Institutionen för datavetenskap
Department of Computer and Information Science

Final Thesis

Front-end design and implementation of a
Web-based Streaming Platform: A User Centered
Approach

by

Fredrik Salin

LIU-IDAlLITH-EX-A--15/071--SE

2015-12-07
Final Thesis

Front-end design and implementation of a Web-based Streaming Platform: A User Centered Approach

by

Fredrik Salin

LIU-IDA/LITH-EX-A--15/071--SE

2015-12-07

Supervisors: Ahmed Rezine, IDA, Linköping University
Albin Carnstam, Mia Clarke, Nil Lakavivat, Netlight Consulting AB

Examiner: Kristian Sandahl, IDA, Linköping University
web 流媒体平台前端的设计与实现：以用户为中心的方法

Front-end design and implementation of a Web-based Streaming Platform: A User Centered Approach

Fredrik Salin

哈尔滨工业大学

Linköping University

2015 年 12 月
Front-end design and implementation of a Web-based Streaming Platform: A User Centered Approach

Fredrik Salin

Kristian Sandahl

Nil Lakavat

Master of Engineering

Software Engineering

Software College

2015

Harbin Institute of Technology
Dissertation for the Master’s Degree in Engineering

**Frontend design and implementation of a Web-based Streaming Platform: A User Centered Approach**

**Candidate:** Fredrik Salin  
**Supervisor:** Associate Prof. Wang Tiantian  
**Associate Supervisor:** Prof. Kristian Sandahl  
**Industrial Supervisor:** Nil Lakavivat  
**Academic Degree Applied for:** Master of Engineering  
**Speciality:** Software Engineering  
**Affiliation:** School of Software  
**Date of Defence:** September, 2015  
**Degree-Conferring-Institution:** Harbin Institute of Technology
摘要
网络在公司和公共场所的大量应用为发展新的服务带来了机遇。近来，网络出现了大量的视频流，并且其数量稳步增加。该领域目前亟需解决的一个问题是，如何在开发软件时充分考虑用户体验，即开发直观、易于掌握和使用的软件系统，使其能够被大多数不同种类的用户所接受和使用。特别是如果采用敏捷开发方法的情况下，该问题变得尤为重要。

本论文描述了一个基于web的流平台的前端的设计与实现。该前端使用JavaScript框架、AngularJS以及HTML5开发，为用户提供上传、共享和讨论视频的功能。

该系统采用用户为中心的设计过程以及敏捷软件开发过程Scrum，在快速开发系统的过程的同时确保优良的用户体验。本文描述了基于Web的流平台的前端的开发实践，为在软件开发过程中结合用户中心的设计思想和Scrum提供指导，进而确保优良的用户体验。

关键词：前端开发，Scrum，用户体验，原型，可用性测试
Abstract

This master thesis investigates how an Agile User Centered approach can be used when designing and implementing the frontend for a web-based on-demand video streaming platform.

Agile and User Experience (UX) are currently two very popular concepts that are frequently discussed in the software development community. Recently there have been a lot of discussions about how to combine Agile development methodologies and UX activities, and it has proven to be a difficult task in practice. This Master Thesis aims at solving this issue by researching methodologies that take both the Agile aspect as well as UX in consideration. Dual-track Scrum, an Agile/User Centered methodology considering UX, was used as development methodology and throughout the development process it was investigated how this methodology, as well as adding UX activities, affected the UX of the streaming platform.

The conclusion from this thesis is that even though the combination of Agile and User Centered processes is difficult, it can be achieved with good results. The thesis presents guidelines for how to successfully combine the two processes as well as what the most important UX activities are and how they can be used in an Agile development project. The thesis also concludes that using a frontend framework, development tools and HTML5 video is very useful for developing the frontend for the streaming platform.

Keywords: Frontend development; Dual-track Scrum; User Experience; UCD; Prototyping; Usability Testing
# Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>摘要</td>
<td>I</td>
</tr>
<tr>
<td>ABSTRACT</td>
<td>II</td>
</tr>
<tr>
<td>CHAPTER 1 INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>1.1 BACKGROUND</td>
<td>1</td>
</tr>
<tr>
<td>1.2 MAIN CONTENT AND ORGANIZATION OF THE THESIS</td>
<td>2</td>
</tr>
<tr>
<td>1.2.1 The thesis project and contributions</td>
<td>2</td>
</tr>
<tr>
<td>1.2.2 Thesis structure</td>
<td>3</td>
</tr>
<tr>
<td>1.3 PURPOSE &amp; AIM</td>
<td>4</td>
</tr>
<tr>
<td>1.4 SCOPE &amp; LIMITATIONS</td>
<td>5</td>
</tr>
<tr>
<td>CHAPTER 2 RESEARCH METHOD</td>
<td>6</td>
</tr>
<tr>
<td>2.1 CASE STUDY</td>
<td>6</td>
</tr>
<tr>
<td>2.1.1 Designing the case study</td>
<td>8</td>
</tr>
<tr>
<td>2.1.2 Data collection</td>
<td>8</td>
</tr>
<tr>
<td>2.1.3 Data analysis</td>
<td>9</td>
</tr>
<tr>
<td>2.1.4 Reporting the findings</td>
<td>9</td>
</tr>
<tr>
<td>2.2 THESIS CASE STUDY</td>
<td>9</td>
</tr>
<tr>
<td>2.2.1 Case Study design</td>
<td>9</td>
</tr>
<tr>
<td>2.2.2 Data collection</td>
<td>10</td>
</tr>
<tr>
<td>2.2.3 Scientific credibility</td>
<td>11</td>
</tr>
<tr>
<td>2.3 BRIEF SUMMARY</td>
<td>12</td>
</tr>
<tr>
<td>CHAPTER 3 THE STATE OF ART</td>
<td>13</td>
</tr>
<tr>
<td>3.1 USER EXPERIENCE</td>
<td>13</td>
</tr>
<tr>
<td>3.1.1 Usability</td>
<td>13</td>
</tr>
<tr>
<td>3.1.2 User Centered Design</td>
<td>14</td>
</tr>
<tr>
<td>3.1.3 User Experience Activities</td>
<td>14</td>
</tr>
<tr>
<td>3.2 WEB USABILITY AND USER EXPERIENCE GUIDELINES</td>
<td>18</td>
</tr>
<tr>
<td>3.2.1 UX guidelines</td>
<td>19</td>
</tr>
<tr>
<td>3.3 MEASURING USER EXPERIENCE</td>
<td>23</td>
</tr>
</tbody>
</table>
3.3.1 Different types of metrics ................................................................. 23
3.4 Software Engineering Methodologies ................................................. 26
  3.4.1 The Waterfall model ................................................................. 26
  3.4.2 Agile software development ....................................................... 27
3.5 Agile Software Development and UX/UCD .......................................... 29
  3.5.1 Similarities between Agile software development and UCD ............... 29
  3.5.2 Differences between Agile software development and UCD .............. 30
  3.5.3 Successful solutions and combined methodologies ....................... 30
3.6 Brief Summary .................................................................................. 36

CHAPTER 4 SYSTEM REQUIREMENT ANALYSIS ........................................ 37
  4.1 The Goal of the System .................................................................. 37
    4.1.1 Functional requirements ......................................................... 37
    4.1.2 Non-functional requirements .................................................. 39
  4.2 Brief Summary ............................................................................... 40

CHAPTER 5 CASE STUDY .......................................................................... 41
  5.1 Interviews ...................................................................................... 41
    5.1.1 Agile and UX ........................................................................ 41
    5.1.2 UX activities .......................................................................... 45
  5.2 Observations .................................................................................. 49
    5.2.1 Sprint 1 ................................................................................ 51
    5.2.2 Sprint 2 ................................................................................ 59
    5.2.3 Sprint 3 ................................................................................ 66
    5.2.4 Sprint 4 ................................................................................ 79
    5.2.5 Sprint 5 ................................................................................ 95
    5.2.6 Sprint 6 ................................................................................ 101
    5.2.7 User statistics ........................................................................ 108
  5.3 Brief Summary ............................................................................... 109

CHAPTER 6 RESULT AND ANALYSIS ...................................................... 110
  6.1 Results from the Usability Tests ....................................................... 110
  6.2 UX Activities .................................................................................. 111
    6.2.1 Sketches .............................................................................. 111
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.2.2 Wireframes</td>
<td>112</td>
</tr>
<tr>
<td>6.2.3 Prototypes</td>
<td>112</td>
</tr>
<tr>
<td>6.2.4 Usability testing</td>
<td>112</td>
</tr>
<tr>
<td>6.3 COMBINING AGILE DEVELOPMENT AND UCD</td>
<td>113</td>
</tr>
<tr>
<td>6.3.1 UX persons in development teams</td>
<td>113</td>
</tr>
<tr>
<td>6.3.2 Close relation/communication</td>
<td>114</td>
</tr>
<tr>
<td>6.3.3 User involvement</td>
<td>114</td>
</tr>
<tr>
<td>6.3.4 Development methodologies</td>
<td>115</td>
</tr>
<tr>
<td>6.3.5 Documentation</td>
<td>115</td>
</tr>
<tr>
<td>6.3.6 Measuring UX</td>
<td>116</td>
</tr>
<tr>
<td>6.4 FRONTEND DEVELOPMENT</td>
<td>116</td>
</tr>
<tr>
<td>6.4.1 Frontend design</td>
<td>116</td>
</tr>
<tr>
<td>6.4.2 Frontend implementation</td>
<td>117</td>
</tr>
<tr>
<td>6.5 BRIEF SUMMARY</td>
<td>118</td>
</tr>
<tr>
<td>CHAPTER 7 DISCUSSION</td>
<td>119</td>
</tr>
<tr>
<td>7.1 METHOD</td>
<td>119</td>
</tr>
<tr>
<td>7.2 RESULTS</td>
<td>119</td>
</tr>
<tr>
<td>7.3 WORK IN WIDER PERSPECTIVE</td>
<td>120</td>
</tr>
<tr>
<td>7.4 FURTHER RESEARCH</td>
<td>121</td>
</tr>
<tr>
<td>CONCLUSION</td>
<td>122</td>
</tr>
<tr>
<td>REFERENCES</td>
<td>124</td>
</tr>
<tr>
<td>APPENDIX A</td>
<td>129</td>
</tr>
<tr>
<td>CASE STUDY INTERVIEW QUESTIONS</td>
<td>129</td>
</tr>
<tr>
<td>APPENDIX B</td>
<td>130</td>
</tr>
<tr>
<td>SUS QUESTIONS</td>
<td>130</td>
</tr>
</tbody>
</table>
Chapter 1 Introduction

1.1 Background

In the mid 1990s Internet came to the world and at this time the number and diversity of computer users grew at an incredible pace [1]. Internet opened a lot of new doors and enabled new opportunities. It made it possible to develop websites and different web based systems. It was possible to very quickly publish things on the web where it was made available for a large audience. This also led to a number of usability and development challenges. The users of the Internet were different from the users of traditional computer systems; they had different backgrounds, native languages, computer experience, network connection speed and so on.

Usability was discussed already before the dawn of Internet, but grew in importance after the founding of it [1].

Today, new Web-based applications are constantly being developed and the complexity of these applications is ever increasing [1]. As the complexity of these products increase, as well as the fact that technology takes a bigger and bigger role in our everyday lives, UX becomes more and more important [1][2]. In 1990 Donald Norman stated in his book “The design of everyday things” that the next challenge for computer software development would not be related to the software or hardware of computer systems, but the usability of these systems [3].

The art of designing software systems that is intended to be human-faced is a relatively new concept [4]. Consumers are putting higher and higher demands on usability of software products. It is therefore important that software development companies focus on developing products that are intuitive and easy to use [4]. The fact that poor usability is the single most common cause of software project failure has been much stressed in written media [5]. Bad usability can for instance result in increased customer training costs, employees become less efficient and bad publicity towards customers. In order to ensure good usability a few things are required: You have to have a good understanding of the users and their needs, being able let the users test the product you are developing and knowledge of the domain. Usability activities lead to good decisions when a given system is to be developed.

For instance in the design-phase of a development project, the high-level key
architecture is developed and how these decisions are made can determine 90% of a product’s cost and performance. It is shown that products that are developed with usability included into their development process can come to the market faster than those without it. They can also be finished on time, with higher quality than projects without usability activities [1].

Agile software development was introduced in the late 1950’s [6] and the popularity has since then increased a lot, especially in the last couple of years [7]. To use Agile methods for software development projects have shown to increase productivity and the quality of the system or product to be developed [7][8][9].

Usability, as a part of User Experience (UX) has a lot of benefits, but in practice the combination of UX and Agile software development has proved to be a difficult task. The problems are mostly related to factors where UX and Agile are different in the aspects of goal, the view on users and documentation [10][11][12]. One common design process is User Centered Design (UCD). UCD is a design process where the needs of the users are the main focus in order to ensure good UX [13]. The combination of UCD and Agile Software Development has been discussed in a number of papers lately [11][14][15][16].

1.2 Main content and organization of the thesis

This section presents the thesis internship project as well as what the contributions of this master thesis were. It also presents what chapters are included in the thesis and how these chapters are structured.

1.2.1 The thesis project and contributions

The master thesis internship project was conducted at Netlight Consulting AB in Stockholm. Netlight Consulting AB is a global IT-consulting company specializing in web development. The company has offices in several countries in Europe including Sweden, Norway, Finland, Germany and England.

The problem as stated by the internship company was that the company had a lot of internal knowledge sharing events including lectures, conferences, seminars, workshops and other events, but only employees who were physically present at the location and time of a given knowledge-sharing event could participate. For instance if an employee was interested in attending an event but it was held in
another country or if the employee could not attend the event this specific date for some other reason the employee would miss out on this opportunity.

The task for the internship was therefore to solve this problem by developing an internal on demand streaming system called “NetTalks” that the company would use to upload, stream, share and discuss internal knowledge sharing events. By recording the presentations and making them available for on demand video streaming, it was possible for all employees to take part in the knowledge sharing whenever they wanted and from any location in Europe. Thereby the developed system improved the knowledge sharing at the company as well as the availability of the knowledge sharing events.

The contributions for this master thesis is a description for how to design and implement the frontend for an on-demand streaming service as well as how to combine Scrum and UX activities to develop a system that has good UX.

1.2.2 Thesis structure

This thesis is structured as follow:
1 Introduction. In the introduction chapter the background of the project and topic are presented as well as the purpose, aim, scope and limitations of the project.
2 Research method. The Research method chapter presents how the research was conducted and how the result can be validated to make sure that the result is accurate.
3 The state of art. The state of art will give present information found in literature about User Experience, Frontend development, Agile methodologies and related research.
4 System requirement analysis. The main goal of the system is presented in this chapter. The functional, non-functional and main use case diagram are listed and described.
5 Case study. The case study chapter presents the result from the conducted case study and includes results from interviews conducted with employees at the internship company. It also includes system design, implementation and testing divided into 6 sprints. During these sprints User Experience activities was added to the frontend development
process and it was analyzed how these activities affected the development process and the UX of the developed system.

6 Result and analysis. The Result and analysis chapter summarizes the findings from literature, interviews and active observations at the internship company and presents relations and differences between these.

7 Discussion. In the discussion chapter the results and method for the thesis are discussed.

8 Conclusion. The final part of the thesis, Conclusion, presents a summary of the thesis findings.

1.3 Purpose & Aim

The internship company had high expectations on the to be developed Web-based streaming platform. The system was required to be developed according to the needs of the users (the employees) and it was going to be a system they would use frequently and for a long time. Therefore, the UX of the system was a very important aspect to consider. The internship company also required that an Agile methodology was to be used for the development process. The combination of Agile and UX processes have been discussed in a number of papers \[5\], but has proven to difficult in practice \[10][12]\.

The aim for this master thesis was therefore to investigate how to include UX activities into an Agile software development process in order to ensure high quality UX when developing the Web-based streaming platform. The main questions for this master thesis were:

1 How should the frontend of a web-based streaming system be designed and implemented?
2 What are the most important factors to consider for a successful combination of Agile Software development and UCD?
3 What and how should UX activities be added to an Agile development process in order to ensure high quality UX?
4 How can it be measured that UX is increasing as the project progresses?
1.4 Scope & Limitations

There are a number of different Agile development methodologies. There are also several different Agile-UX methodologies. The master thesis will be using the Agile-UX methodology Dual-track Scrum for the development of the software system. Other Agile-UX methodologies were investigated but not tested in practice.

There are many frameworks for frontend development that could have been used when implementing the streaming platform during the thesis internship. This thesis report will be using AngularJS as the frontend framework and other frameworks will not be tested.

There are many different UX activities available but because the focus of the report is the combination of UX activities and agile methodologies this thesis report will only try some of the most common UX activities.

The thesis report does not include any project prestudy or research before the start of the development process. The focus was an ongoing development process.
Chapter 2 Research Method

This chapter presents theory for research methods as well as how the research was conducted and what methods that were used. The research was conducted in two parts. The first part included a literature study that was to be the foundation for the theoretical framework. The second part of the research involved a case study that was to be the empirical foundation. The empirical result was analyzed according to the theoretical framework and from the analysis results and conclusions were made.

2.1 Case Study

There are a number of different research methods that can be used to clarify a hypothesis or a question for a given problem [17]. Experiments can for instance be conducted to gain knowledge about how something works or do surveys to understand the general opinion in a given question. Other methods are to look at previously documented events or research products and their functions. Different kinds of research methods are suitable for different kinds of problems. In general it can be said that what kind of research method that should be selected is highly dependent on the proposed research questions [17]. Typical research questions are for instance: Who, what, how or why. From each of these questions there are research methods that are suitable for the given question. Another important aspect is to what extent the researcher has control over the event that happened during the research and if the research is focusing on past or present events [17].

Case studies are usually preferred when the questions “how” or “why” are posed, when the researcher has limited control over the events within the research and when the focus of the research is reality-based events occurring in the present time [17]. The case study research method is suitable when the researcher wants to gain a holistic view and find characteristics of real-time events. These events are for instance the behavior of small groups, processes for organizations or management, changes or performance of organizations. Case study is said to focus on decisions: Why these decisions were taken, how they were implemented and what the result of this implementation was [17]. Case studies are considered to be a suitable research method for software engineering research projects and the amount of studies in this
area has increased [18]. However they are different from case studies in social science in the sense that the study objects are organizations or companies developing software rather than just using it, the studies are often project-oriented rather than function oriented and work process is typically complex engineering work rather than a standard everyday process [18].

There are four different types of research methodologies that serve different purposes [18]. These are: Exploratory, Descriptive, explanatory and improving. The exploratory approach is used when the researcher wants to describe what is happening, when seeking new insights to provide ideas and hypotheses for new research. The descriptive purpose is used for portraying a certain phenomenon or event. The explanatory approach is used when the researcher wants to find an explanation to a given problem or event. When the researcher wants to improve a certain aspect of what is studied the improving approach is typically used [18]. Case studies can be of either single- or multiple-case. For single-case studies only one specific case is researched and from this case conclusions are made [17]. Single case-studies are usually conducted for these types of case studies according to [17]:

- When researching a well-constructed theory
- When researching a unique or extreme case
- For a representative or typical case

Multiple-case studies are, as the name suggests, a case study that involves the studying of multiple cases and from these cases see similarities or differences and draw conclusions. Multiple-case studies are usually to prefer over single-case studies because evidence from several cases is more compelling [17].

The case study process can also be characterized as flexible or fixed. If all parameters are defined already before the case study is started the study process is said to be of a fixed design process. It is more common that case studies are of flexible type, that is, the key process parameters might change during the time of the study [17].

A case study consist of five steps according to [18]:

1. Designing the case study
2. Data collection preparation
3. Collecting data
4. Analysis of collected data
5. Reporting the findings as well as study quality discussion

2.1.1 Designing the case study

Proper planning and design of the case study is vital for the quality of the study [17][18]. The case study design should include the goals or objective of the case study, the unit of analysis, the research questions, a description of how data was collected and where the data was found [17][18].

2.1.1.1 The quality of the case study

Four common components are important in order to establish high quality research design: Construct validity, internal validity, external validity and reliability [17][18].

Construct validity is about making sure that the study measures what it is suppose to be measuring according to the research questions. Construct validity can be increased by using multiple sources of evidence, establishing a chain of evidence and having case study report drafts reviewed by external key informants. Internal validity is about making sure that when investigating whether one factor affects a second factor, it should not be affected by a third factor. Internal validity is mainly used for multiple-case and explanatory case studies. External validity is about to what extent the findings can be generalized and to what extent the findings are relevant to other cases. Reliability is about that it should be possible for another researcher to conduct the same case study with the same results [17][18].

2.1.2 Data collection

The data collected in the study can be of two types: Quantitative or qualitative [18]. The Quantitative involves numbers and the collected data is analyzed using statistics. The qualitative approach involves words, pictures and diagrams and is analyzed using categorization and sorting. Usually case studies use the qualitative data collection approach as this approach provides a deeper description of the issue. Often a combination of the qualitative and quantitative data gives the researcher a better understanding of the phenomena [18].
2.1.3 Data analysis

The data analysis part of the case study is the least developed part of case study research and therefore the hardest one \cite{17}. Data analysis takes practice and there are no clear directives for how to perform an analysis. The quality of the data analysis is highly dependent on the empirical thinking and analytical skill of the researcher \cite{17}. The most important part of the data analysis is to make sure that it is possible to follow the extractions of results and conclusions from the collected data \cite{18}.

2.1.4 Reporting the findings

The case study report should include the findings of the study as well as judge the quality of the case study \cite{17}. It is important to think about who will read the report and adapt the content to this audience \cite{17,18}.

2.2 Thesis case study

This section describes how the case study was performed for this thesis report.

2.2.1 Case Study design

The case study was a single-case flexible exploratory study where the development process of the internal streaming service was the study object. Both qualitative and quantitative data collection was performed. The qualitative data collection was in form of interviews and active observations and the quantitative data collection was in form of a survey.

To get a better understanding of UX and agile methods the research started by doing a literature study. This study was conducted to get a better understanding of what UX activities there are and how they can be integrated into agile software development projects. The study also involved finding existing agile processes that included or had the user in center.

An empirical study in the form of a case study was also conducted, containing both interviews with people at the internship company having knowledge in the research area, as well as active observations, when a proposed process developed by the researcher was tested in practice by the researcher on a live project at the
Thesis for Master’s Degree at HIT and LiU

internship company. By combining information from theory, interviews and observations an optimal process for how to add UX activities to agile software development project is believed to be possible to present.

The case study was performed in order to make sure answers to the main questions of this thesis report would be found. The following objectives were set up for the case study that would work as a guideline for the when the case study was performed:

- Find out what the most common software development methodologies are for ensure good UX
- Find out what are the most frequently used UX activities in practice and how they are used in an agile environment
- Try the found UX activities and methodologies in a real software development project to find out if they are improving the UX, and if they result in high quality UX at the end of the project. High quality UX is defined by:
  - An SUS value of over 85 (which is considered as the highest grade) at the end of the project
  - A 100% completion rate and only low severity usability issues at the end of the project
- What frameworks and tools for frontend development are there and in what way do they improve and/or simplify the development process?

2.2.2 Data collection

The collected data were based on the result from literature studies, active observations and interviews.

2.2.2.1 Observations

The case study observations were of active sort, meaning that the researcher participated actively in the frontend development process at the internship company and from this participation observed and reflected about this process. This was the main source of data collection. UX activities were added into the development process during the time of the internship and the result of adding these activities were evaluated by the researcher. During the active observations the effect of using
UCD and Dual-track Scrum was also analyzed. Chapter 5.2 presents the results of the active observations.

2.2.2.2 Interviews

Interviews were another source of data and these interviews were conducted with people at the internship company to get a better understanding of how people at a software development company usually do their frontend development as well as how they work with UX and agile processes in practice at live project. The respondents were consultants at the internship company having experience from a number of different projects where they have been working with Agile processes and UX. Table 2.1 shows the name, dates and roles of the people interviews as part of the case study. The interviews were of semi-structured nature and all interviewees were asked the same questions except the interviewee on the last interview where follow-up questions were asked on the first conducted interview with the same person. Chapter 5.1 presents the results of the conducted interviews.

<table>
<thead>
<tr>
<th>Name</th>
<th>Date</th>
<th>Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sandra Valett</td>
<td>2015-02-12</td>
<td>UX designer / Frontend developer</td>
</tr>
<tr>
<td>Fredrik Fischer</td>
<td>2015-03-18</td>
<td>Project Manager / Agile Coach</td>
</tr>
<tr>
<td>Lisa Ring &amp; Daniel Andersson</td>
<td>2015-03-24</td>
<td>Interaction designers / Frontend developers</td>
</tr>
<tr>
<td>Niklas Skaar</td>
<td>2015-03-27</td>
<td>Interaction designer</td>
</tr>
<tr>
<td>Sandra Valett</td>
<td>2015-04-16</td>
<td>UX designer / Frontend developer</td>
</tr>
<tr>
<td>Sabina Sonning</td>
<td>2015-06-04</td>
<td>UX designer</td>
</tr>
</tbody>
</table>

2.2.3 Scientific credibility

Construct validity, external validity and reliability have been considered in order to increase the credibility of the case study results. Internal validity has not been considered because it is not a main concern for the chosen type of case study.
2.2.3.1 Construct validity

The construct validity for this thesis was increased by having multiple sources of evidence. Various interviews were conducted with employees at the internship company, active observations were conducted and literature in the subject was researched. A chain of evidence has been established and maintained by matching the thesis research questions, the case study objectives, data collection and the analyzing process.

2.2.3.2 External validity

The case study was of single-case type and these are often hard to generalize. However, general information about UX, UX activities and agile methodologies were gathered from literature studies and from interviews with employees at the internship company having experience from several projects where they had been using this in practice. This information was then used during the active observations and therefore the result from the case study should be applicable to similar projects.

2.2.3.3 Reliability

Reliability has been achieved for the case study by developing a case study protocol describing the case study in detail and including instructions for how to perform the case study. By using the case study protocol another researcher should be able to repeat the procedure and arriving at the same results.

2.3 Brief summary

This chapter begins with describing different research methods that can be used. The case study research method is then described more in depth because this is the method chosen by the researcher of this thesis. Lastly the research structure for this thesis is presented.
Chapter 3 The State of Art

3.1 User Experience

User Experience is defined as “a person's perceptions and responses that result from the use or anticipated use of a product, system or service” by the international ISO standard [19]. All of the aspects of how the users perceive, feel about, use and interactive with a product or system are included in the concept of UX [19][20]. Some of the most important factors that are affecting the quality of the UX are how useful the product is, usability, the design and layout and how easy the product is to learn [20].

3.1.1 Usability

Usability, as a part of UX, is according to the ISO standard Usability defined as “the capability of a software product to be understood, learned, used, and attractive to the user, when used under specified conditions” [21].

Usability is therefore a part of the concept that is UX. Usability is related to the ability of a given user to use a system or product in order to successfully perform a certain task [22]. It can also be seen as quality attribute for how easy a user interface is to use. Usability can be defined to consist of 5 quality components [23]:

- **Learnability** - For a first time user, how easy is it to perform basic task in the system?
- **Efficiency** – How quickly can user that have learned the design perform the tasks?
- **Memorability** – Will the users remember how to perform the tasks when they return to the design after a certain period of time?
- **Errors** – Do users make a lot of errors? How severe are these errors?
- **Satisfaction** – Are the users feeling satisfied and confident when they use the design?
3.1.2 User Centered Design

Quite often when a system is designed the most important thing to consider, the end user, is forgotten \[13\]. Not uncommonly the focus is instead what features to add to the system, or business goals, but if the people that are actually going to use the system, are not considered the system does not add that much value \[13\]. Instead of deciding on the design of a system and then forcing the users to adapt to the behavior of the system, the User centered design (UCD) approach starts from the needs and beliefs of the users in order to ensure high quality UX. This is something that is also considered throughout the whole process of designing and developing the system. By taking this design approach the resulting system is more user-friendly \[13\][24].

There are three basic principles of UCD according to \[24\]:

1. Focus on the users early on in the process
   a. Collect data in a structured and systematic way
   b. Have direct contact between users and the designers

2. Measure and test the system
   a. It should be easy to use and learn the system
   b. Testing should be conducted on actual users

3. Iterative design process
   a. Designers should always be prepared to discard or remake designs
   b. The system should be tested continuously

3.1.3 User Experience Activities

This section lists some of the most frequently used UX activities and explains how they are typically used.

3.1.3.1 Sketches

Sketching is a rapid way of drawing that is performed without any intention of creating something to be considered the finished product \[25\]. Sketching is a tool used to separate the process of making and the actual design work. Sketches are mainly used in the initial stages of creating a design from an idea, when a lot of things are still uncertain and different alternatives and concepts are to be explored.
The deliverables from sketching are a rough foundation for the actual design work. Sketches should not require long periods of time to create, the creator should be ready to dispose of them, they should be minimally detailed and they should lead to discussion rather than present the final design decisions [25].

3.1.3.2 Wireframes

From the sketches that are decided to be further developed, wireframes can be created [25]. A wireframe is a document describing all the components a web page consists of and how these components act and look together [26][27]. Wireframes can be used as blueprints to be used when the page is to be implemented and the level of detail can vary depending on the project. The level of detail should be adapted to the needs of a given project. For some projects it might be sufficient with a sketch on a piece of paper and some post-its. This is usually the case when interaction designers and developers work close together in a small team. But for some other projects when there are more people involved and there might not be that close connection between the developers and the interaction designers, more detailed wireframes might be required. Wireframes are used to plan and make sure that the high level architecture and the required interface functionality of the system are included. If the system or product to be developed is of low complexity usually or only one wireframe is needed that can be used for all page, but often there is a need for multiple wireframes to clarify the functionality of the different pages and especially when the pages in the system are very different. Wireframes are not only a good aid for programmers and designer having a visual map for the functionality of the system, but also people at management level can use the wireframes to make sure the product to be developed is in line with their view and goals of the system [26].

Often having wireframes leads to discussion when people with different backgrounds and different skillsets work with the wireframes [26]. An interaction designer might focus on making the product easy to use, intuitive and visually good looking, while developers might be focusing on the functionality and that the solutions should be technically viable. Having to collaborate and figuring out the details together can help to reduce problems before the implementation of the product has even started [26].
3.1.3.3 Prototypes

Prototypes are used later on in the design process, when final design suggestions are starting to take form [25]. At this stage sketches (and wireframes if these are used) have been iterated and although the designs are not final, the result from creating sketches and wireframes can now be presented as a few design suggestions [25].

A prototype is a draft version of a design that enables ideas to be explored and the design concepts and intentions to be shown to the users and tested in usability testing sessions before any real development work have been started [28]. This way time and money can be saved by only developing designs approved by real users; It’s much more expensive to change to product when it’s developed than early on in the development process when the design is still only on the prototype stage [28].

Prototypes are represented in two types: High- and low-fidelity prototypes [28]. High-fidelity prototypes are similar to the final product in terms of appearance and function and are as close as possible to the real user interface. They are computer based and usually allow user interaction such as clicks and manipulation of data [28][29].

Low-fidelity prototypes are often made of paper, either hand-made or printouts, and do not allow user interaction. Low-fidelity prototypes are created more quickly than high-fidelity prototypes. One advantage of using low-fidelity prototypes is that the users might feel more comfortable to give feedback on these rough sketches rather than prototypes that look much like the final design [28][29].

3.1.3.4 Usability testing

The most frequently used method for user research (regarding usability) is usability testing and there are a number of different usability testing techniques [26]. Usability testing can for instance be users testing the finished system or product, or users testing ideas by being shown a sketch or interact with a prototype of a system. For large-scale projects prototypes are often use throughout the entire project timeframe [26].

In general usability testing involves giving a user scenarios to perform in the developed system, product or prototypes and observe how the system is understood and if the features in the system are easy to perform [29][30]. This is useful for finding problems with the design and how it can be improved. The participant is encouraged
to talk aloud about how he or she perceives the system and how he or she thinks when performing a given task. Usability testing can be performed using advanced methods such as video recordings and eye tracking but also using simpler methods that only requires a few users and a paper prototype\textsuperscript{[29]}.

It is important to properly plan a session before the test is conducted\textsuperscript{[29]}. Before the test is conducted the goals should be specified and what there is to learn. A few scenarios that involve tasks to be performed should also be decided on and how these tasks should be presented to the user. It should also be settled what level of prototype fidelity to use. When a plan is set up it should be decided how extensive the test should be. How much time and money can be put on testing the solution? If only quick answers are required, a few hours or a couple of days might only be required. If a more comprehensive test is to be conducted including multiple tasks this might require a month or more\textsuperscript{[29]}.

Designing the tasks for usability testing is very important. Good tasks should present what works well when using an interface and what issues there might be with the design\textsuperscript{[31]}. Good task design for usability testing includes the following aspects\textsuperscript{[31]}:

- The task comes from goals that is important to the user
- Includes questions that are important to the success of the product or system to be evaluated
- The tasks have a scope that is not too broad or too tint as well as a finite and manageable set of solutions
  - Task takes about 5-30 minutes including questions
- The user should be able to understand what the end point of the task is
- Tasks should not just make the user talk about what his or her opinion of the interface is, it should make them interact with it

It is important to start testing early and then keep doing this on a regular basis\textsuperscript{[32]}. According to Steve Krug it is much more effective to test one user early in the project than 50 user at the end of a project\textsuperscript{[32]}. Errors occur more frequently at the requirement and design stages and it is much more expensive to do something about them at a later stage than in the beginning. It is also important to iterate the testing process; things that were design and tested should be tested again. This is important because there might be problems that weren’t found during the first test session.
because of some other problems. Just as a developer is unfit to test his or her own code it a designer is also biased to his or her design. A designer knows how the application should work; but independent testers or visitors don’t have this information [32].

3.1.3.4.1 A/B testing

Using A/B testing different versions of a product, system or feature are presented to different users to get an understanding of which version that is the most suitable and effective one [33][34]. The idea of A/B testing is to present one version to some users and another version the other users during the same period of time and then compare the results. Usually the results are presented as for instance increased sales or increased signups. A/B testing was used in the 2007 presidential campaign for the U.S. elections. For this occasion A/B testing was used to optimize the website for Barrack Obama in order to get more email signups and increase the number of voters and donations [33].

It is important to define the goal of the test and what it actually is you want to test. To avoid doing mistakes in the process of A/B testing, five steps are proposed according to [30]:

- Define what to improve and success metrics
- Identify the biggest bottlenecks
- Make assumptions about user interaction and form a hypothesis
- Prioritize the testing variation candidates based on the hypothesis.
- Run the test.

3.2 Web Usability and User Experience Guidelines

Even though there might be no silver bullet for ensuring good UX there are guidelines based on extensive research to optimize the user experience of a web application [26]. Websites and applications can be complicated and sometimes hard to use. Websites lack instruction manuals, training seminars and personal guides to help the user through for instance a website, which other services might have. Therefore it is very important to design websites and applications so that they are
user friendly and easy to use, or else people lose their interest and stop using the site [26].

3.2.1 UX guidelines

In this section a number of common UX guidelines and best practices are listed and explained.

3.2.1.1 Remove possible question marks

If the architecture and navigation of the website are not intuitive the user might be unsure of how to use the system [26]. If the website isn’t easy to navigate and understand quickly by the user, the user will leave and search for other alternatives [35]. The structure of the site should be obvious, the user should be “guided” through the site by adding elements on the site showing the user how to navigate and links to other pages should be easy to find [26][35]. Figure 3.2 displays an example of a website that might cause questions. The design of the site in general is simple and intuitive, but the user will focus on the three large statements to the left because users usually explore websites in a “F”-pattern. It is a bit unclear what these statements mean. Instead the image and content to the right explains and presents the company’s products and this section block should be on the left instead for a better user experience [35].

![Example of questionable design](image_url)

Figure 3.1 Example of questionable design [35]
The website presented in figure 3.3 has a similar structure to the site in figure 3.2 but avoids the question marks by presenting hands on content explaining what the site is about. This website also has clearly visible links to where it is possible to try out and download their system. When reducing the number of question marks users have when visiting the website, the users will gain a better sense of orientation and increasing trust towards the company behind the site. The less thinking that is required by the user the better is the user experience [35].

3.2.1.2 Manage the focus of the users

It is important to think about what you want the user to see and steer the attention of the users to these elements and content [35]. The content on the page that is not as important should not be the first thing the user sees, if the user should see this at all. When a user visits a website they scan the page for interesting elements, they don’t read the text [26][35].

For instance images will attract more attention then text and characters in bold is more prominent for the user then plain text [35]. Figure 3.4 displays a good example of the website Humanized.com, which uses the principle of focus very good. The word “Free” is highlighted and instantly attracts the attention of the users while still being informative and not annoying for the user. The site doesn’t include distracting elements or irrelevant information and the button to the right stating “Get it now” is easy to find for the user and easily guides the user to the download. By using focus on different important visual elements it is possible to guide the user from point A to point B [26][35]. A study conducted by Jakob Nielsen shows that 77% of first time visitors on a website do not scroll, meaning that they will only see the
content visible on the area on the screen when the scrolling function is not used \cite{36}. This information shows the importance of arranging and highlighting the most important information and elements on the site \cite{26,37}.

Figure 3.3 Example of managing the focus of the users \cite{35}

3.2.1.3 Keep the design simple

It is very unusual that the users come to a site because the site has a nice design \cite{35}. Instead, the users come for the content; the page should therefore be designed keeping simplicity in mind rather than complexity \cite{26,35}. When designing a site the “keep it simple” principle should be the starting point. The Finch website presented in figure 3.5 follows the “keep it simple” design principle and by clearly presenting the users with site information as well as options without having content that is not required \cite{35}. According to a number of studies about user’s perception of website’s credibility there is a strong connection between the design of a website, how it is structured, if the design is consistent, colors and so on and the credibility of the website \cite{37}. Other important factors are the content quality, the error rate, how often the site is updated with new information \cite{37}.
3.2.1.4 Make use of white space
To make use of white space, having empty space between images, buttons, paragraphs and other elements, makes it easier for the user to get an understanding of the page as well as makes it easier for the user to perceive the information on the page [35]. When a new user visits the site the first step this user takes is to categorize and divide the content into sections of information that the user can understand, by using white space it is possible to do this for the user. It is usually better to make use of white space than for instance dividing sections with a line [35].

3.2.1.5 Use conventions
Using conventions – ideas, elements and design that has been tested and are used by other sites are a good way of reducing the time it takes for a user to learn how to use a given website [35]. Users usually expect certain navigation, structure and search functionality based on existing conventions. By finding out what these are and using them in new websites to be developed make users more confident and strengthen the reliability and credibility of your site. As a rule it is better to come up with new innovative ideas only if you know it is better than the convention, otherwise use the conventions [26][35]. It is also good practice to make the interface consistent and keep elements, design and navigation similar through different views. Keeping the interface consistent throughout the design will enable the users to “reuse” what they have already learned when previously using the system when this feature comes around again [26].
3.3 Measuring user experience

Measuring user experience can be a very useful for gaining knowledge about how the design decisions are perceived by the users of a product or system \[2\]. It reveals information about user-system interaction. Measuring the UX provides information about how effective the UX is, if the users are able to complete a given task in the system. The result of the measurement can present an estimate of how serious found errors are as well as how often they occur. It might also give information about the required effort to complete the task and user satisfaction. Measuring user experience can also provide the researcher with valuable information about what parts that needs to be improved and provide the researcher with suggestions for improvement. If the UX is continuously measured it also shows if the given design decision are improving the UX as a product or system are released in new versions \[2\].

There are number of different UX metrics that can be used to measure UX \[2\]. These metrics could be used to measure for instance task success, occurring problems or user satisfaction. When using UX metrics the same kind of measurement should be used when something is measured to make sure that the outcomes can be compared. It should also be possible to observe a given metrics, for instance if a task is completed or not. Metrics used for measuring UX must also be quantifiable, that is, it must be possible to put numbers on a given metrics. For instant a UX metric could show that 80% of the users managed to complete a given task in the system. Something as simple as changing how a data input field works could result in reducing the number of input errors by 85% \[2\].

When measuring UX the behavior and attitude of the users are measured \[2\]. These users can be very different and this can make the use of UX metrics more complicated. For this a confidence interval can be used to show how much the data can vary \[2\].

3.3.1 Different types of metrics

There are a number of different metrics that can be used depending on what the researcher wants to measure. Some common types of UX metrics are Performance metrics, Issue-based metrics and Self-reporting metrics \[2\].
3.3.1.1 Performance metrics

Performance metrics is used to measure user behavior or interaction with a system or product and are highly dependent on scenarios or tasks [2]. Using performance metrics you measure for instance if a user managed to complete a certain task or if a user managed to complete a certain task in a certain amount of time. Table 3.6 shows how these metrics can be presented [2].

<table>
<thead>
<tr>
<th>Task completion rate diagram</th>
</tr>
</thead>
<tbody>
<tr>
<td>Participant 1</td>
</tr>
<tr>
<td>Participant 2</td>
</tr>
<tr>
<td>Participant 3</td>
</tr>
<tr>
<td>Participant 4</td>
</tr>
<tr>
<td>Participant 5</td>
</tr>
<tr>
<td>Participant 6</td>
</tr>
<tr>
<td>Participant 7</td>
</tr>
<tr>
<td>Participant 8</td>
</tr>
<tr>
<td>Participant 9</td>
</tr>
<tr>
<td>Participant 10</td>
</tr>
<tr>
<td>Task completion rate</td>
</tr>
</tbody>
</table>

3.3.1.2 Issue-based metrics

Issue-based metrics notices occurred usability issues and might give information about how frequently they appear, which type of issue it is and also how severe they are [2]. For instance, issue-based metrics might present the number of unique usability for a number of iterations on a design. Figure 3.7 presents the number of usability issues and the severity of these for a number of tasks [2].
3.3.1.3 Self-reported Metrics

When getting information about what the users thoughts, perception and experience of a system are the metrics are called self-reported metrics [2]. Self-reported metrics may indicate how users feel about the system, which might be the most important thing to measure. One of the most important things to accomplish when developing a product is to make the users like the product and feel good about using it. Users are much more likely to remember how they felt and if they liked using a product, then how long the process of performing tasks or the if the number of clicks were more than necessary [2].

The information could be gathered by creating a form with rating scales, list of attributes the user can choose from or just ask questions directly to the users about what they liked most about the system [2].

Rating scales are one of the most commonly used ways of collecting self-reported data. To be able to analyze and use the result in an easy way it is common practice to assign a number to each answer. This allows an average to be calculated and compared across tasks, focus groups and similar [2].

Post-session ratings are useful to get an overall score for the usability of a product or system. This is the most typical use of self-reported metrics. This overall
score can then be used to compare how the usability of the system changes over time. It can also be used to compare overall scores for different design alternatives [2].

3.3.1.3.1 System usability scale

The System Usability Scale (SUS) is one of the most commonly used tools for getting an overall value for the UX of a system [2]. The SUS consists of 10 statements, 5 positively worded and 5 negatively worded, which the users are suppose to rate according to their level of agreement. Each statement has a five-point scale and the rating for each statement can be combined to provide an overall score between 0 and 100, where 100 is the highest possible score [2].

3.4 Software Engineering methodologies

Especially in software engineering projects, it can be hard for people outside of the engineering process to gain visibility, understand and control [6]. Traditionally the product or system to be developed is not visible until it is almost finished, making it difficult to know if the project is on the right track, respond to changes and to know when the project will be finished. Therefore methodologies and methods have been introduced in order to solve these problems [6].

3.4.1 The Waterfall model

The Waterfall model was first mention in an article in 1970. Using the Waterfall model the process is divided into a number of phases where each phase has to be completed before the next phase can be started. The 6 phase of this model are [38]:

1. Gathering system requirements
2. System analysis
3. System design
4. Implementation
5. Testing
6. Maintenance
The Waterfall model is preferably used in large, complex or mission critical project or in organizations including engineers with limited skills\textsuperscript{[38]}

### 3.4.2 Agile software development

Rather than the traditional way of developing software using the Waterfall model, where the software is developed in a number of sequential steps, the idea behind agile software development is that the development process is done iterative and incrementally\textsuperscript{[6]}. Each step of the Waterfall model (requirement gathering, system design, implementation and testing) is performed but in shorter periods of time (iterations) that are then repeated. After each iteration, a working product is demonstrated to the stakeholders. This enables quick adaption to changes in the requirements from the customer. There are traces of incremental software development methods from as early as 1957\textsuperscript{[6]}.

The \textit{Manifesto for Agile Software Development} was published in 2001 and defines what today is called “Agile Software Development”. The Manifesto includes twelve principles that are characteristic for agile software development. The principles include customer satisfaction, to welcome changing requirements, close connection between business and developers and simplicity. The teams using this methodology are usually self-organizing and cross-functional, that is each member should take responsibility themselves and it is believed each member knows how to solve their own problem in the most suitable way without having someone telling them how to do things. Cross-functional is a term meaning that agile teams often consist of members with different specialties\textsuperscript{[6][39]}. Some examples of agile methods include Scrum, Extreme Programming (XP) and Kanban\textsuperscript{[6]}.

#### 3.4.2.1 Scrum

The most popular agile software development methodology to use is Scrum\textsuperscript{[7]}. Scrum is shown to increase the productivity and quality of software development project as well as increase customer satisfaction and lower the costs of the project\textsuperscript{[7][8][9]}. Scrum is a lightweight framework that as most other agile methods is performed in an iterative and incremental fashion. The development process is divided into smaller periods of time called “sprints” where every sprint consists of all of the steps in the software development life cycle (SDLC). When using Scrum
the focus is to continuously improve the product, but also the working process using retrospective meetings. Scrum is performed in smaller groups of about seven people usually with a different set of knowledge; in a software development project there might be people with UX skills, a backend developer and a tester. Using Scrum a Scrum board is used to visualize the workflow \[40\].

A scrum group consists of a Scrum Master (SM), a Product Owner (PO) and team members with different sets of skills. The PO represents the customer, communicates the vision of the product to the rest of the Scrum team and controls in which order the requirements should be implemented and decides when the product fulfills the requested functionality \[40\].

The SM is commonly taken for the project leader, but the role is more of a coach than a leader. The SM is the Scrum expert, responsible for coaching the Scrum team in the Scrum methodology. The SM should teach the team about the practices and how to apply Scrum to maximize the performance of the team \[40\].

The Scrum team members should be able to collaborate and self-organizing. Each team member has his or her own responsibility to finish the task they have been assigned to and they should choose themselves how to best solve the task, there is no project leader telling them what to do or how to best solve it. Often each team member do tasks in his or her area of expertise, but sometimes they have to work with tasks that is not in their area of expertise. This is typically if there is a task that contributes more to completing the assignments of the current sprint then other tasks \[40\].

3.4.2.2 Kanban

Kanban was first introduced by Toyota as a way of making their manufacturing process more efficient \[41\]. Kanban is a type of Lean approach, having the goal to eliminate waste in order to deliver value to the customer in an optimal way. It has grown to be one of the most popular Lean approaches \[41\].

Kanban is used in software development projects to enable teams to work more effectively together and focus on early and continuous release of software. Kanban teams are self-managed and collaboration is a key factor. A Kanban board is often used in order give the team a clear view of the workflow for a specific project \[42\]. The board and this way of visualizing the development process is a central part of Kanban (the word Kanban can be translated to “Visual Card”) \[43\].
When using Kanban one of the basic principles is to balance the workflow, the amount of work in process should be kept to a minimum so that the team doesn’t take on too many tasks at the same time. Another principle is that the workflow should be enhanced. The enhanced workflow can be achieved by prioritizing tasks so that when one task is finished the most prioritized task is the one being picked next [42].

One difference between Kanban and other agile development methods such as Scrum is that the development process is not divided into sprints and that the tasks can be added at any time, not only at the beginning of a new development period. Another difference is that in Scrum a given task must fit into the interval of a sprint whereas in Kanban this is not required and a task requiring a longer time can be added at any point [43].

3.5 Agile software development and UX/UCD

Lately there has been a growing interest in researching how User Experience (UX) design can be included in Agile software projects. A number of papers have been published in the subject recently [5]. Studies have shown results that combining agile methods and UCD have improved the development process and the quality of the software product [11][14][15][16]. However, in practice it is not always an easy task [10][11][12].

3.5.1 Similarities between Agile software development and UCD

One important factor for even considering integrating agile software development and UCD is that they are similar in some sense, which makes it possible to combine the two without running into too many problems. Agile software development and UCD are similar in a few basic principles within the two concepts [11]:

Both use an iterative process for development, where information gathered from previous iteration is used in iterations later on [11].

Using for instance Scrum the software is developed gradually through a series of sprints where the software is continuously improved based on customer feedback. One of the basic principles of UCD is iterative design [11].

Another core principle in UCD is to involve the user from the start and continuously involve the user throughout the development process. This is also
practiced in agile methodologies. Using for instance Scrum the software is continuously review by the users in demo sessions. In a Scrum team also includes a product owner that is responsible for managing and prioritizing the requirements [11].

Team communication and coherence are important aspects in both practices. Both approaches have activities that the whole team should participate in [11].

3.5.2 Differences between Agile software development and UCD

Even though there are a number of principles and core concepts that agile software development and UCD share there are also quite a few differences that make the two concepts harder to combine [10][12].

The focus of the two concepts is one of the main differences. The goal for agile methods is to develop useful software from a set of features specified by the customer. UCD on the other hand aim to develop usable software, that is, the focus for UCD is to fulfill the goals and needs of the end users of the system [11].

Both UCD and agile methods involves the user in some way. While UCDs’ main focus is the end user and making the end user like a certain product, agile methodologies’ focus is fulfilling the needs of a customer. The customer is often a representative of the given company, but the customer might not be an actual user of the product, and therefore the end user might not be involved in testing the product when using the agile approach. The key for getting the system accepted is to understand the customer, but the key for designing the system for good usability is to understand the users [10].

UCD requires the creating of design sketches, prototypes and other documentation in order to communicate design within the team and to developers to implement the software. Agile methods however dictate minimum documentation and a focus on working software instead [6][39][11].

3.5.3 Successful solutions and combined methodologies

Some different ways of solving this problem has been proposed and tested. Common findings for a successful integration of agile methods and UCD are [11][15][16].
• Involving the user in the development process and continuously getting feedback from the user

• The development team should consist of both UX designers and developers and these should have close and continuous communication

• Adapting UCD to agile iterations. Designers must be able to deliver prototypes and designs to the developers in a way that works with the agile method cycle. It is important that developers don’t have to wait for the designers to deliver prototypes and vice versa.

• A common understanding and vision for the project. It is important that all team members are included in the bigger picture for the project. This can be done by sharing for instance documentation, sketches and prototypes between team roles

3.5.4 Agile Methods focusing on UX

There are agile methods that suggest development processes for how to combine agile methods and UCD.

3.5.4.1 Dual-track Scrum

Using Dual-track Scrum the basic concepts of Scrum is adopted but the focus is to discover the customer needs and making sure that these needs are understood using evidence and prototypes [44]. Dual-track Scrum addresses the problem that a lot of features developed when using Scrum are hardly or never used as a result of not validating the real need of the user and that insufficient attention is placed on user experience. The process of Dual-track Scrum is presented in figure 3.8. Using Dual-track Scrum there are two tracks, one “Discovery Track” focusing on finding out what product to build, and one “Delivery Track” focusing on finding out if the product is built in the right way [44][45]. The Product owner, lead developer and UX/UI persons are included in the Discovery track while developers and testers focuses on producing the product or system. In this way the design and implementation can be iterated separately in parallel. Dual-track Scrum deals with the problem that the product should be designed and prototypes should be developed and tested before the coding begins. The prototypes are usability tested at
least one sprint in advanced that the implementation is to be performed and when the implementation is done the code is usability tested\[44\].

Even though there are two parallel tracks the developers and the interaction designers are in close contact and should work close together, especially when the interaction designers presents the design to be implemented to the developers and when the developers presents the developed system that the interaction designers are going to perform usability testing on \[44\][45].

![Figure 3.6 The Dual-track Scrum development process](image)

### 3.5.4.2 The Lean startup

The Lean Startup was created for making startups get their products faster to the market, to teach startups when the product need to be changed and when it is on the right track \[46\]. The Lean startup also provides a way of growing a business at the fastest pace possible \[46\].

There are five principles of Lean Startup, and these are \[46\][47].

1. Entrepreneurs are everywhere. The definition of a startup is the creation of new services and products when the conditions are extremely
uncertain. A startup doesn’t have to be a few people that just started a company and works in a garage; it can be applied to any company of any size. Even large companies that already have products might take the initiative to create completely new services and products.

2. Entrepreneurship is management. A startup is an institution, not just a product, and therefore the institution requires a new type of management strategy adapted for the conditions of extreme uncertainty.

3. Validated learning. Startups exist not only to deliver new products and services to customers, they also exist to learn how a sustainable business can be created. This learning is achieved by frequently running experiments to test the visions of the entrepreneurs.

4. Build-Measure-Learn. The main process of the Lean Startup is to from ideas build a product, measure how the customers respond to the product and from the response learn whether to pivot, change the product, or persevere, keep the product as it is. It is very important for the institution to try to accelerate the pace of moving through this loop. The Build-Measure-Learn loop can be seen in figure 3.9.

5. Innovation accounting. In order to make entrepreneurial outcomes better progress need to be measured, milestones setup, work need to be prioritized and entrepreneurs need to be held accountable. This requires a new startup specific kind of accounting.
A central part of Lean Startup is to create a minimum viable product (MVP) and the use of continuous deployment. A MVP is a version of a new product with the least amount of features possible that brings as much validated learning as possible from testing it on customers. The Lean Startup process using MVP can be seen in figure 3.10. From the testing feedback new assumptions can be declared and a new MVP is developed and so on until it meets the needs of the users and can be released as a final product. Using Lean Startup there is frequent contact with the end users to make sure the right assumptions about the final products are made and by taking this approach waste is reduced and the product can be shipped to the market more quickly.
3.5.4.2 Lean UX

The foundation of Lean UX is based on theories from Lean and Agile development as well as traditional UX activities. Lean UX includes concepts from Lean Startup such as the learning loop (Build-Measure-Learn) as well as the MVP. The purpose of using Lean UX is to eliminate waste; both waste in the process but also wasteful implemented features. Everything that does not provide value for the end user should be removed. The purpose is also to have fast time to market and practice design thinking, that is, methods to design in order to satisfy people’s needs and increase value. Using Lean UX there should be cross-functional teams with different competences and an important part of Lean UX is team collaboration and communication between team members. Another important part of Lean UX is to keep the deliverables in low fidelity and editable. The Lean UX process for ensuring UX is presented in figure 3.11.
3.6 Brief summary

This chapter begins by presenting information about User Experience, Usability and User Experience activities. It then continues by presenting information about UX guidelines and how to measure UX. Agile software development is also covered and the chapter end with presenting how agile and UX can be combined as well as related research.
Chapter 4 System Requirement Analysis

Before the researcher started this research the internship company had conducted a prestudy and set up goals and high-level requirements for the system to be developed. These goals and requirements are presented in this chapter.

4.1 The goal of the system

In order to increase the availability of knowledge sharing events within the company, the internship company has requested a web-based system for video streaming. The system is supposed to enable any employee to upload, stream, share and comment on posted internal video files. This thesis will be focusing on the frontend part of the system and more specifically how the frontend can be implemented to ensure good UX. Additional requirements will be gathered mainly from the Product Owner during demo sessions that will be conducted at the end of each sprint and from continuous user tests. The user tests will include users from the whole organization because all employees are potential users. A broad variety of users during the usability testing will give the best picture of how the system best should be implemented and might also give feedback on different parts of the system.

4.1.1 Functional requirements

This section presents the main functional requirements for the on demand streaming system. The related detailed requirements for each high-level functional requirement are also presented in this section. These detailed requirements are presented in detail for each sprint period in chapter 5.2.

Figure 4.1 presents the use case diagram for the system. These main high-level functional requirements are:

(1) The user shall be able to upload video files as well as related video content such as Powerpoint-presentations, to the system. The related detailed requirements are UM.1, UM.1.1, UM.1.2, UM.1.3.

(2) The user shall be able to in an easy way share a video from the system to other users either by E-mail or to the company social media platform. The
related detailed requirements for this high-level requirement are WM.1, WM.1.1 and WM.1.2.

(3) The user shall be able to comment on videos he or she has watched, and the comments should be visible to other users. The related detailed requirements are CM.1, CM.1.1 and CM.1.2.

(4) The user shall be able to stream videos from the system as well as use functions such as play, pause and seek in the video he or she is currently watching. The related requirements are WM.1, WM.1.1, WM.1.2, WM.1.3.

(5) Related video content shall be presented to the user of the system and the user should be able to download this content to his or her personal computer. The related detailed requirements are WM.3, WM.3.1, WM.3.2.

(6) The user shall be able to edit a video he or she has uploaded. It should be possible to edit the metadata such as video title, description and the speakers in the video as well as edit the video if it needs to be shortened or parts needs to be taken out from the given video. The related detailed requirements are UM.2 and UM.2.1.

Figure 4.1 Main use case diagram of the on demand streaming system
4.1.2 Non-functional requirements

The main non-functional requirements are:

(1) The system shall be available for access from any geographical location provided a given user has Internet access. One important goal with developing the system is that it shall improve knowledge sharing between offices in different countries. It is therefore important that users in all of the countries that the internship company has offices in can access the system.

(2) Only authorized users shall be able to gain access to the system. Users shall only be able to gain access through the Netlight AD. The system will contain material that involves company secrets as well as internal presentations of new frameworks and techniques that shall not be available to competitors.

(3) The User Interface shall be intuitive and easy to use. The system should be user friendly and it shall not be hard to understand how to use the system of where to find the information a given user is requesting.

(4) Personal information stored in the database shall be encrypted. There might be sensitive information in the database such as passwords and social security numbers that if an attack should occur shall not be available in plain text.

(5) The system shall be possible to use both from a laptop and from mobile devices. Many of the employees will be using mobile devices to access the system and therefore this needs to be supported.

(6) The system shall be well documented and the code shall be commented. The language of the documentation and the comments will be written in English. The system is intended to be developed further by the developers at the internship company after the Master Thesis project. Therefore, it is important that it is easy to understand how the architecture of the system is structured. This is also important for future maintenance.

(7) The system shall be able to support a minimum of 50 users streaming video at the same time. Because of the size of the company and the fact that they are expanding quickly it is likely that there will be a number of users requesting video material at the same time and it is therefore important that the system can handle concurrent users.

(8) The client side part of the system shall follow the Model View Controller (MVC) design architecture pattern. By using the MVC pattern the system will have a good and reliable structure and will enable future extension and reuse.
4.2 Brief summary

This chapter starts by introducing the goal of the system. Then the functional, non-functional as well as the main use case diagram for the system are presented.
Chapter 5 Case Study

This chapter presents the results from interviews and active observations conducted at the internship company.

5.1 Interviews

The interviews were conducted at the internship company where questions regarding frontend development and how to combine agile processes and UX were asked to the respondents. The respondents were employees at the internship company with experience in the given subject from several projects.

5.1.1 Agile and UX

The respondents all thought that successful, modern software companies should develop their products using an agile methodology [53][54][55][56][57]. To introduce agile methodology in an organization is not always easy though, and sometimes companies think they are working agile when they really are not. It can be hard to adapt to fast-paced agile methodology and sometimes an organization might not be experienced enough to accept this way of working. One problem that arises might be that management wants to plan what should be done in detail a couple of months in advanced, and this is in opposition to the agile way of working. Another problem that might occur is that when using agile methodology, you often have to accept that you don’t know what needs to be done and then work with finding this out, but this might be hard to sell to management. It gets even harder when the needs of the users has to be taken in account and UX activities are to be added to the existing agile process [53][54][55]. Some respondents thought UX is often included into agile processes that are not adapted for this [56].

Usually UX activities are conducted or at least supervised by people with UX as their speciality and not something that “regular” developers do [56]. Traditionally people working with UX don’t belong to a specific project, they might be working in several project but never as a part of a team in those project. UX persons are often brought in at the beginning of a project to help understand the users, set up system flows and designing the GUI, but in recent trends the people working with UX are almost always included as a part of an agile team where they are working
close to the developers and continuously improving the design, system flow and conducting usability tests in close relation to the users of the system to be [56][57].

5.1.1.1 The UX person role

All respondents thought that it is preferable have people experienced in UX as a part of the development team rather than a resource that is brought in on demand [54][55][56][57]. According to the respondents this is mainly because it is almost impossible to think about all the details in advance, and because it is hard to be completely finished with a design, the design can always be improved [54][56]. One respondent even said that:

“It is useless to think that design and UX activities is something that can be done externally from a development project” [54].

Some companies might not see the value of having an UX designer as a part of a development team, and the respondents expressed that they thought companies mainly excluded UX designers because of economical and budget reasons. The respondents thought that if there are more then two developers in a team a UX designer should be added as well as a rule. Adding a UX designer always pays off and there are always things to do for this person – Often UX designer can work with frontend development as well as UX tasks, or perform usability tests on previously implemented features and research new features [54][56].

5.1.1.2 Team communication

A close relationship between persons working with UX and developers is preferred during the whole project [54][56][57]. It is important to find a well functioning way of communicate between the developers and the UX persons and to make sure that the team is always moving forward. The developers are dependent on the work of the UX persons and vice versa. Therefore it is preferred to find a way of working in parallel so that the developers don’t have to wait for a prototype for instance and vice versa [54][56][58]. Sometimes the activities might come off sync, for instance it could be required to do usability tests on a prototype before choosing what solution to select and then implement. This could result in that the developers become blocked in their work because they have to wait for the tests to be
performed before they can start the implementation \cite{56}. To solve this problem, it might be possible for the developers to design and setup the database while the UX persons start to develop the sketches and flow of the system or that the developers fix bugs in the system while waiting for the new design. Another way to deal with this problem is to keep focus on creating low-fidelity sketches and prototypes; these should be developed as fast as possible. Sometimes it might not even be necessary for the UX persons to develop prototypes. Maybe it is enough to do a quick sketch on paper or just explain the new feature to the developers. The prototype can then be refined in parallel. Maybe the developers don’t even need a prototype to develop a certain feature. The relationship should be dynamic and it should be possible to trust each other, the process don’t always have to be to first develop a prototype and then the implementation work can start \cite{54,55,56}.

It is important that all team members participate in the UX work \cite{54,58}. It is not only UX designers who own the system, it is the whole team and all team members should feel responsibility and come with their own ideas and suggestions. Often half an hour or one hour is enough to have a meeting and bring up thoughts. It is important that UX persons have some kind of technical knowledge to know if a given solution is viable, but when starting on a new feature input from developers is important to know if the given design is possible to do \cite{54,58}.

5.1.1.3 The role of the users

In all interviews the importance of involving the user in the development process were stressed. The user should be included when deciding upon the requirements, the design, the system flow as well as continuously testing the system when new features and versions of the system are developed \cite{53,54,56,57}.

It is important to have a physical contact with the end users and be able to present and get feedback on design ideas \cite{58}. Sometimes it can be hard to get access to users for testing though. Often the contact is with the customer, but the customer might not be the user of the system. If users to test on are not available one option might be to test on someone similar to the focus group, this could be a friend a relative. The important thing is to test the system on someone that is in a similar situation as the users and that is not familiar with the given system. To what extent the tests are conducted are highly dependent on the amount of time and resources available. Test as much as possible, but quite often time and resources for this is
very limited in practice. In school you are taught to do surveys, user tests in several iterations but when starting to work in real projects you have to make due with what is available. Sometimes the only available option might to be ask someone who is sitting beside you at the office if they for instance would understand a feature if it was represented by this certain button [58].

5.1.1.4 Development methodologies

All respondents stated that they did not follow a certain agile methodology to the letter [53][54][55][56][57]. However, all respondents had some kind of agile methodology as their base for their development process. As one of the respondents stated:

“Nobody does Scrum as it is supposed to be done” [56].

Two respondents used a mix or Scrum and Kanban when conducting their development work, which UX activities were added to, [56][57] two other respondents used a version of Dual-track Scrum [54][57]. Lean UX was used by one of the respondents [55] and one of the respondents used the method called “Lean Startup” [53].

5.1.1.4.1 Thoughts on other development methods

When asked about other Agile processes some respondents were skeptical to the Dual-track Scrum approach [55][56]. Two respondents stated that this would not work in their teams because they thought it would divide the members of the team. They stated that wanted to have a close relationship between developers and interaction designers. Another problem they expressed was the difficulty of when and how to communicate. According to these respondents it is hard for interaction designers to prepare a prototype for the next sprint when things change all the time and a lot of work is done in parallel to the development work. They also expressed that this way of working might cause the common understanding of why and what to develop to be lost [56].

One respondent currently not working with Lean Startup expressed many positive feelings about this way of working [55]. The respondent’s main points to
why this method was good was that it shortens the time to production, which the respondent thought was important. Another preferable thing with Lean Startup he said was that only the basic things are developed for a feature and then the user is consulted to get feedback on this solution to make sure that the right thing is developed. This respondent saw some problems with this method though. When using this method it is important to keep it at the MVP level, but often more features, layout and design are added than are necessary or required to test the specific feature.  

5.1.2 UX activities

This section presents UX activities mentioned by the respondents in the interviews and how they work with these activities in their development process.

5.1.2.1 Sketches and prototypes

Sketches were something that all respondents used in their everyday development work. Sketches are good because they can be created quickly and also altered or modified in a fast sense. Sketches are good both when shown to users and customers to get a common understanding of what to develop but also during the everyday sprints to present ideas to team members. Sketches can be drawn and altered when sitting together with the users; this allows the user to participate in the early design of the system to be. In this way both the users and the project team can agree on a common design. By doing this an image of how the system will look is already placed in the users’ mind and it reduces the rework that might have to be done if the users are not pleased with the final design of the system even though the final system might be great. It is important to really understand what the users want.

The respondents stressed that it is important that the sketches are not to “good looking” because this does not give as much feedback as if the sketches are more rough. The closer the team members work the lower fidelity the sketches are required to have. When working closely it’s possible to sit together as designer and developer and just point and discuss how things should look. Using this approach, it is usually not required to iterate the prototypes either. If it is a
remote team or if developers and designers are not situated at the same place a higher demand is put on the degree of detail of the sketches. It is incredibly expensive to update sketches and prototypes in the same way as code is updated. The design should only be done as simple as everybody still understands, and as soon as possible designers and developers should communication directly instead of keep updating sketches and prototypes. Many UX designers put a lot of time in creating good looking prototypes with animation and transitions but the time put in this doesn’t provide enough result. If a high-fidelity Photoshop image is shown, most people will not be that inclined to give feedback or change the design. This might also cause the customer to think that the system is done and start to wonder why the system hasn’t been delivered yet. The customer might not understand that nothing is built; it is just a frame without any really content, data or functionality.

Handmade sketches are good in the beginning, but the longer in the development process the project progresses more should be shown to the users and the customer. The users should be involved from the beginning of the project and not be involved to late requesting changes when parts of the system are already done. In the beginning sketches are great, but later on sketches should process to mock-ups or wireframes first without color and then with color to show how the system is being created. This allows the team to show the whole process to the users and how the project is progressing over time.

Another good thing about sketches is that you don’t need to be a skilled drawer or have a number of hours experience in some program to create these, they are just a way to show the users how your system design is like. Sketches are also good as “agreement documentation”. If the sketches the users agreed on at the beginning of the project are saved, they can be fetch and used to show to the users if they claim later on that the resulting system is not in line with what you agreed on early on. This might cause the customer to say; “Yes, now I remember that this was what we agreed on”. It is also good to store these sketches to continuously compare that the developed system looks like the sketches and thereby is in line with the customers’ requirements.

It is important to think about the system flow, and that features should be included when designing the system. Should it be possible to search the page? Is there a need for an extra menu? Should dropdowns be used? Text fields? Colors,
shapes and sizes are easily changed but the basic entities (text field, dropdown, menus) are important to decide on in the beginning, nobody likes starting again from the beginning \[58\]. Another useful way of getting an initial design is to look at how competitors have solved the given problem; this might give inspiration for how the problem can be solved \[54][58\].

Some of the respondents had a team session called “Design studio” where all team members participate to set up goals and in smaller groups develop sketches for the new system or feature \[55][56\]. The sketches are presented within the team and this process is iterated until the team has combined the ideas and finally agreed on a final best design. This design is then tested on users to be able to get feedback and take design decisions from this. How much time to put on the Design studio sessions is dependent on the size of the new system or feature. The session could be a couple of hours to a whole day or two \[54][56\].

Prototypes were used by the respondents to test the design on users before the implementation work had started. This was an efficient way of making sure that the right thing is developed before even starting the development work and making sure not to develop features that the users don’t want \[53][58\]. Prototypes can be used together with A/B-testing when different prototypes are shown to different users to get an understanding of what design version is the most suitable \[54][58\].

5.1.2.2 Wireframes

The respondent said that wireframes was developed by interaction designers in the teams and then shown to developers and management \[53][54][56][57\]. Wireframes are a good way of presenting design ideas, getting people to understand the design and structure of the system. Having wireframes visible and showing them to stakeholders often lead to discussion. For instance showing wireframes to developers might cause discussions about if a given design is technically doable or not. When showing the wireframes to management they can check if they think it fulfills the needs of the customer. Producing wireframes are therefore a good activity to get input on developed design and making sure that everybody understands the design decisions and developers and management are able to present difficulties, if there are any, with the given design \[53][54][56][57\]. Wireframes should be kept updated until they are not needed anymore, that is, when
stakeholders and developers already understands the design and have a clear image of the system to be developed [53][54].

5.1.2.3 Usability testing

All respondents used usability testing in their projects to test sketches and prototypes as well as the implemented features according to the needs of the users [53][54][55][56][58]. Usability testing was used continuously throughout the development process. Most respondents tested both the sketches and prototypes before the implementation work started and also after the feature had been implemented on users [53][54][56][57]. One respondent put almost all focus on prototype testing before the implementation started and only tried the implemented features herself. Feedback from the users was received after the feature was publically available rather then tested in traditional usability testing sessions [54].

In the interviews many respondents expressed that the best way to conduct usability testing was to be physically present together with the users and observe them as they interact with the system. A good way of doing this is to have a test leader and an observer from the development team [55][56][57]. This way the tester can notice how the user perceives the system and see if the user has any problems fulfilling the specified tasks. Usability testing should be done continuously through the development process and not just at the end of a project. This way problems can be noticed earlier, which minimizes the risk of having to delay or redo a given design or feature [56][57]. During the testing sessions the tests were conducted by the respondents in different ways. Some respondents said they gave the user predefined scenarios and tasks to perform in the system, sometimes they just showed different design alternatives to get the best alternative from user feedback and sometimes the user could try the system freely [53][54][56][58]. Some respondents said that it is good to test the system on different users each time usability testing is performed to get the maximum result out of the testing session because the user are not biased and haven’t seen the system before that might cause them to have formed a picture of how the system should look [56][57].
5.1.2.3.1 A/B testing

Three respondents used A/B testing in their development process [53][54][55]. This was used when requirements from the users were unclear or when the development team wasn’t sure about what design to pick from a set of design alternatives. It is important to not put to much detail in the prototypes used in A/B testing because only one version will be the final design and the other versions will not be designed and implemented at all [53][54]. Using statistics from for instance Google Analytics the respondents then could see what version best fulfilled their KPI (key performance indicator) [54][55].

5.1.2.4 Measuring the UX

The respondents had a number of ways of measuring the quality of the UX within their system. All respondents conducted usability testing that can be seen as a sort of measurement [53][54][55][56][58].

Many of the respondents used analytics tools for measuring how the users were using the service – what they click on, how much video they are playing and so on [54][56]. One respondent said that the company she worked at had their own department that was working with analyzing what the users think of their products. This department conducts analysis via phone for instance [56].

One respondent stated that for quantitative data he would email to the whole company let them try the system or a given feature (for internal tests) and then fill in a Google form to give feedback [53].

All respondents were able to measure the UX via usability test sessions, but the metrics were not written down in excel sheets and made graphs on. Rather they would get a feeling about what the user thought of the tested features and kept this in mind. However, they would write down what the issues were in order to fix these later on [53][54][55][56][57].

5.2 Observations

In this part of the chapter the active observations from the development work conducted at the company are presented for every sprint. The Scrum development methodology was used for the development process. For the first two sprints “vanilla” Scrum was used and didn’t include any UX activities or combined agile
UX methodology. The remaining sprints Dual-track Scrum as well as UCD was used as development methodology and UX activities were added to see how these activities affected the development process. The activities added were taken from the proposed activities from the literature study as well as interviews with employees working with UX and agile methods at the internship company. For every sprint requirements, design, implementation, testing and reflections from retrospectives are presented as well as observations.

The development team consisted of one developer (the researcher of this report) focusing on frontend development and UX, one developer focusing on backend development and a product owner responsible for defining and prioritizing the team backlog.

The product owner had set up a few guidelines that we followed throughout the project:

- The system was developed to be used, and to be used as soon as possible
- The system was to be released early and new versions should be continuously released
- Focus was on creating value rather than a nice design
- New features were implemented according to MVP, only the minimum possible features and design were added to each version, in order to release more often and test if the design was feasible

Figure 5.1 shows the user story backlog (at the top) and the product/sprint backlog (at the bottom) for the project as it were at the beginning of sprint 1. Each row in the user story backlog represents the minimum requirements for a release and each column represent stories for one of major features within the system. The top row represents the first and most critical release; the criticality decreases for each row down. The tasks were dealt in the sequence from left to right for each row. During the project at least one new version of the system was released for every sprint on the internship company intranet; sometimes there were several releases per sprint.
5.2.1 Sprint 1

The initial sprint included a lot of tasks to get the development work up and running. It also included setting up the high level architecture of the system and deciding upon the structure of the API to allow communication between the frontend and backend part of the system.

5.2.1.1 Requirements

The tasks set up for the first sprint was to develop some of the most basic and important features for the system. These requirements are described in detail in table 5.1.

<table>
<thead>
<tr>
<th>Requirement ID</th>
<th>Description</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>WM.1</td>
<td>The user shall be able to watch a video file</td>
<td>High</td>
</tr>
</tbody>
</table>
The system shall use the API to request and set the video source

The user shall be able to press play to start watching a video file

The user shall be able to pause a video while watching it

The user shall be able to seek in a video the user is currently watching

The user shall be able to get a list of all available media files on one page

The system shall be able to fetch all video objects via the API

The user shall be able to select a video from the list of videos and get routed to the specific page for this video

The system be able to set and get current selected video object

The user shall be able to share a video

The user shall be able to bookmark a video page and start watching this video at a later time

<table>
<thead>
<tr>
<th>Module</th>
<th>Requirement</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>WM.1.1</td>
<td>The system shall use the API to request and set the video source</td>
<td>High</td>
</tr>
<tr>
<td>WM.1.2</td>
<td>The user shall be able to press play to start watching a video file</td>
<td>High</td>
</tr>
<tr>
<td>WM.1.3</td>
<td>The user shall be able to pause a video while watching it</td>
<td>High</td>
</tr>
<tr>
<td>WM.1.5</td>
<td>The user shall be able to seek in a video the user is currently watching</td>
<td>Medium</td>
</tr>
<tr>
<td>MM.1.1</td>
<td>The system shall be able to fetch all video objects via the API</td>
<td>High</td>
</tr>
<tr>
<td>MM.2</td>
<td>The user shall be able to select a video from the list of videos and get routed to the specific page for this video</td>
<td>High</td>
</tr>
<tr>
<td>MM.2.1</td>
<td>The system be able to set and get current selected video object</td>
<td>High</td>
</tr>
<tr>
<td>MM.3</td>
<td>The user shall be able to share a video</td>
<td>High</td>
</tr>
<tr>
<td>WM.2.1</td>
<td>The user shall be able to bookmark a video page and start watching this video at a later time</td>
<td>High</td>
</tr>
</tbody>
</table>

**5.2.1.2 Design**

A big part of this sprint was about collaborating with the other developer in the team to set up the architecture for the whole system and decide upon what information that would be available in the system via requests to the API. The frontend was divided into two modules – Main Module (MM) and Watch Module (WM). The MM included the landing page, where all available videos were to be presented to the user. When the user selects a video from the list in the MM the user would be transferred to a “watch view” that was represented by the WM. This module would be where the users watched a specific video as well as see information about this video. The requirements for each module can be shown in table 5.1.
In the first sprint the whole team had a session where it was discussed how the high-level architecture for the entire system should be structured. The resulting high-level architecture for the system is presented in figure 5.1. The system follows the client-server model for distributed applications structure. The JavaScript framework AngularJS was selected to be used to develop the frontend.

AngularJS was selected because it is a comprehensive frontend framework that provides a structured way of developing frontend application using the Model View Controller (MVC) architecture pattern \([59][60]\). It includes services, directives, routing and provides testability. Using AngularJS the application has a number of HTML views each with its own controller where functions related to the content on this view will be implemented \([60][61]\). The frontend also had services that were used to share information between the different controllers and to keep track of user specific data when the view changed \([61]\).

A RESTful API would be used to structure the communication between the client and the server; it specified a number of requests that would be made by the users and what the responses to these requests would be. The server would handle these requests to either supply the requesting user with information from the database, videos from the file storage or by inserting information to the database.

![Figure 5.2 High-level system architecture](image)
5.2.1.3 Implementation

For this sprint the basic structure of the system was implemented according to the architecture that was designed. A scaffolding tool called Yeoman was used to generate the project. More specifically the Yeoman generator “AngularJS Full-Stack generator” was used.

The generator created the basic structure for a modern web application using AngularJS as the frontend framework and NodeJS/ExpressJS on the backend [62]. The generator also installed Bower, which was used for frontend package and dependency manager [63], as well as Grunt that was used to run tasks like minification, compilation, unit testing and deployment [64]. Bootstrap was also included by the generator, which set up a basic layout and design for the system and includes design templates for typography, grid system, forms, buttons, navigation, icons and Javascript extensions [65]. Using Yeoman it was possible to for instance create a new route in the terminal without having to create any files or write any code manually. For instance it was possible to write: “yo angular-fullstack:route main” to create the MM. This command would generate a new route called “main” that would include a controller, a controller unit test file, a HTML-view file, a CSS-file and register the route at the specified URL. Using Grunt it was possible to build the project using the “grunt” command in the terminal. Commands were also available for previewing the project by writing “grunt serve” and deploying it using “grunt serve:dist”. This gave the project a “kick start” and enabled the development work to be much more effective.

During the first sprint views and controllers were added for the “Main view” (the index page) showing a list of available videos and the “Watch View”, the view for playing video files.

Using AngularJS it is easy to perform get requests to the API. Figure 5.3 shows a get request made in the MM controller to get all video metadata and save these videos in a “scope-variable”, making this object available both in the MM controller but also in the MM view.
Using Angular it was easy to loop through the array with video objects and display them in the view. Figure 5.4 shows how to use the AngularJS directive “ng-repeat” to loop through all videos in the “$scope.videos” object that was requested as shown in Figure 5.3 and display each video's title and description. How this is presented to the user can be seen in figure 5.5. Figure 5.4 also shows one example for when Bootstrap can be useful. Setting the span class to “glyphicon glyphicon glyphicon-play-circle” will display a “play symbol” as can be seen in figure 5.5.
When the user selects a video to watch the system has to keep track of which video was selected and this information must also be available in the WM. To solve this problem a service was used. This way different controllers can share common variables. Figure 5.6 shown how this is implemented. When the user selects a video to watch from the MM view the “goToVideo”-function is called that calls the video service’s “setVideoObject”-function to make the video object available for the WM. The next line uses Angular location to transport the user to the WM.
In the controller for the WM the selected video object can be received by calling the “getVideoObject”-function of the video service as shown in Figure 5.7.

Figure 5.7 Getting the video object from the video service

Figure 5.8 shown how the HTML5 video tag was used to display videos. When the selected video object was received from the service HTML5 video could be used to display the video by setting the video source as the media_url name of the video object. HTML5 video includes a lot of the features that was required to implement for this sprint such as video playback, seeking and pause for video \(^{[66]}\). Figure 5.9 displays the result when playing a video in the system.

Figure 5.8 Using HTML5 video tag to display videos
The easiest possible solution was implemented for bookmarking and sharing a link was to use the Angular location service to add the video id parameter to the url on this form: “/watch?video=3”. This enabled the user to copy the url and then share it or bookmark it using the browser.

5.2.1.4 Testing

The implemented features were only tested manually during the first sprint. A demo was performed for the PO and the thesis project supervisors at the internship company. They were pleased with the current result and concluded that a first version of the system had been produced - a first version that could be continuously
improved over time. The product owner and the supervisors proposed that the next step for the development work would be to focus on:

- Make it possible to download PowerPoint- and Keynote presentations for each video
- The information shown on the frontpage should be:
  - Title
  - Speaker
  - Duration
- The information presented for each video on the watch view should be:
  - Title
  - Description
- Before each video is played the user shall be notified that the videos are for internal use only and not to be spread outside the company
- It should be possible to leave feedback on the system

5.2.1.5 Retrospective

At the first retrospective meeting all team members thought that the development process had started to take shape and that it was good that all team members were involved in designing and structured the basic architecture for the whole system. This gave all team members a clear view of the system to be and enabled all team members to present input. The team members felt that they had been developing a first working version of the system fairly quick. The fact that testing procedures both for usability testing and unit testing was lacking was something that was brought to attention and it was unclear of the users would like the first version of the system.

5.2.2 Sprint 2

In the second sprint much of the features created in the first sprint were improved and extended. A feature for downloading PowerPoint presentations for each video was implemented as well as a feature for the users to leave feedback on the system in order to receive thoughts and suggestions for improvement. AngularJS unit tests were researched.
5.2.2.1 Requirements

What requirements to focus on in this sprint were based on what the PO and the internship supervisors stated in the demo for the previous sprint in section 5.2.1.4. Some new requirements such as the feedback feature and the notice to show before each video stating it is for internal use only were also added after the demo session. Table 5.2 shows these tasks in detail.

Table 5.2 Sprint 2 detailed requirements

<table>
<thead>
<tr>
<th>Requirement ID</th>
<th>Description</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>MM.1.2</td>
<td>Thumbnails shall be presented to the user for each available video</td>
<td>Low</td>
</tr>
<tr>
<td>MM.1.3</td>
<td>The user shall be able to see title, speakers, a thumbnail and duration for each video on the frontpage</td>
<td>High</td>
</tr>
<tr>
<td>WM.2</td>
<td>The user shall be able to get information about video title and description in the watch view</td>
<td>High</td>
</tr>
<tr>
<td>WM.3</td>
<td>The user shall be able to manage related media content such as Powerpoint presentations</td>
<td>High</td>
</tr>
<tr>
<td>WM.3.1</td>
<td>The user shall be able to see related video content if this exists for a video</td>
<td>High</td>
</tr>
<tr>
<td>WM.3.2</td>
<td>The user shall be able download related video content if this exists for a video</td>
<td>High</td>
</tr>
<tr>
<td>MM.4</td>
<td>The user shall be able to leave feedback on the system.</td>
<td>High</td>
</tr>
<tr>
<td>MM.4.1</td>
<td>The system shall use the API to save the feedback in the database and sent it by email to the development team.</td>
<td>High</td>
</tr>
<tr>
<td>WM.1.4</td>
<td>Before each video is played a short notice video should play stating the videos should not be shared publicly</td>
<td>High</td>
</tr>
</tbody>
</table>
5.2.2.2 Design

During the second sprint an improved design for the frontpage of the system was designed where each video listed were presented using a thumbnail image from the video file, a bold title and people speaking in the video were listed. This look was inspired by the look of similar streaming services such as Youtube and TedTalks to make it easier to use because the users would recognize the design and structure of the view. This also follows the “use conventions”, “Manage the focus of the users” guidelines as mentioned in section 3.3.6. Title and description was added in the watch view along with a link to the Powerpoint file if there was one for the given video. Presentation files (such as PowerPoint presentations) was frequently requested by the employees at the internship company and therefore it was important that this content should be easy to find for the users. The “focus guideline” mention in section 3.3.6.3 was followed to make sure that the users would find this content. The Watch view was divided into sections where the video player and presentation files were listed next to each other as this was believed to make it easier for the customer to find the presentation file while still being able to watch a video. The result can be seen in figure 5.14. For the feedback feature other systems usually uses a popup window to fill in this information and therefore this was to be added where the user should be able to add information in a form. The popup would be opened from a button in the footer.

5.2.2.3 Implementation

Figure 5.10 shows how thumbnail and duration was displayed dynamically for each video in the MM. The thumbnail image for each video was displayed by performing a request to the API using the “media_thumbnail”-variable in the videoMetaData object. It also shows another example of how Bootstrap can be used. Setting each media object div class to “col-xs-3” will result in a 4-column grid. Figure 5.11 presents the resulting layout of the page that was displayed to the user.
Figure 5.10 Using Angular and the API to get thumbnail and duration for video

Figure 5.11 Showing videos with thumbnails and duration
The NgDialog module was used to display the requested feedback field that enabled the user to post feedback about the service. A feedback form was added in the footer of the page as a button with the label “Give feedback”. Figure 5.12 shows how the feedback field looked.

Figure 5.12 The feedback form
A new column was added next to the HTML5 video player in the WM for displaying related content such as PowerPoint presentations if these are available for a certain video.

The video object received from the API now included an array of related content objects that could be looped through using the AngularJS ng-repeat directive and displayed to the user as shown in Figure 5.13. For each iteration the index value can be received by using the $index variable which AngularJS sets. This variable is used to differentiate the different presentation files displayed in the view of the WM as shown in figure 5.14.

```html
<div ng-repeat="content in videoObject.related_content">
  <a target="_self" download="{{content.url}}" ng-href="/api/videos/getContent/{{content.url}}"></a>
  <span class="glyphicon glyphicon-file"></span>{{videoObject.media_title}} presentation {{$index+1}}
</div>
```

![Figure 5.13 Presenting additional content for videos](image)

![NetTalks: On demand streaming for knowledge sharing at Netlight](image)
5.2.2.4 Testing

During this sprint UX activities were researched as well as AngularJS unit tests. However, no tests had been written yet. The implemented features were tested manually and approved by the product owner and stakeholders at a demo session at the end of the sprint. From the demo session requirements regarding how the system could be improved and what to implement next was gathered and added to the backlog. These requirements were as following:

- It should be possible for the user to see what spoken language each video is in
- It should be clearer what the system is and what features it includes. It should be stressed that feedback on the system is requested from the users
- It should be possible to see user behavior statistics, such as how many times a video has been watched
- Because many employees will most likely use the system on mobile devices, the system should also work on these devices
- With an increasing number of videos it is important that it is possible to filter videos based on a set of attributes

5.2.2.5 Retrospective

At the retrospective meeting conducted after this sprint it was concluded that we were able to produce new system features at a fast pace. However the team felt that there still were some problems with having a structured process to follow, especially when designing and testing the system. Quite often only the requirement gathering and implementation process were properly executed. Tasks were not assigned to any specific person and their implementation decisions were not really discussed in the team, quite often a given developer took decisions without consulting the other team members first, which have been causing problems during the sprint. A more structured process was decided upon and written down for all team members to follow focusing on clear steps for the design and testing phase of each sprint. For the design phase it was concluded that sketches and prototypes
should be use before starting the development work, and that these should be tested on the users of the system before the implementation work should be started. For the testing phase it was concluded that unit test should be written before a task could be considered as “done”. There should also be usability testing involving real users to test the features that was added in a sprint. For this to work the team concluded that they would try to use the Dual-track Scrum development process.

5.2.3 Sprint 3

In Sprint 3 the development process was changed from regular Scrum to Dual-track Scrum. This was done because it was believed to be a better developing process to achieve good UX. During this sprint it was also concluded what and how information about videos should be presented to the user. The sprint also included deciding upon how to save and handle user statistics. In Sprint 3 both unit and usability testing was first introduced and as well as a more structured way of designing the system using flow charts.

5.2.3.1 Requirements

Quite a lot of requirements for this sprint came from the demo at the end of the previous sprint. As can be seen in Table 5.3 presentation text was to be added for the system. It should be clear what language each media file was in. The feedback field could be mistaken for feedback on media files and this needed to be clarified. Statistics was to be save about user behavior such as which videos they are watching to be able to present the users with the most popular videos. The system is requested to be able to work on mobile devices.

<table>
<thead>
<tr>
<th>Requirement ID</th>
<th>Description</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>MM.1.3</td>
<td>The user should be to see the spoken language of each video</td>
<td>Medium</td>
</tr>
<tr>
<td>MM.5</td>
<td>The user shall be able to read information about the system</td>
<td>Medium</td>
</tr>
<tr>
<td>SM.1</td>
<td>Information about user activity within the system shall be stored</td>
<td>High</td>
</tr>
<tr>
<td>SM.1.1</td>
<td>Statistics about how many times a video</td>
<td>High</td>
</tr>
</tbody>
</table>
The system shall be able to retrieve user activity statistics

The system shall be able to fetch the number of views each video has

The user shall be able to watch videos on mobile devices (Android and iPhone)

The user shall be possible to filter and sort videos based on given metrics

The user shall be able to search for a video using title or speaker through a search field

5.2.3.2 Design

In the third sprint system flow charts were developed for the first time to get a clear overview of how the modules should work and the features that should be included. A new module for statistics was added called Statistics Module (SM). Requirements for this module can be seen in table 5.3. A new video player was to be added to provide support for watching videos on mobile devices.

A plan for usability testing was developed that included both quantitative and qualitative tests. The plan involved physically meeting users and ask them to perform tasks from scenarios that involved using the implemented features in order to get an understanding of how it was perceived and if the developed features were in line with the users’ vision for the system. The users were asked to talk aloud about what they were doing, how they saw the system and if something was unclear. After the test they filled in a SUS-survey that would work as quantitative tests to make sure that the system was at an acceptable level of usability and to enable UX statistics over time.

A new module for called Filter and sorting module (FSM) was created to handling the functionality regarding filtering and sorting videos on different variables such as video title, speaker and office. Flow charts were created for the more complex system flows (when the user is about to watch a video and when a video is uploaded to the system). These flow charts can be seen in figure 5.15 and
5.16. The idea with creating flow charts was to get a clear picture of how the system should work and to solve uncertainties before the implementation work was initiated.

![Flow chart of the Watch module](image)

Figure 5.15 Watch module flow chart.
5.2.3.3 Implementation

Quite a lot of features in the system were improved in the system. Following the vision set up in Sprint 1 (that the system should be highly modularized to enable reusability and extensibility). The feedback form was put in a separate component and it was changed how it was presented to the user so it would be clear what its purpose was. Previously all code for the feedback field had been placed in the index.html file but now only the controller was specified according to Figure 5.17. In the feedback controller it was then possible to define an external HTML template using the ngDialog module. How this was implemented is shown in Figure 5.18.
It was also possible to add external HTML files to a given HTML file using the `ng-include` directive. Figure 5.19 shows how the video player and related content component (listing related files such as PowerPoint presentations in the Watch Module) was moved to external components and then added to the view of the WM.

JW Player was implemented instead of the previously used standard HTML5 video player. This new player also used the HTML5 video framework but also enabled playback on Android and iOS devices [67].

To present language for a specific video in the MM speech bubbles were added to the bottom left corner of each video thumbnail. The result is shown in Figure 5.20. Figure 5.21 shows how this is implemented in the MM view. The first class value for the `span`-tag is “speech-bubble-{{metaData.media_language}}”, which dynamically fetches the language of a certain video from the video object and sets the language at the end of this attribute. The second class value is set to “speech-bubble”, which hold CSS rules that is common for all speech bubbles. The CSS rules for these classes can be seen in Figure 5.22. Figure 5.21 also shows how easy it is to implement filtering features using AngularJS. By specifying “ng-model='searchString'” for the input field the current value in the input field is available globally and will be updated whenever the user alters this field. By using “filter:” on this variable in the ng-repeat section videos will be filtered depending on the value of the input field.
**NetTalks**

NetTalks is an interactive platform for increased knowledge sharing at Netlight. The idea is that any Netlight employee should be able to upload, comment, discuss and enjoy video content from all Netlight’s knowledge sharing sessions.

What you’re currently using is a VERY BETA version! We are releasing this early version because we want you to try this new system and give us feedback in order for us to make it better!

Try it out and send us some feedback about what you think and if you have any thoughts for improvement!

### Videos

| 00:36:34 | Hjalmar Moa
Architecture and Object
Oriented Design With Solid Principles and Onion Architecture |
|----------|---------------------------------------------------------------|
| 00:14:33 | Fredrik Rosenqvist, Fredrik Salin
NetTalks Presentation |
| 00:14:13 | Ernil Cofstson
Matching Algorithm Presentation |
| 00:44:36 | Johanna Lundén, Karin Sigge
Sustainable Consultants & Time Management |
| 00:31:38 | Peter Lindgren, Fredrik Schersten, Charlotte Kastengren
Hot Topics Within Agile |
| 00:44:50 | Mia Clarke
You look good today and other pitfalls |
| 00:44:50 | Fredrik Rosenqvist, Fredrik Salin
NetTalks Promotion Video |
| 00:05:41 | Martin Wenneberg
Lightning Talk - Martin Wenneberg |

Figure 5.20 The new look of the main view
Figure 5.21 Implementation of Main View

```html
<div class="inner-addon right-addon">
  <i class="glyphicon glyphicon-search"></i>
  <input type="text" ng-model="searchString" class="form-control" placeholder="Search presentations..." />
</div>
<div class="video_row">
  <div class="col-xs-3" ng-repeat="metadata in videoMetaData | filter:searchString">
    <span class="thumbnail">
      <a href="" ng-click="goToVideo(metadata)">
        <img ng-src="/api/thumbnail/{{metadata.media_thumbnail}}" />
      </a>
      <div ng-repeat="speech in metadata.speech-bubble | filter:searchString">
        <span class="speech-bubble" ng-repeat="language in metadata.media_language | filter:searchString">
          {{metadata.media_length}}
        </span>
      </div>
    </span>
  </div>
</div>

Figure 5.22 CSS rules for main view

```
```
.speech-bubble {
  background-repeat: no-repeat;
  background-size: 2.5em auto;
  bottom: 0.5em;
  color: white;
  display: block;
  height: 2.5em;
  left: 0.5em;
  position: absolute;
  width: 2.5em;
}
.speech-bubble-Swedish {
  background-image: url("/assets/images/swedish_speech_bubble.png");
}
.speech-bubble-English {
  background-image: url("/assets/images/english_speech_bubble.png");
}
.speech-bubble-Finnish {
  background-image: url("/assets/images/finnish_speech_bubble.png");
}
.speech-bubble-Norwegian {
  background-image: url("/assets/images/norwegian_speech_bubble.png");
}
.speech-bubble-German {
  background-image: url("/assets/images/german_speech_bubble.png");
}
```
To save and manage statistics for users two options were available. Either save and manage this information in the system, or use a third party service to solve this problem. It was decided to use Google Analytics for saving this data because it was easy to implement, it was not required to save and come up with a strategy for how to solve this in the system, and Google Analytics had a very good CMS and API to fetch the data later on \textsuperscript{[68]}[69]. The only thing required to get it working was to add a JavaScript tracking code to the MM and the WM.

5.2.3.4 Testing

Unit testing was conducted in this sprint for the frontend components. Both white- and black box testing was performed on the watch-, main module. Seven usability testing sessions were held that showed that the users were quite pleased with the current functionality and look of the system. A few issues were found, mainly involving the limited ability to search and filter the videos.

5.2.3.4.1 Unit tests

Table 5.4 presents a mapping between the conducted unit tests during this sprint and the requirements they are testing.

<table>
<thead>
<tr>
<th>Test Case</th>
<th>Test Case Identifier</th>
<th>Tested Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tests that all video objects are received from the database and are in the correct format</td>
<td>MMT.1</td>
<td>MM.1.1</td>
</tr>
<tr>
<td>Checks that the user if routed to the correct video view when selecting a specific video to watch</td>
<td>MMT.2</td>
<td>MM.2</td>
</tr>
</tbody>
</table>

AngularJS was created with testability in mind. For unit tests it follows the Separation of concerns principle – controllers, services and even functions should be possible to test in isolation without having dependencies to other parts of the system. Karma, was used to execute AngularJS tests via the command line and the Jasmine testing framework was used to write and structure the unit tests \textsuperscript{[70]}. Using
the Grunt task runner all tests could be executed running “grunt test:client” in the terminal, which will trigger Karma to run the tests.

Figure 5.24 presents the definition of a group of tests for the MM controller. Using Jasmine framework the “describe”-function groups tests together. Within the tests the application is loaded and the controller is injected along with other services that are needed to run the tests like a scope, a mock http backend and the location service.

```
'use strict';

describe('Controller: MainCtrl', function () {

    // load the controller's module
    beforeEach(module('netTalksApp'));

    var MainCtrl, scope, httpMock, controller, location;

    // Initialize the controller and a mock scope
    beforeEach(inject(function ($controller, $rootScope, $httpBackend, $location) {
        scope = $rootScope.$new();
        controller = $controller('MainCtrl', {
            $scope: scope
        });
    }));

    it('should get videos from api request', function () {
        // Define response for calling api and then perform the call
        scope.getVideos();
        httpMock.expectGET('/api/metaData/allEntries').respond([{
            "media_id": 1,
            "media_title": "Architecture and Object Oriented Design",
            "media_upload_date": "2015-02-16T23:00:00.000Z",
            "media_uploader_id": 1,
            "speaker": "Hjalmar Noa"
        }]);
        httpMock.flush();
        // Check if the api call has set the scope variable videoMetaData media_id to 1
        expect(scope.videoMetaData[0].media_id).toEqual(1);
    });
```

Figure 5.24 Defining the test and injecting required services

When the group of tests are defined each individual test case is defined within an “it”-call. Figure 5.25 presents a test case that checks whether the function “getVideos” sets the variable videoMetaData correctly for a given response from the API. The test case in Figure 5.26 checks whether a user is being routed to the correct url when the function “goToVideo” is called.
5.2.3.4.2 Usability tests

The usability tests were conducted with people at the internship company with different backgrounds and roles. Because this was the first conducted usability test a larger amount of employees (6 persons) were selected to participate. Table 5.5 lists the test cases and what requirements they are testing.

<table>
<thead>
<tr>
<th>Test Case</th>
<th>Test Case Identifier</th>
<th>Tested Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Checks that the user understands that the search field can be used to search for title of video and speakers. The test case also checks that a user can select a specific video and start to stream it in Google Chrome. At the end of the test the user is asked to pause the video</td>
<td>UT.1</td>
<td>WM.1, WM.1.2, MM.1, WM.1.3, WM.1.4</td>
</tr>
<tr>
<td>Test that the user understands how to bookmark a video to watch it later and how to share a video</td>
<td>UT.2</td>
<td>WM.2.1, MM.3</td>
</tr>
</tbody>
</table>
The test checks whether the user can find and download PowerPoint presentation for a given video

Checks whether the user can leave feedback for the system

<table>
<thead>
<tr>
<th>Test person</th>
<th>Task 1</th>
<th>Task 2</th>
<th>Task 3</th>
<th>Task 4</th>
<th>Task 5</th>
<th>Task 6</th>
<th>Task 7</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>5</td>
<td>7</td>
<td>7</td>
<td>7</td>
</tr>
</tbody>
</table>

Task completion rate

|                  | 100%   | 100%   | 100%   | 71.43% | 100%   | 100%   | 100%   |

5.2.3.4.2.1 Performance based metrics

Table 5.6 shows whether a given test person (TP) managed to complete a certain task as well as the task completion rate for all test persons. If a task was complete it was marked with a one (1), and if the task was not completed it was marked with a zero (0), in the table below.

5.2.3.4.2.2 Issue based metrics

In Table 5.7 presents identified problems for each task during the usability testing sessions as well as their frequency and severity. This is also visualized in Figure 5.29.
### Table 5.7 Issue based metrics

<table>
<thead>
<tr>
<th>Task</th>
<th>Problem</th>
<th>Frequency</th>
<th>Severity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>The search field on the front page is a bit hard to find</td>
<td>2</td>
<td>Low</td>
</tr>
<tr>
<td>1</td>
<td>It is unclear if the link below the thumbnail for a video and the thumbnail link are links to the same page</td>
<td>1</td>
<td>Low</td>
</tr>
<tr>
<td>2</td>
<td>It is hard to find where to download Powerpoint presentations</td>
<td>4</td>
<td>Medium</td>
</tr>
<tr>
<td>3</td>
<td>The search field on the front page is a bit hard to find</td>
<td>1</td>
<td>Low</td>
</tr>
<tr>
<td>3</td>
<td>There is limited information about what it is possible to search for</td>
<td>1</td>
<td>Low</td>
</tr>
<tr>
<td>4</td>
<td>It is hard to find where to filter videos by language</td>
<td>6</td>
<td>Medium</td>
</tr>
<tr>
<td>5</td>
<td>It is not that easy to share videos you think is interesting</td>
<td>6</td>
<td>Medium</td>
</tr>
<tr>
<td>6</td>
<td>It is unclear how the search results are connected to the search term</td>
<td>5</td>
<td>Medium</td>
</tr>
<tr>
<td>7</td>
<td>The feedback window does not disappear after feedback is sent</td>
<td>2</td>
<td>Low</td>
</tr>
</tbody>
</table>
Figure 5.29 Graph showing severity and frequency for the issue based metrics

5.2.3.4.2.3 Self reporting metrics

The following values presented in Table 5.9 are the result from a questionnaire that every test person answered after completed usability testing session. The questionnaire was based on the SUS and the SUS value for each test person as well as the average SUS score is presented in the table.

<table>
<thead>
<tr>
<th>Question number</th>
<th>TP1</th>
<th>TP2</th>
<th>TP3</th>
<th>TP4</th>
<th>TP5</th>
<th>TP6</th>
<th>TP7</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>3</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>9</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Total</td>
<td>39</td>
<td>35</td>
<td>35</td>
<td>36</td>
<td>31</td>
<td>34</td>
<td>36</td>
</tr>
<tr>
<td>SUS value</td>
<td>97,5</td>
<td>87,5</td>
<td>87,5</td>
<td>90</td>
<td>77,5</td>
<td>85</td>
<td>90</td>
</tr>
<tr>
<td>Average SUS value</td>
<td>87,86</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

5.2.3.5 Retrospective

At the retrospective meeting the team members felt that pace of which the system was developed had slowed down a bit. This was partly because a lot of work had been placed on making sure that the processes presented in the last sprint were followed. Another reason was that the usability testing sessions took quite a while to plan, conduct and evaluate. The researcher had almost no experience of usability testing from previous projects. The usability test were concluded to be a very good way of making sure that the developed system was in line with the users’ view of how the system should be. Quite a lot of new requirements came up from these
sessions, even though the users seemed to be quite pleased with the first version of
the system. These requirements were pretty specific and easy to add to the sprint
backlog of the next sprint.

It was also concluded that getting user feedback via the implemented feedback
field only resulted in feedback like “Good work”, and “Looks nice”, that was not
really useful for improving the system.

5.2.4 Sprint 4

In Sprint 4 advanced filtering features was added, the statistics gathered in
Sprint 3 was fetched and presented to the users, and comment functionality was
added to the system.

5.2.4.1 Requirements

Quite a few requirements for this sprint came from the usability testing
sessions, or they provided more specific information about how to implement
planned features more in detail. This involved tasks about filtering videos based on
language, office and tags. Some tasks were also added related to the structure and
look of the main menu and position of search field and text on the pages. A feature
for retrieving statistics from the Google Analytics API was also to be implemented
to be able to present the most popular videos to the users. Another requested feature
was to be able to fetch and post comments from the internal social network from
this system. Using the results from the usability testing tasks for altering the UI
design could be added and the UI could be altered accordingly.

<table>
<thead>
<tr>
<th>Requirement ID</th>
<th>Description</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>FSM.1.2</td>
<td>The user shall be able to filter videos based on language, office and tags</td>
<td>Medium</td>
</tr>
<tr>
<td>SM.2.2</td>
<td>The user shall be able to see the number of views each video has</td>
<td>High</td>
</tr>
<tr>
<td>CM.1</td>
<td>The system shall be integrated with Yammer to gain comment functionality</td>
<td>Medium</td>
</tr>
</tbody>
</table>
CM.1.1 The user shall be able to watch comments for a given video Medium

CM.1.2 The user shall be able to post comments on a given video Medium

WM.2.2 The user shall be able to share a video by mail or copy link using a share button Medium

5.2.4.2 Design

A new module was designed called Comments Module (CM). This module was created to receive and post comments to/from the internship company social media platform. From the interviews it was understood that it was common for the whole team to be involved in sketching and designing new features and therefore all team members were involved in creating sketches and wireframes for all pages of the website during this sprint. The common design work lead to team discussions about how to solve different features to be implemented and what technical issues there might be with the current design. These sketches were created an iterated on the whiteboard and the final design, in form of wireframes, which all team members had concluded was the best one was taken photos of and saved as documentation. A prototype was also developed for the upload feature that was to be used in sprints to come. The prototype was created using a web-based tool on the page weld.io. Using this tool it was easy and fast to create an interactive prototype that included the wanted interactions of the upload feature. The prototype was developed keeping the previously created flow charts in mind. At least for the upload feature that was yet to be developed the flow chart helped when developing the sketches, wireframes and the prototype.

From the usability testing sessions it was determined that the purple banner in the header section, as can be seen in figure 5.20, was too large and unnecessary, the describing text under the banner was also not necessary and this should be fixed in this sprint. The header banner would therefore be removed and the text under this banner would be moved to a separate page called “About”, as can be seen in the wireframe of the MM view in figure 5.30. Figure 5.30 also shows how the user can use dropdowns to filter on office, language and tags. TedTalks filtering layout inspired the design for the filters area. The listing of videos was divided into two
sections, one showing the most popular videos and one displaying the newest videos. The idea of this layout comes from Youtube who also have sections for popular, new and recommended videos on their site.

Figure 5.31 shows the wireframe for the upload feature, where users were going to be able to post new videos to the system. The design was based on the standard layout of an html form, where the user would start at the top with the most important required information and then work its’ way down as information is added. The form would give the user information for each field, such as that the video file must be in mp4 format, and would give the user feedback saying what was wrong if the user enters incorrect information for any of the fields. At the bottom there would be a button for canceling the upload and a button for submitting the form. The submit button would not be activated until all required information was added.

Figure 5.32 displays the wireframe for the WM view. This wireframe was quite similar to what the system looked like before, the only difference being that a section for comments was added below the video information section and that below the listing of PowerPoint presentations there would be a section for recommended video.

From the upload wireframe a prototype was created showing the look and system flow for this new feature. The prototype can be seen in figure 5.33-40. Prototypes for the other modules were not created because these features were already implemented and the users seemed to be quite pleased with how they worked.
Figure 5.30 Main view wireframe.
Figure 5.33 Upload view wireframe.
Figure 5.34 Watch view wireframe.

![NetTalks](image)

**Upload video**

Fields marked with an asterisk (*) are required fields

- **Choose Video**
- **Choose Presentation**
- **Title**
- **Speaker/Speakers**
- **dd/mm/yyyy**
- **Description**
- **Language**

Select categories applicable for topic:
- Frontend
- Backend
- UX
- BI
- Agile development

![Upload form](image)

Figure 5.35 First step of upload view
Figure 5.36 Step two of the upload feature
Figure 5.37 Step three of the upload feature
Figure 5.38 Step four of the upload feature
Figure 5.39 Step five of the upload feature
5.2.4.3 Implementation

The sketches created during this sprint gave a clear image of what to develop and that all members of the development team had the same vision and idea about the layout and flow of the system simplified the development process. Statistics were added for getting information from Google Analytics about the number of views for each video. This was implemented according to Figure 5.41. First statistics were fetch via the API and then for each video in the video metadata object the corresponding number of views was added as “videoViews”. This way the viewing statistics are available both in the MM view and controller. Figure 5.42 shows how tags are fetched from the API and then made available in the MM controller and view by creating a new scope variable called “filterOptions.tag”. Another variable was also added for keeping track of the number of available tags. This could later on be used when displaying the tags in the view.
Figure 5.41 Including viewing statistics to the video metadata object

Figure 5.42 Getting tags from the API and making them reachable for the view

The AngularJS module ngTagsInput was added to handle showing tags in the view in an easy way. ngTagsInput introduces a directive “tags-input” as well as an “auto-complete” directive for making it easier to select a tag from the list of available tags. Figure 5.43 shows how this was implemented in the MM view and also shows how “ng-options” can be used to loop through an array and show these elements in a dropdown list. The “loadTags($query)” function passes the user input to the controller and compares the input to the available tags fetched in Figure 5.41. If the user input is a substring of any of the available tags, these tags will be returned and display to the user in the dropdown list of the view. Figure 5.44 shows how to add multiple filters to filter the results in the video object loop as well as how to order the videos by number of views using the AngularJS “orderBy” filter component. Figure 5.45 shows the new look of the MM view when it is filtered by
office “Stockholm” and language “English”. It also shows the autocomplete feature for the tags input filter.

5.43 Filters for tags, office and language

5.44 Filtering results in the video object loop
When implementing the new look of the watch view the researcher tried an approach where the video would stretch for the whole width of the page according to figure 5.46. This was concluded to be better than the design in figure 5.34 created during the design session and therefore the wireframe and the layout was changed to this redefined design.
5.2.4.4 Testing

The developed prototype for the upload module was tested on a few users (not in official usability testing sessions though, more to get a feeling of if they liked the design or not) before the implementation work started to make sure the idea we had about the to be developed upload feature were in line with the users’ idea for this feature. Unit tests were written for the newly created statistics module according to Table 5.11.

Table 5.11 Sprint 4 Unit tests

<table>
<thead>
<tr>
<th>Test Case</th>
<th>Test Case Identifier</th>
<th>Tested Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tests that the feedback uses http post to pass the data to the server via the API</td>
<td>MMT.3</td>
<td>MM.4.1</td>
</tr>
<tr>
<td>Checks that it is possible to use the</td>
<td>MMT.4</td>
<td>MM.2.1</td>
</tr>
</tbody>
</table>

Figure 5.46 New look of watch view
created video service to set and get current video object

Tests that the media source is correctly set for the video player

<table>
<thead>
<tr>
<th></th>
<th>WMT.5</th>
<th>WM.1.1</th>
</tr>
</thead>
</table>

5.2.4.5 Retrospective

As a whole we found that we had a more structured workflow. Using wireframes and prototypes made the whole team get a better picture of what to develop and issues could be found at an earlier time. We also found comfort in having the same vision for the system and that it improved our communication. The design work in this sprint (sketching, prototyping, wireframes) took quite some time to perform, which resulted in that less development work was conducted. To create prototypes and wireframes for the already existing views should probably have been more useful if this was done earlier, and this is also the way to do it if we should have been using Dual-track Scrum as it should be used. Another reflection was the changed design of the watch view during the sprint. According to Dual-track Scrum the design should be done one sprint ahead, but if the design needs to be changed during a sprint this is not followed. This posed questions regarding if designing one sprint ahead actually was such a good idea.

Because not that many new features were added to the system during this sprint and the fact that, as was concluded in the previous sprint, usability testing takes quite a lot of time, this was not performed during this sprint.

The team found it good that we had been taking to our product owner and asked her to priorities the tasks to do ahead. This made us not having to ask her all the time what we should do next and what to focus on. By adding the statistics it was also possible to see that the system was already being used by people at the internship company as can be seen in the graph in figure 5.47 which is taken from Google Analytics.
5.2.5 Sprint 5

The upload feature was developed during this sprint. The second usability testing session was performed on previously developed features.

5.2.5.1 Requirements

For this sprint the upload function was started to be developed. The user should be able to upload a video, a presentation file and add metadata for the video through a form on a separate upload page. As can be seen in Table 5.11 most requirements for this sprint were connected to the upload feature of the system.

<table>
<thead>
<tr>
<th>Requirement ID</th>
<th>Description</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>UM.1</td>
<td>The user shall be able to upload media files to the streaming server</td>
<td>High</td>
</tr>
<tr>
<td>UM.1.1</td>
<td>The user shall be able to select a video file to upload through a form</td>
<td>Medium</td>
</tr>
<tr>
<td>UM.1.2</td>
<td>The user shall be able to select a related content file such as Powerpoint presentations through a form</td>
<td>Medium</td>
</tr>
<tr>
<td>UM.1.3</td>
<td>The user shall be noticed when the upload is successful</td>
<td>Medium</td>
</tr>
<tr>
<td>---------</td>
<td>------------------------------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>UM.2</td>
<td>The user shall be able to add metadata for a specific video so it can be presented to other users</td>
<td>High</td>
</tr>
<tr>
<td>UM.2.1</td>
<td>The user shall be able to add information such as title, speaker, location, language and tags for a specific video</td>
<td>High</td>
</tr>
</tbody>
</table>

### 5.2.5.2 Design
A new module was created called Upload Module (UM) developed from the prototypes created in the previous sprint. This module was going to involve functionality for uploading new videos to the system as well as adding metadata such as title, speaker and description for this video.

### 5.2.5.3 Implementation
The implementation of the upload feature was developed keeping the wireframes and prototypes created in sprint 4 in mind. Already having the design prepared enabled the development work to start quickly. Angular File Upload was implemented for handling the file upload. Angular File upload is module for AngularJS that supports drag-n-drop upload, upload progress, validation and upload queue. A form was used for the file upload and this form is presented in figure 5.48. An important part for making the form user friendly was to add form validation and telling the user how to use the upload form and give feedback if the user did something wrong during the upload process.
5.2.5.4 Testing

Both unit tests and usability tests were conducted and these tests are described below.

5.2.5.4.1 Unit tests

The unit tests for sprint 5 are presented in Table 5.12.

Table 5.12 Sprint 5 Unit tests

<table>
<thead>
<tr>
<th>Test Case</th>
<th>Test Case Identifier</th>
<th>Tested Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tests that the feedback uses http post to pass the data to the server via the API</td>
<td>MMT.3</td>
<td>MM.4.1</td>
</tr>
<tr>
<td>Checks that it is possible to use the</td>
<td>MMT.4</td>
<td>MM.2.1</td>
</tr>
</tbody>
</table>
created video service to set and get current video object

| Tests that the media source is correctly set for the video player | WMT.5 | WM.1.1 |
| Fetches user statistics and checks that this data is in the correct format | SMT.1 | SM.2, SM.2.1 |
| Checks that comments are received from Yammer and that comments are shown if these exists for a specific video | CMT.1 | CM.1.1 |

### 5.2.5.4.2 Usability tests

Table 5.13 lists the test cases and what requirements they are testing. The usability tests were conducted with people at the internship company with different backgrounds and roles. This was second time usability testing was performed and because of this as well as the fact that there had not been that many new features and system flows implemented in the system, only three employees were selected to participate.

<table>
<thead>
<tr>
<th>Test Case</th>
<th>Test Case Identifier</th>
<th>Tested Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tests that the user can post comments to Yammer for a video</td>
<td>CMT.2</td>
<td>CM.1.2</td>
</tr>
<tr>
<td>Tests that the user can filter videos based on tags, office and language as well as the user understands that this is possible</td>
<td>MMT.4</td>
<td>FSM.1.2</td>
</tr>
</tbody>
</table>
5.2.5.4.2.1 Performance based metrics

Table 5.14 shows whether a given test person (TP) managed to complete a certain task as well as the task completion rate for all test persons.

<table>
<thead>
<tr>
<th>Test person</th>
<th>Task 1</th>
<th>Task 2</th>
<th>Task 3</th>
<th>Task 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>3</strong></td>
<td><strong>2</strong></td>
<td><strong>3</strong></td>
<td><strong>2</strong></td>
</tr>
<tr>
<td><strong>Percent completed</strong></td>
<td><strong>100%</strong></td>
<td><strong>66,67%</strong></td>
<td><strong>100%</strong></td>
<td><strong>66,67%</strong></td>
</tr>
</tbody>
</table>

5.2.5.4.2.2 Issue based metrics

In Table 5.15 presents identified problems for each task during the usability testing sessions as well as their frequency and severity. This is also visualized in Figure 5.51.

<table>
<thead>
<tr>
<th>Task</th>
<th>Problem</th>
<th>Frequency</th>
<th>Severity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>The video should not start to play automatically</td>
<td>2</td>
<td>Medium</td>
</tr>
<tr>
<td>2</td>
<td>It is not possible to sort videos by number of views</td>
<td>3</td>
<td>Medium</td>
</tr>
<tr>
<td>2</td>
<td>It is not clear that Yammer is the comment feature</td>
<td>1</td>
<td>High</td>
</tr>
<tr>
<td>3</td>
<td>It is possible to add your own invalid tags to the tag search field</td>
<td>1</td>
<td>Medium</td>
</tr>
<tr>
<td>3</td>
<td>It is confusing to present related video content besides the playing video.</td>
<td>1</td>
<td>Low</td>
</tr>
<tr>
<td>3</td>
<td>It should be possible to see all video information when entering the watch view</td>
<td>1</td>
<td>Low</td>
</tr>
</tbody>
</table>
It should be possible to sort videos by date

5.2.5.4.2.2 Self reporting metrics

The following values presented in Table 5.16 are the result from the SUS questionnaire performed after each usability testing session. The SUS value for each test person as well as the average SUS score is presented in the table.

Table 5.16 SUS answers and value

<table>
<thead>
<tr>
<th>Question</th>
<th>Test person 1</th>
<th>Test person 2</th>
<th>Test person 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>9</td>
<td>5</td>
<td>3</td>
<td>5</td>
</tr>
</tbody>
</table>
5.2.5.5 Retrospective

During the retrospective meeting the team concluded that much more task points had been finished compared to the previous sprint. The team felt that the testing procedures that in previous sprints had been unclear now were understood and structured. The team felt confident with the work process as a whole. However, the team still took on too much task scores, which left task unfinished at the end of the sprint. For the coming sprints it was therefore concluded that at the sprint planning the team should not add task so it exceeded the number of completed task points from the previous sprint. Another problem that was noticed was that team members tend to start new tasks instead of completing tasks that were on the code review and testing lane on the sprintboard. This was something that the team needed to stop doing; tasks should be completely finished before starting new ones.

5.2.6 Sprint 6

In the final sprint a lot of small design tasks were performed to tweak the design to fit the needs of the users. The system was designed to have a more simplistic and clean design.

5.2.6.1 Requirements

The requirements for sprint 6 is shown in table 5.17.

<table>
<thead>
<tr>
<th>Requirement ID</th>
<th>Description</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>FSM.1.3</td>
<td>The user shall be able to filter videos based on number of views and date</td>
<td>Medium</td>
</tr>
</tbody>
</table>
5.2.6.2 Design

From the usability tests at the end of the previous sprint suggestions to have “areas” for popular and new videos were made. It was also mentioned by the test persons that all filtering functionality needn’t be visible all the time, it was enough showing the search field and giving the user the option to use more advanced filtering alternatives on demand. Some users also thought that the GUI should be simpler and have more “air” between the different sections on the page. These requests are in line with the guidelines in chapter 3.3.6.5 (Keep the design simple) and 3.3.6.6 (Make use of white space). Therefore these requests were taken in consideration and design in this sprint.

5.2.6.3 Implementation

Figure 5.52 shows the new improved look of the GUI. This refined design is simpler and more “air” has been added between the newly added popular videos and newest video sections. The advanced filtering alternatives were moved to just be displayed when the user chooses to do this by pressing the “Filters” button to the right of the search field. When the user uses any of the filtering alternatives the popular and newest sections are not displayed any more and instead the search results are displayed as shown in figure 5.53. Figure 5.53 also shows how the user can sort the videos based on view count or date using a dropdown box. The same type of design was also implemented in the watch view presented in figure 5.54.
Figure 5.52 Improved look of main view including video sections
5.2.6.4 Testing

Usability test focusing on testing the upload feature was performed during this sprint.

5.2.6.4.1 Usability tests

The usability tests were conducted with people at the internship company with different backgrounds and roles. This was third time usability testing was performed
and because of this as well as the fact that there had not been that many new features and system flows implemented in the system, only three employees were selected to participate. Table 5.19 lists the test cases and what requirements they are testing.

Table 5.19 Sprint 6 Usability tests

<table>
<thead>
<tr>
<th>Test Case</th>
<th>Test Case Identifier</th>
<th>Tested Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Checks that it is possible to filter videos based on language, office and tags</td>
<td>FSMT.1</td>
<td>FSM.1.2</td>
</tr>
<tr>
<td>Tests that it is possible to upload a presentation with given files and metadata</td>
<td>UMT.1</td>
<td>UM.1, UM.2, UM.1.1, UM.1.2</td>
</tr>
</tbody>
</table>

5.2.6.4.3 Performance based metrics

Table 5.20 shows whether a given test person managed to complete a certain task as well as the task completion rate for all test persons.

Table 5.20 Task completion rate

<table>
<thead>
<tr>
<th>Test Person</th>
<th>Task 1</th>
<th>Task 2</th>
<th>Task 3</th>
<th>Task 4</th>
<th>Task 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Total</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Percent completed</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
</tr>
</tbody>
</table>

5.2.6.4.4 Issue based metrics

Table 5.21 presents identified problems for each task during the usability testing sessions as well as their frequency and severity. This is also visualized in Figure 5.57.
Table 5.21 Issue based metrics for each task

<table>
<thead>
<tr>
<th>Task</th>
<th>Problem</th>
<th>Frequency</th>
<th>Severity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Not having a pointer when hovering over a video makes it unclear if it is clickable</td>
<td>1</td>
<td>Low</td>
</tr>
<tr>
<td>1</td>
<td>It is not possible to list all videos on one page in an easy way</td>
<td>1</td>
<td>Low</td>
</tr>
<tr>
<td>1</td>
<td>It would make more sense to have newest videos over popular videos</td>
<td>1</td>
<td>Low</td>
</tr>
<tr>
<td>2</td>
<td>It is sometimes hard to see if the videos were filtered</td>
<td>1</td>
<td>Low</td>
</tr>
<tr>
<td>4</td>
<td>The dropdowns, buttons and search fields do not look the same. This confuses the user</td>
<td>1</td>
<td>Low</td>
</tr>
<tr>
<td>5</td>
<td>It is unclear if that the buttons are used to select a file</td>
<td>1</td>
<td>Low</td>
</tr>
<tr>
<td>5</td>
<td>The user shouldn't be able to add his or her own tags</td>
<td>2</td>
<td>Medium</td>
</tr>
<tr>
<td>5</td>
<td>It is not possible to see all available tags in the dropdown for tags in the upload form</td>
<td>1</td>
<td>Low</td>
</tr>
<tr>
<td>5</td>
<td>The cancel button can should be red, can be mistaken for some kind of error</td>
<td>1</td>
<td>Low</td>
</tr>
</tbody>
</table>

**Number of usability issues based on severity**

![Chart showing number of usability issues based on severity]
5.2.6.4.5 Self-reporting metrics

The following values presented in Table 5.22 are the result from the SUS questionnaire performed after each usability testing session. The SUS value for each test person as well as the average SUS score is presented in the table.

<table>
<thead>
<tr>
<th></th>
<th>TP1</th>
<th>TP2</th>
<th>TP3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>9</td>
<td>5</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Total</td>
<td>38</td>
<td>38</td>
<td>36</td>
</tr>
<tr>
<td>SUS value</td>
<td>95</td>
<td>95</td>
<td>90</td>
</tr>
<tr>
<td>SUS average</td>
<td>93.33</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

5.2.6.5 Retrospective

At the retrospective meeting for this sprint there were a lot of positive things that were brought up. The team concluded that most of the problems that had existed in previous sprints were fixed. The team did not start new tasks when some tasks still had to be code reviewed or tested, the team completed a lot more task points then previous sprints, all team members had a feeling that the system felt more “complete” then after previous sprints and it felt very good that the whole upload system flow was working (this was one of the most complex parts of the system).
5.2.7 User statistics

From the gathered statistics on the page it can be shown that the system had already been started to be used, and from all countries where Netlight has offices. Table 5.23 presents these results.

Table 5.23 Number of sessions from different countries on NetTalks

<table>
<thead>
<tr>
<th>Country</th>
<th>Sessions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sweden</td>
<td>429</td>
</tr>
<tr>
<td>Norway</td>
<td>8</td>
</tr>
<tr>
<td>Finland</td>
<td>7</td>
</tr>
<tr>
<td>Germany</td>
<td>4</td>
</tr>
<tr>
<td>United Kingdom</td>
<td>4</td>
</tr>
</tbody>
</table>

Figure 5.58 presents the diversity of roles for employees participating in the usability testing sessions. As can be people of different backgrounds and skillsets were selected for the testing sessions in order to get better feedback on the system.

Figure 5.58 Roles of employees that were involved in the usability tests
Figure 5.59 presents statistics for how often the employees participating in the usability testing sessions used similar streaming services. This was asked in order to get information about how experienced the users were with using these kinds of services.

![Frequency of using streaming services](image)

**Figure 5.59 Frequency of using streaming services for employees that were involved in the usability tests**

### 5.3 Brief summary

First of all, this chapter presents the result of the interviews conducted at the internship company including information about how they work with frontend development, UX and agile methods professionally at the internship company. The second part of this chapter presents the findings from developing the frontend for the on demand streaming application using the combination of agile methods and UX activities. This section presents the detailed requirements, system design, implementation, testing and reflections from each sprint throughout the development process at the internship company.
Chapter 6 Result and Analysis

In this chapter the results and the analysis from the case study are presented based on the literature study, interviews with employees at the internship company, as well as the researchers’ own reflections from active observations when trying the proposed theories from the literature study and the interviews. The results and learnings about frontend development are also presented.

6.1 Results from the usability tests

As can be seen in figure 6.1 the SUS value (the overall value for the systems’ UX) has increased a lot from the start of the development process to the end of this period, and it increased for each iteration. At the end of the project the SUS value was 93, which is higher than the goal value of at least 85. This shows that the design and implementation decisions taken were good and that the users thought that the system had good UX.

When analyzing the task completion rate from the case study it can be seen that during the first two usability testing sessions there were some tasks that all test persons weren’t able to complete. At the last usability testing session however, all test persons managed to complete all tasks. The number of usability issues has decreased during the sprints as well and during the last session the severity of most issues were low, only one issue had medium severity. The goal of 100% completion rate and only low severity usability issues has therefore also been completed, except one usability issue on the medium severity level.
6.2 UX activities

The most important User Experience activities found were: Sketches, prototypes, wireframes and usability testing. This section presents the findings for how

6.2.1 Sketches

During the research it was found that sketches are good in the initial design of features and views. Sketches should not be put to much time in creating and they should have low fidelity, because it should be possible to quickly alter or throw them away. Having low-fidelity sketches can also provide more feedback from users than a high fidelity sketch. Iterating sketches within the team until the team can conclude that the optimal design was met I found was very useful, and this was something that many of the interviewees at the internship company also practiced during their everyday work. Involving all team members in the sketching processes also results in a better holistic picture of the project for all team members.

When setting the initial design I found that it was very useful to follow the guidelines stated in chapter 3.2.2 as well as to look at similar features that other
companies have developed to get inspiration for the new design. This was something that was practiced by the respondents at the internship company as well.

6.2.2 Wireframes

Wireframes are usually an enhanced version of sketches, where the system flow and navigation are added as well as more detailed interaction elements. I felt that wireframes were good to have as a blueprint that it was possible to use to develop according to. Keeping these as documentation was useful because it made it possible to look at them whenever a team member was unsure about how to implement something to fulfill the design. As mentioned in the state of art as well as during the interviews wireframes can except for being used for communicating design between UX designers and developers also be used to communicate design decisions to management and users. This was something that was not tried during the active observation part of the case study.

6.2.3 Prototypes

During the active observations I felt that prototypes were useful when you want to test user interaction on a certain design. It’s a good way of testing a design without having to first implement it. As for sketches and wireframes it’s important to think about what fidelity to use when developing prototypes. Using high-fidelity prototypes, it might present the user with a better picture of how the system will look but a low-fidelity prototype might provide more feedback to the development team.

6.2.4 Usability testing

Usability testing was something that was practiced by all interviewees at the internship company. Usability testing shall be performed early on and continuously. From the active observations I felt this was probably the most important UX activity because practicing it before implementing a feature and made sure the design would work which minimized the development of features that the users don’t want or don’t understand. Performing usability testing on prototypes before implementing
the feature also resulted in less issues when usability testing was performed after the feature had been performed.

From the active observations I felt that usability testing is best performed when sitting down with a user and observing him or her when letting this user perform specified tasks within the system. This way it is possible to see if the user manages to perform the tasks, what problems their might be if the user does not manage to perform the task, and also lets the user state how it feels to use the system and if the user thinks that something is unclear or needs to be changed. This feedback that was received during the usability testing sessions was usually very specific and easy to add to the backlog of the next sprint enabling the issues to be fixed quickly. Performing usability testing in this way was suggested both by literature and during the interviews.

6.2.4.1 A/B testing

A/B testing was not tested during the active observations. It was used by people at the internship company and it seems like a good way to test designs in production when it might be hard to get hold of users to test the design on, when there is no time for physical usability testing, or as an extra source of feedback besides usability testing.

6.3 Combining agile development and UCD

Both in literature, during the interviews and when trying out the combination by myself in a real project I found that it’s possible to combine Agile software development and UCD with positive results. Some common success factors were found during the case study.

6.3.1 UX persons in development teams

All respondents during the interviews had UX designers as a part of their development team and stated that this person/these persons should be a part of the team for the whole duration of the project. This was also something found in literature and when trying it during the active observations having a UX designer in the team especially improved the design and testing phases during the development work.
6.3.2 Close relation/communication

Having a close and continuous communication between developers and designers is essential because developers are dependent on for instance wireframes and sketches from designers and designers are dependent on for instance working code from the developers to test it on real users. It is important that this exchange of deliverables is working properly so no team member has to wait for some other team member, which might cause a certain member to be blocked in this person’s work.

It is important for all team members to feel responsible for the product together and to do this it is important that all members participate in UX work. By doing this they get a holistic picture of the project and can also come up with their own ideas. For the given internship project I worked as an UX designer and the sole frontend developer by I felt when including a backend developer I could still get feedback and input on a given design and this person might also come with input from the backend part of the system that might affect the given design.

6.3.3 User involvement

As stated in the literature study one of the biggest differences between agile development and UCD is their focus. In agile the focus is working software and customer satisfaction, while in UCD the focus is making the software usable for the end users. One of the biggest shortcomings in agile is not considering the end user and this is one of the most powerful aspects that is added when combining agile and UCD. During the active observations involving the users in the design phase and during usability testing sessions before and after implementing a certain feature provided essential feedback that could quickly be used to alter the system to fit the goals of the users, the ones that are actually going to use the system. It was also concluded that getting feedback from the demo sessions and having a feedback form on the live system supplied insufficient feedback. During the interviews all respondents stressed the importance of involving the user and doing this continuously and on a regular basis.
6.3.4 Development methodologies

As mentioned in the interviews a variety of development methodologies were used by the respondents. Some respondents used Scrum/Kanban but with added UX activities to the development process. These respondents mentioned that using this approach it to an extent felt like UX was just added to a process that was not adapted for UX. Two respondents used Dual-track Scrum with good results. Two respondents used Lean UX/Lean Startup as their development methodology.

Using Dual-track Scrum for the internship company project worked fairly well. I think having a methodology that considers both the agile aspect and UX helps when there is a need for adding UX activities to the development process. Dual-track tries to solve one of the biggest problems when combining agile and UCD that is handling deliveries and making sure that no team member has to wait for the other to conduct their work. However designing one sprint ahead I think can be problematic sometimes because for instance if requirements changes between sprints and something has been “wastefully” developed. As stated by some of the respondents this methodology might not always be that flexible and when following it strictly it could prohibit the developers and designers from having a close communication and might cause problems.

When using Dual-track Scrum think it is very important to keep the two tracks mentality informal and keep a close communication between designers and developers mostly exist on paper. One way to do this is to have design sessions where the whole team participates, which was used during the active observations as well as mentioned by respondents during the interviews.

Because we used concepts coming from Lean Startup like MVP and continuous delivery this methodology might have been more suitable to use for the project.

6.3.5 Documentation

Even though the agile manifest values working software over documentation, some documentation is required in order to get agile to work with UCD. This is mainly design artifacts like wireframes, prototypes and results from usability testing sessions. This documentation is documentation that is required by developers in order to perform development tasks. The documentation should be kept at a level of as low fidelity as possible so it is possible to communicate the design to developers
but it should require too much time or it might not fit in agile iterations or developers might have to wait for these deliverables.

6.3.6 Measuring UX

It is important to make sure that the users are responding well to your design decisions. During the active observations a number of UX metrics was used in order to measure the UX: Task completion rate, issues frequency and severity as well as SUS. This provided information about how many users managed to complete the given tasks, what issues that existed with the design and how severe they were as well as an overall value of the UX. This gave an indication about what features that needed improvement and in what way as well as if the design decisions made had a positive or negative impact on the average UX.

Documenting this information like in the active observations and making graphs and tables were not performed by any of the respondents during the interviews. This seems to be because it took too much time. The respondents instead kept this in their mind and focus on making sure the found errors were fixed. However, most of the respondents used usability testing as a way of measuring the UX.

Other ways of measuring the UX as proposed during the interview were to have people at the company contact users and ask them what they think about the system, or to perform surveys.

Quite a few of the respondents used tracking in their systems to gather statistics about user behavior as a way of measuring the UX, something that was not performed during the active observations.

6.4 Frontend development

This section presents findings from designing and implementing the frontend for the system as presented in chapter 5.2.

6.4.1 Frontend design

During the start of the active observations it was very beneficial to as a team together discuss and set up the high level architecture for the system. This provided a solid foundation to start the development work from as well as it made sure that
the team had the same vision and goal with the system to be developed. It was also good to define the data and the routes of the API because this allowed the frontend and backend to be developed in parallel. Another factor that was found to be very important when designing the system was to always strive to develop the system in smaller modules. This provided structure as well as a loose coupling between different system parts. This for instance made it very easy for us to switch from the standard HTML5 video player used at the start of the project, to the more advanced JW-player without having to make a lot of changes in our code. User experience activities also played a big part in the system design. Sketches, wireframes and prototypes played an important part in specifying the initial design for the features to be implemented and could also be used to test our ideas on the users before starting to implement these. Usability testing provided a very important feedback loop from the users and worked an indication of how good our design ideas were. For coming up with the initial designs the design guidelines in chapter 3.2.2 were a great aid throughout the design process. These guidelines provided us with the best practices for designing for the web. The “use conventions”-guideline was especially important for us because we got a lot of inspiration and aid from looking at how similar systems, like Youtube and TED talks, had structured their sites and interface.

6.4.2 Frontend implementation

Throughout the development work the use of frameworks and tools was very beneficial. The tool Yeoman was useful in the beginning of the project for generating the basic structure of the project according to best practices. This made it possible to get the development work up and running much faster and provided a solid foundation to start the development of the system from. Using a package manager as Bower makes it easier to manage frontend libraries and using a task runner as Grunt was very useful to automatize build flows and tasks like minification, running tests, previewing project changes and distributing the project. When building the project with Yeoman Bootstrap was also included. Bootstrap provided useful design templates, icons and much more. I used AngularJS as frontend development framework. This framework provided a good well-known and tried MVC-architecture, routing, services, testability, directives and modules as a
few examples that made the development process easier, quicker and provided structure.

Using HTML5 video it was very easy to implement the playback functionality for the system. The only required thing was the HTML5 video tag as well as to provide a source for the video file. There are also a lot of third party HTML5 video players that can be used to play video files.

6.5 Brief summary

This chapter starts of by presenting the connection between literature, conducted interviews and observations when doing the frontend development at the internship company. It is then described what UX activities were found to be the most important and how UX activities and Agile software development can be combined. The last part of this chapter presents findings about frontend development.
Chapter 7 Discussion

7.1 Method

The chosen method enabled validation of the findings in this report in a good way. By combining results from theory, interviews and active observations the result presented in this paper had a robust foundation. It would be interesting to see this case study replicated and used in a different project to see if the results would be the same. It would also be interesting to see what the results would be if a different methodology like Lean UX was used.

If this research would be conducted again I would probably suggest conducting interviews with people from a number of different companies. The interviews conducted during this thesis were all with employees from the same company. Even though the respondents were all consultants with experience from consulting at a number of different companies a different perspective and different experiences might be found when interviewing people employed at other companies.

The internship project was conducted using a small development team. It would have been interesting to do the case study using a larger development team and over a longer period of time.

I worked both as the UX designer and frontend developer while at the same time writing on this thesis report. Even though it was interesting to take active part in the case study it would also have been interesting if the observations were made by someone not directly included in the development process. This might provide the researcher with more objective material and this way the researcher might have found things that were missed because of the active participation in the development work.

7.2 Results

The developed system had an increasing SUS value for each sprint it was measured and at the end of the project it ended at a very high level. The users of the system were employees at an IT consulting firm though with fairly good computer skills. As can be seen in Figure 5.59 most of the employees participating in usability testing sessions were quite familiar with video streaming services from before.
Deploying the system in a different organization having less computer skills and experience of streaming services might have resulted in a lower SUS value. However, as can be seen in Figure 5.58, I used employees from all parts of the company (HR, recruiters, sales as well as developers and designers) for the usability testing sessions. This was done to give an as fair value for the UX as possible.

During the internship project the researcher had good access to users to conduct usability tests on. The usability testing sessions provided a vital source of feedback from the end users. During the interviews finding users to perform usability tests was said to be hard for some projects, which caused problems for the project. If finding users to test the implemented features on had been hard during the internship project, I think this would have made it much harder to ensure good UX. If this was the case alternative users would have had to be found and these users might not have provided as good feedback as we got when performing usability tests with the actual users of the system during the internship project.

### 7.3 Work in wider perspective

The developed streaming platform makes it possible for the employees at the internship company to take part in knowledge sharing events through on demand video streaming even though they are not physically present at this event. This is very beneficial for the company, but it might also result in less employees meeting up at these events when they can watch the event later on instead in the comfort of their home for instance. This is an unwanted effect because it might result in less employee-to-employee interaction and it is possible that less internal group discussions might occur when the employees are not physically being present at the knowledge sharing event. It is therefore important for the internship company to “sell” the system as a something to use when an employee is unable to attend an event or when an employee wants to see the presentation again, rather than a system to use instead of attending a knowledge-sharing event.

Making videos available for the whole company increases the communication and knowledge sharing for all employees across offices and countries. This makes it possible for an employee holding an event to record it and then distribute it on the platform. This reduces the need for having to hold the event several times and on several places, something that is practiced today. Having this opportunity therefore
reduces the need for employees to travel between countries to hold knowledge sharing events and thereby might also reduce negative environmental effects and also reduce travelling costs.

7.4 Further research

As mentioned in chapter 7.1 I felt that using Dual-track Scrum might not have been the best option for development methodology. It would have been interesting if a similar case study was performed but using a different methodology, for instance Lean Startup or Lean UX, and see if the results would be better.

I only tried a few UX activities and one UX activity that I did not try out, A/B-testing, seems to be quite interesting. There are a lot of new tools available that makes A/B-testing simpler and effective.

Another interesting field of study is how statistics can be used to measure and get insights on the UX. A few of the respondent used statistics to for instance analyze user flows in their systems and I think this is an interesting field of study and that there are many possibilities for this to be further investigated.
Conclusion

For the frontend design of the web-based streaming platform it’s important to start by setting up a high-level architecture for the whole system and to involve all team members in this process. This to have a common understanding and goal when it’s time to start the development work. Another important thing to consider is to structure and develop the system in modules to make it easier to reuse or replace the modules if needed. When designing the frontend, it’s also very important to make use of UX activities such as sketches, wireframes and prototypes as well as following web design guidelines and best practices.

When implementing the streaming platform, it was concluded that using a scaffolding tool is a great way of generating the basic structure of a project and enables the development work to start much quicker. Another important factor for simplifying and gain structure when doing frontend development is to use a frontend framework. For the thesis project AngularJS was used and this framework provided a clear system architecture, routing, services, directives and testability. The HTML5 video element made video playback easy to implement and add to the system.

Combining Agile software development and UCD is a very good way of both developing working software as well as software that has high quality UX. However, it has been concluded in this thesis that to combine these two processes is not that easy to do in practice. The main problem is that the two processes are different in the aspects of goal, the view on users and documentation. To solve these problems and make the combination successful the following guidelines have been found: Including UX designers in Agile teams, a close communication between UX designers and developers, just enough documentation, a continuous user involvement, metrics for measuring the UX and using combined an UCD-Agile methodology.

In this thesis it has been investigated what UX activities to include in an Agile development process, and the most important ones were considered to be sketches, wireframes, prototypes and usability testing. Sketches, wireframes and prototypes are good tools to use when starting to design a new feature and it’s beneficial if the whole team is included in this process to get a common understand and goal as well
as to at an early stage find potential problems. Usability testing should be practiced from the beginning and then continuously throughout the development process. It’s important to perform usability testing on sketches/prototypes before starting the development work to make sure that you don’t develop features that the user does not want as well as afterwards to make sure that the developed feature is in line with the needs of the user. The best way to conduct usability tests is by physically being present with the user and gives the user tasks to perform in the system and evaluate the result.

In this thesis usability testing sessions was used to collect input on and measuring the UX of the system. Using different metrics during these sessions gave a good indication about whether the user was able to complete the defined tasks, what problems the user had as well as provided an overall score for the UX of the system. By continuously conducting usability tests both on newly implemented features as well as including old ones it was possible to measure and get a good indication on how the UX changed over time as the development project progressed. Other ways of measuring the UX is to perform user surveys on implemented features and to have tracking in the system to collect information about user behavior that later on can be analyzed.
References


[58] Valett, S. Frontend developer/Interaction designer Netlight Consulting AB.
Interviewed 16th May 2015.


Appendix A

Case Study interview questions

1. What is your current role at the company?
2. How have you worked with UX in previous projects?
3. What are your reflections on combining Agile and UX?
4. How should people working with UX be included into the development team?
5. Which UX activities do you think are the most important ones and how do you use them in your development process?
6. What development methodology do you currently use?
7. What are your thoughts on methodologies made to combine Agile and UX/UCD (Dual-track Scrum, Lean Startup, Lean UX)?
8. How should the user be included in the development process?
9. How do you measure the UX of the system/product over time?
Appendix B

SUS questions

1. I think that I would like to use this system frequently.
2. I found the system unnecessarily complex.
3. I thought the system was easy to use.
4. I think that I would need the support of a technical person to be able to use this system.
5. I found the various functions in this system were well integrated.
6. I thought there was too much inconsistency in this system.
7. I would imagine that most people would learn to use this system very quickly.
8. I found the system very cumbersome to use.
9. I felt very confident using the system.
10. I needed to learn a lot of things before I could get going with this system.
På svenska

Detta dokument hålls tillgängligt på Internet – eller dess framtida ersättare – under en längre tid från publiceringsdatum under förutsättning att inga extra-ordinära omständigheter uppstå.

Tillgång till dokumentet innebär tillstånd för var och en att läsa, ladda ner, skriva ut enstaka kopior för enskilt bruk och att använda det oförändrat för ickekommersiell forskning och för undervisning. Överföring av upphovsrätten vid en senare tidpunkt kan inte upphäva detta tillstånd. All annan användning av dokumentet kräver upphovsmannens medgivande. För att garantera äktheten, säkerheten och tillgängligheten finns det lösningar av teknisk och administrativ art.

Upphovsmannens ideella rätt innefattar rätt att bli nämnt som upphovsman i den omfattning som god sed kräver vid användning av dokumentet på ovan beskrivna sätt samt skydd mot att dokumentet ändras eller presenteras i sådan form eller i sådant sammanhang som är kränkande för upphovsmannens litterära eller konstnärliga anseende eller egenart.

För ytterligare information om Linköping University Electronic Press se förlagets hemsida http://www.ep.liu.se/

In English

The publishers will keep this document online on the Internet - or its possible replacement - for a considerable time from the date of publication barring exceptional circumstances.

The online availability of the document implies a permanent permission for anyone to read, to download, to print out single copies for your own use and to use it unchanged for any non-commercial research and educational purpose. Subsequent transfers of copyright cannot revoke this permission. All other uses of the document are conditional on the consent of the copyright owner. The publisher has taken technical and administrative measures to assure authenticity, security and accessibility.

According to intellectual property law the author has the right to be mentioned when his/her work is accessed as described above and to be protected against infringement.

For additional information about the Linköping University Electronic Press and its procedures for publication and for assurance of document integrity, please refer to its WWW home page: http://www.ep.liu.se/

© Fredrik Salin