions with the website	Erik	20
	10.3	Testing	Erik	8
13	As a customer I want to be able to collect products in a shopping cart (M)			24
	Task ID	Task description	Responsible	
	13.1	Implement shopping cart	Mats	10
	13.2	Integrate the shopping cart with the checkout process	Mats	8
	13.3	Implement saving of shopping carts	Mats	4
	13.4	Testing	Mats	2
11	As a registered customer I want to be able to subscribe to journals (H)			56
	Task ID	Task description	Responsible	
	11.1	Let admin create journals and issues	Joakim	20
	11.2	Implement functionality for subscribing to journals	Joakim	16
	11.3	Implement functionality for unsubscribing to journals	Joakim	16
	11.4	Testing	Joakim	4
9	As an administrator I want to see statistics for the website (M)			46
	Task ID	Task description	Responsible	
	9.1	Create an admin interface for viewing statistics	Olav	8
	9.2	Create SQL queries for generating the statistics	Olav	14
	9.3	Generate graphs from the statistics	Olav	20
	9.4	Testing	Olav	4
15	As a customer I want to be able to make compendiums (L)			14
	Task ID	Task description	Responsible	
	15.1	Implement an option for making a shopping cart into an compendiu	Mats	4
	15.2	Make it possible to share the compendium	Mats	6
	15.3	Testing	Mats	4
12	As an admin I want the products to be watermarked (M)			28
	Task ID	Task description	Responsible	
	12.1	Use Zend Pdf or another PDF library to add watermarking	Erik	24
	12.2	Testing	Arne	4
16	As a registered customer I want to see my order history (L)			36
	Task ID	Task description	Responsible	
	16.1	Implement a "My account" section for customers	Arne	8
	16.2	Add an order history section to the my account section	Arne	12
	16.3	Make it possible to change personal data in my account	Arne	12
	16.4	Testing	Arne	4
14	As an administrator I want to give discounts (L)			38
	Task ID	Task description	Responsible	
	14.1	Make an admin qui for creating discounts	Olav	14
	14.2	Integrate the discount rules with the checkout and payment system	Olav	20
	14.3	Testing	Olav	4
	Total:			290

Table 7.1: Sprint 3 backlog

7.3 Design

In figure 7.1 is the updated database model. Some tables and fields were added during this sprint. A log table is added for the new functionality for logging and statistics. Due to the shopping cart implementation, the table named "order" has new fields. The same goes for the journal table, where 16 different prices need to be given for each journal.

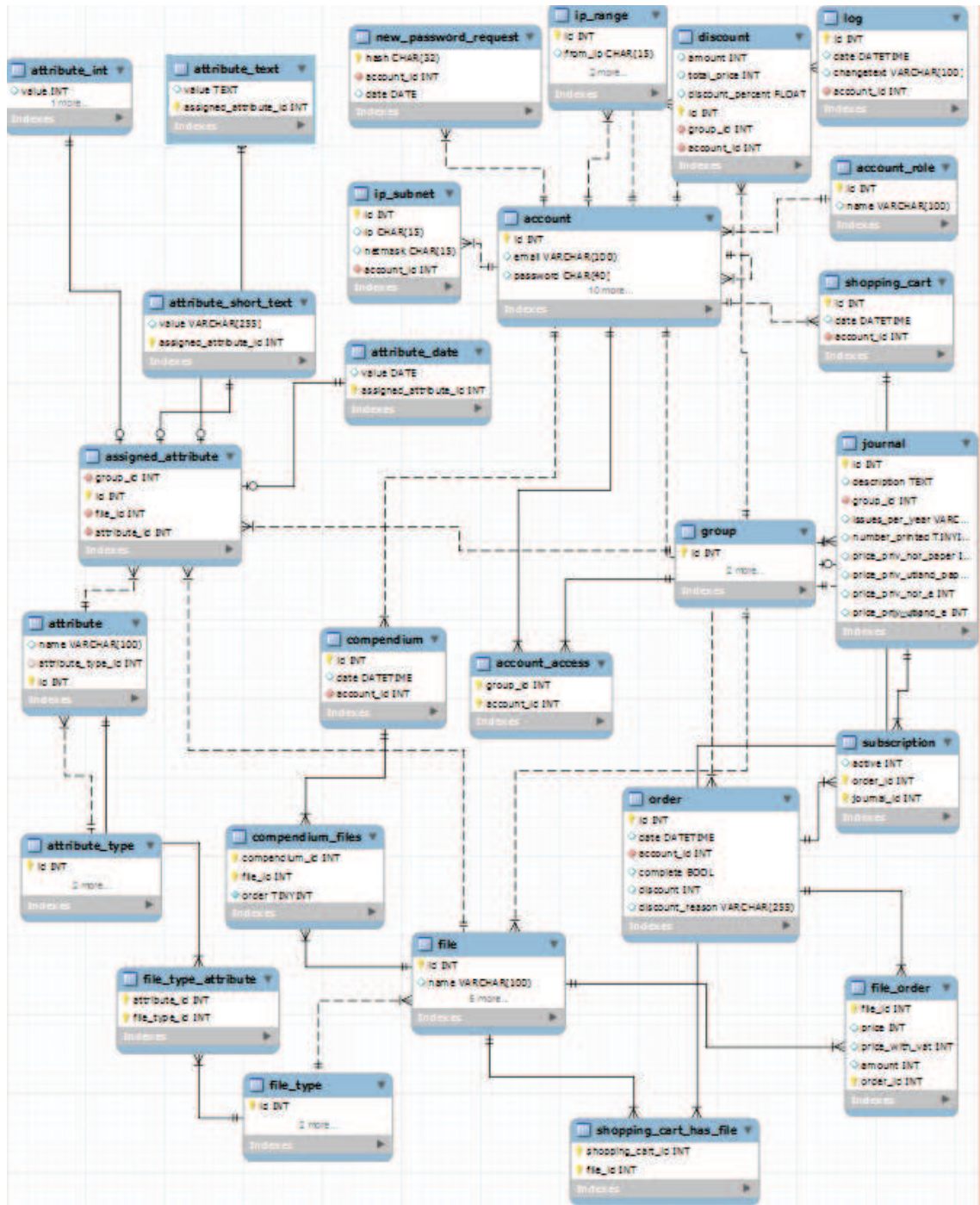


Figure 7.1: Updated database model

7.4 Implementation

7.4.1 My Account Interface

This section documents the "My Account" interface creation. It belongs to User Story 16: "As a registered customer I want to see my order history" from the sprint 3 backlog.

Access to the sites

To access these sites the user can follow the two links displayed in the left account bar. There exists one link for personal details and one for order history. For each of the interfaces made for "My Account" there is an access check. There are two ways to get access to the sites. Either the account id for the requested details and the currently logged in id have to match, or the currently logged in account role must be set to admin. If this check doesn't give a positive response no details will be fetched from the database. This is for preventing that a user tries to brute-force url's to get access to information. If the access check gives a positive response the customer can check and edit his personal details and see complete order history and details, and the administrator can get access to customers order history and details.

Personal details

The screenshot shows the 'tapir akademisk forlag' website. The header includes the logo, the name 'tapir akademisk forlag', and links for 'English' and 'Kontakt'. Below the header, it says 'Din ip adresse er 129.241.104.250'. The main content area is titled 'Kontodetaljer' and contains a table with the following data:

Bruker	
Brukernavn	bjorgan@stud.ntnu.no
Navn	Arne Bjørgan
Telefon	76767676
Fakturaadresse	
Gate	Osloveien
Postnummer	7050
Poststed	Trondheim
Leveringsadresse	
Gate	Kløbu
Postnummer	7030
Poststed	Trondheim
Tilhørighet	
Logget inn som	Administrator

Below the table, there is a link: 'Endre kontodetaljer eller passord'. On the left side, there is a search bar and a navigation menu with the following items: 'Søk', 'Du er logget inn som Arne Bjørgan', 'Logg ut', 'Brukermeny', 'Kontodetaljer', 'Ordrehistorikk', 'Administrasjonsmeny', 'Kontoer', 'Produkter', 'Grupper', 'Tidsskrifter', 'Ordrer', 'Statistikk', and 'Rabatt'.

Figure 7.2: Personal details

This site is made so that the user can check his personal information. If the user wants to edit his information or change his password this can be done by pressing the link under the table. He will then be routed to the user edit page with his account id that is used to fill out the form basis.

User edit

Rediger kontodetaljer

Fullt navn: *

Epost: *

Passord:

Skriv inn nytt passord her for å endre det. Må bestå av minst 6 tegn. Lar du dette feltet stå tomt vil du fortsatt ha det gamle passordet.

Gjenta passord:

Telefon:

Faktura-adresse

Gate: *

Postnr: *

Poststed: *

Leverings-adresse

Gate:

Postnr:

Poststed:

[Avbryt](#)

Figure 7.3: User edit page

Here the form is filled out, and the user can edit the details that he wants to change. If the password field is left open the password will stay the same. If the user wants to change password he can write a new password in the password field. When the user saves the changes he is routed back to the personal details page with his account id.

Order history

In this interface the user can see a complete history of orders. The page displays two tables, one for file orders and one for subscriptions, both active and non-active. Only the basic attributes for the file orders are displayed to keep the page clean. The attributes present are order id, date for purchase and total price. If the user wants to see more details he can press the link displayed in the back of each row. The subscription table displays order id, date, name of journal, subscription price and the active attribute, which tells if the subscription is currently active.

Ordrehistorikk

for bruker Arne Bjørgan

Varekjøp

OrdreID	Dato	Total	Se detaljer
5	2029-10-20 09:00:00	446,-	Ordredetaljer
8	2030-10-20 09:00:00	155,-	Ordredetaljer

Abonnement

OrdreID	Dato	Navn	Abonnementspris	Aktiv
5	2029-10-20 09:00:00	FoU i praksis	250,-	Nei
8	2030-10-20 09:00:00	Etikk i praksis	300,-	Ja

Figure 7.4: Order history

Order details

The link from the file order table routes the user to this page. Here the user can see more details about which files the order contains. The attributes displayed are file id, file name, price without vat and price with vat. If the user has gotten discount for this order, it is displayed and subtracted under the file rows. The discount information given are a short description of why the discount is given, and the total discount given for this order. The bottom row displays the total price for this order with the VAT included and the discount subtracted. Each file name is displayed as a link that routes the user to a page displaying the product details for this file.

Ordredetaljer

for ordre 5

ID	Navn	Pris u/mva	Pris m/mva
38	Opplastingstest	345,-	456
40	Nok en opplastingstest.	34,-	40
	Kunderabatt NTNU, 10 %		-50
Total pris			446

[Tilbake](#)

Figure 7.5: Order details

7.4.2 Journals

This part of the sprint is about both the admin part and the user part of the journal. The admin part is to let admin publish and administer journals. A customer should be able to subscribe and unsubscribe to a journal by using our system.

A journal consists of issues which again consist of articles. Because of this inheriting the group system with its attributes that we defined in the previous sprint is well suited for this representation. When a user wants to subscribe to a journal it should be easy to do this. We have two forms with check-boxes and then one where personal information is sent by email to Tapir together with the earlier selected alternatives. The choices the user makes will determine the products chosen and price and the price displayed to the user at the end of the subscription process. If the customer is an institution, a text area displays together with the other personal info text boxes, where the customer can enter IP ranges.

The admin part is quite similar to admin part for products. First is a menu where he can add a completely new journal, or click in on an existing journal. Then comes different issues of the chosen journal. Here he can both add a new issue, and add a new article to an already added issue.

Tittel	Legg til ny fil	Rediger	Slett
Etikk i praksis - Utgave 1, 2009	Legg til ny fil	Rediger	Slett
Etikk i praksis - Utgave 2, 2009	Legg til ny fil	Rediger	Slett
Etikk i praksis - Utgave 2, 2008	Legg til ny fil	Rediger	Slett
Etikk i praksis - Utgave 1, 2008	Legg til ny fil	Rediger	Slett

Figure 7.6: The user interface for a given journal

Tidsskrifter

Huk av ønskede tidsskrifter *

Etikk i praksis - alle utgaver, fra kroner, 2(mai og november) utgaver i året

FOU i praksis - alle utgaver, fra kroner, 2 utgaver i året

, fra kroner, 2 utgaver i året

Norsk tidsskrift for migrasjonsforskning, fra kroner, utgaver i året

Nordic Journal of Religion and Society, fra kroner, utgaver i året

Norsk tidsskrift for misjonsvitenskap, fra kroner, utgaver i året

Privatperson

Institusjon

Norge

Utland

Send mail til post@tapirforlag.no eller ring + 47 73 59 32 10 hvis du har spørsmål angående abonnering på tidsskrift.

Figure 7.7: The form for subscribing to a journal

7.4.3 Discounts

This section documents the implementation of discounts that belong to User Story 14: "As an administrator I want to give discounts". In the database we added a discount table with the following fields:

- **Id:** Each discount has a unique id
- **Total_price:** If a shopping cart's total sum exceeds this integer value, then a percent discount or an amount discount is given
- **Amount:** An integer value describing how much discount (in NOK) is given if the shopping cart exceeds the total_price.
- **Discount_percent:** A float, e.g. "0.15", that is the percent discount. Either Amount and Discount_percent can be used at a time, the other has to be "null".
- **Account_id:** If set, the discount counts only for that specific account, for example a "NTNU customer account". If this field is empty the discount counts for all customers.
- **Group_id:** If set, the discount counts only for files under a specific group. If empty, the discount counts for all files.

The discount table is quite complex. In the simplest case a discount is given for the amount you buy for in a single order. Discounts for this scenario is given in table 7.2. In this simple case, some value is given in the total_price column, called "Ordre over (kr)" in the table. In addition to this, a value is set in either the amount or discount_percent column. The discount given to the customer is the highest valid discount. As an example, if the total cost is 1200 kr, then a 150 kr discount is given. If the total cost is 1700 kr, 170 kr is given in discount as 10% of the total sum is higher than the fixed discount of 150 kr.

Rabatt id	Ordre over (kr)	Rabattbeløp (kr)	Rabattprosent	Slett
33	2000		20.00 %	Slett
32	1000	150		Slett
31	500		10.00 %	Slett

Table 7.2: Discounts for an order

7.4.4 Watermarking of PDF files

Watermarking was implemented by using Zend Frameworks PDF library. With this library it was fairly easy to write a text on the bottom of every page in the PDF document. A wrapper, called `Tapir_Watermark`, was made for making this watermark and can be found in the `Tapir_Library`. Below is a screen-shot the watermark made by the system on a PDF.

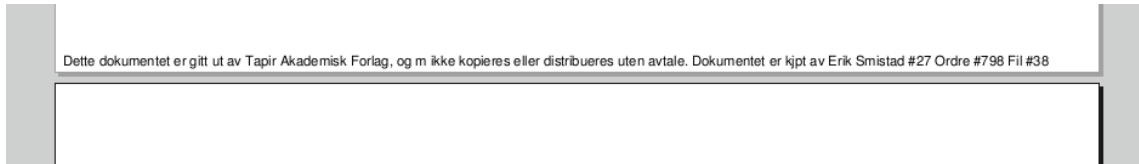


Figure 7.8: Watermarked PDF file

7.4.5 Checkout process and PayEx integration

The checkout process starts after the user has pressed checkout while viewing the shopping cart. If the user is logged in a summary of the order will be shown, if the user is not logged in he will be requested to log in or register a new account. See 7.9

When the user has confirmed that he wishes to proceed with the order by clicking on a button in the summary. Then the web-server will send an WSDL request to the payex server and get an URL address in return. Then the user is redirected to the URL given by payex which contains an order reference. On the website the user was redirected to, see 7.10, a credit card payment dialog is displayed.

After the user has entered all his credit card credentials he will be redirected back to our website. Our website will then send another WSDL request to the payex server and receive information about the payment that was made. Then we check the amount that was paid and if everything checks out we display the download dialog were the user can download all the files within three hours after payment, see 7.11, The user also receives an receipt by email with a link to the download location.

For å gå videre må du enten logge inn eller registrere deg

Logg inn Epost: <input type="text"/> Passord: <input type="password"/> <input type="button" value="Logg inn"/>	Gå til registreringskjema Glemt passord?
--	---

Figure 7.9: Dialog for summary when not logged on

The screenshot shows a payment form for 'SIT Tapir Digital' with a reference number of 796. The product is 798, and the article is 'Ordre nummer 798'. The total amount is 54.00 NOK, including 10.80 NOK in VAT (25.00%). The form is titled 'Betal med Kredittkort' and includes the following fields:

- Korttype:** A dropdown menu set to 'Visa'.
- Kortnummer:** An empty text input field.
- Navn på kortholder:** An empty text input field.
- Utløpsdato:** Two dropdown menus for month and year, currently set to '01' and '09'.
- CVC:** A text input field with a small card icon and a link 'Hva er CVC?'.

At the bottom of the form, there are two buttons: 'Avbryt' (grey) and 'Fullfør betaling »' (green). To the right, there are logos for MasterCard, MasterCard SecureCode, and VISA VERIFIED by VISA. A 'PayEx' logo is also present in the bottom right corner.

Below the form, there is a note: 'Du vil bli sendt tilbake til butikken når betalingen er fullført. Dersom ditt kort krever en verifisering vil du bli sendt videre til din kortutsteder. [Les mer »](#)'. A link 'Velg språk...' is also visible in the bottom right corner.

Figure 7.10: PayEx payment form

Nedlasting

Du kan laste ned filene ved å trykke på dem under.

NB: Filene må lastes ned innen tre timer. I løpet av disse tre timene kan du laste filene ned så mange ganger du ønsker

[Opplastingstest](#)

[Boka mi](#)

Figure 7.11: Download section

7.4.6 Statistics

This section documents the implementation of statistics that belong to User Story 9: "As an administrator I want to see statistics for the website". The statistics to be recorded was site traffic statistics and sale statistics, as well as a log of important events and user actions. In the statistics section of the requirement chapter, 4.3, we had some initial ideas of how the statistic page should look like. Together with the customer we changed our priorities and decided to use Google analytics as much as possible. The only thing we log on the site is the number of times a product page has been visited, and the number of times a file has been downloaded. In addition we have two separate logs: one for customer actions and one for administration actions. The overview of the statistics site is given in figure 7.12.

Brukerstatistikk
 - Se antall besøk og nedlastninger for filer

Google analytics
 - Logg inn med brukernavn " " og passord " " (uten hermetegn)

Adminlogg
 - Liste over tidligere endringer utført av admin

Brukerlogg
 - Liste over registrerte brukere, nedlastede filer osv

Figure 7.12: Statistics main page

In figure 7.13 each file is listed together with the amount of downloads and hits for that file. The implementation is done by using a "helper" in Zend. Anywhere in the code you can call `this->_helper->logger->incVisits($id)` where `$id` is the file id, and it will be updated in the database. There's another function, `incDownloads`, that is used to increase the download count by one. As for logging `$this->_helper->logger->logCustomer($text)`; and `->logAdmin($text)` can be called with a text string describing what was changed. The `logAdmin` will automatically check which admin that performed the change. As for the customer, the customer name should be sent in the string if it is wanted in the log.

Navn	Nedlastninger	Besøkstreff
Leder: Forskning i teori og praksis	2	8
Student som læremiddel	2	3
Den store Vitenskapsartikkel	1	82
ty	0	0

Figure 7.13: File visits and download counter

Administrator	Dato	Endring
Joki	2009-11-02 19:58:23	Lastet opp: "slett"
Joki	2009-11-02 19:52:31	Lastet opp: "slett"
Joki	2009-11-02 19:46:43	Publiserte filen: "Lyd i praksis"
Joki	2009-11-02 19:43:06	Lastet opp: "Lyd i praksis"
Joki	2009-11-02 19:25:18	Slettet filen: "slettmege"

Figure 7.14: Administration log

Opprett ny kunderabatt

Kundenavn:

Totalbeløp - alle handlevogner som overstiger dette beløpet får rabatten:

Velg hvilken rabatt du vil bruke:

Prosent

Beløp

Prosent - skriv "0.20" for 20%, uten hermetegn

Beløp - "300" vil bety at det gis 300 kr i rabatt for alt som overstiger totalbeløpet

Figure 7.15: User interface for adding a new customer discount

Figure 7.15 shows the form for making a new discount for some customer. There's another form for making a discount just for a group of files as well. An example of a discount table for customers and groups is given in table 7.3. If the total amount ("Ordre over") is empty, then the discount is applied no matter the value of the product. The more information is set in a discount row, the more restrictive the discount is. If both an account and a group is given on the same time, then the discount is applied for that customer for that specific group of products.

Ny beløpsrabatt

[Legg til en ny kunderabatt](#)

[Legg til en ny grupperabatt](#)

Eksisterende kunderabatter

Kunde id	Kundenavn	Ordre over (kr)	Rabattbeløp (kr)	Rabattprosent	Gruppe id	Gruppe navn	Slett
0				10.00 %	19	FOU i praksis - alle utgaver	Slett
30	Olav Undheim			12.00 %	4	Undergruppe	Slett
30	Olav Undheim	4444	44		0		Slett
30	Olav Undheim	10000		30.00 %	0		Slett
25	UIO	55555	43434		0		Slett

Table 7.3: Customer and group discounts

7.5 Tests and results

Testing was performed as in sprint 2, with the use of unit testing for the smaller parts of the system. In addition, we performed integration testing, which is testing of modules put together. It was performed by adding a new functionality one by one and testing that it did not introduce new errors.

The demonstration of the sprint results was held on Monday the 2nd of October. The group member responsible for each backlog item showed the customer how it was implemented. We had a close contact with the customer throughout the whole sprint, and quite a lot of functionality had already been shown to the customer before the final sprint meeting. The end result was that 4 items was completely finished, 3 items had a little work left and 1 item was not started on.

The tasks that were completely finished in the sprint were (in order of priority):

- 10: As a customer I want to be able to purchase products
- 9: As an administrator I want to see statistics for the website
- 12: As an admin I want the products to be watermarked
- 16: As a registered customer I want to see my order history

The tasks not completely finished were:

- 13: As a customer I want to be able to collect products in a shopping cart
- 11: As a registered customer I want to be able to subscribe to journals
- 14: As an administrator I want to give discounts

The task not started on was:

- 15: As a customer I want to be able to make compendiums

The tasks not completely finished just needs a little extra work, which will be done in sprint four.

7.6 Sprint evaluation

As was somewhat expected, we did not have time to implement all of the user stories in the sprint backlog. Table 8.3 lists the hours used for each of the backlog items. Watermarking of PDF files took less time than expected, mainly because we did not really know what we needed to do when we estimated the hours for it. The second backlog item that took less time than expected was statistics. The sole reason for this was the use of Google analytics instead of implementing all the statistics ourself, including graphs for displaying the results. Implementing the shopping cart and the journal system took more time than expected, and still needs a little more work to be completely finished.

In total we worked just above 200 hours for the sprint. When adding time used for meetings and report writing, it ends up close to the budget estimate of 240 hours. The downside was that we did not manage to completely finish all the tasks. As can be seen in the burndown chart on the next page, figure 8.13, we still have about 43 hours left. If we add the 43 hours we believe we have left to the 202 hours used, we end up with 245 hours in total, which is 45 hours less than initially predicted. We were worried that the estimates could be way off compared to the actual implementation time, but a rough 15% difference is, in our opinion, not too bad for such a short-lived project.

Sprint 3		19. October - 1. November			
Story ID	Description			Estimated implementation time	Actual implementation time
10	As a customer I want to be able to purchase products (H)			48	40
	Task ID	Task description	Responsible		
	10.1	Create a checkout qui	Erik	20	22
	10.2	Integrate Pavex payments solutions with the website	Erik	20	14
	10.3	Testing	Erik	8	4
13	As a customer I want to be able to collect products in a shopping cart (M)			24	29
	Task ID	Task description	Responsible		
	13.1	Implement shopping cart	Mats	10	16
	13.2	Integrate the shopping cart with the checkout process	Mats	8	10
	13.3	Implement saving of shopping carts	Mats	4	2
	13.4	Testing	Mats	2	1
11	As a registered customer I want to be able to subscribe to journals (H)			56	35
	Task ID	Task description	Responsible		
	11.1	Let admin create journals and issues	Joakim	20	18
	11.2	Implement functionality for subscribing to journals	Joakim	16	15
	11.3	Implement functionality for unsubscribing to journals	Joakim	16	0
	11.4	Testing	Joakim	4	2
9	As an administrator I want to see statistics for the website (M)			46	28
	Task ID	Task description	Responsible		
	9.1	Create an admin interface for viewing statistics	Olav	8	12
	9.2	Create SQL queries for generating the statistics	Olav	14	12
	9.3	Generate graphs from the statistics	Olav	20	2
	9.4	Testing	Olav	4	2
15	As a customer I want to be able to make compendiums (L)			14	0
	Task ID	Task description	Responsible		
	15.1	Implement an option for making a shopping cart into an compendiu	Mats	4	0
	15.2	Make it possible to share the compendium	Mats	6	0
	15.3	Testing	Mats	4	0
12	As an admin I want the products to be watermarked (M)			28	14
	Task ID	Task description	Responsible		
	12.1	Use Zend Pdf or another PDF library to add watermarking	Erik	24	11
	12.2	Testing	Arne	4	3
16	As a registered customer I want to see my order history (L)			36	28
	Task ID	Task description	Responsible		
	16.1	Implement a "My account" section for customers	Arne	8	7
	16.2	Add an order history section to the my account section	Arne	12	13
	16.3	Make it possible to change personal data in my account	Arne	12	4
	16.4	Testing	Arne	4	4
14	As an administrator I want to give discounts (L)			38	28
	Task ID	Task description	Responsible		
	14.1	Make an admin qui for creating discounts	Olav	14	16
	14.2	Integrate the discount rules with the checkout and payment system	Olav	20	10
	14.3	Testing	Olav	4	2
	Total:			290	202

Table 7.4: The sprint backlog with hours used

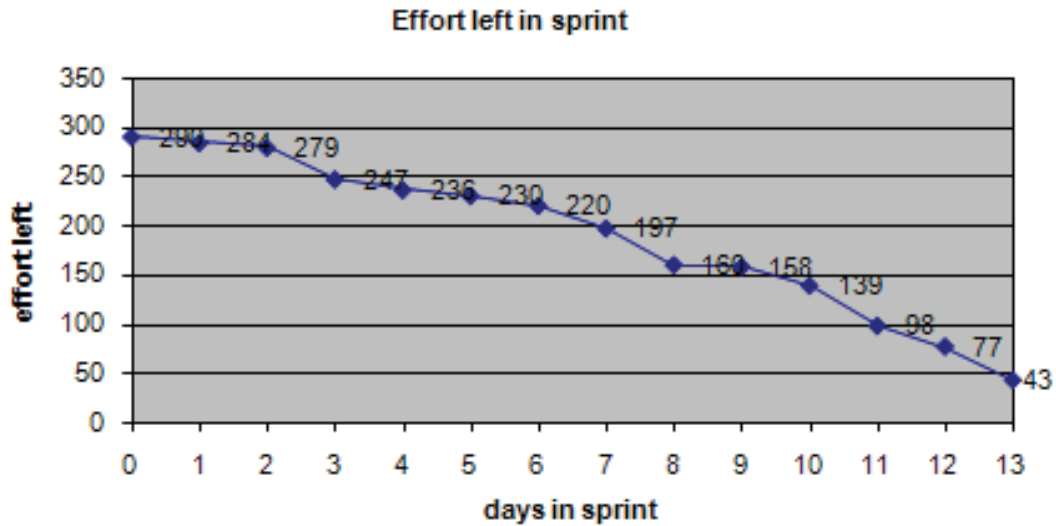


Figure 7.16: The burndown chart of this sprint

Positive experiences

- Managed to do most of the tasks, it felt like we had good progress with the project
- Group work sessions are used a lot and makes implementation easier
- We did not do any "shortcut" implementations just because we are coming close to the project ending

Negative experiences

- Only half the product items were completely finished
- The hours put into this sprint was in the lower scale of what should be acceptable
- We should have updated the risk table more frequently

Ideas for improvement

- Use more hours working on the sprint
- Make more reasonable time estimates for each task
- Update risk table when new risks are defined

CHAPTER 8

Sprint 4

Sprint duration: 2th November - 8 November(7 days)

This chapter documents the work done and the artifacts created throughout this sprint. This is the fourth, and last, sprint. This sprint lasts for one week, and the main goal in this sprint is to finish the product: add the remaining functionality, remove code errors and improve the appearance of the web site.

Chapter overview

The chapter of the third sprint contains the following sections:

- Chapter 8.1 Sprint plan
This section describes the planned work process of this sprint.
- Chapter 8.2 Sprint backlog
This section gives the goals for this sprint that was decided at the planning meeting.
- Chapter 8.3 Design
This section contains the updated database model.
- Chapter 8.4 Implementation
This chapter contains the implementation of this sprint.
 - Chapter 8.4.1 The front page
This section describes the changes we did to the front page.
 - Chapter 8.4.2 Journals
This section describes the remaining functionality we implemented in the "Journals" module
 - Chapter 8.4.3 Managing subscriptions
This section describes how we added a way for administrators to manage subscriptions
 - Chapter 8.4.4 Shopping cart
This section describes the integration of discounts with the shopping cart
 - Chapter 8.4.5 Discounts
This section describes the integration of discounts with the shopping cart
- Chapter 8.5 Acceptance testing
The testing performed with the customer at the end of the sprint.
- Chapter 8.6 Results
A summary of the sprint; what tasks were completed and what was postponed.
- Chapter 8.7 Sprint evaluation
A retrospective of what went good and bad in the sprint and what can be done to improve the process.

8.1 Sprint plan

Sprint 4 will last one week, from November 2nd to November 9th. This leaves us with 10 days from the end of sprint 4 to the final presentation. The purpose of this sprint will be to finalize the product. We have almost all the functionality in the system already, so this week will be used to a) finalize the leftovers from sprint 3, b) integrate every module together and perform system testing, c) find and remove logical errors and bugs, and d) improve the appearance of the site.

In the early stages of the project, we planned to only have three sprints. This would lead to a planned finish of the web site at the 2nd of November, meaning we would have a lot of time left to work on the report and the final presentation. In the start of sprint 3, however, we noticed that we would not need that much time for the report, while we could still use some more time for implementation and testing. Making changes to a plan is often looked upon in a negative way, but we feel that the benefits of a fourth sprint will outweigh the costs. The costs in this case, is having less time for writing the report and preparing for the demonstration.

We plan to work the same way as before, with many group sessions where everyone is working together, and everyone taking part in every activity: design, implementation, testing and report writing.

8.2 Sprint backlog

Sprint 3 had three backlog items that were not completely finished, which are listed in table 8.1. The remaining discount implementation can only be done after the shopping cart is implemented. The total time estimated for the remaining items is about 29 hours. In addition to the three items, we have added a task called "finalizing the web site". This will be our last work on the web site, so whatever we do not have time to fix will be listed in a "future work" section later in this report.

Finalizing the web site is estimated to about 70 hours. This is basically the amount of time we think we will work, after the three backlog items are finished. We wish to perform an acceptance test with the customer to get some feedback on possible improvements. In addition, we want to upload the product to the server it will run on in the future, to test if everything is working as planned.

Sprint 4		2. November - 9. November			
Story ID	Description				Estimated implementation time
13	As a customer I want to be able to collect products in a shopping cart (M)				12
	Task ID	Task description	Responsible		
	13.1	Implement shopping cart	Mats		3
	13.2	Integrate the shopping cart with the checkout process	Mats		6
	13.3	Implement saving of shopping carts	Mats		1
	13.4	Testing	Mats		2
11	As a registered customer I want to be able to subscribe to journals (H)				12
	Task ID	Task description	Responsible		
	11.1	Let admin create journals and issues	Joakim		2
	11.2	Implement functionality for subscribing to journals	Joakim		5
	11.3	Implement functionality for unsubscribing to journals	Joakim		4
	11.4	Testing	Joakim		1
14	As an administrator I want to give discounts (L)				5
	Task ID	Task description	Responsible		
	14.2	Integrate the discount rules with the checkout and payment system	Olav		4
	14.3	Testing	Olav		1
A	Finalize the web site				70
	Task ID	Task description	Responsible		
	A.1	Fix small issues and known bugs	Arne		30
	A.2	Improve the appearance of the page	Erik		14
	A.3	System testing	Olav		8
	A.4	Fix issues discovered in the system testing	Mats		10
	A.5	Acceptance testing with customer	Joakim		4
	A.6	Upload to server domeshop.no	Erik		4
	Total:				99

Table 8.1: Sprint 4 backlog

8.3 Design

The database model was updated in sprint 4, and now has 30 entities. All the tables are shown in figure 8.1, but the attributes are hidden in order to increase readability. A table called "lastdownloads" was added to keep track of which files were last downloaded, which is used in the "top 10 downloads" on the front page. Two more entities, "attribute_category" and "product_category" so that a file can belong to a category like mathematics, physics, ethics or any other category.

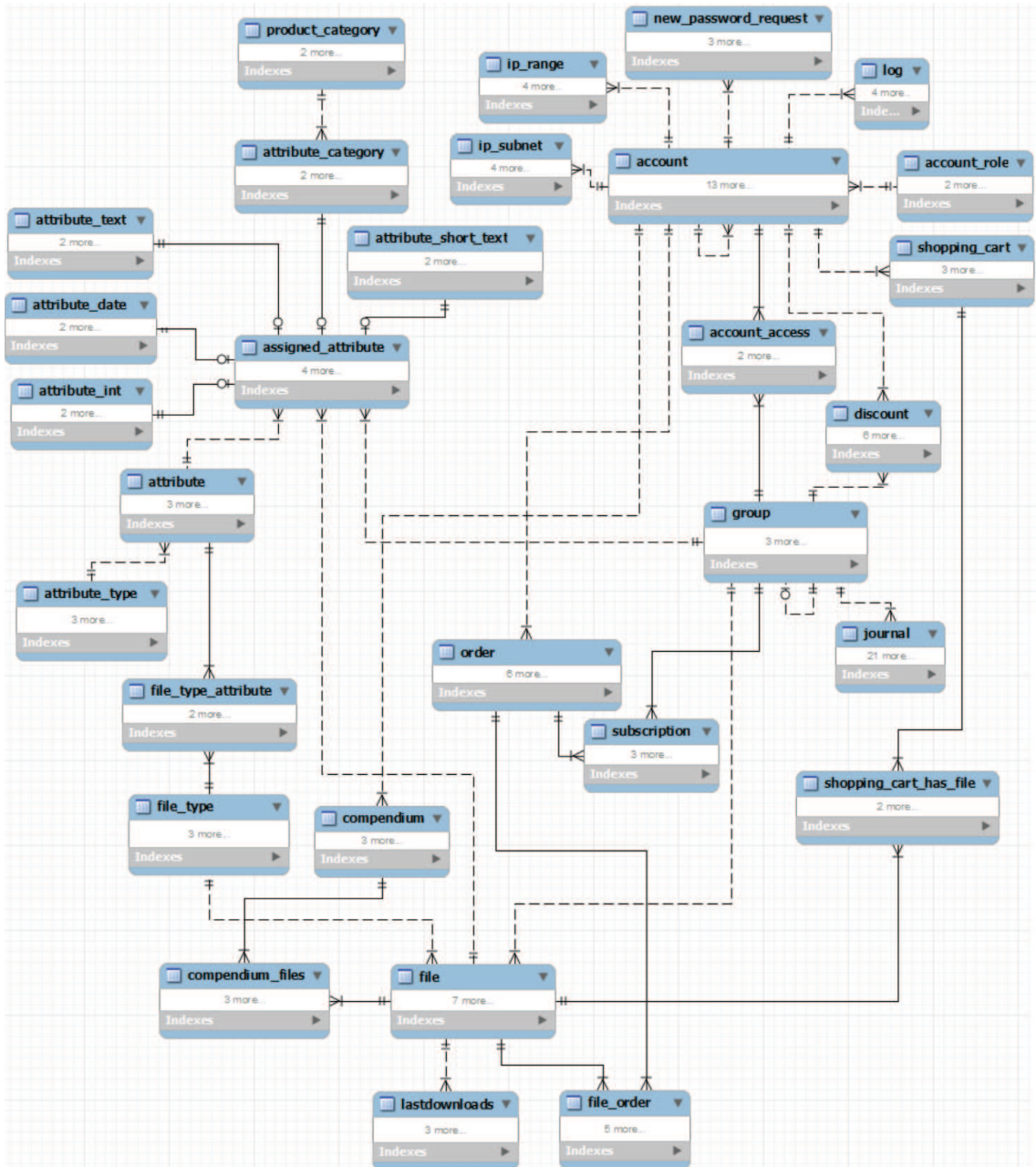


Figure 8.1: Database model

8.4 Implementation

8.4.1 The front page

The front page is the first page the customer will see. It has to make the customer want to browse the rest of the page. The products should be accessible with few clicks. At the top of the page the customer can search through all the published products. We have also added a link to advanced searching for those who want that. The main focus is anyway on the basic search field. Under we have added a text that describes what the customer can expect from the system. Under we have added tables that should give an overview of files that other users have liked. The first tables contains the ten most popular products. In a technical sense the most popular files are the most downloaded products, defined by a counter in the database. The next table contains the ten last published products. In case the customer have waited for a product to be published, he can find it here. The third table contains the ten last bought products, and the fourth table the ten last downloaded products. These two latest tables gives the customer an insight into which products that other customers are currently buying. At the bottom of the page we have a table that contains links to the existing journals, and the latest publications from these.

Søk
Avansert søk

Velkommen til Tapir Akademisk Forlag sin digitale bokhylle. Her vil du til en hver tid finne de siste elektroniske utgivelsene våre, både tidligere utgivelser i digital versjon og helt nye publikasjoner. Tjenesten er ment som et tilbud til våre kunder om å kunne kjøpe enkeltdeler av våre utgivelser til en lavere pris, på et format som sparer miljø, arbeid og distribusjon. Utvalget vil være begrenset i starten, men vi jobber kontinuerlig med å tilby nye produkter på denne digitale plattformen, og vil derfor oppfordre til å kikke innom jevnlig for å holdes oppdatert på vårt innhold. Vi ønsker hele tiden å bli bedre, og vil gjerne høre fra deg dersom det er stoff du ønsker tilgjengelig digitalt, eller dersom du har tilbakemeldinger på tjenesten.

Se også vår facebookside for siste nytt fra forlaget, samt vår ordinære nettbokhandel på <http://butikk.tapirforlag.no>. God fornøyelse!

Mest populære produkter	Nyeste produkter
1. Leder: Forskning i teori og praksis (Artikkel)	1. Lyd i praksis (Lydbok)
2. Student som læremiddel (Laeremiddel)	2. Lenketest (Artikkel)
3. Den store Vitenskapsartikkel (Artikkel)	3. Kategoritest (Artikkel)
4. Utfordringer og dilemmaer i starten av et aksjonsforskningsprosjekt (Artikkel)	4. fhf (Artikkel)
5. tyu (Artikkel)	5. Etikkens kvaler (Artikkel)
6. Etske spørsmål (Artikkel)	6. Etske spørsmål (Artikkel)
7. Etikkens kvaler (Artikkel)	7. tyu (Artikkel)
8. fhf (Artikkel)	8. Utfordringer og dilemmaer i starten av et aksjonsforskningsprosjekt (Artikkel)
9. Kategoritest (Artikkel)	9. Er det å forske på praksis viktig for praksisfeltet? (Artikkel)
10. Lyd i praksis (Lydbok)	10. Leder: Forskning i teori og praksis (Artikkel)

Figure 8.2: Front page part 1

Siste kjøpte produkter	Siste nedlastede produkter
1. Opplastingstest (Artikkel)	1. Student som læremiddel (Laeremiddel)
2. Nok en opplastingstest. (Artikkel)	2. Opplastingstest (Artikkel)
3. Opplastingstest (Artikkel)	3. TestNavn (Artikkel)
4. Nok en opplastingstest. (Artikkel)	4. NavnEksempel (Ebok)
5. Student som læremiddel (Laeremiddel)	5. Utfordringer og dilemmaer i starten av et aksjonsforskningsprosjekt (Artikkel)
6. Lær middel å kjenne (Laeremiddel)	6. Er det å forske på praksis viktig for praksisfeltet? (Artikkel)
7. Leder: Forskning i teori og praksis (Artikkel)	7. Leder: Forskning i teori og praksis (Artikkel)
8. TestNavn (Artikkel)	8. Boka mi (Ebok)
9. Student som læremiddel (Laeremiddel)	9. Nok en opplastingstest. (Artikkel)
10. TestNavn (Artikkel)	10. Test (Artikkel)






	Tidsskrift	Siste utgave
	Etikk i praksis	Siste utgave
	FOU i praksis	Siste utgave
	Norsk tidsskrift for migrasjonsforskning	Siste utgave
	Nordic Journal of Religion and Society	Siste utgave
	Norsk tidsskrift for misjonsvitenskap	Siste utgave

Figure 8.3: Front page part 2

8.4.2 Journals

Because the journal part did not finish in sprint 3 this was added in the fourth and last sprint. Most of the work that remained were small fixes. Like the format of the email sent to Tapir with the subscription order from the forms. Some of the logic behind the schemes that also remained was added. Displaying details about a chosen journal was implemented. This allows the user to view the meta data of a journal before subscribing to it. This matches the displaying of other products on the web site.

The user story of journals also included to let a customer unsubscribe from a journal. To do this, the customer sends an email to the staff at Tapir, and they do the managing. The customer is informed about this on the main page for journals at the web site.

8.4.3 Managing subscriptions

An administration GUI for managing subscriptions was made in this sprint. In this GUI an admin user can view, add, edit and delete subscriptions. A subscription is linked to a specific user account and a group to which the user should get free access to through the subscription. A subscription also has an ending date, which is the date when the subscription ceases to be valid.

Below is a screen-shot of the overview of all subscriptions, divided into active and inactive subscriptions.

Abonnement

[Legg til et nytt abonnement](#)

Aktive abonnement

Abbonnementskonto:	Tilgang til gruppen:	Sluttdato:	Rediger:	Slett:
Erik Smistad	Etikk i praksis - alle utgaver	2009-11-26	Rediger	Slett
Erik Smistad	FOU i praksis	2009-11-26	Rediger	Slett

Inaktive abonnement

Abbonnementskonto:	Tilgang til gruppen:	Sluttdato:	Rediger:	Slett:
NTNU	Alle	2009-11-01	Rediger	Slett

Figure 8.4: Personal details

Below is a screen-shot of the form for adding new subscriptions. When specifying account name and group name an Ajax auto-complete helper searches through the database and gives a list of suggestions below the text-box to help the user find and select items in the database. The ending date field also has an Ajax helper for selecting a valid date. The way this works is that when you select the date field with your cursor a calendar pops up which you can select a date with. All this Ajax functionality is implemented with the ZendX jQuery library.

Legg til et nytt abonnement

Kontonavn:

Sluttdato:

Velg gruppen du vil gi tilgang til:

- 16 **Etikk i praksis - alle utgaver** tilgang til alle produkter
- 17 **Etikk i praksis - Utgave 1, 2009**
- 18 **Etikk i praksis - Utgave 2, 2009**
- 25 **Etikk i praksis - Utgave 2, 2008**
- 26 **Etikk i praksis - Utgave 1, 2008**

Figure 8.5: Adding a subscription

8.4.4 Shopping cart

Since the shopping cart took quite a bit longer than expected to finish we had to add this section to the sprint 4 chapter.

Design

The starting point of the implementation is based on the cart implementation in [35], which is based upon the principle fat model skinny controller where most of the code is put into the model.

The first time the shopping cart is invoked a mechanism to create a unique session that is based upon the model class is used. The session holds a list with the items in the cart. When a new product is added/removed the cart session variable is called and the list is updated accordingly.

Products added to the cart are "intelligent" objects that calculate their own price according to the set conditions. ex. calculates their own discount. This is done by creating a new cart item resource which basically is an object containing variables and methods for getting/setting the necessary values.

Shopping cart GUI

On the store pages right menubar on the cart summary is shown. This informs about how many items there is in the cart and the total price including VAT. When the shopping cart is not empty a link to the cart view is shown. This is supposed to be the main entrypoint for the purchasing process.



Figure 8.6: Shopping cart summary

The shopping cart index lists all the items the user have added. The individual product prices are shown including possible discounts. The total price is shown. The shopping cart model receives data from each shoppingcart item and calculates total price, total VAT and subtotal.

Produkt	Pris	Slett
Student som læremiddel - Produktrabatt	kr 25,00 - kr 2,50	<input type="checkbox"/>
Totalt:	kr 22,50	
Herav mva (25%):	kr 4,50	
Totalt eks. mva:	kr 18,00	

Buttons: [Til Kassen](#) [Oppdater](#)

Figure 8.7: The shopping cart index

As seen from Figure 8.7 there are two options for manipulating the cart. The user can delete the whole cart at once or remove individual items by using the checkboxes and the update button. The changes are reflected in the cart summary menu.

If there are items in the cart a checkout button will appear. By clicking this the system checks if the user is logged into the site. If this is not the case a form is displayed prompting the user to log in or register. On the other hand if the user is logged in he/she is redirected to and the buying process takes over. After a successful purchase the cart is emptied.

The user has an option to save the shopping cart for later when he/she is in the shopping cart overview. See Figure 8.7. Since the session is lost after a pre-determined time this means the cart is lost also. By offering a service like this the user gets the opportunity to access the cart later or even in different browsers/computers.

If there exists a shopping cart in the database the user can access it by going from the link in either the shopping cart index (Figure 8.7) or from the left user menu on the page (Figure 8.8). The user is then presented with a list with the previously saved carts named after the date the user saved it (Figure 8.9). Here the user can either load a cart or delete it. If a user loads a saved cart the currently cart session data is deleted and replaced with the items in the saved cart.



Figure 8.8: The user can enter saved shopping carts from the user menu

Lagrede handlevogner


Lagringsdato	Slett
2009-11-09 11:52:48	Slett
2009-11-09 12:47:57	Slett
2010-11-09 16:19:17	Slett

Figure 8.9: Overview over saved shopping carts

Adding items to the shopping cart

Each product not being offered for free has a buy form attached which consists of a button. The buy button fires a method in the controller which add the item to the cart. The system then gives the user feedback by displaying a message and the cart summary in the right menu is updated to reflect the changes.

Student som læremiddel

Pris	25 kr	
Tilknyttet bok	Student i vekst	
Filtype	Testfil	
Generell lenke	www.loa.no	
ISBN/ISBN2	123214324234/123432132423	
ISSN/ISSN2	219213921933/293423942934	
Redaktør	Arne Hoel	
Teknisk kontaktperson	Koriel Myrez	

Pris: 25,-

Legg i handlevogn

Figure 8.10: Product summary with buy button

Since this is digital distribution there is no need to have a quantity option for each product; therefore it is not possible to add the same item more than once in the cart. A user trying to do this will be informed adding a product more than once is not possible.

8.4.5 Discounts

The remaining part of the discount functionality was to integrate it with the shopping cart and the order process. Figure 8.11 shows a shopping cart with three products. The two last products have a discount because they belong to a certain group that has a discount, in this case group 14 with a discount of 25%. In addition, there's a "big buyers discount". This is a discount given when the value of the shopping cart exceeds a set value. For this example, a NOK 20 discount is given when the order sum is over NOK 200.

HANDLEVOGN
[Slett Handlevognen](#) [Lagre Handlevognen](#)

Produkt	Pris	Slett
Lær middel å kjenne	kr 250,00	<input type="checkbox"/>
Student som læremiddel	kr 25,00	<input type="checkbox"/>
- Produktrabatt	- kr 6,25	
Leder: Forskning i teori og praksis	kr 19,00	<input type="checkbox"/>
- Produktrabatt	- kr 4,75	
Storkjøpsrabatt	- kr 20,00	
Totalt:	kr 263,00	
Herav mva (25%):	kr 52,60	
Totalt eks. mva:	kr 210,40	

[Oppdater](#) [Til Kassen](#)

Figure 8.11: Shopping cart showing an order with multiple discounts

The customer can watch his former order history. Figure 8.12 shows the order details for the shopping cart of figure 8.11. We made a choice to just save one value for the discount amount, which is the sum of all discounts. This is done because the discount is saved in the "Order" table in the database. In addition, a text string is made explaining the discounts so that is possible to keep track of why the discount was given.

Ordredetaljer

for ordre 831

ID	Navn	Pris u/ mva	Pris m/ mva
12	Student som læremiddel	20.00,-	25.00
25	Lær middel å kjenne	200.00,-	250.00
42	Leder: Forskning i teori og praksis	15.20,-	19.00
	20 kr rabatt gitt for ordresum 283 for kunde id 30. 25% rabatt gitt for fil "Student som læremiddel" med pris 25 som til hørte gruppe 14, kunde id 30. 25% rabatt gitt for fil "Leder: Forskning i teori og praksis" med pris 19 som til hørte gruppe 14		-31.00
Total pris			263

[Tilbake](#)

Figure 8.12: The order history showing order with multiple discounts

8.5 Acceptance testing

Usability test with test persons from Tapir Academic Press

Test specification

The user test was conducted at Tapir Academic Press' facilities at Nardo. Four persons were tested using a set of tasks covering a broad selection of the functional requirements. The test persons were:

- Yngve Syrtveit – Editor and developer of digital learning facilities and customer contact in this project.
- Anne Dahle – Publishing consultant
- Vebjørn Andreassen – Editor for the social sciences
- Tina Skjærvik - Language consultant and responsible for the periodicals.

At the end of the test the test person was given the opportunity to give a general feedback on the usability and usefulness of the system. The test took from 15 minutes to 40 minutes depending on the users speed when performing the tasks and the users willingness to comment on the system during testing, and the amount of feedback given by the user at the end of the test. During the test, notes were made from the users comments and actions.

Tasks

Task 1 – Customer role

1. Register in the system and log in.
2. Find a scientific paper which is "Lederen" in the last edition of the periodical "FoU i praksis" without using the search functionality.
3. Add this paper to the shopping cart.
4. Find the article with the title "Er det å forske på praksis viktig for praksisfeltet?" by using the search functionality.
5. Download this paper.
6. Check that this paper was watermarked with a text at the bottom of each page describing who bought it and Tapirs ownership to the material.
7. Subscribe to "Etikk i praksis"

Task 2 – Administrator

1. Log in using the following account: Email: test@tapir.no; Password: 123123
2. Check which previous purchases this user have made.
3. Edit the network area of the institution NTNU. Set the startadress to 0.0.0.1.
4. Upload a file from your computer onto the system and register at least 2 attributes.
5. Find the uploaded product and edit one of the attributes you entered.
6. Add this product to the group "Supergruppe".
7. Give the user you made in Task 1 a 10% discount on all products, and grant the user administrator rights.
8. Check the statistics of how many times "Er det å forske på praksis viktig for praksisfeltet?" have been visited.

Correspondence between the tasks in the usability test and the functional requirements in the product backlog:

Task in the usability test	Product Backlog Item
1.1	2
1.2	1
1.3	10, 13
1.4	3
1.5	
1.6	12
1.7	11
2.1	
2.2	16
2.3	6
2.4	4
2.5	5
2.6	8
2.7	7, 14
2.8	9

Table 8.2: Correspondence between tests and backlog items

Not covered by the test because of missing implementation: 15

General feedback:

Some of the test persons commented that some of the functionality should be accessible from more than one place. This was pointed out after they tried to access for instance discount from the user menu or subscription from product menu.

Three out of four test persons also commented that the page looked unfinished. This was partly due to missing information in some of the pages, and partly that few products were uploaded.

Feedback for task 1 - Customer role

1. Only Anne had problems with this task, mainly because she wrote the wrong information in the registration process and was prompted by the system to enter valid information. She concluded that she had written the wrong information because she had forgotten her glasses.
2. All the test persons first click the FoU-link under periodicals. They find it in the product section after looking there.
3. All users managed this without errors.
4. All users managed this without errors.
5. Two of the users have trouble as their computer won't let them open the pdf in adobe reader. They download the file and manage to read it. Old version of IE is probably the reason for this. The same two comment on the filename of the paper which should be the title of the paper, not a random text string.
6. All the test persons comment that Norwegian characters are not supported.
7. Two of the test persons comment that when you click "Subscribe this periodical" from the info page of a periodical, one should not have to specify which periodical in the ordering sequence. They also miss the confirmation email. One test person commented that there should be an extra "Name" field for each address in case the addressee is not the same person ordering the subscription. The same person wanted some more information on the subscription and English text.

Feedback for task 2 - Administrator

1. All users managed this without errors.
2. Two of the users entered the Account section before they found the information at the Order History section.
3. Two of the users commented that the users should be sorted by name, not registration date. One of them also mentioned the need for a search function when the number of users becomes high.
4. Two of the test persons thought the unpublished files in the systems was on the local computer. They discovered that this was not the case by them selves, but it took some time of looking for the “upload a new file” link.
5. One of the test persons tried to search for the uploaded file to edit the attributes. Uploaded files are not open to the search functionality, and the test facilitator had to tell her this.
6. All but one of the test persons had problems with this. They first had problems understanding that groups was a way of organizing the products in the system, and after they where told what the mission of the group system was two of the test persons had trouble finding the correct link to add a product to a group. They both looked in the “Edit attribute” part of the product before finding the right link elsewhere.
7. This task was a problem for all the test persons. The discount section requires the user to remember the name of the user correctly with no possibilities to look up users without leaving the discount section. It also requires the test person to know the id of the group which is only visible if you have database access. Three of the test persons also entered the percentage wrong, and had to try again. Granting administrator rights to a user didn’t involve any problems for the test-persons.
8. Only one test person had trouble with this because she read the statistics of the “Leder” rather than the specified file.

Quality of the test:

The set of tasks was constructed to cover a broad selection of functional requirements. In practice the tasks involves all but one functional requirement, but does not cover the entire span functionality of each task. Constructing a test covering the systems entire functionality would yield a vast amount of time both for the group and the test persons at Tapir, and was therefore not an option. As long as we are aware that the test only covers a selection of functionality it has limited validity risk and is valuable to us.

The reliability of the test involves how general the test is, and whether the feedback from the users can be believed to have relevance also for other users. Since the great majority of feedback came from more than one person, these errors should be seen as a general problem.

The value of using the customer contact as a test person was limited as he had good insight in the system, and had used the system before. He did however encounter some problems which gives the test person some value for the test.

Conclusion:

The test persons mainly discovered the same usability flaws which is an argument for correcting the most common feedbacks. The feedback was concrete and often came with suggestions on how to improve the functionality, which makes the feedback more useful.

The test persons are the people who will be involved in the use of the system, and their feedback must therefore be regarded as useful and important.

8.6 Results

The sprint ended with the demonstration meeting on Monday the 9th of October. The customer got to see how the web page looked after the sprint, and was informed about what could be improved in further work.

The tasks that were finished in the sprint:

- 11: As a registered customer I want to be able to subscribe to journals
- 13: As a customer I want to be able to collect items in a shopping cart
- 14: As an administrator I want to give discounts

The remaining functionality was completed. Fixing small issues and known bugs, improving the appearance of the page and performing testing was done as well. The only thing missing, was uploading the code to the server it will run on in the future. A list of improvements that could be implemented was used throughout the sprint, and was shared with the customer. The downside is that there is still many of the improvements that were not completed in this sprint. In addition, the testing with the customer showed some more issues that should be resolved before the web site is taken into use. These remaining improvements will be listed in the further work section later in the report.

8.7 Sprint evaluation

This sprint only lasted for one week. Because of that, it was easier to make good estimates for how much time we would use. As can be seen in table 8.3 we estimated 99 hours and used 92 hours. The most significant part of this sprint was improving graphical user interface. In former sprints when we focused on functionality, some of the extra work of putting the different modules together was transferred to the next sprint.

In this sprint we have managed to put in as many hours as we wanted. Still, there is quite a lot of small improvements left. For this product we tried to implement as much as the functionality as the customer wanted. As it turned out, only the backlog item with using compendiums was left out. However, to make a net store with a lot of functionality both for customers and administrators takes a lot of time.

Sprint 4		2. November - 9. November			
Story ID	Description			Estimated implementation time	Actual implementation time
13	As a customer I want to be able to collect products in a shopping cart (M)			12	13
	Task ID	Task description	Responsible		
	13.1	Implement shopping cart	Mats	3	4
	13.2	Integrate the shopping cart with the checkout process	Mats	6	5
	13.3	Implement saving of shopping carts	Mats	1	2
	13.4	Testing	Mats	2	2
11	As a registered customer I want to be able to subscribe to journals (H)			12	10
	Task ID	Task description	Responsible		
	11.1	Let admin create journals and issues	Joakim	2	3
	11.2	Implement functionality for subscribing to journals	Joakim	5	5
	11.3	Implement functionality for unsubscribing to journals	Joakim	4	1
	11.4	Testing	Joakim	1	1
14	As an administrator I want to give discounts (L)			5	4
	Task ID	Task description	Responsible		
	14.2	Integrate the discount rules with the checkout and payment system	Olav	4	3
	14.3	Testing	Olav	1	1
A	Finalize the web site			70	65
	Task ID	Task description	Responsible		
	A.1	Fix small issues and known bugs	Arne	30	35
	A.2	Improve the appearance of the page	Erik	14	12
	A.3	System testing	Olav	8	8
	A.4	Fix issues discovered in the system testing	Mats	10	8
	A.5	Acceptance testing with customer	Joakim	4	2
	A.6	Upload to server domeshop.no	Erik	4	0
	Total:			99	92

Table 8.3: The sprint backlog with hours used

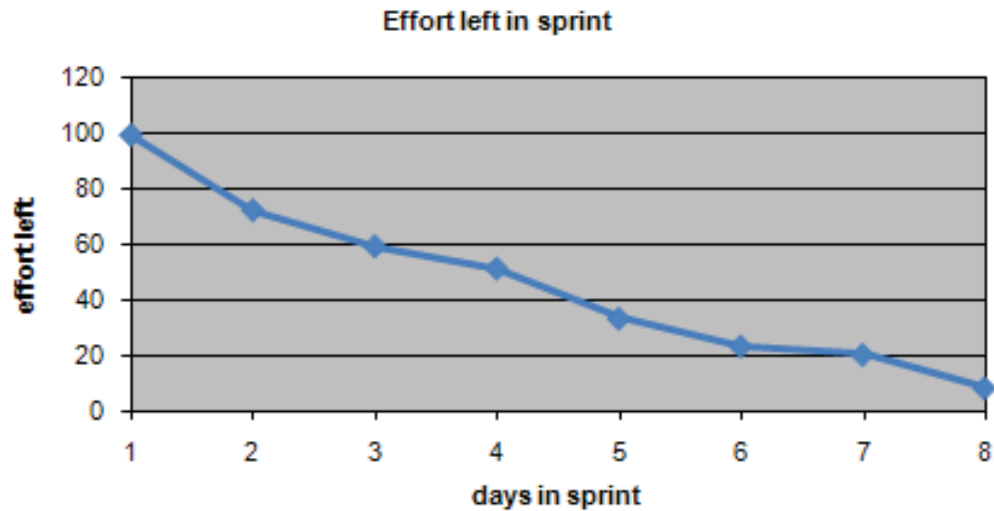


Figure 8.13: The burndown chart of this sprint

Positive experiences

- In addition to the bigger tasks we managed to fix a lot of bugs and fix the layout of the page
- We felt that we worked efficiently with the tasks in this sprint
- We saw that the page is beginning to take form and is useful for the customer

Negative experiences

- We still had some fixes to do when the sprint was over
- Since everybody were working on different bugs it could be messy at times
- We should have estimated more hours for fixes and layout issues

Ideas for improvement

- Communicate better about what we are working on
- Make better time estimates

CHAPTER 9

Overview of system structure

Chapter overview

This chapter is meant to give an overall explanation of the system and goes more into technical details than the sprint chapters.

The chapter contains the following sections:

- Section 9.1 System design
This section gives a general overview of how the different parts of the system and how it is built up.
- Section 9.2 Structure
This section defines the structure of the website, i.e. where the different files of code are located etc. and how this file structure is logically built.
- Section 9.3 Modules
This section lists all the modules in our system and explains how the different features in the modules are implemented.
- Section 9.4 Database
This section displays how the entire database is built up.
- Section 9.5 Security
This section explains how the different security issues such as protection of the product files and access control are handled in the system.

9.1 System structure

Figure 9.1 shows the entire system and how all of the parts interact together. The user interacts against two websites, Tapir's website and PayEx payment website, used only for paying for products. Tapir's website has two main layers: The application layer is upper layer which defines this website in general. This layer contains all of the modules of the website. Then we have the bottom layer which is called the library. This library contains code which can be used by the entire application and works like a common codebase. The Zend library defines the framework that the website is using, see [19] for more detailed information on this framework. This framework has several packages or modules. Though only three are listed in the figure, Zend framework has many more. The system uses the DB module as an abstraction layer for accessing the database, see [36]. So all the communication between the database goes through this module. The Tapir library contains code written by the group, which is general code not entirely specific to this project, and available to the entire application. And the PayEx library contains code and WSDL documents used for communicating with PayEx's web services for managing payments.

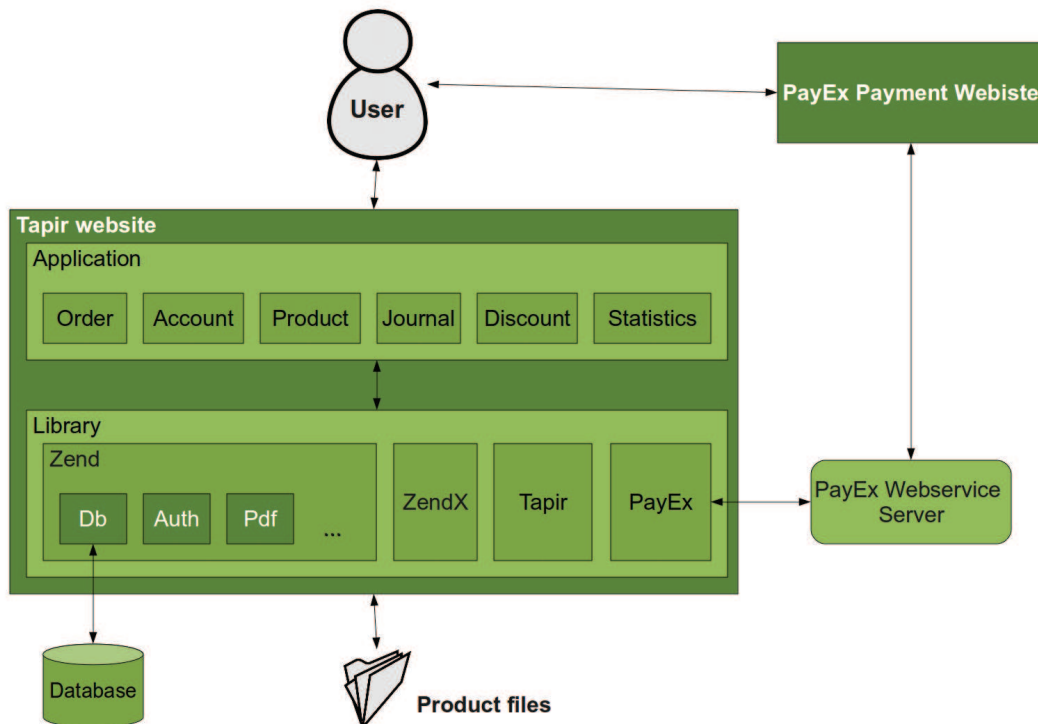


Figure 9.1: Diagram of the entire system

In the websites root folder there are five main folders. Below is a figure of how the file structure in the project folder is.

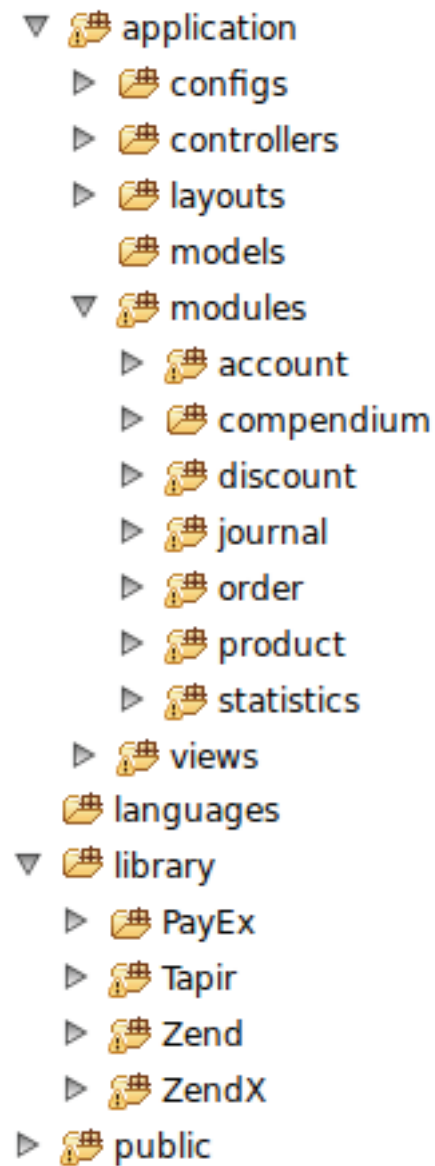


Figure 9.2: Filestructure

Application

The application folder contains all of our own written code, this is code related to this specific website. Inside this folder you will find several other folders. There are two special files in this folder. One called Bootstrap.php and the other is called routes.ini. The bootstrap file is used by the system in the startup of a request and initiates things like routes, layouts etc. The routes file define routes which does not belong to any module like the front page and the contact page.

In the configuration folder there is an application.ini file which contains all the settings of the website. Settings like the database connection information is stored here. It is important that this file is protected from any other users, since the database password is written in plain text in this file.

In the layouts folder we have a folder called scripts which contains view files which defines the looks which are present on all subsites, like header, footer and sidebars.

The modules folder contains all the modules. Every module is in its own subfolder with the name of the module. In each module subfolder there are a controllers, forms, models and views folders which contain controllers, form objects, models and views respectively. Each module also has a Bootstrap.php and routes.ini file. These module bootstrap files can be used to initiate some functionality that the module needs. The routes file defines all the routes that belongs to the module and its controllers.

We have a controllers, models and views folders also in the application folder which define the pages that does not belong to any module. These are pages like the contact page and the front page.

Languages

In this folder language files defining translations of the website are stored.

Library

The library folder contains common code that can be used by the entire application. In the library folder we have four folders:

- Zend: Contains all of the Zend Framework code
- ZendX: Contains optional Zend wrappers for other libraries. The system uses the ZendX jQuery javascript library in this folder
- Tapir: Contains code written by the group that is common for the entire application
- PayEx: Code supplied by PayEx for the integration of the payment system with a PayEx wrapper

Public

The public folder contains all the files that should be available through the web browser. This includes the index.php file which all requests are sent to, images, CSS and javascript.

This folder also contains another folder called upload. This is where all the product files are stored, as well as the product images. This folder and all the subfolders has to be readable and writable by the web server.

Tests

In this folder unit tests are stored.

9.2 Modules

The system has six different modules as well as a common codebase for all the modules called the library. In this section a documentation of all the modules is presented.

9.2.1 Account

Description

The account module handles everything related to an account on the website. There are three different types of accounts in the system and they are defined in the database entity `account_role`. The three different types are:

- Customer. This is a regular customer and is the default role. Users who register manually on the site get this role.
- Institution. This role can have a set of IP ranges connected to it, and is usually used for institutions who pay for having a part or all the products available to their members through IP ranges.
- Administrator. This is mainly the role of the employees at Tapir who has privileged access to the administration part of the website.

The account database entity defines an account and also has a class associated with it called `Account_Model_Account`. This class represents an account and has two important methods:

- `createNewPasswordRequest()`. Sends a new password request to the users email and stores the requests in a separate database entity called `new_password_request`
- `hasAccessTo(file)`. Checks if this user has free access to the given file through a subscription

This module has three controllers:

- Account: Handles all the events related to account except the tasks that are administrator tasks

- Admin: Handles all the events related to the administration of accounts
- IpFiltering: Handles everything related to managing IP filtering for institution users

Authentication system

Zend_Auth is used for implementing the authentication system, i.e. the login system. Details on the Zend_Auth package can be found at [37]. A custom authentication adapter was made and is located in library/Tapir/Auth/Adapter.php. This adapter takes in an email and a password to the constructor and will authenticate the user and store the authenticated account in a php session.

You can always retrieve the logged on account by using:

```
$auth = new Zend_Auth::getInstance();
if($auth->hasIdentity()) {
    $account = $auth->getIdentity();
}
```

New password function

This is used when a user request a new password on the "forgot password" site. The user supplies the system with the email address which the user has a registered account with at the website. The system looks up the account by using the submitted email address. The createNewPasswordRequest method is called on the account object and a row with a unique hash and the account ID is created in the new_password_request table in the database, then an email is sent to the email address given with a link to the website with the unique hash as a parameter in the URL. When the user goes to this URL he his prompted to write a new password which is then stored in the database.

IP filtering

Clients connecting to the web page initiates a new server session variable created in the init method in the action controller in the Tapir library. This variable stores the client IP address. Fetching of this address is done through the apache server calling: `$_SERVER['REMOTE_ADDR']` or `REMOTE_ADDR` depending upon whether the server has globals enabled or not. Matching against two tables in the database: `ip_range` and `ip_subnet`. If there is a match a new session variable is created containing the user ID corresponding to the match and the name of the recognized institution is displayed on the page.

The `ip_range` table contains rows with starting IP address and the ending IP address. Here the system checks if the client IP is between (including) these two fields.

Each row in the `ip_subnet` table contains the starting address of a specified subnet and the corresponding subnet mask. Matching of the client IP against this table is done by converting the client IP address and the subnet mask field in the `ip_subnet` table. The binary AND operation is then applied to these two binary

strings. The result is converted back to 10base and checked if the result is equal to the corresponding subnet address in the `ip_subnet` table.

Both `ip_subnet` and `ip_range` tables are in a one-to many relationship to the user account table. This gives the possibility of one user account to have several subnets and IP ranges associated with each user.

9.2.2 Product

Description

The product module handles everything concerning the product, i.e. product uploading, browsing and managing their attributes and groups. This module does not include the shopping cart and payment system, these features are part of the order module.

Product types

The system has support for several different product types. There is built in support for customizing the display of each product type and which attributes should be available to the different types. This is implemented in the way that we have an entity in the database called `file_type` which contains a list of all the different product types. So to add a new product type you have to add a new row to this table, in this table you also have to specify the name of the view files which the system shall use to define the appearance of the product. Each product type has two view/html files in the folder `views/scripts/product/type/` in the product module. For instance for the type "Vitenskapelig artikkel", the view file name is `article`, so there are two files in the folder called `article.phtml` and `article-small.phtml`. The first one is used for defining the looks of the product when viewing the details of one specific product and the other one is used for defining the looks for a product when browsing a list of products.

To define which attributes should be default to one specific product type there is a table called `file_type_attribute` which has two keys one being the `file_type_id` and the other being the `attribute_id`.

Attribute system

The attribute system is probably the most complex system of this website. It supports different types of attributes, inheritance and overriding of attributes using groups to define a hierarchy. The attribute types which are currently in the system are:

- **ShortText:** Used for small text strings like author and has a limit of 255 characters
- **Text:** Used for large texts like abstract and has no limit in size
- **Date:** Used for defining dates like the date an article was published
- **Integer:** Used for numbers like price

- Category: Used for assigning a product or a group to a specific product category which are all defined in the product_category table
- Image: For adding an image to products

All of these attribute types are defined in the table attribute_type together with a class name which is the name of the class associated with the type. These classes all inherits from the abstract super class called Product_Model_AttributeType. The class diagram in figure 9.3 displays how these classes relate to each other. All of the subclasses has to define the function createFormElement() which return a Zend_Form_Element needed for making the input element in the HTML form for submitting the attribute value itself. This is needed because the input type varies for each attribute type.

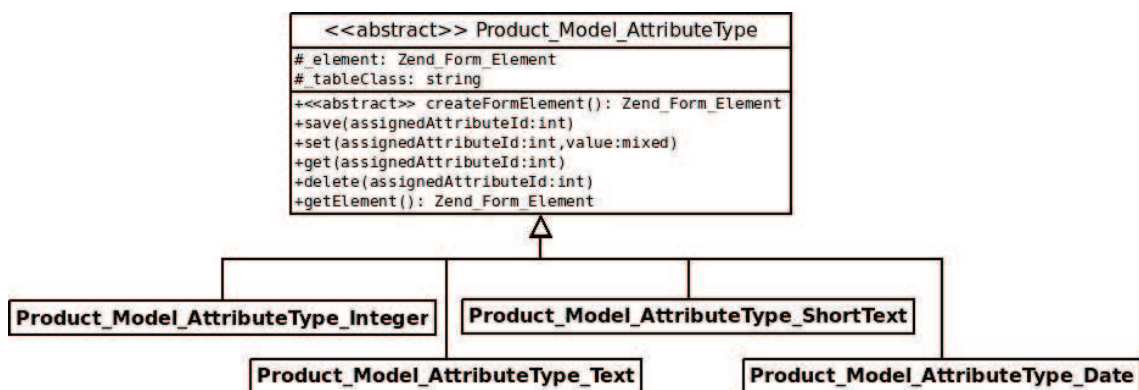


Figure 9.3: Attribute type class diagram

Since all of the different attribute types also need to be stored in different ways, all of the attribute types have one entity in the database each, except the image attribute which store its images on the in the folder public/upload/image/ instead. The attribute system is built up in the database in the way that there is one entity called attribute which defines all of attributes that are available, like author, price, abstract etc. These attributes all have a foreign key which defines which attribute type they are. The assignment of a specific attribute to specific product or group is stored in the database in the assigned_attribute table. While the value itself is stored in either one of the different attribute type tables. In figure 9.4 is an ER diagram displaying the relations between all the entities in the attribute system and the entire product module.

The figure 9.5 shows the way the inheritance of attributes work in a group hierarchy. We see that group 1 defines values of two attributes, price and author. Group 2 and 3 inherits from group 1 indicated by the arrow. Group 2 overrides the value of price inherited from group 1 and sets it to 1 instead. And group 3 overrides the value of author. In bottom of the diagram we see that we have several files or products. Attributes can also be overridden on the files themselves as file 2 and 3 does. Under the files we see which attribute values the files get. Note that the attribute value the file gets is the defined closest to the file itself.

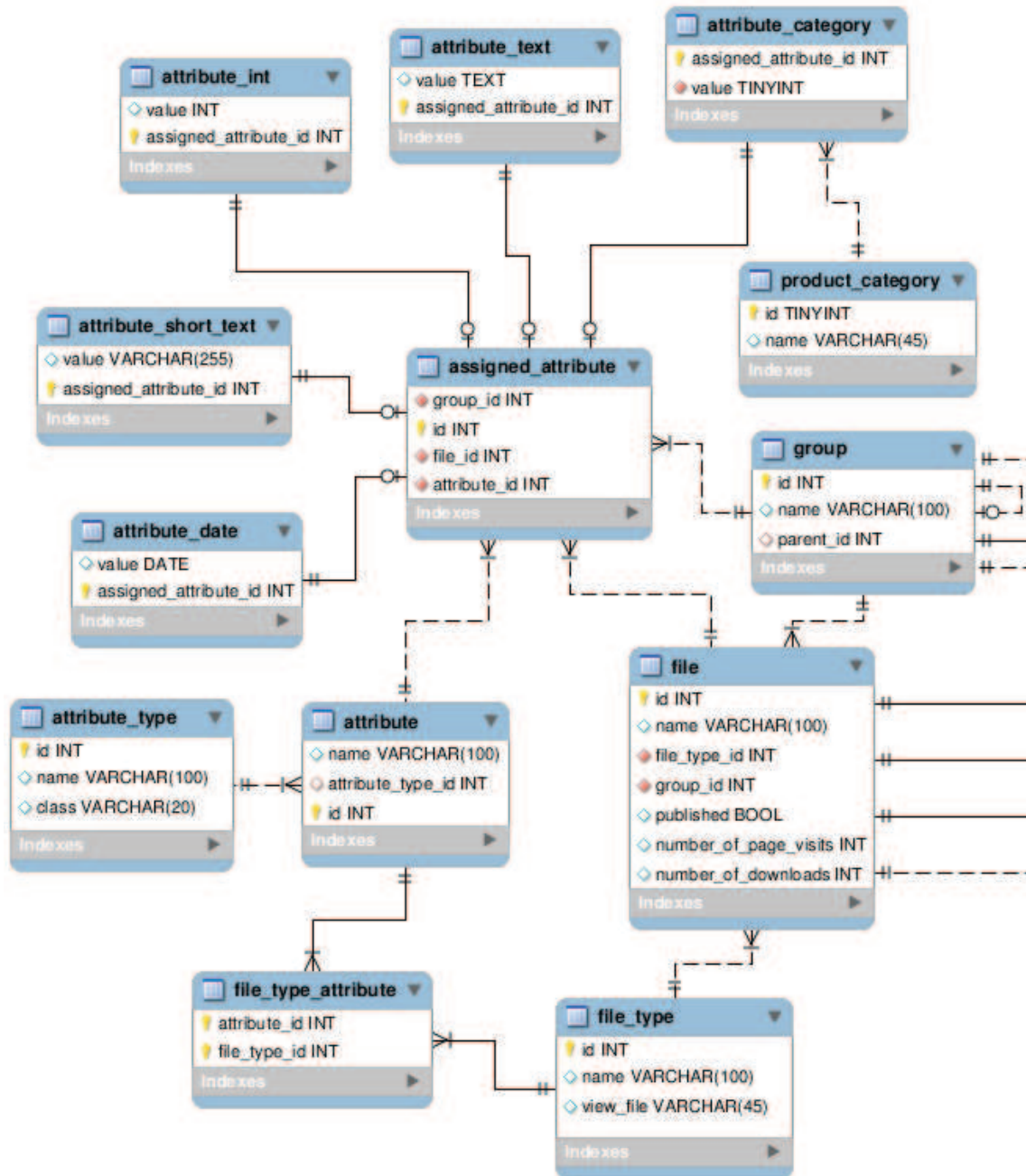


Figure 9.4: ER diagram of the attribute system and product module

Watermarking

The watermarking of PDF files is implemented in the class called `Tapir_Watermark` and uses the `Zend_Pdf` library for handling the PDF files. Files are watermarked when they are requested to be downloaded by a user. The class takes 3 arguments: the file path to the PDF, the file path to store the watermarked PDF, and the string to watermark the PDF with.

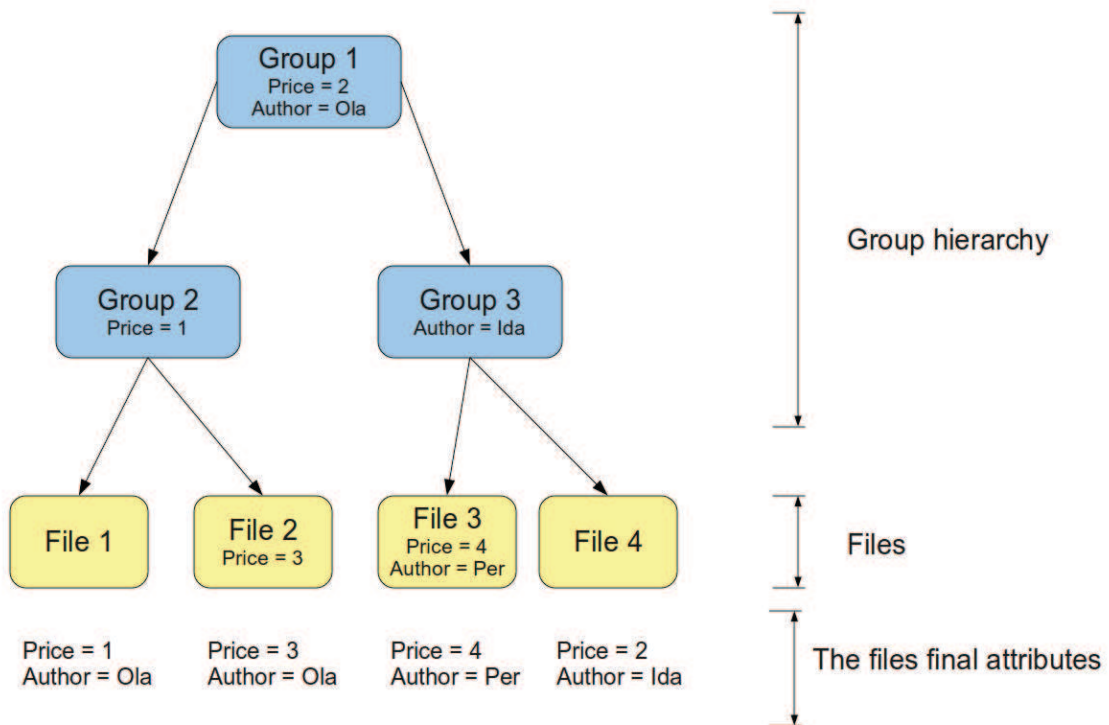


Figure 9.5: Group

9.2.3 Order

Description

The order module contains all the features belonging to the processes of selecting which products to buy and the payment and download of these products. This module also contains functionality for managing subscriptions for administrators, i.e. giving access to certain product groups to certain users.

This module has four controllers:

- CartController handles everything related to the shopping cart
- CheckoutController handles everything related to the checkout and payment process. For more information on the integration of the payment system PayEx, see 9.4
- DownloadController handles the download of products.
- SubscriptionController handles the managing of subscriptions for administrators.

Shopping cart

The shopping cart is implemented in the module called order. The `Order_Model_Cart` model, which stores the products the user wishes to buy, stores cart items in a session variable. The cart class implements `SeekableIterator`, `Countable`, and `ArrayAccess`

interfaces. These interfaces are defined in PHP SPL Library [38] and the reason for using these is to provide a good way for interacting with the data in the cart.

SeekableIterator interface allows the access to the cart data in the following ways:

- iteration over the cart items using foreach: `foreach($cart as $item)`
- seek for an item at a specific position: `$cart->seek(2)`
- standard iterate access such as: `$cart->rewind()`, `$cart->next()` and `$cart->current()`

Countable interface allows for counting of items in the cart:

- `count($cart)`

ArrayAccess interface allows for access to the cart using:

- `$cart[3];`

There are several methods that the three interfaces require. For more information go to [38] or look in `Order_Model_Cart`.

The Cart model class properties:

- `$_items`: array of the cart items
- `$_subtotal`: : total price of the cart items before VAT
- `$_total`: total price of the cart items including VAT
- `$_sessionNamespace`: session store

Cart model class methods:

- `init()`: called during construct and loads the session data
- `addItem($product)`: adds items to the cart and updates it
- `removeItem($product)`: removes item from the cart
- `setSessionNs(Zend_Session_Namespace $ns)`: sets new session instance to use for storage
- `getSessionNs()`: gets current session instance
- `persist()`: saves cart to session
- `loadSession()`: loads a stored session
- `calculateTotals()`: calculate total cost
- `calculateVatTotal()`: calculate vat total
- `getSubTotal()`: get total ex. VAT

- getTotal(): get total inc: VAT
- getVatTotal(): total VAT
- inCart(): checks if a item is in the cart

When adding an item to the cart a new Order_Resource_Cart_Item is created. The cart resource item implements the values and methods necessary to correctly calculate it's cost according to properties set by administrator.

The cart controller has the following actions:

- add: adding cart item to the cart session
- view: view the cart contents
- update: update cart contents. Inside this method the code coupling the cart to the checkout process is located
- delete: empty the cart contents
- save: save the cart to the database
- index: view all carts saved in the database
- savedPopulate: empty cart session and load cart from database into cart session.
- savedDelete: delete a saved cart from the cart database

There are two forms associated with the shopping cart:

- Order_Form_Cart_Add: form added to the product view which contains the buy button
- Order_Form_Cart_Table: form displaying a cart item information

The cart view has a cart view helper called Tapir_View_Helper_Cart. This helper contains code for generating the cart summary, the add form in the products view and a method for adding one form for each product in the shopping cart to the cart contents view.

9.2.4 Discount

The discount module is used for setting discounts. A discount can be set on a group of files or for a sum of an order. The discount can be specified just for one customer, or for everyone. Originally it was planned to have this functionality as a part of the "Order" module, but we decided to have it as a separate module to reduce the size of the "Order" module.

The discount module has a controller called DiscountController, with actions for adding, editing and removing discounts. The module has two forms. The first form

is for adding a new discount connected to the total sum of a shopping cart. The second form is for giving a discount to a group of files.

In the database there is a discount table that stores the discounts. When a new product is added to a shopping cart, a check is done to see if there is any discount given to this product. If the customer is logged in, discounts that are set to everyone, meaning `account_id` in the discount table is set to 0, and discounts just to this customer are found. The highest discount is then returned. The check for a discount is performed by the function `getFileDiscount` in `modules/order/models/Resource/-Cart/Item.php`.

A discount can be given based on how much the total sum of the order is. This check is done by the function `getShoppingCartDiscountQuery` in `modules/order/-models/Cart.php`. All the discounts are summarized to a decimal number and stored in the "Order" table in the database, together with the reason for the discount.

9.2.5 Statistics

The statistics module contains the code for displaying the administration log, the user log and the file statistics. The actual code for logging an action is stored in `Tapir/Controller/Action/Helper/Logger.php`. There is one table in the database that stores all the log items, both for the customer and the user actions. The actions are distinguished as the `account_id` of an administrator action is logged, while the customer's id is not stored explicitly.

Each file has an integer for how many times the file page was visited, and how many times the file was downloaded. The counter can be incremented by calling `incVisits` and `incDownloads` in the `Helper/Logger` class, such as `$this -> _helper -> logger -> logAdmin($msg);`

9.2.6 Journal

Description

The journal module contains the functionality for managing journals, their issues and the products linked to them. It also contains functionality for letting customer view a journal and fill out a form for subscribing to a journal.

The journal module is essentially built on the group system. A journal itself is stored as a group and each issue is stored as group. A journal also has some extra fields related to it, such as description and prices for each of the subscription types. There is a journal entity in the database which has all of these extra fields and a foreign key indicating which group it is linked to. The module only has two controllers:

- `JournalController` handles all the events related to a journal as well as the subscription form
- `IssueController` handles all the events related to an issue of a journal

9.3 Database

The database has a total of 28 entities, which are all needed to implement the functionality of this system. Figure 9.6 has been made using the Crows foot notation [34] and is made in the MySQL DBMS [39]. Zend's database abstraction layer, [36], is used exclusively for communicating with the database server. By using this abstraction switching from one DBMS to another should not be a problem. Each table in the database has a `Zend_Db_Table` class related to it. These classes are always found in the folder `models/Table` in each module and extends `Zend_Db_Table_Abstract`. When these classes are instantiated they work like a representation of the table itself and you can then run methods on this object for fetching, inserting, updating and deleting rows. See [40] for more information on how to use these table objects. When retrieving rows from the database using this abstraction layer, it returns objects which have all the fields of the table as member variables. It is possible to specify that a row should be returned as a specific object which one can define manually. These objects are always located in the `models` folder of each module and has to extend `Zend_Db_Row_Abstract`. These objects will also have all of the fields of the table as member variables, but the real advantage of this is that one can also define methods to operate on these variables.

As we can see on the figure 9.6, the attribute system has most entities. This is because every different attribute type has to be stored with different data types.

9.4 PayEx integration

PayEx was integrated with our system using the re-direct method as explained in the preliminary study. The figure below explains how this re-direct method works with PayEx and our system.

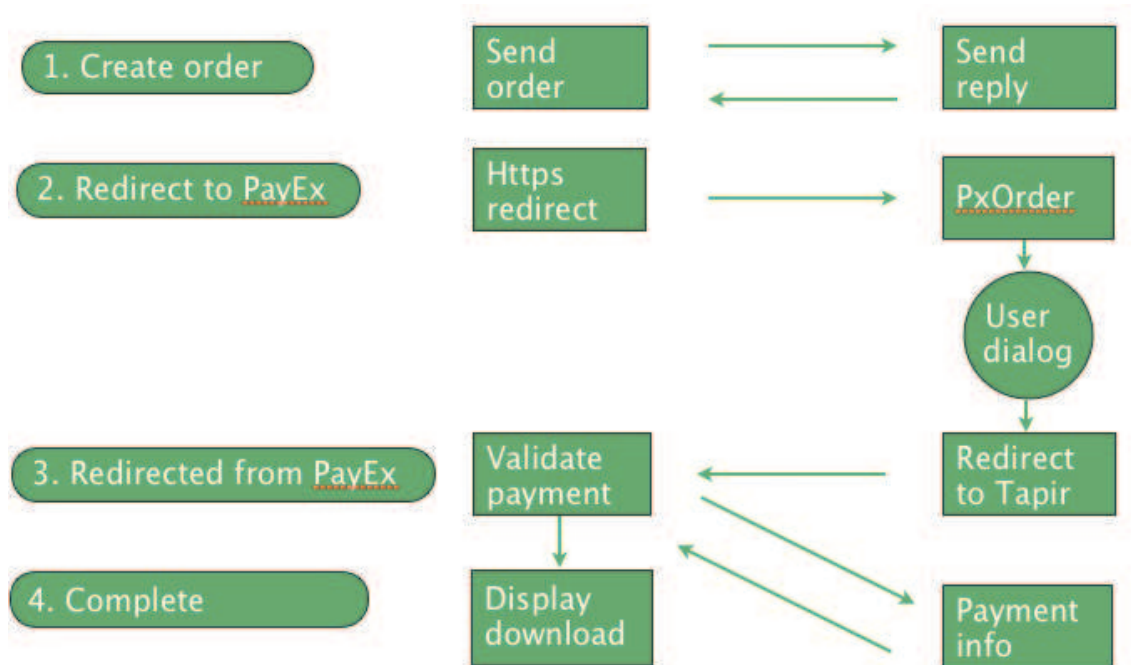


Figure 9.7: Sequence diagram of payment

1. After the user has confirmed the order at the checkout, and order is created in our database and a XML document is sent to PayEx using SOAP. This document contains information like the order ID in our database, the amount the user have to pay, our account number at PayEx, return URL etc.

After this is sent, a response is received from PayEx with a status code and an order reference

2. Using the order reference received from PayEx the system redirects the users browser to PayEx secure payment site
3. When the user is finished doing the payment at the PayEx site the user is redirected back to our website by using the return URL specified in the XML document in the first step.
4. After the user has returned to our website we send a new XML document to PayEx using SOAP with the order ID and our account number at PayEx. In response we receive all the details of the transaction. In this step we check that the transaction status was OK, and that the amount paid is the same as the amount the customer was instructed to pay. If all of this checks out the

user gets a receipt to his email and is shown a download dialog, where the user can download all the files.

A wrapper was made for PayEx source code to ease the integration of the payment system. The wrapper and all of PayEx source code is located in the library folder. The wrapper class is called PayEx_Transaction and contains the following methods:

- start(array params) Sets up a new order/transaction at PayEx. I.e. step 1 and 2 in 9.7
- validate(orderReference, accountNr) Validates the transaction against the order in our database. I.e. step 4 in 9.7

9.5 Security

9.5.1 File security

The products themselves are stored in the public/upload/file folder. In this folder there is a .htaccess file which is set up to deny any outside requests from the web server. To actually open and download a file, a user has to go through the download controller in the order module.

9.5.2 Account security

Account passwords are stored encrypted in the database using the SHA algorithm, preventing anyone gaining read access to the database from stealing password to accounts.

9.5.3 Payment security

When the user is going to pay for an order, the user is redirected to PayEx payment website. This website uses SSL encryption of the traffic sent back and forth, hence encrypting sensitive information like credit card details.

9.5.4 Access control

All the controllers inherits a method called authorize from Tapir_Controller_Action. This method is used to check that the user is logged on and that the user has the correct role needed. And if the user has the wrong role he is shown a message saying "Access denied". If the user is not logged on he is redirected to the login page.

CHAPTER 10

Evaluation

This chapter contains the evaluation of the project as a whole. The purpose of this chapter is to describe the working process, the project results and relations between the stakeholders in the project. A discussion of further improvements on the product will be given in the end of the chapter, together with an evaluation of the course.

Chapter overview

The chapter of the first sprint contains the following sections:

- Chapter 10.1 Work process
Describes the development process and the hours worked on the project
- Chapter 10.2 Results
The product and how much of the product backlog was completed
- Chapter 10.3 The customer and the project
Information about the customer and the project task
- Chapter 10.4 The supervisors
Information about the supervisors for this project
- Chapter 10.5 Further work
A discussion of what improvements can be made on the system'
- Chapter 10.6 Suggestions for improvements
A discussion of what can be improved in the TDT4290 course
- Chapter 10.7 Concluding remarks
A conclusion of the evaluation chapter

10.1 Work process

This section gives an overview of how we worked with the project. This includes a discussion about the development method we used, how much time was spent on each phase and the total amount of hours used on the project.

The development process

We used Scrum as our development process model. When we chose to use Scrum we had multiple reasons for doing so. The first was that the requirements were likely to change as the project progressed. If we had used a sequential software development process, such as the waterfall model, changes to the requirements would most likely lead to a lot of extra work. As it turned out, the requirements were fairly stable after sprint 1, mainly because the customer had a good idea about what he wanted and that one of the group members had former experience developing similar web sites.

Another reason for choosing Scrum was that it better suited a student project where the participants do not have assigned roles, such as designer, programmer or tester. It was greatly appreciated by everyone in the group to have the possibility to work with all aspects of the development process. Having assigned development roles for each group member could lead to more efficient work, but then we would not have learned as much from different areas.

We used the roles, the artifacts and the sprint meetings. The most significant difference from the work process of this project and the Scrum methodology was the use of daily sprints. The schedules of the group members were very different, so it was difficult to find a time where everyone were present early in the day. Instead, we had a dedicated group room where we sat together and worked. It worked well to just keep the project status meetings informal. A reason for why it worked so well could be that the group were relatively small, 5 people, and nobody were shy to speak their mind.

The sprint length was two weeks for the three first sprints and then one week for the last sprint. The shorter the sprints, the easier it is to predict how much work can be done for that sprint, but the deliveries will be small as well. Sprints lasting two weeks worked very well. It was short enough to keep things simple, but still long enough to design and implement tasks of some size. Another important point is that all group members have other subjects as well, and a two week sprint in this course corresponds to a one week sprint if you worked on a full-time project. To sum it up; the use of Scrum in this project was a success, and we can recommend it for this course in the years to come.

Hours spent on each phase

The total amount of hours we used on each phase of the project was a good match to the estimate. Table 10.1 shows the difference between what was planned and what actually happened. At the start of the project it was difficult to register the hours properly, to know what work should be under which phase, especially for the phases project management, self study, project planning and preliminary study. Once we started with the sprints it was easy to keep track of the exact hours used for each phase. The reason the project evaluation is a significantly higher than planned is that we worked a lot on the report after sprint 4. These hours used on the report between sprint 4 and the deadline was registered in the the "Project evaluation" phase.

Phase #	Phases	Planned hours	Used hours	Difference	Difference %
1	Project management	140	166	26	18,6 %
2	Lectures and self study	120	157	37	30,8 %
3	Project planning	120	108	-12	-10,0 %
4	Pre study	120	97	-23	-19,2 %
5	Requirements specification	215	146	-69	-32,1 %
6	Sprint 1	240	238	-2	-0,8 %
7	Sprint 2	240	224	-16	-6,7 %
8	Sprint 3	240	202	-38	-15,8 %
9	Sprint 4	0	92	92	
10	Project evaluation	75	124	49	65,3 %
11	Presentation and demonstration	50	70	20	40,0 %
	Total:	1560	1624	64	4,1 %

Table 10.1: Hours used on each phase

The most significant change we made to the project plan was adding a fourth sprint. We had a lot of time from the end of sprint 3 to the final presentation, so by adding a fourth sprint we had some more time to work on the product. In total we used 756 hours on the sprints, compared to 720 hours as was planned.

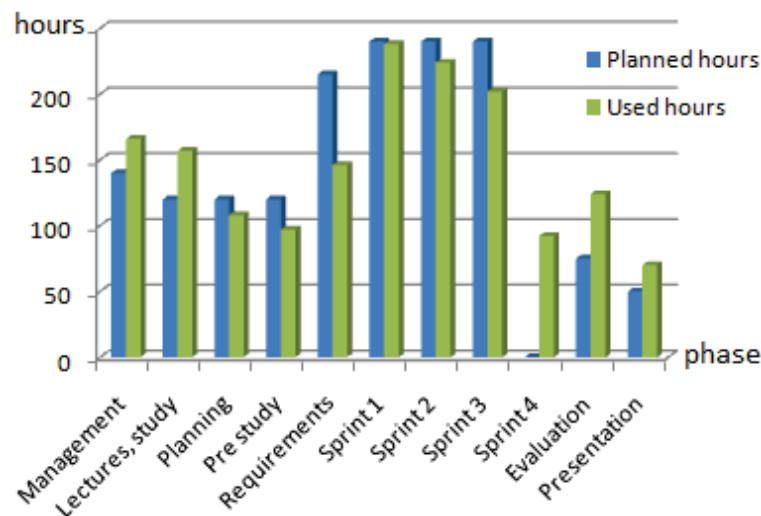


Figure 10.1: Bar chart of the phases

Hours worked in total

The budget for this project was 1560 working hours and we spent 1624 hours. How much each group member worked with this course is displayed in figure 10.2. As can be seen, the work distribution is fairly close for all the group members. Everyone were within a 20% range of each other, which means that everyone contributed to the project.

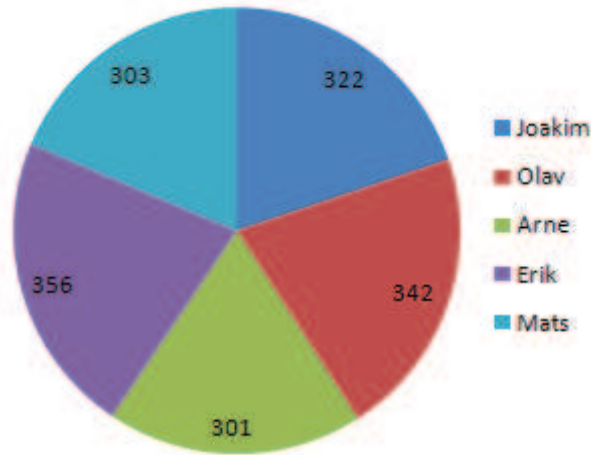


Figure 10.2: The total hours used for each group member

Hours worked per week is given in figure 10.3. This graph was made using the hours registered at the status report that was sent to the supervisors each week. Week 47 is missing, which are the last 4 days before the presentation. Most people were around the expected 24 hours, but there were some variations, as expected, due to traveling, illness, higher workload in other classes etc.

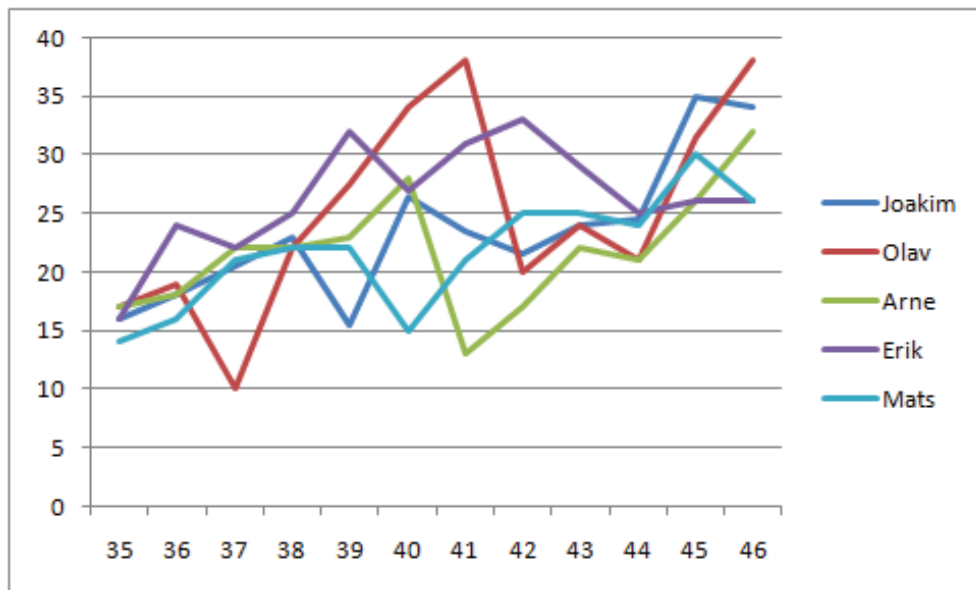


Figure 10.3: Hours worked per week

Work activity

The activity level in the group can be measured by the commits per day, which is collected by SVN. Figure 10.4 shows the number of commits just for the web site. The highest commit activity is on Saturday and Monday. This is likely to be because all the sprints ended on a Monday, and everything has to be committed before the sprint review meeting. The second thing that is noticeable is how little work has been done with the product on Tuesdays. The supervisor meetings was held every Wednesday. Tuesdays were used as a day to write on the report and writing status report, agenda and other items for the supervisor meetings. Figure 10.5 shows the amount of commits on the report, which clearly shows the amount of work done with the report on Tuesdays.

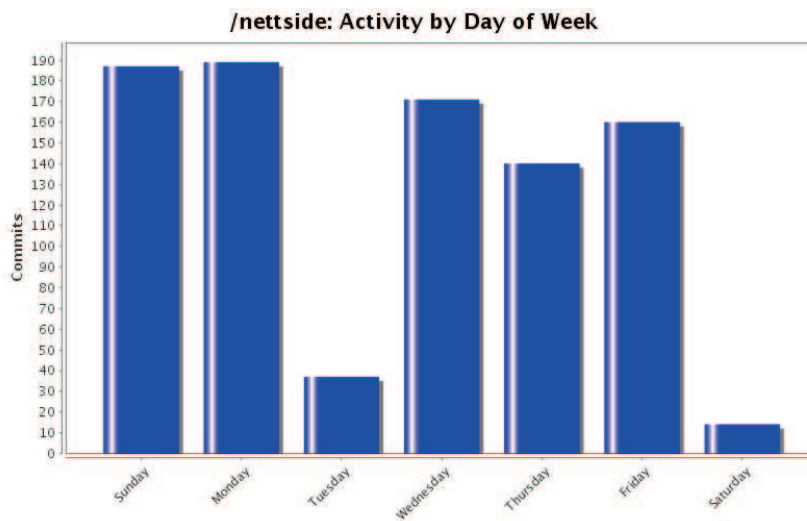


Figure 10.4: Activity on design and implementation

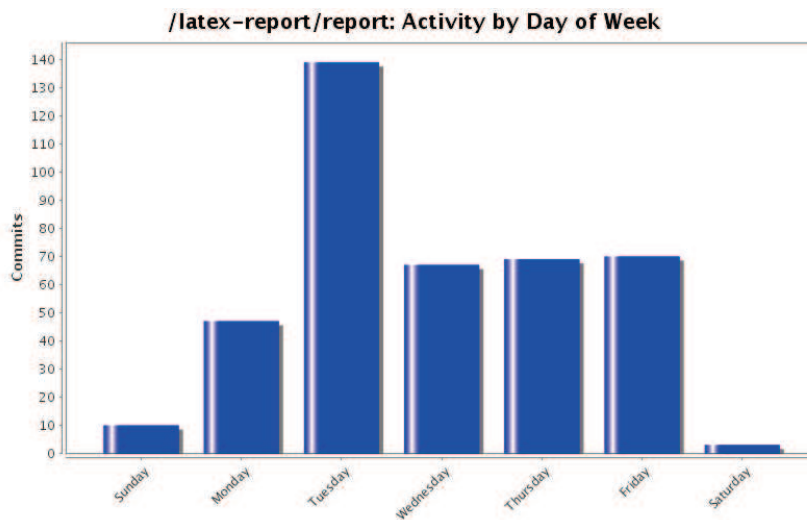


Figure 10.5: Activity on the report

As most of the group members had no experience developing web pages and with PHP it was crucial to have group sessions as often as possible. This meant sitting together and being in range of each other to reduce the effort it took to ask questions regarding the implementation, and to keep up to date on what everyone was working on. Figure 10.6 shows the commits per hour of the day for the web site. The highest number of commits are from 12 to 15 each day. Although we usually started working at 10:15 each day, the commits are only carried out after some changes are made, thus there is a delay between when you start working and what you have committed.

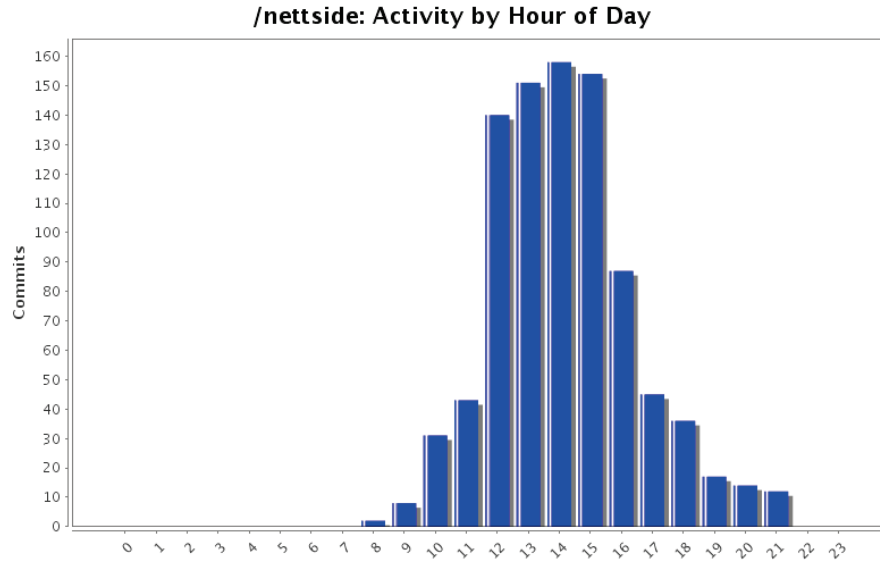


Figure 10.6: Activity per hour on the implementation

Implementation process

The implementation was a steady process. The first lines of code were committed at the end of sprint 1, at the 4th of October. In figure 10.7, the most noticeable is the straight line from Friday 23rd of October to Thursday 29th of October. The week before, we had focused more on implementation than the report. As a consequence, we had to make an all-out effort to finish the sprint 2 chapter in the report. After the 29th we had a steady work flow where we managed to balance implementation and report writing.

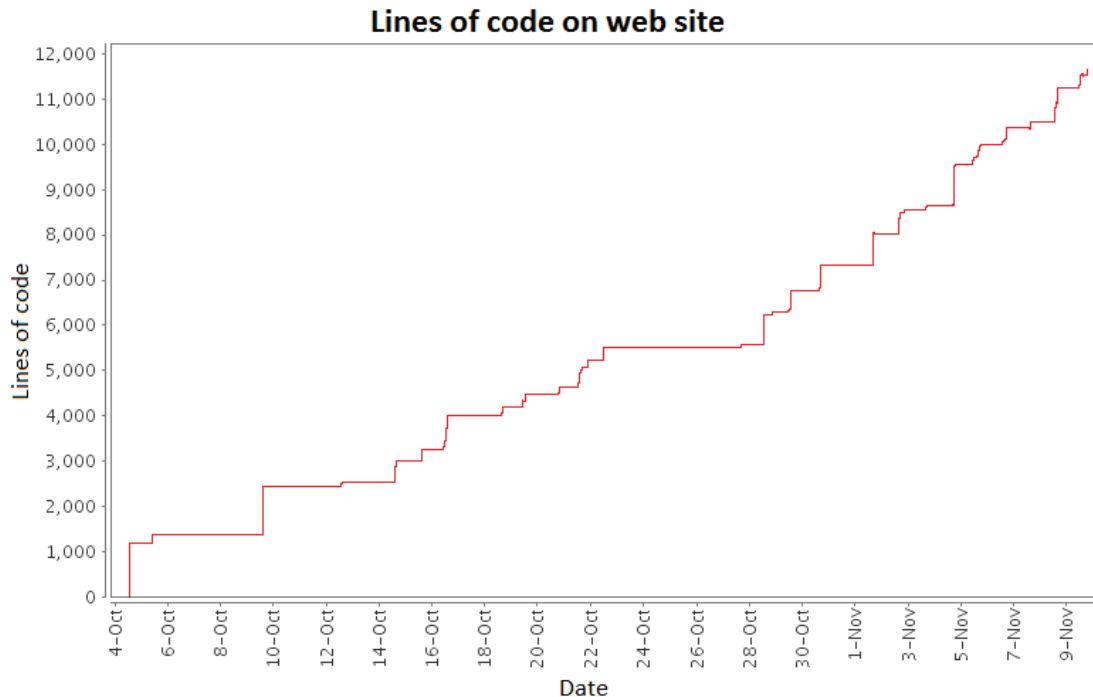


Figure 10.7: Lines of code on the web page

Table 10.2 shows the lines of code for each group member. We wrote about 11 500 lines of code, although the table says over 14 000 as some libraries were included in this number. The group member with experience of web development using the same framework made almost half of the implementation. The good thing is that everyone took part in the implementation and made at least 10% of the code.

Author	Author Id	Changes	Lines of Code	Lines per Change
smistad	smistad	494 (55.0%)	6585 (45.8%)	13.3
bjerkhei	bjerkhei	140 (15.6%)	2274 (15.8%)	16.2
matsgora	matsgora	74 (8.2%)	2059 (14.3%)	27.8
olavu	olavu	95 (10.6%)	1783 (12.4%)	18.7
bjorgan	bjorgan	95 (10.6%)	1670 (11.6%)	17.5
Totals		898 (100.0%)	14371 (100.0%)	16.0

Table 10.2: Lines of code for each group member

10.2 Results

The assignment given by Tapir Academic Press was to develop a prototype of a net store for distributing electronic articles. After discussions with the customer about what functionality the net store should have, the product item was finalized with 16 items. As can be seen in 10.3 all the items except one were finished. The result is a complete net store that can run on the specified server, domeshop.no.

Story ID	Story name	Priority	Sprint finished
1	Customer browsing products	H	2
2	Customer registers account	H	1
3	Customer searches for products	H	2
4	Admin uploads products	H	1
5	Admin change product attributes	H	2
6	Admin changes IP range	H	2
7	System admin controls access of users	H	2
8	Admin manages groups of products	H	2
9	Admin views statistics	M	3
10	Customer purchases product	H	3
11	Customer subscribes to journals	H	4
12	Admin watermarks file	M	3
13	Customer shopping cart	M	4
14	Admin gives discounts	L	4
15	Customer makes compendiums	L	-
16	Customer order history	L	3

Table 10.3: Items in the product backlog

The product can be used as a stand-alone product, or can be integrated with the existing book store currently in use, which is also written in PHP. The resulting net store of this project is highly modular. As such, it is possible to reuse parts of the code, add new code, or remove some of the code in an easy way if necessary.

10.3 The customer and the project

Tapir had a dedicated person, Yngve Syrtveit to take care of this project. The second meeting with him took place at Tapir, where we got to introduced to about ten of the workers there.

The task was a wide spanning one, so we had a great amount of things to implement. Even though much of it was not that complex, there was lots of things to do and we got a challenge of keeping track of the different parts of the project. If we saw that there were things we would not be able to implement we told the customer in advance and then we discussed how to handle it. The customer were understanding and flexible when such issues arise.

We had a meeting with the customer two times every week. This routine quality assured the dialog with the customer regarding the product. Both parties could also invite to a discussion by email and get an answer in seldom more than a couple of hours.

The customer has been willing to help, and has been giving valuable feedback on both the product and the report.

10.4 The supervisors

Every week we had a meeting with the supervisors, who was Basit A. Kahn and Bian Wu. This meeting we gave status on the project and the report. Before this meeting we sent an agenda for the meeting, an updated sprint backlog and a status report including the amount of hours each group member had worked the previous week. We also sent the report together with a change log, meeting minutes from customer and other meetings. All this gave a pressure on working hard. Most often the message from the supervisors was to write more on the report. This way we felt the pressure from them regarding the report, and at the same time knowing we should be doing implementing to satisfy the customer. Even though this situation could be frustrating during the project, we can now when we are close to the final delivery thank the supervisors for pushing us to get most report work done early. They also gave us valuable feedback on the report as it was evolving.

10.5 Further work

There are some extensions to the system that can be made in the future. In this section we present some of the extensions we recommend.

Caching for increased performance

Instead of retrieving and processing the same data each time a user requests it, and the result of the request is the same every time for every user, one can store the result and display it to the next user who requests it, without any processing or data querying. This technique is called caching and increases performance and use less resources.

For instance the attributes of products will not change often and is therefor ideal to cache, especially since traversing through the entire group tree of a product to find its attributes is time consuming.

For more information on how to implement this using the Zend framework, see [41].

Search Engine Optimization

Since most people will probably find this site through a search engine by searching on any of the products, the website would benefit from a proper search engine optimization. For example by using more meta tags when displaying product and have more information in the title of each product page could increase the hit rate to the website.

Support for multiple languages

Currently the entire website is in Norwegian. Zend framework has support for translating the static text parts of the website and can easily be implemented with the site. Basically this is done by putting every static text string on the site through a translation function. This function looks for a translation and if no translation is found the input string is used instead. See [42] for more information on how to implement this functionality.

The translation of the dynamic content, i.e. the content in the database like the name of the products and all the attributes, is much more complex to implement. To implement this one would have to create extra fields in the database to support multiple languages.

Compendium feature

This was the one user story which we did not have time to implement. But we think that this feature could be easily implemented by using the already implemented feature of saving shopping carts. This feature would also need functionality for setting the order of products, use `Zend_Pdf` to concatenate the articles and make the compendium accessible by a specific URL.

10.6 Suggestions for improvements

This section provides ideas for improving the course.

Group size

The groups that were defined before the course started didn't exactly work out because a lot of students take their fourth year in another country. The course responsible should have had a list of students actually present and taking the course. Since there were a lot of students missing the course responsible had to move students between groups and this led to that some groups got smaller than they should have been. Our group consist of five people, which was ok, but given the workload of this project it would have been better with 6 or 7 people in each group.

Lectures

The lectures were mostly good, but sometimes we feel that they came at the wrong time. For example the technical writing seminar were set to the 1st of October. By that time we had already written a lot in the report. When we got the tips in the lecture, we had to go back in the report to correct it. If the lecture had been given earlier we could have saved some hours that could have been used to implementation or further report work. In addition we think that some of the lectures that took four hours were too long. We are used to lectures going in a relatively fast speed, but in this course some of the lectures could have been completed in less than four hours. Some lectures were dragged on a little too much, as if to just fill the time slot of four hours.

Time

We suggest that the kick-off for this course is held in the start of the semester instead of waiting for a week. In such a big project we need all the time we can get to satisfy in delivering a good product and a well-written report. Of course the issue with time is combined with us being only five students.

Supervisor and customer

The supervisor and assistant supervisor did a good job guiding us through this project. One thing that could have improved our report work were if the supervisors could have been more consistent with the info booklet when setting requirements. We also feel that groups with different supervisors had some differences to the report requirements. It would have been better if all the supervisors and the info booklet were consistent in all areas. When it comes to the customer we were lucky and got a customer that really cared about our work and contributed with ideas for improvement throughout the whole project. Some of the other groups might not have the same experience. It is important that the customer genuinely cares about the project and takes part in every aspect of it.

10.7 Concluding remarks

This course has been a great experience and extremely informative. Being put together in a group with people you don't know and work on a project you have no former experience with, is a nice challenge. It creates a hectic work situation, where we can put our theoretical knowledge into use. Decision and problem solving, coordination, management, report writing and especially contact with a real customer will be valuable experiences. The team feels this project has been of a tremendous educational value, even if the workload at times was high.

We are all satisfied with the final product. It took time to learn the programming language and the file structure in Zend framework, and as such little implementation was done in the first sprint. This meant more work for the other sprints, but with the added fourth sprint we managed to get done all of the product items except one. The result was a web shop with tailored for electronic articles and journals.

Glossary

Glossary

Notation	Description
abstract	Context:book. Summary of content, 73
acceptance test	Here: testing done by the project customer prior to accepting transfer of ownership, 139
acceptance test	See page 20, 20
Access control	A system which enables an authority to control access to areas and resources in a given computer-based information system, 3
access control list	See page 48, 47
accountability	what as subject has done in the system, 47
ACL	See access control list, 47
adapter	a software component that converts transmitted data from one presentation form to another, 153
admin	See administrator, 51
administrator	Context: Project. See page 54, 51
AJAX	asynchronous javascript + XML. A group of interrelated web development techniques used on the client-side to create interactive web applications., 134
apache	Apache HTTP server. A popular open source web server, 37
API	Short for application programming interface. An interface that a software program implements in order to allow other software to interact with it, 24
attribute	a specification that defines a property of an object, element, or file, 103

Notation	Description
authentication	confirming the identity of a person or system, 47
authorization	specifying access rights to resources, 47
backlog	a prioritized set of high level requirements of work to be done, 10
bibtex	reference management software for formatting lists of references, 73
burndown chart	a graphical representation of work left to do versus time, 91
cascading stylesheets	a style sheet language used to describe the presentation semantics of a document written in a markup language., 76
checkout	the location where a transaction occurs, 120
CIDR	See classless inter-domain routing, 45
class diagram	a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, and the relationships between the classes., 82
classless inter-domain routing	A methodology of allocating IP addresses and routing Internet Protocol packets, 45
CMF	See Content management framework, 39
CMS	See content management system, 39
compendium	Here: compilation of scientific papers into a book, 82
concurrent versions system	a revision control system for open source and commercial software development, 29
constructor	a special block of code in a class consisting of statements called when an object is created, either when it is declared or when it is dynamically constructed on the heap through the keyword new, 153

Notation	Description
content management framework	an application programming interface for creating a customized content management system, 39
content management system	a collection of procedures used to manage work flow in a collaborative environment, 39
controller	See page 80, 81
CSS	See cascading stylesheets, 76
customer	Context:Project. See page 54, 51
CVS	See concurrent versions system, 29
Daily scrum meeting	See page 26, 25
data-flow diagram	a graphical representation of the flow of data through an information system, 30
database management system	a set of computer programs that controls the creation, maintenance, and the use of the database in a computer platform or of an organization and its end users, 162
datagram	Another name for message. In information technology a datagram is a discrete package of data and headers which contain addresses (which is the basic unit of transmission across an IP network). People also calls it packet, 46
DBMS	See database management system, 162
development methodology	See Software development methodology, 24
DFD	See data-flow diagram, 30
digital distribution	The practice of providing content in a purely digital format, which is downloaded via the internet straight to a consumer's home., 137
Digital watermarking	See page 42, 41
DRM	Short for Digital Rights Management. Acronym for technologies which provide access control for digital media, 3

Notation	Description
e-commerce	Electronic commerce. Consists of the buying and selling of products or services over electronic systems such as the Internet and other computer networks, 39
eBook	Short for Electronic Book aka digital book., 3
eLearning	Short for electronic learning encompasses technology-enhanced learning or very specific types of technology-enhanced learning such as online web-based learning, 3
endnote	a commercial reference management software package, used to manage bibliographies and references when writing essays and articles, 73
entity	either a thing in the modeled world or a drawing element in an entity relationship diagram, 154
ER diagram	Entity relationship diagram. An abstract and conceptual representation of data., 84
FEIDE	See page 48, 47
framework	a re-usable design for a software system. See software framework, 24
functional requirement	may be calculations, technical details, data manipulation and processing and other specific functionality that define what a system is supposed to accomplish., 10
Gantt-chart	A type of bar chart that illustrates a project schedule. Gantt charts illustrate the start and finish dates of the terminal elements and summary elements of a project., 10
Gantt-diagram	See Gantt-chart, 10
graphical user interface	type of user interface item that allows people to interact with programs in more ways than typing, 76

Notation	Description
groups	Context:Project. See page 97, 98
GUI	See graphical user interface, 76
hash function	any well-defined procedure or mathematical function which converts a large, possibly variable-sized amount of data into a small datum, 166
HTML	HyperText Markup Language. The predominant markup language for web pages., 24
HTTP	Hypertext transfer protocol. The standard protocol for retrieving inter-linked resources on the internet, 44
implementation	a realization of a technical specification or algorithm as a program, software component, or other computer system, 50
institution	Context:Project. A special user that can have several other users connected to it. This special user can have a special IP range of subnet resulting in users connected from these networks are granted privileges of this special user, 106
integration test	See page 20, 20
interface	A contract declaring that every class extending this class must implement the predefined rules in the given interface, 153
IP range	Series of IP addresses defined by a starting and an ending address, 3
IP-filtering	A mechanism that decides which types of IP datagrams will be processed normally and which will be discarded. An IP filter operates mainly in layer 2, of the TCP/IP reference stack, 3

Notation	Description
IPv4	the fourth revision in the development of the Internet Protocol (IP) and it is the first version of the protocol to be widely deployed., 45
IPv6	the next-generation Internet Protocol version designated as the successor to IPv4, 45
ISBN	International Standard Book Number is a unique numeric commercial book identifier based upon the 9-digit Standard Book Numbering, 73
ISSN	International Standard Serial Number is a unique eight-digit number used to identify a print or electronic periodical publication, 73
Iteration	Context: Scrum. See Sprint, 9
javascript	an object-oriented[2] scripting language used to enable programmatic access to objects within both the client application and other applications, 134
journal	In context to this project a journal is a compilation of scientific papers, following the requirements given for scientific papers, published regularly, 2
loopback	a virtual network interface implemented in software only and not connected to any hardware, but which is fully integrated into the computer system's internal network infrastructure., 46
model	See page 80, 81
model view controller	See page 80, 81
Modular	Design of a system in parts, 9
module	Code put together in one folder/place dividing the complete system in smaller parts so it is easier to manage, 154

Notation	Description
module test	See page 20, 20
MVC	See model view controller, 39
MySQL	My Structured Query Language. A relational database management system, 162
NAT	See network address translation, 46
netmask	See network mask, 45
network address translation	the process of modifying network address information in datagram packet headers while in transit across a traffic routing device for the purpose of remapping a given address space into another, 46
network mask	a 32-bit mask used to divide an IP address into subnets and specify the networks available hosts, 45
non-functional requirement	a requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behaviors, 10
object	any entity that can be manipulated by the commands of a programming language, such as a value, variable, function, or data structure., 103
paper prototype	a widely used method in the user-centered design process, a process that helps developers to create software that meets the user's expectations and needs - in this case, especially for designing and testing user interfaces., 85
payex	See page 45, 44
PDF	See portable document format, 24
PHP	Hypertext Preprocessor, is a widely used, general-purpose scripting language that was originally designed for web development, to produce dynamic web pages., 7

Notation	Description
portable document format	a file format created by Adobe Systems in 1993 for document exchange., 24
product	Context:Project. Items that are offered by the network store, 51
Product backlog	a high-level document for the entire project. It contains what will be built, backlog items, 10
Product owner	See page 15, 15
Project management	See. 18, 17
prototype	A first of preliminary model of something, 2
proxy server	a server that acts as an intermediary for requests from clients seeking resources from other servers., 47
QA	See quality assurance, 19
quality assurance	refers to planned and systematic production processes that provide confidence in a product's suitability for its intended purpose., 19
Requirement specification	Software Requirements Specification (SRS) is a complete description of the behavior of the system to be developed., 9
revision control system	a software implementation of revision control, the management of changes to documents, programs, and other information stored as computer files, that automates the storing, retrieval, logging, identification, and merging of revisions., 29
Scrum	See page 25, 24
Scrum master	See page 15, 15
scrum sprint	See page 26, 25
search engine	Aka web search engine. A tool designed to search for information on the World Wide Web. Ex google, 23

Notation	Description
search engine optimization	the process of improving the volume or quality of traffic to a web site from search engines via natural or un-paid search results, 23
secure sockets layer	cryptographic protocols that provide security for communications over networks such as the Internet, 43
SEO	See search engine optimization, 23
session	a session is a semi-permanent interactive information interchange, 106
session variable	See session, 106
SHA	A set of cryptographic hash functions designed by the National Security Agency, 166
shopping cart	a digital counterpart to the physical version, 138
short message service	See page 45, 44
SMS	See short message service, 44
SOAP	Short for Simple Object Access Protocol, is a protocol specification for exchanging structured information in the implementation of Web Services in computer networks., 44
social-website	sites where people socialize and interact with eachother, 73
Software architecture	Structure of a program or computing system, 81
Software development methodology	The documented collection of policies, processes and procedures used by a development team or organization to practice software engineering, 24
software framework	A re-usable design that may include support programs, code libraries, a scripting language, or other software to help develop and glue together the different components of a software project. Various parts of the framework may be exposed through an API, 24

Notation	Description
sprint	See scrum sprint, 25
Sprint backlog	A document containing information about how the team is going to implement the features for the upcoming sprint., 10
Sprint planning meeting	See page 26, 25
Sprint review meeting	See page 26, 25
SSL	See Secure sockets layer, 43
SSL certificate	an SSL (See SSL) electronic document which uses a digital signature to bind together a public key with an identity, 43
subnet	See subnetwork, 45
subnetwork	See page 46, 45
subscription	Context:Project. Annual payment so the user(s) can download all the subscribed articles/journals published, 133
subversion	a revision control system for open source and commercial software development, 29
SVN	See subversion, 29
sysadmin	See system administrator, 51
system admin	See system administrator, 51
system administrator	Context:Project. See page 54, 51
system test	See page 20, 20
Team members	Context Scrum. See 15, 15
UML	Unified Modeling Language, an object modeling and specification language used in software engineering, 82
unit test	See page 20, 20
UNIX	a computer operating system originally developed in 1969 by a group of AT&T employees at Bell Labs, 16

Notation	Description
URL	Uniform Resource Locator, a subset of the Uniform Resource Identifier (URI), 120
usability test	See page 20, 20
use case	a description of the system behavior as it responds to a request that originates from outside of that system., 33
utf-8	a variable-length character encoding for Unicode. It is able to represent any character in the Unicode standard, yet is backwards compatible with ASCII., 16
VAT	value added tax, 135
VCS	See revision control system, 29
version control	See page 29, 28
Version control system	See version control, 28
view	See page 80, 81
Waterfall model	a sequential software development process, in which progress is seen as flowing steadily downwards (like a waterfall) through the phases., 24
watermarking	See Digital watermarking, 41
wrapper class	wrapper class is any class which encapsulates the functionality of another class or component. These are useful by providing a level of abstraction from the implementation of the underlying class or component, 120
WSDL	Short for Web Services Description Language. An XML-based language that provides a model for describing Web services., 44
XML	Extensible Markup Language. A set of rules for encoding documents electronically, 134

Notation	Description
Zend Framework	open source, object-oriented web application framework implemented in PHP 5 and licensed under the New BSD License., 16

References

- [1] Methods and Tools. [online].
www.methodsandtools.com/archive/scrum1.gif, 09.2009.
- [2] Connexions Waterfall. [online].
cnx.org/content/m28927/latest/graphics1.png, 09.2009.
- [3] Wikipedia Subversion. [online].
[en.wikipedia.org/wiki/Subversion_\(software\)](http://en.wikipedia.org/wiki/Subversion_(software)), 10.2009.
- [4] Enode MVC pattern. [online].
<http://www.enode.com/x/markup/tutorial/mvc.html>, 10.2009.
- [5] Wikipedia Subnetwork. [online].
en.wikipedia.org/wiki/Subnetwork, 10.2009.
- [6] Tapir Academic Press. [online].
<http://www.tapirforlag.no/>, 09.2009.
- [7] Wikipedia Scrum. [online].
no.wikipedia.org/wiki/Scrum, 09.2009.
- [8] Eric J. Braude. *Software Engineering: An Object-oriented Perspective*. Wiley, 2000.
- [9] Free Software Foundation. [online].
<http://www.nongnu.org/cvs/>, 09.2009.
- [10] Inc. CollabNet. [online].
<http://subversion.tigris.org/>, 09.2009.
- [11] Scott Chacon. [online].
<http://git-scm.com/>, 09.2009.
- [12] PHP Group. [online].
<http://www.php.net/>, 09.2009.
- [13] Microsoft. [online].
<http://www.asp.net/>, 09.2009.
- [14] Python Software Foundation. [online].
<http://www.python.org/>, 09.2009.
- [15] The Perl Foundation. [online].
<http://www.perl.org/>, 09.2009.

REFERENCES

- [16] The World Wide Web Consortium (W3C). [online].
<http://www.w3.org/>, 09.2009.
- [17] Wikipedia Digital Watermarking. [online].
<http://en.wikipedia.org/wiki/JavaScript>, 09.2009.
- [18] Adobe. [online].
<http://www.adobe.com/flashplatform/>, 09.2009.
- [19] Zend. [online].
<http://www.php.net/~helly/php/ext/spl/>, 10 2009.
- [20] Django Software Foundation. [online].
<http://www.djangoproject.com/>, 09.2009.
- [21] Wikipedia Digital Rights Management. [online].
no.wikipedia.org/wiki/Digital_rights_management, 09.2009.
- [22] Wikipedia Digital Watermarking. [online].
http://en.wikipedia.org/wiki/Digital_watermarking, 09.2009.
- [23] Paragallo AS. [online].
<http://paragallo.com/>, 09.2009.
- [24] DIBS. [online].
<http://www.dibs.no>, 09.2009.
- [25] Payex. [online].
<http://payex.no>, 09.2009.
- [26] Paypal. [online].
<http://paypal.com>, 09.2009.
- [27] Sendega. [online].
<http://sendega.com>, 09.2009.
- [28] Wikipedia Digital Rights Management. [online].
http://en.wikipedia.org/wiki/Classful_network, 09.2009.
- [29] Wikipedia Digital Rights Management. [online].
http://en.wikipedia.org/wiki/Network_address_translation, 09.2009.
- [30] Inc. Advameg. [online].
http://www.faqs.org/docs/linux_network/x-087-2-firewall.filtering.html, 09.2009.
- [31] Wikipedia Digital Rights Management. [online].
http://en.wikipedia.org/wiki/Proxy_server, 09.2009.
- [32] Wikipedia Digital Rights Management. [online].
http://en.wikipedia.org/wiki/Access_control, 09.2009.

- [33] UNINETT. [online].
<http://feide.no/>, 09.2009.
- [34] Janine Bernat. [online].
<http://www2.cs.uregina.ca/~bernatja/crowsfoot.html>, 10.2009.
- [35] Keith Pope. Zend framework 1.8 web application development, 2009.
- [36] Zend. [online].
<http://www.framework.zend.com/manual/en/zend.db.html>, 11 2009.
- [37] Zend. [online].
<http://framework.zend.com/manual/en/zend.auth.html>, 11 2009.
- [38] Zend. [online].
<http://www.php.net/~helly/php/ext/spl/>, 10.2009.
- [39] MySQL. [online].
<http://www.mysql.com/>, 11 2009.
- [40] Zend. [online].
<http://www.framework.zend.com/manual/en/zend.db.table.html>, 11 2009.
- [41] Zend. [online].
<http://framework.zend.com/manual/en/zend.cache.html>, 11 2009.
- [42] Zend. [online].
<http://framework.zend.com/manual/en/zend.translate.html>, 11 2009.
- [43] Cake Software Foundation. [online].
<http://cakephp.org/>, 09.2009.
- [44] Mountaing Goat Software User stories for product backlog. [online].
www.mountaingoatsoftware.com/presentations/79-user-stories-for-your-product-backlog, 10.2009.

APPENDIX A

Project directive

A.1 Contact information

	Name	E-mail	Phone number
Tapir			
	Yngve Syrtveit	yngve.syrtveit@tapir.no	45448105/(735) 98441
	Lasse Postmyr	lasse.postmyr@tapir.no	95992842/(735) 98478
Supervisors			
	Basit Khan	basit@idi.ntnu.no	(735) 50435
	Bian Wu	bian@idi.ntnu.no	(735) 91726
Group			
	Joakim Bjerkheim	bjerkhei@stud.ntnu.no	40452569
	Arne Bjørgan	bjorgan@stud.ntnu.no	97670391
	Olav Undheim	olavund@gmail.com	92660199
	Mats-Gøran Karlsen	matsgora@gmail.com	45287973
	Erik Smistad	smistad@idi.ntnu.no	90524585

Figure A.1: Contact list

A.2 Meeting notice

*TDT4290 Customer Driven Project, Group 5
Tapir Academic Press*

Agenda of Meeting

Date: 08.10.2009

Time: 12:15-13:15

Type of meeting: Customer

Place: ITS 236

Purpose: Project planning

Agenda for the weekly meeting with the supervisor

1. Gå gjennom møtereferat fra forrige møte
2. Status på avtaler og signeringer
3. Oppdatering på betalingsløsning og servertilgang. (Yngve)
 1. Payeks info sendt til gruppen, skal prøve å få gunstigere pris gjennom SIT-tapir som benytter payex.
 2. Server hos domeshop kan være en løsning, venter på pristilbud fra IT-Sjefen på pirsenteret.
4. Oppsummering av innholdsportering og testgruppe (Yngve)
 1. Trolig vil 2 av tidsskriftene bli publisert i sin helhet via dette systemet. (komplett historikk)
 2. De andre tidsskriftene vil vi publisere de 2-3 siste utgivelsene.
 3. Redaksjonene for tidsskriftene vil bli forespurt om å sitte i testpanel når spesifikasjonen for denne jobben foreligger. Prototype-walkthrough o.l. vil være mest hensiktsmessig å foreta med forlagets ansatte på Nardo, mens funksjonell testing kan være mer relevant for testpanelene?
5. Innhold sprint (Yngve)
 1. Det er ønskelig å legge de delene av systemet hvor det er størst sansynlighet for lisenskostnader eller hylleware til sene sprinter.
 2. Potensielle deler kan være: betalingsdelen, dmm, cms
6. Eventuelt
7. Planlegge neste møte

Attachments:

1. Minutes of the last customer meeting

Contact: kpro5@idi.ntnu.no

Figure A.2: Notice of meeting

A.3 Meeting notice - supervisors

TDT4290 Customer Driven Project, Group 5
Tapir Academic Press

Notice of Meeting

Date: dd.mm.2009

Time: 14:15-15:00

Type of meeting: Supervisor

Place: ITS 236

Purpose: Work on the project directive

Agenda for the weekly meeting with the supervisor

1. Approval of agenda
2. Approval of minutes of meeting from last advisory meeting
3. Comments to the minutes from last customer meeting or other meetings
4. Approval of the status report, which may be structured as follows:
 - 4.1 Summary
 - 4.2 Work done in this period
 - Status of the documents that are being created
 - Meetings
 - Other activities
 - 4.3 Problems – what is interfering with the progress or taking resources?
 - Problems are often risks that have taken effect.
 - 4.4 Planning of work for the next period
 - Meetings
 - Activities
 - 4.5 Other
5. Review/approval of attached phase documents
6. Other issues are listed here...
7. Other issues

Attachments:

1. Minutes of the last customer meeting

Figure A.3: Notice of meeting, supervisor

A.4 Meeting minutes

*TDT4290 Customer Driven Project, Group 5
Tapir Academic Press*

Minutes of Customer Meeting

Date: 05.10.2009

Time: 12.30-13.20

Secretary: Arne Bjørgan

Purpose: Customer

Participants

<i>Project group:</i>	Joakim Bjerkheim	Present
	Arne Bjørgan	Present
	Erik Smistad	Present
	Olav Undheim	Present
	Mats Gøran Karlsen	Present
<i>Tapir:</i>	Yngve Syrtveit	Present

Meeting summary:

1. The agenda was approved.
2. We show the class diagram to Yngve.
3. We demo the functionality to Yngve. Log in and the possibility to retrieve a forgotten password.
4. Tapir doesn't want to have two different payment solutions for subscribers. The customer pays for the first year through our payment solution, for the following years an email should be sent from our system to tapir, so that they can bill the subscriber for the next year and so on. We should not make functionality for charging the subscriber automatically.
5. We discuss sprint 2. Yngve thinks that the planned work for this sprint is ok.
6. We discuss some issues with connecting to NTNU's SQL server with Windows computers.
7. Yngve might come to Gløshaugen early on Thursday to join the project work.

Specific tasks to be performed:			
Id	Task	Responsible	Deadline
T1	Work on sprint 2. See backlog	All	19.10.2009

Figure A.4: Meeting Minutes

A.5 Weekly status report

GROUP 5
SUMMARY STATUS REPORT
Week 44
21.October - 27.October

1 Introduction

This status report documents what group 5 has done during the second half of week 43 and the first half of week 44. We plan to finish sprint 3 on Sunday, and start a Sprint 4 lasting one week.

2 Progress summary

We have worked at sprint 3, details are found in the tables. We divided the sprint 3 backlog items so that each person is working on one item, which is what have been done this week. Details for the responsibility can be found in the sprint 3 backlog document.

Detailed group member work is shown below (week day, date, activities, hours):

Joakim Bjerkheim

Week 44	Wednesday	21-oct-2009	Supervisor meeting, report	4
	Thursday	22-oct-2009	Sprint 3	5
	Friday	23-oct-2009		
	Saturday	24-oct-2009	Sprint 3	3,5
	Sunday	25-oct-2009	Sprint 3	3
	Monday	26-oct-2009	Customer meeting, sprint3	5
	Tuesday	27-oct-2009	Sprint 3, meeting preparations	4
Week total -->				24,5

Figure A.5: Weekly status report

A.5. WEEKLY STATUS REPORT

Olav Undheim

Week 44	Wednesday	21-sep-2009	Supervisor meeting, sprint 3 startup	3
	Thursday	22-oct-2009	google analytics	5
	Friday	23-oct-2009		0
	Saturday	24-oct-2009		0
	Sunday	25-oct-2009		0
	Monday	26-oct-2009	admin log	8
	Tuesday	27-oct-2009	sprint 3 work, statusreport writing	5
Week total -->				21

Arne Bjørgan

Week 43	Wednesday	21-oct-2009	Dentist	0
	Thursday	22-oct-2009	Sprint 3 work, my account	5
	Friday	23-oct-2009	Sprint 3 work, My Account	4
	Saturday	24-oct-2009		
	Sunday	25-oct-2009	Sprint 3 work, Order History	3
	Monday	26-oct-2009	Customer meeting, Sprint 3 work, My Account, Order History	6
	Tuesday	27-oct-2009	Sprint 3 work, order history	2
Week total -->				20

Figure A.6: Status report, part 2

Erik Smistad

Week 44	Wednesday	21-oct-2009	Meeting and work with the report	3
	Thursday	22-oct-2009	Working on user story 12	6
	Friday	23-oct-2009	Working on user story 12	5
	Saturday	24-oct-2009		
	Sunday	25-oct-2009	Working on user story 10	3
	Monday	26-oct-2009	Working on user story 10	6
	Tuesday	27-oct-2009	Working on user story 10	3
Week total -->				25

Mats Goran Karlsen

Week 44	Wednesday	21-oct-2009	Supervisor meeting, shopping cart	4
	Thursday	22-oct-2009	Shopping cart	7
	Friday	23-oct-2009	Shopping cart	4
	Saturday	24-oct-2009		
	Sunday	25-oct-2009	Shopping cart	1
	Monday	26-oct-2009	Meeting,	1
	Tuesday	27-oct-2009	shopping cart	7
Week total -->				24

Figure A.7: Status report, part 3

3 Changelog

We wanted to make some changes in the report, e.g. writing the testing section in the sprint 2 chapter. We prioritated the sprint 3 work this week, and will make the changes to the report after sprint 3 is finished.

4 Planned work for next period

Finish sprint 3
 Write sprint 3 chapter in the report
 Start sprint 4 (one week sprint)

Figure A.8: Status report, part 4

Sprint 1

B.1 GUI sketches

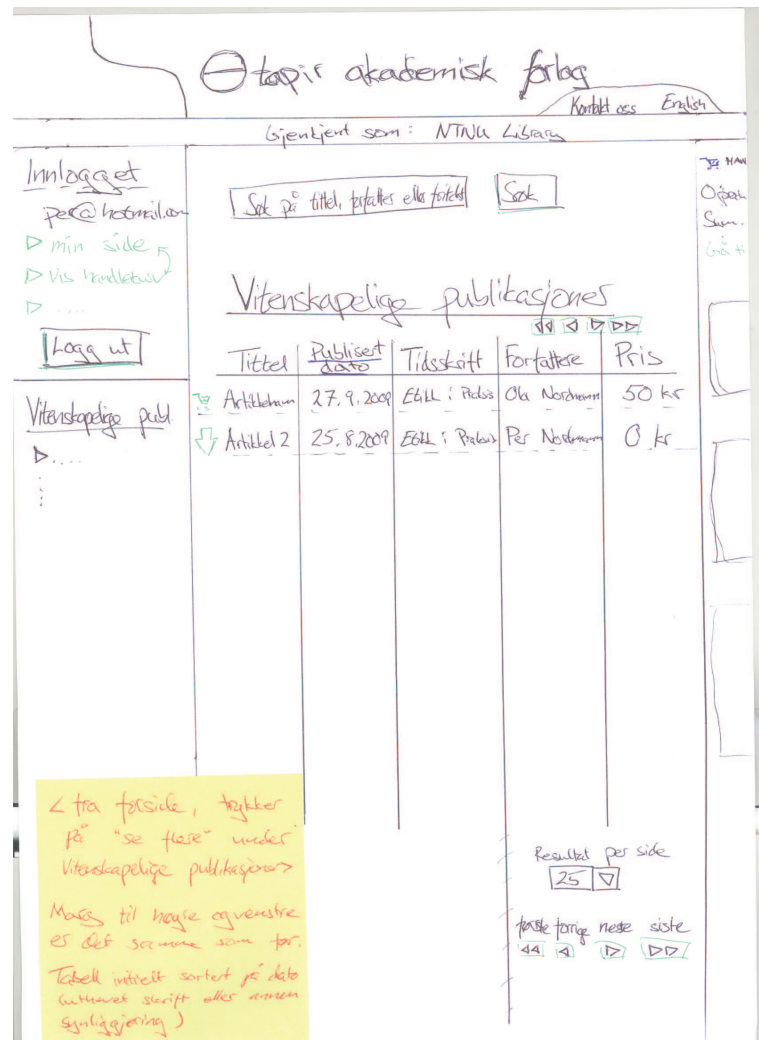


Figure B.1: Page listing scientific publications

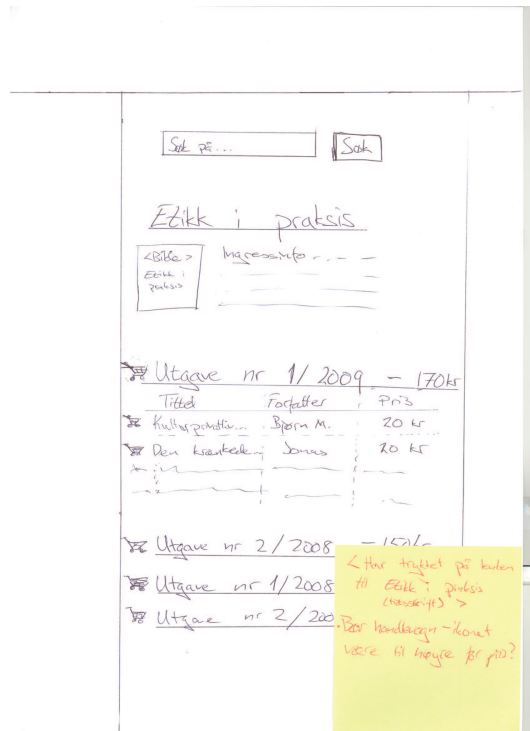


Figure B.2: Page for a specific journal

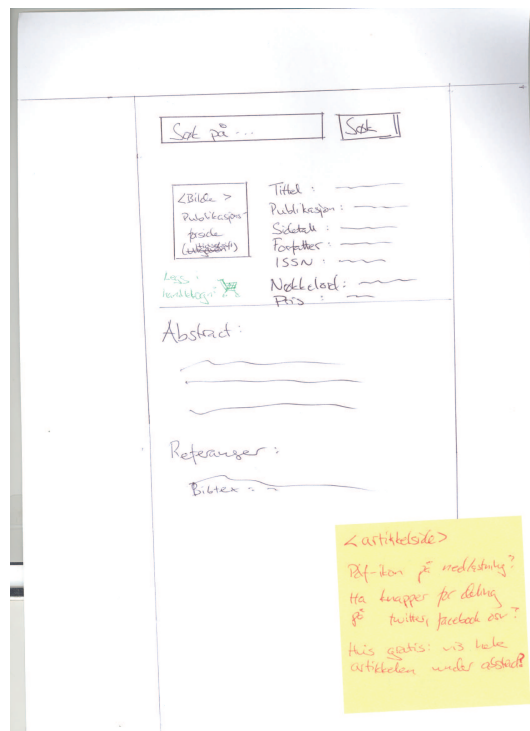


Figure B.3: Page of a specific article

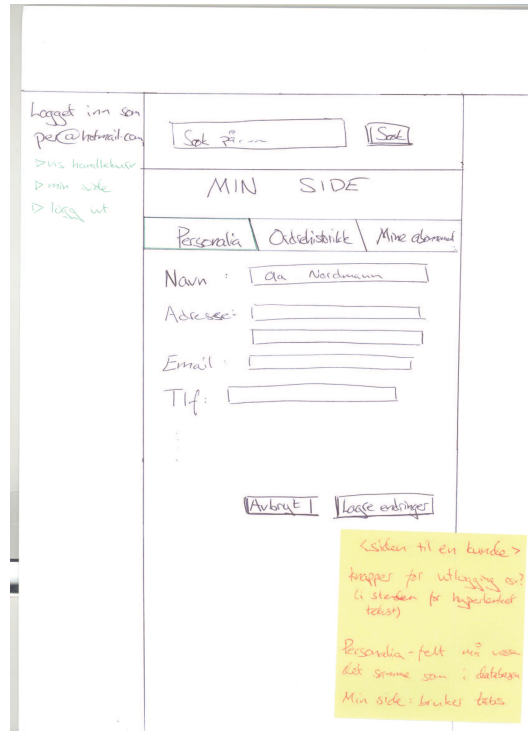


Figure B.4: Customer's personal information page

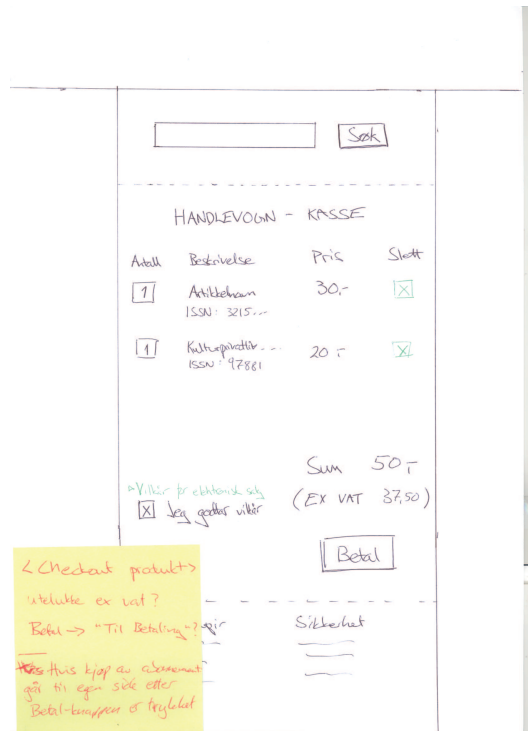


Figure B.5: Shopping cart checkout

APPENDIX C

User manual

This chapter explains the most important functions on the web site. It is a user manual written for administrators of the site, and not regular customers. In general the site should be intuitive enough to use without any form of external guidance. This guide is written to explain the functionality in simple steps and can be used as needed.

Chapter overview

The user manual contains the following sections:

- Section C.2 Log In
This section describes how you log in as an administrator.
- Section C.3 Account
This section describes how to edit a users role and an institutions IP-range
- Section C.4 Product
This section describes how to add a new file and add attributes to it.
- Section C.5 Groups
This section describes how to add a subgroup to an existing group, change attributes of a group and add an existing product to a group.
- Section C.6 Journal
This section describes how to add an issue to a journal and add an article to an issue.
- Section C.7 Subscription
This section describes how to assign a user subscription.
- Section C.8 Statistics and log
This section describes how to access statistics and log for the page.
- Section C.9 Discount
This section describes how to add discount for a total shopping price, and add discount for a group.

C.1 Log in

1. Insert username and password in the text field
2. Press "Logg inn" button

After having logged in, the menu that displays is showed in C.6.



Figure C.1: Admin menu

From this point on, we assume that you are logged in.

C.2 Account

Give a user administrator rights

1. Follow the link to accounts



Figure C.2: Admin menu

2. Press edit user

Oversikt over brukere

[Legg til bruker](#)

Brukernavn	Navn	Rolle	Rediger bruker	Rediger nettverksområde	Ordrehistorikk	Slett bruker
joa@joakim.no	Joki	Admin	Rediger		Ordrehistorikk	Slett
ntnu@ntnu.no	NTNU	Institusjon	Rediger	Rediger	Ordrehistorikk	Slett
nytest@nytest.com	nytest	Kunde	Rediger		Ordrehistorikk	Slett
nyadmin@nyadmin.com	nyadmin	Admin	Rediger		Ordrehistorikk	Slett
hei@hallo.se	dsfsf	Institusjon	Rediger	Rediger	Ordrehistorikk	Slett
ntnu@idi.no	NTNU Nytest	Institusjon	Rediger	Rediger	Ordrehistorikk	Slett
kunde@kunde.no	Vanlig Kunde	Kunde	Rediger		Ordrehistorikk	Slett
UiO@UiO.no	UiO	Institusjon	Rediger	Rediger	Ordrehistorikk	Slett
yngve.syrtveit@tapir.no	Yngve Syrtveit	Admin	Rediger		Ordrehistorikk	Slett

Figure C.3: Choosing edit user

3. Choose administrator in the drop-down menu and choose save

Rediger bruker

Fullt navn: *

Epost: *

Passord:

Må bestå av minst 6 tegn

Gjenta passord:

Telefon:

Faktura-adresse

Gate: *

Postnr: *

Poststed: *

Leverings-adresse

Gate:

Postnr:

Poststed:

Rolle: *

[Avbryt](#) [Save](#)

Figure C.4: Choosing admin and save

Give customer an IP range

Follow the previous manual. Instead of clicking on edit user, click edit network area. You will then get this page:

Kunde: NTNU

Registrerte subnett
 Legg til subnett

Subnettadresse	Netmaske	Rediger	Slett

Registrerte IP-områder
 Legg til IP-område

Startadresse	Sluttadresse	Rediger	Slett
0.0.0.0	0.0.0.1	Rediger	Slett

Figure C.5: Ip range page

The you just choose to add an IP-range and you fill in start- and end-address and choose save. The IP-range is now saved for this account.

C.3 Product

Click "Produkter" in the admin menu to go to the product menu. The product part here assumes you are in the product menu when doing these steps.



The screenshot shows the admin interface for Tapir Akademisk Forlag. At the top, there is a header with the logo and navigation links for "English" and "Kontakt". Below the header, the main content area is titled "Filer" and includes a link "Legg til en ny fil". A search bar is present with the text "Søk etter en fil:" and a "Finn" button.

There are two tables of files:

Upubliserte filer

Navn	Type	Publiser	Rediger	Slet
Er det å forske på praksis viktig for praksisfeltet?	Vitenskaplige artikler	Publiser	Rediger	Slet
Kjempetest	Vitenskaplige artikler	Publiser	Rediger	Slet
Wee	Vitenskaplige artikler	Publiser	Rediger	Slet
Weee2	Vitenskaplige artikler	Publiser	Rediger	Slet

Publiserte filer

Navn	Type	Depubliser	Rediger	Slet
Lær middel å kjenne	Læremidler	Depubliser	Rediger	Slet
NavnEksempel	E-bøker	Depubliser	Rediger	Slet
TestNavn	Vitenskaplige artikler	Depubliser	Rediger	Slet
Student som læremiddel	Læremidler	Depubliser	Rediger	Slet
Den store Vitenskapsartikkel	Vitenskaplige artikler	Depubliser	Rediger	Slet
Test i ebok	E-bøker	Depubliser	Rediger	Slet
Opplastingstest	Vitenskaplige artikler	Depubliser	Rediger	Slet
Test	Vitenskaplige artikler	Depubliser	Rediger	Slet

Figure C.6: Admin menu

Add a new file for sale

1. Click "Legg til ny fil"
2. Write the name of the file(will be displayed in views on the page), choose file type, choose the file and click "lagre". The file is now in the database
3. Give desired attributes. Be aware that all attributes that are set will be shown on the web site. Press "Lagre" when finish
4. You can still edit, delete and add attributes to the file. The file will not be displayed on web pages until it is published. Press "Publiser denne filen" to do this.

Add attributes to a file

1. Choose the desired file either by clicking on it in the list or by using the search field on top of page
2. Click "Legg til attributter til denne filen"
3. Choose the wanted attribute and "Fortsett"
4. Repeat number 2-3 for all attributes you want to set

C.4 Groups

The group system is to be found under "Grupper" in the admin menu. Groups are used to manage files that belong together or have some common attributes. An attribute could be the year it was published, the language of the text or even the price of the file.

Add a subgroup to an existing group

1. Press "Lag en ny gruppe"
2. Write the name of the group you want to create
3. In the drop down menu, choose the parent group you want your new group to inherit from
4. Press "Lagre og legg til attributter"
5. The attributes you don't specify here that are set in the parent group will get the same values as the parent group

Change attributes of a group

1. Click the title of the group you want to edit attributes for
2. Press "Rediger" in the attribute table to the right for the attribute you want to change
3. Assign the new value and click "Lagre"

Add a existing product to a group

1. Go to the group menu and click the name of the group
2. Click "Legg til en eksisterende fil i denne gruppen"
3. Decide whether the group attributes should overwrite the file attributes
4. State the name of the product and press "Lagre"

C.5 Journal

The journal part is quite similar to the product part, but some differences occur due to issues (editions).

Etikk i praksis - Utgaver

Legg til ny utgave

Tittel	Legg til ny fil	Redi
Etikk i praksis - Utgave 1, 2009	Legg til ny fil	Redi
Etikk i praksis - Utgave 2, 2009	Legg til ny fil	Redi
Etikk i praksis - Utgave 2, 2008	Legg til ny fil	Redi
Etikk i praksis - Utgave 1, 2008	Legg til ny fil	Redi

Figure C.7: Journal admin view

Add an issue to a journal

1. Click the title of the journal the issue belongs to
2. Press "Legg til ny utgave"
3. Write the name of the issue and click either "Lagre og avslutt" or "Lagre og legg til attributter"

Add an article to an issue

1. Press "Legg til ny fil"
2. Write the name of the article and find the file on your computer by clicking "Velg"
3. State the attributes you like and press "Lagre"

C.6 Subscription

This is found under "Abonnement" in the admin menu. When a mail is generated from the subscription forms and sent to Tapir, you want to register the request here.

Assign a user subscription

1. Click the link "Legg til et nytt abonnement"
2. Write the name of the account you will give access
3. State the date the subscription expires
4. Write the name of the group the user should get access to

C.7 Statistics and log

1. Choose the statistics link in the administrator menu. You will then arrive at a page with four links.
2. The first link routes you to a user statistics page. This page displays number of downloads per file and number of users that have watched the details page for a file.
3. The second link routes to Google Analytics, which is a tool for generating site statistics. Here you can get a lot of information, like average time on site, bounce rate and so on. You also get access to other useful information, for example you can see how many percent of the customers that came directly or arrived from a search engine. Here is a capture of a Google Analytics report for the Tapir page:

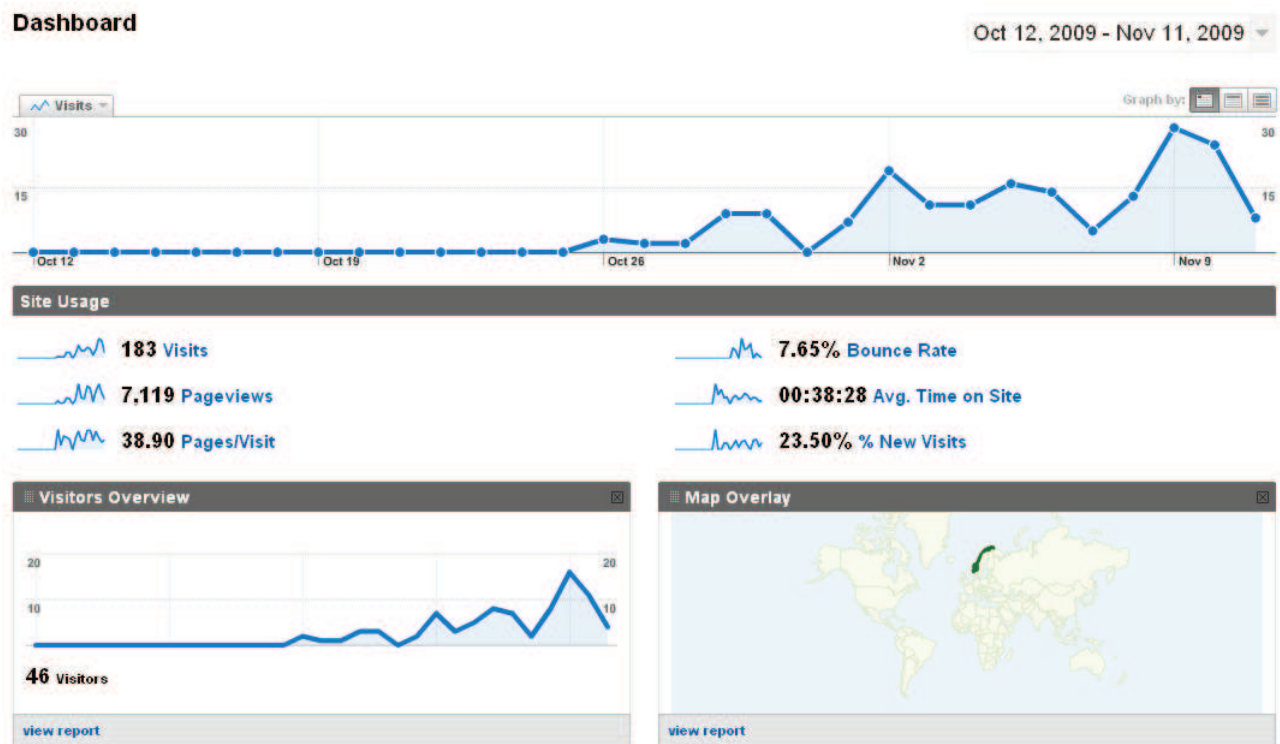


Figure C.8: Google Analytics

4. The two bottom links give you an administrator log and a user log. The attributes found in the table are user, date and event.

C.8 Discount

You have possibility to add discounts to customers and groups. To access the discount menu follow the link named "Rabatt" in the left bar. You will then get two links displayed.

Discount for total shopping price

1. Follow the top link. You will first get a table with existing discounts of this type displayed.
2. To add a new one press the link above the table.
3. You can then fill in a specific user that should get the access, or leave this field open to give the discount to all users.
4. Then you fill out the total price that should give a discount(every shopping cart above this total will get the discount).
5. You then choose if you want to give a static discount or a percentage discount on the total.
6. Press save and check that the discount is correctly displayed in the table

Discount for a group

1. The bottom link routes you to the discount page for groups and file groups.
2. First you will be displayed existing discounts of this type.
3. To add a new discount press the link above the table.
4. If the customer name field is set empty the discount will count for all customers.
5. Then you can define a group id and percentage discount for this group.
6. Press save, and check that the new discount in the table is correct.