

Camilla Velvin

# Application for sharing books within second home areas

Master's thesis in Industrial Cybernetics

Supervisor: Sverre Hendseth

June 2022

**NTNU**  
Norwegian University of Science and Technology  
Faculty of Information Technology and Electrical Engineering  
Department of Engineering Cybernetics



Camilla Velvin

# **Application for sharing books within second home areas**

Master's thesis in Industrial Cybernetics  
Supervisor: Sverre Hendseth  
June 2022

Norwegian University of Science and Technology  
Faculty of Information Technology and Electrical Engineering  
Department of Engineering Cybernetics





# Preface

This master's thesis is written as part of the Master's Degree Programme in Industrial Cybernetics at the Norwegian University of Science and Technology (NTNU). The project was conducted in the spring of 2022 and is based on the specialization project accomplished fall of 2021.

I wish to thank all the test participants that took the time to participate in the usability tests. Thank you for the feedback and input. I would also thank my supervisor Sverre Hendseth for an exciting assignment, insightful input, and guidance during this instructive period. Finally, I want to thank family, friends, and *kanelbolleonsdag* for listening to ideas, contributing valuable feedback, and motivating.



---

Camilla Velvin

May 27, 2022  
Trondheim



# Abstract

Books are a large part of Norwegians' life. It is approximated that 25% of the population reads daily. To face the constant need for "new" books and considering the climate changes, an application for book sharing, Hyttebiblioteket, is created. This application is created as a supplement to the existing solutions and differs by focusing on sharing books within second home areas.

The project, in its entirety, has consisted of two parts, a specialization project [1] and this master's thesis. To realize the idea of book sharing within second home areas, the specialization project started by determining the requirement specifications for a web application. Then, with the knowledge gained from interactions and literature review, a web application for Hyttebiblioteket was developed.

In this master's thesis, a mobile application for Hyttebiblioteket is created. This application is created to improve Hyttebiblioteket. The master's thesis started by conducting usability tests on the web application created in the specialization project. These usability tests, literature reviews, and personal experiences were the basis when the mobile application requirements were defined. The app was then designed and developed. The developed app is a cross-platform application developed using Ionic with Capacitor. A cross-platform application is an application created for mobile devices, that function with both Android and iOS. The backend solution for the application is created by using Firebase as Backend-as-a-Service (BaaS) instead of creating a dedicated backend. At the end of the project, another usabil-

ity test was conducted. This test contained five different participants, and its objective was to test the developed mobile application and its functionality. The results were then analyzed and evaluated, and used to suggest changes to make a more usable application. The usability test results also showed that the developed application could be used to share books within second home areas and possibly evolve to other areas.

At the current time, the mobile application, *Hyttebiblioteket*, is a functioning software ready for release. However, based on the costs of releasing a mobile application on App Store and Google Play, the release was postponed.

# Sammendrag

I dag er bøker en stor del av nordmenns liv, og det er tilnærmet 25% som leser daglig. For å møte det konstante kravet til ”nye” bøker, og samtidig tar klimaendringene i betraktning, er det laget en applikasjon for bokdeling, Hyttbiblioteket. Denne applikasjonen er laget som et supplement til de eksisterende løsningene, og skiller seg ut ved at det hovedsakelig fokuseres på bokdeling i hyttefelt.

Prosjektet har i sin helhet bestått av to deler, et fordypningsprosjekt [1] og selve masteroppgaven. For å realisere ideen om bokdeling i hyttefelt, startet fordypningsprosjektet med å definere krav til en webapplikasjon. Deretter, ble det med bakgrunn i litteratursøk og interaksjon med potensielle brukere laget en webapplikasjon for Hyttbiblioteket.

I denne masteroppgaven har det blitt laget en mobilapplikasjon for å forbedre Hyttbiblioteket. Masteroppgaven startet ved å gjennomføre brukervennlighetstester av webapplikasjonen laget i fordypningsprosjektet. Resultatene fra disse brukertestene, litteratursøk og personlig erfaring ble grunnlaget for kravspesifikasjonen til mobilapplikasjonen. Mobilapplikasjonen ble deretter designet og utviklet. Denne mobilapplikasjonen ble laget som en kryssplattform applikasjon ved bruk av Ionic og Capacitor. En kryssplattform applikasjon vil si at applikasjonen fungerer på mobile enheter med både Android og iOS. Applikasjonen som er laget bruker Firebase som Backend-as-a-Service (BaaS), noe som vil si at det ikke ble laget en dedikert tjenerløsning. I slutten av prosjektet ble en ny brukervennlighetstest gjennomført. Denne testen bestod av fem ulike deltakere, og målet med testen var å

teste mobilapplikasjonen som var laget. Resultatene fra denne testen ble deretter analysert og evaluert, før det ble presentert mulige endringer for bedre brukervennlighet. Resultatene fra brukertestene viste også at den utviklede applikasjonen kunne bli brukt for å dele bøker i hyttefelt, og senere, også i andre områder.

På nåværende tidspunkt er mobil applikasjonen, Hyttebiblioteket, en vel-fungerende programvare klar for publisering. Men, på bakgrunn av kostnadene for publisering av mobil applikasjoner på App Store og Google Play, er det valgt å utsette publiseringen.

# Contents

<b>Preface</b> . . . . .	<b>iii</b>
<b>Abstract</b> . . . . .	<b>v</b>
<b>Sammendrag</b> . . . . .	<b>vii</b>
<b>Contents</b> . . . . .	<b>ix</b>
<b>Figures</b> . . . . .	<b>xiii</b>
<b>Acronyms</b> . . . . .	<b>xv</b>
<b>1 Introduction</b> . . . . .	<b>1</b>
1.1 Background and motivation . . . . .	1
1.2 Problem description . . . . .	2
1.3 Thesis outline . . . . .	2
<b>2 Theory</b> . . . . .	<b>5</b>
2.1 Existing solutions . . . . .	5
2.1.1 Finn . . . . .	5
2.1.2 Little Free Library . . . . .	6
2.1.3 BookCrossing . . . . .	7
2.1.4 Booksharing.app . . . . .	8
2.2 Software development methodology . . . . .	8
2.3 Database . . . . .	11
2.4 Backend-as-a-Service . . . . .	12
2.5 User experience . . . . .	12
2.5.1 Interaction design . . . . .	13
2.5.2 Usability testing . . . . .	15
2.5.3 User interview . . . . .	16
2.6 Mobile application . . . . .	17
2.7 Application monetization . . . . .	17

2.8	The specialization project . . . . .	19
2.8.1	The project's objective . . . . .	19
2.8.2	Method . . . . .	19
2.8.3	Results . . . . .	20
<b>3</b>	<b>Requirement specification . . . . .</b>	<b>23</b>
3.1	Functional requirements . . . . .	23
3.2	Non-functional requirements . . . . .	29
<b>4</b>	<b>Implementation and design . . . . .</b>	<b>31</b>
4.1	Choice of software development methodology . . . . .	31
4.2	User interface . . . . .	34
4.2.1	Color selection . . . . .	34
4.2.2	Inspiration for the user interface . . . . .	34
4.2.3	Prototyping . . . . .	35
4.3	Usability tests . . . . .	36
4.3.1	Test participants . . . . .	36
4.3.2	Preparations and conducting . . . . .	37
4.4	Technical implementation . . . . .	39
4.4.1	Development framework . . . . .	39
4.4.2	Libraries . . . . .	41
4.4.3	Backend solution . . . . .	43
4.5	Chosen app monetization strategy . . . . .	46
4.6	Deployment . . . . .	47
4.6.1	Android studio . . . . .	47
4.6.2	Xcode . . . . .	47
4.6.3	Release of the application . . . . .	48
<b>5</b>	<b>Results . . . . .</b>	<b>49</b>
5.1	Status of application at delivery time . . . . .	49
5.1.1	Design of the application . . . . .	49
5.1.2	Achievement of functional requirements . . . . .	64
5.1.3	Achievement of non-functional requirements . . . . .	66
5.2	Usability tests . . . . .	67
5.2.1	First usability tests, testing of the web application . . . . .	68
5.2.2	Second usability tests, testing of the mobile application . . . . .	72



<b>6 Discussion</b>	<b>77</b>
6.1 Analysis of usability test results	77
6.1.1 Software bugs	77
6.1.2 Critical problems	78
6.1.3 Serious problems	79
6.1.4 Minor problems	80
6.2 Evaluation of process	81
6.2.1 Design phase	81
6.2.2 Development phase	81
6.2.3 The accomplishment of usability tests	82
6.3 Evaluation of the technical implementation	83
6.3.1 Usage of Ionic and Capacitor	83
6.3.2 Backend	83
6.4 Evaluation of the product	84
<b>7 Conclusion</b>	<b>87</b>
<b>8 Further work</b>	<b>89</b>
<b>Bibliography</b>	<b>91</b>
<b>A Digital attachment</b>	<b>105</b>
<b>B User guide for downloading the application on an Android device</b>	<b>107</b>
<b>C User guide for running the application in a simulator</b>	<b>109</b>
C.1 Environment setup on machine	109
C.2 Clone the repository from Github	109
C.3 Run code	110
<b>D Results of first usability test</b>	<b>111</b>
<b>E Results of second usability test</b>	<b>123</b>



# Figures

2.2	A Little Free Library converted from an old dollhouse [10].	7
2.3	Waterfall- and agile methodology with different phases . . .	11
2.4	Field of User experience (UX) design [42, p. 21]. . . . .	13
2.5	Implemented interface in specialization project . . . . .	22
4.1	General milestones for the project, created in TeamGantt [78].	32
4.2	Activities for two weeks in Trello [79]. . . . .	33
4.3	Chosen color palette from Colors [81]. . . . .	34
4.4	Inspiration for Hyttebiblioteket user interface . . . . .	35
4.5	A simplified model of how a mobile application with Capacitor and web app is built. [102]. . . . .	41
4.6	Structure in Cloud Firestore [114] . . . . .	44
4.7	The data structure in Cloud Firestore for Hyttebiblioteket .	45
4.8	Diagram for interaction between client and BaaS for login and saving a new book. . . . .	45
5.1	Front page of the app. A page containing a slider with information explaining the concept of the application, registration and log in. . . . .	50
5.2	Registration page for the mobile application. . . . .	51
5.3	Login page for the mobile application. . . . .	52
5.4	A page for forgotten passwords in the mobile application. .	52
5.5	List of available books in the mobile application. . . . .	53
5.6	Map with available books in the mobile application. . . . .	54
5.7	Books registered or posted by the user. . . . .	55
5.8	Page to register a new book in the mobile application. . . .	56

5.9	Page to edit a registered book. . . . .	57
5.10	Information about a specific book. . . . .	58
5.11	Page with an overview of borrowed books. . . . .	59
5.12	Page showing information about a specific borrowed book. . . . .	60
5.13	Home page in the mobile application . . . . .	61
5.14	Open modals on the home page. . . . .	63
5.15	The application in dark mode. . . . .	64
5.16	Some screenshots from Hyttebiblioteket on an iPhone 12. . . . .	67

# Acronyms

**BaaS** Backend-as-a-Service. v, vii, xiii, 12, 20, 43, 45, 83

**CSS** Cascading Style Sheets. 41, 42

**HCI** Human-Computer Interaction. 13

**HTML** HyperText Markup Language. 41, 42

**IDE** Integrated development environment. 47

**IxD** Interaction design. 12–14

**MVP** minimum viable product. 19

**NoSQL** Non Structured Query Language. 11, 12, 44

**SQL** Structured Query Language. 11, 12

**UX** User experience. xiii, 12, 13, 15



# Chapter 1

## Introduction

### 1.1 Background and motivation

This master's thesis is based on the web application created in the specialization project, fall 2021 [1], and addresses Hyttebiblioteket. Hyttebiblioteket is an idea concerning book sharing within second home areas. This idea originates from the climate changes the world is facing. By creating a solution for sharing books, we will achieve a more circular economy and make a small contribution to the climate. Books are a large part of Norwegian lives, and Statistics Norway states that 25% of Norwegians read printed books on an average day [2]. A solution for sharing books will therefore reduce the amount of newly printed books, resulting in an increase in the capture of carbon dioxide as the usage of wood pulp is reduced, and the forests can be left standing [3]. Furthermore, when the forests are reserved, it will also preserve the biodiversity which is essential for us humans due to, for example, pollination [4].

Today, a market for book sharing exists, but demand in the market was noticed from experience with the different solutions. The motivation for this application has been to create a book-sharing application aimed at second home areas, as my family and I have several times been sitting at our second home longing to read a "new" book. A web application to handle this demand was created in the specialization project. However, it was

in this master's thesis desirable to create a mobile application for accessibility and user experience as 88% of mobile users' prefers mobile apps over mobile websites [5].

## 1.2 Problem description

This project overall goal was to help the planet to capture more carbon dioxide by reducing the use of wood pulp for producing books, by using elements from the circular economy. To make this happen the main idea is to create an application, so book readers could share books instead of buying new ones. The mobile application made in this project is an improvement of the idea "Hyttebiblioteket" that evolved in the specialization project in the fall of 2021.

Through the subtasks outlined below, the mobile application for sharing books within secondary home areas was created.

1. Plan and implement usability tests for the web application created in the specialization project. Then analyse and evaluate these results to define the functional requirements and the design.
2. Specify the functional and non-functional requirements for the application.
3. Explore different software development methodologies for use in a development project.
4. Study the development of mobile applications and find the best approach for the given requirements.
5. Design, develop and implement a mobile application for sharing books within second home areas with the given specific requirements.
6. Plan and implement usability tests of the developed application.

## 1.3 Thesis outline

This master's thesis contains of eight chapters with associated subchapters.



- The first chapter introduces the background for the assignment and the problem description.
- In the second chapter the theory behind the assignment is presented.
- In the third chapter the requirement specification are specified.
- In the fourth chapter, the choice of technology and method used in the assignment are justified.
- The fifth chapter presents the usability test results and the status of the application at delivery time.
- In the sixth chapter, the different results are discussed, and also divided into four subchapters:
  1. Evaluation of process
  2. Evaluation of the technical implementation
  3. Analysis of usability test results
  4. Evaluation of the product
- The seventh chapter concludes the accomplishment of the problem description.
- In the eighth chapter a view of the requirement for broaden out the system is presented.



# Chapter 2

## Theory

In this chapter the existing solutions and how they differ from my problem is presented. Later, necessary theory for developing an application that contribute to the the climate with elements from circular economy will be presented.

### 2.1 Existing solutions

Today it exists several different digital solutions for lending and borrowing books. The following presents and examines some of these solutions.

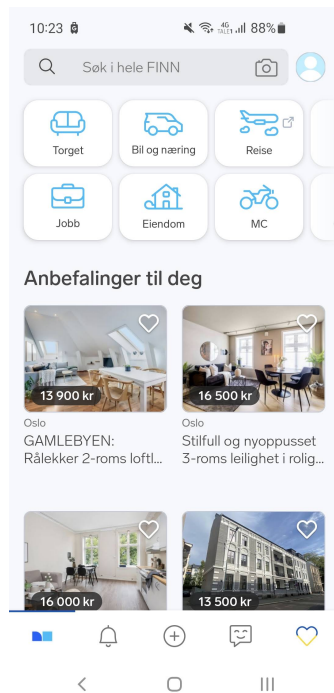
#### 2.1.1 Finn

In March 2000 *Finn.no AS* was established. Finn.no has specialized in classified advertisements and services for purchase and sale between private persons and small and large companies [6].

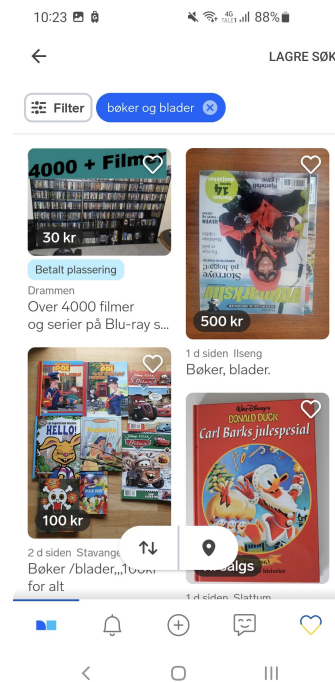
Initially, Finn launched as a web application, but in 2012 they also launched an mobile application for iOS and Android. Today, Finn is the largest marketplace in Norway and has around a million searches per day and about seven million visitors every week [7].

When entering Finn, the page presents the user with different categories

for ads (Figure 2.1a). This way, the user can find relevant ads (Figure 2.1b). The user can then scroll through the available ads, and if something is interesting, they can contact the person publishing the ad to arrange hand over and perhaps payment.



(a) The main page of Finn.no.



(b) Choice of products given at the search of "books and magazines".

Finn is available for people over 15 years for selling and purchasing, and it is free for private persons to publish ads for Finn Market Place (Torget). When publishing an ad, the user can choose the price they want for their product. This way, the products at Finn vary in price, and some products might even be free as giveaways.

### 2.1.2 Little Free Library

In 2009 the first Little Free Library was built in Hudson, Wisconsin. The Little Free Library was a bookcase built as a tribute to the mother of the founder Todd Bol, as his mother loved to read. The built bookcase was filled with books, and everyone who wanted could take or leave a book.

The idea evolved rapidly, and in 2010 the name Little Free Library was established to share books and bring communities together [8].

Today there exist more than 100 000 Little Free Libraries around the world [9]. This means that more than 100 000 bookcases or other containers (see Figure 2.2) are placed in different locations with the purpose of people taking and leaving books.



**Figure 2.2:** A Little Free Library converted from an old dollhouse [10].

Everyone can establish a new Little Free Library by creating a container for books and registering it online. By registering the container, the mobile app or web page will help other users locate the container.

### 2.1.3 BookCrossing

In 2001 Ron Hornbaker got the idea of BookCrossing. This idea is that people can place books in any public place in the way that other users can find the books and read them. When a person finds a BookCrossing book, they should register it online at BookCrossing.com. This way, it is possible to follow the book's wandering, where it has been, and opinions about the book [11].

Since the beginning, BookCrossing has expanded the way to share books. Now BookCrossing.com also offer the users to post their books on the website and others to browse the posted books and contact the owner for hand-over or sending of the book.

BookCrossing is a popular website, there are currently over 1.9 million users, and in 2005 it was awarded People's voice for best community websites and mobile sites [12].

#### **2.1.4 Booksharing.app**

*The following paragraph is copied from the specialization project [1, p. 2]*

Another solution for book sharing under development is Booksharing.app. The concept here is that users can post printed books that they no longer need, and by doing so they will earn "Bookcoins". When a user has earned Bookcoins, he/she gets access to other people's books in the city. When finding an exciting book, the user can contact the book holder and send a request to take the book. If the book holder confirms the request, it can be arranged a meeting and handing off the book, as well as payment using Bookcoins [13].

## **2.2 Software development methodology**

In software development projects, the methodology refers to planning, developing, testing, and deploying a project to create well functional software [14]. The methodologies contain a collection of procedures, techniques, tools, and documentation aid that helps improve the management and control, and decrease the complexity of the software development process [15].

Several different software development methodologies exist, and new ones are represented rapidly. As each project has a different goal and scope, no methodology fits all, and each methodology will therefore contain advan-

tages and disadvantages for the software project.

The two most famous categories underneath software development methodologies through time are *Traditional methodology* and *Agile methodology*. The traditional methodology is the oldest method, and the most famous is the *Waterfall method* [16]. The traditional methodology is still used, but other methodologies have dominated the software development processes in recent times [17]. The basis of the traditional- and waterfall method is a sequential development process. A sequential approach divides the software development life cycle into phases. Specific to the waterfall method, one phase can start after the previous phase is completed with no overlap between the phases [18]. The different phases and the order of them are the following (see Figure 2.3a) [19, 20]:

1. Requirements - communicate with users and find the requirements for the project.
2. Design - by using the requirements, the system is designed. In addition, the specification regarding programming language or hardware is established.
3. Implementation - write the code for a functional product by taking information from the previous phases.
4. Testing - systematically testing the completed code and reporting all errors.
5. Maintenance - fix problems and mistakes found by the customer after the application is rolled out.

The advantage of the waterfall method is that it uses clear and defined steps compared to other methodologies. The precise steps make the methodology easy to understand for developers and customers. It is also easy to create milestones, and delivery timelines with rigid and well-defined phases, and a well-documented process and goals [21]. Waterfall method,

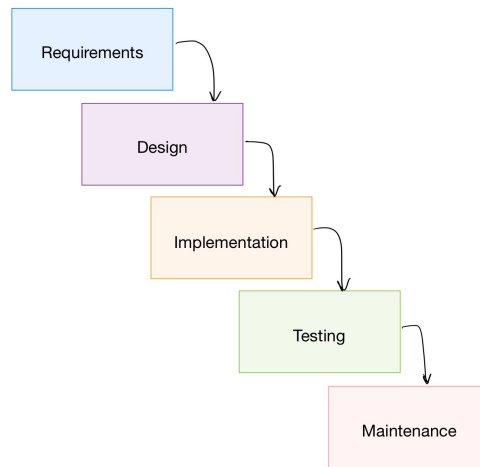
therefore, fits short projects where the product definition is stable [22]. The waterfall method's disadvantage is that the sequential approach does not have room for unexpected changes [23]. Furthermore, as most current projects are more extensive, complex, and constantly evolving, it is harder to define all requirements in advance. As a result the traditional, linear methods are replaced by agile, iterative methods [24].

Agile methodology is a definition for many iterative and incremental software development methodologies that share common principles and practices [25]. Since the 1940s, agile- and lightweight methods have been used to a small extent [26]. However, in the 1990s, the methodology was widely accepted among mainstream software developers and became more popular [24]. In 2001 a group of well-known software developers created the *Agile Manifesto*. The Agile Manifesto contradicts the traditional linear models and their focus on planning and documentation [27]. From software development experience, they changed the management process by guiding the projects based on the following values [28]:

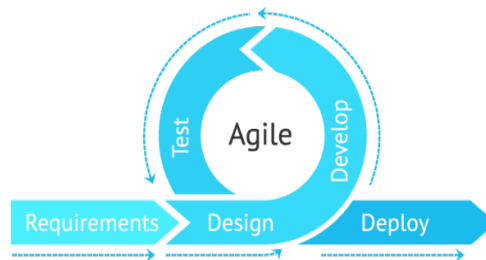
- Individuals and interactions over processes and tools.
- Working software over comprehensive documentation.
- Customer collaboration over contract negotiation.
- Responding to change over following a plan.

Agile methodology's highest priority is to satisfy the customers with rapid and continuous delivery of small and useful software [24]. This way, there is room for feedback and changes to evolve and develop the product as new iterations are delivered (See Figure 2.3b).





(a) Waterfall methodology [20].



(b) Agile methodology [29].

**Figure 2.3:** Waterfall- and agile methodology with different phases

## 2.3 Database

*The next paragraphs is copied from specialization project and adjusted due to relevance [1].*

A database is a structured collection of data, generally stored and accessed from a computer. The primary purpose of a database is to deliver accurate and consistent data to the users and secure the stored data [30]. Today there exist several types of databases. The main two are relational (*SQL*)- and non-relational (*NoSQL*) databases.

A relational database is based on the relational model where the data collections are organized into tables with predefined relationships [31]. These

tables use a primary key to uniquely identify an element and find the relations between the different tables by adding a foreign key, the primary key of the related element [32]. Structured Query Language (SQL) is a language supported by all relational databases and a tool for managing the database and performing different operations on the data [33].

On the other side, we have non-relational databases. Instead of storing data in tabular schemes, these databases optimize the storage model based on the requirements of the type of data [34]. This indicates that a project with flexible data models fit better in a NoSQL database, while SQL fits better with structured data. Because of the ability to scale NoSQL databases, it is often used in *Big Data* applications. In addition, can some queries become faster with NoSQL.

## 2.4 Backend-as-a-Service

In mobile applications, cloud-hosted solutions called Backend-as-a-Service (BaaS) have frequently been used in the last couple of years [35]. The idea with BaaS is that the solution provides all necessary backend solutions, like authentication, storage, and insight [36]. This simplifies the development process, and the developers can use more time improving the application instead of using time and energy to plan, create and maintain a custom backend.

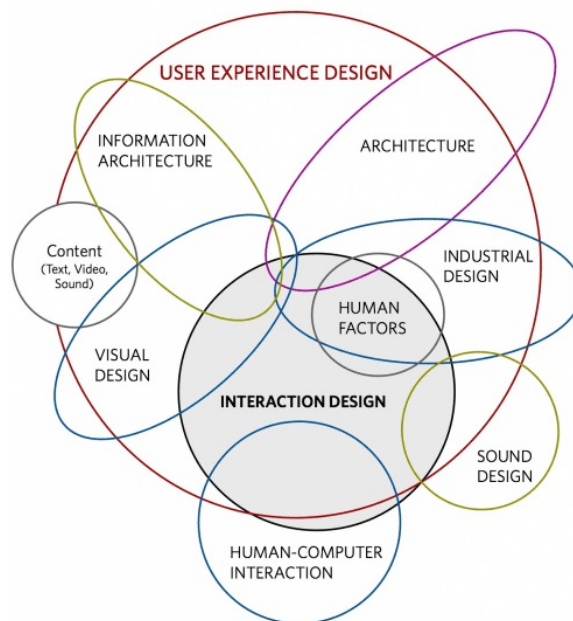
The top cloud providers, Amazon, Google, and Microsoft [37], offer similar BaaS. For example, Amazon offers *AWS Amplify*, Microsoft offers *Azure Mobile Apps Service*, and Google offers *Firebase* [38].

## 2.5 User experience

User experience (UX) is a term referring to how a product is experienced and used [39]. UX design is a large field often misinterpreted as Interaction design (IXD). Even though there is a significant overlap between the two

fields (see Figure 2.4), the crucial difference is that IxD focuses on user interactions while UX accounts for all user aspects [40].

A vital field relating to UX is Human-Computer Interaction (HCI). HCI is the study regarding how humans interact with computers. The discipline HCI overlaps a lot of other disciplines, and contains methods for developing and designing new interfaces, techniques for evaluating existing interfaces, and theoretical models and theories of Human-Computer Interaction [41].



**Figure 2.4:** Field of User experience (UX) design [42, p. 21].

### 2.5.1 Interaction design

Interaction design (IxD) is the practice of designing interactive digital products and services [40]. More specific, John Kolko describes it as [43]:

*Interaction Design is the creation of a dialogue between a person and a product, system, or service. This dialogue is both physical and emotional and is manifested in the interplay between form, function, and technology as experienced over time.*

This description means that IxD is a design science focusing on understanding the user interaction and how to design systems and objects to achieve the best user- interaction and experience. To achieve the best user interaction is concrete methods, techniques, and frameworks used [44]. Therefore, the field interaction design is significant for developing applications even though it is relatively new and constantly evolving.

The process of Interaction design involves four basic activities [45, p. 50]:

1. Establishing requirements
2. Designing alternatives
3. Prototyping
4. Evaluating

These activities intend to inform one another and be repeated, indicating that IxD is an iterative process. The process ensures that the system gets feedback and can improve the user- interface and experience. A good user interface is essential as a lack of usability will cause frustration, and the system will not get used [44].

For the best user- interface and experience, there are different design principles one should follow [44–46]:

- **Visibility.** The functionality of a system must be visible to show that they are present and how it is used. In contrast, when functions are out of sight, it is hard for the user to find and figure out how to use them.
- **Feedback.** To understand what is going on, the system should provide information to the user. This information should contain the state of the system. For example, if a user presses a button, the system should indicate that the press is registered.
- **Constraints.** This principle refers to the concept of constraining the user interactions at a given moment. This way, the system limits errors made by the user. An example of constraints in a system is that the system greys out options that are not available.
- **Consistency.** Is the idea of creating similar things in similar ways

such that users save time and thinking, and the system gets easier to learn and use.

- **Affordance.** Is the interaction the user feels that he/she can perform in the system. These ideas are based on the system's design and the users' experiences with other systems. For example, a door handle affords to pull or twist.

In Norway, the government has also stated in the Norwegian act relating to equality and a prohibition against discrimination (§§17 to 19) that all systems regarding information and communication technology (ICT) must follow the universal design [47]. Therefore, websites and mobile applications must follow the minimum requirements of the Web Content Accessibility Guidelines (WCAG) standard 2.0 [48].

## 2.5.2 Usability testing

Usability testing is important for designing good User experience (UX). It is said that even the best UX designers cannot design a good enough user experience without iterative design driven by user interactions [49]. There is no official definition to characterize usability testing. Steve Krug, an expert on UX defines usability testing as [50, p. 13]:

*Watching people try to use what you're creating/designing/building (or something you've already created/designed/built), with the intention of (a) making it easier for people to use or (b) proving that it is easy to use.*

A usability test usually identifies problems in the design, shows what users do instead of what the developers think they will do, and helps the developers with new inputs and ideas [51]. It is normal to categorize usability testing into two categories; quantitative and qualitative testing.

With quantitative usability testing, participants perform critical tasks in the system while specific metrics that describe the user experience and performance are collected [52]. The goal of quantitative usability tests is

to find the usability of a design by collecting measurable data such as success rate, time-on-task, or the number of errors [53]. Therefore, this form of testing has a well-defined, structured test protocol that needs to be followed consistently for all participants. To conclude statistically from this form of testing, it requires a more significant amount of participants [50, p. 13].

A qualitative usability test consists of observational findings that identify design features that are easy or hard to use [53]. These usability test findings help with insight into how the product can improve. As the goal is not to collect measurable data, the qualitative usability tests can be more informal, and the test can change protocol mid-test if necessary [50, p. 14]. In addition, these tests require fewer participants, where each participant gets assignments he/she should solve. At the same time, should the participant tell about his/her interaction with the application, a so-called think-aloud protocol, such that the observer can note the participant's thoughts [50, p. 14].

### 2.5.3 User interview

An interview is a way to collect information about users' needs and wishes, and learn what a user is trying to do, what is working and not [54]. There are different ways to conduct an interview. The interviews are, based on their structure, categorized into three categories [55]:

- **A structured interview** is where everything is planned. The interviewer has a list of questions and writes the answer to each question. The advantage of this interview form is that all users get the same questions.
- **A semi-structured interview** has some questions planned in advance but the interviewer can follow up with questions based on the user's answers. This form of interviews can help the interviewer gain new information.
- **An unstructured interview** is where the interviewer typically has a

chosen theme and then lets the conversation go freely.

## 2.6 Mobile application

A mobile application, usually called a mobile app, is designed to run on a mobile device such as a smartphone or a tablet [56]. The main difference between mobile- and web apps is that mobile apps are normally downloaded and installed through an app store. These apps could have access to the system resources, such as GPS and camera. In contrast, web applications are accessed via the internet browser [57]. Mobile apps also have the advantage of being faster, but require the user to download updates while web apps update themselves.

Today, most smartphones use Android from Google or iOS from Apple Inc. as their operating system [58]. As a result, the two most common ways to develop a mobile app is by using a native or a cross-platform framework [59]. With native development, the app is created and optimized for a specific operative system. For example, to create an app for iOS, one can either use Objective-C or Swift as a programming language, while for Android, it is most common to use either Java or Kotlin [60]. Cross-platform development is creating an app with the same codebase for several platforms. Cross-platform applications can be developed using tools like React Native, Ionic, Flutter, or NativeScript, and the application developed can be deployed on both iOS and Android [61].

## 2.7 Application monetization

Monetizing an application refers to how a mobile app earns money [62]. There are different strategies to generate revenue from a mobile app. The most used strategies are "in-app advertising", "in-app purchases" and "paid download"[63].

Using an advertising-based model, it is vital that the advertisement is rel-

evant and targeted to the users. With irrelevant, low-quality, or intrusive ads, the ads can cause harm to the user experience resulting in a lack of engagement and usage of the app [64]. Typically, it is used four different types of in-app advertisement [64–66]:

- **Interstitial ads.** These ads cover the full screen and appear at natural pause points, for example, when the site is loading.
- **Native ads.** Native ads are ads that is integrated into the app's interface. These ads usually look like another post in the application.
- **Reward ads.** The reward ads are popular in games. These ads offer the user a reward for watching or interacting with it.
- **Banner ads.** Usually found at the top or bottom of the page are the banner ads. These ads use text, images, and pop-ups to draw the users' attention.

In the "in-app purchase" model, the application is free but allows users to purchase items within the app. This model is adequate for mobile games, particularly when users need, for example, extra lives [64]. There are four types of in-app purchases, and each app can offer one or multiple types [67, 68]:

- **Consumable** is the most common type of in-app purchase and refers to the consumable, meaning that a purchased item gets deleted when it is used. An example of consumable is lives in a game or power-ups.
- **Nonconsumable** are features that are purchased once and do not expire. An example can be a filter in a photo app or characters in a game.
- **Auto-renewable subscriptions** are subscriptions that allows the user to access some premium features. These subscriptions charge the users regularly until they cancel. An example of an auto-renewable subscription is Spotify Premium.
- **Nonrenewing subscriptions** provides access to content for a limited amount of time. After the period has ended, the user can choose to renew the subscription manually.



The paid app model or the "paid download" strategy makes the user pay to use the application, usually in the app store, before downloading [69]. This model provides a one-time fee and gives access to full functionality.

## **2.8 The specialization project**

This master's thesis is based on the specialization project, Hyttebiblioteket, by Camilla Velvin, conducted the fall 2021 [1]. In the specialization project it was created a web application for sharing books within second home areas. To improve Hyttebiblioteket, conducted in the specialization project, it was in this master's thesis desirable to create a mobile application based on the functionality of the web application.

### **2.8.1 The project's objective**

The specialization project focused on creating an minimum viable product (MVP) of the concept, Hyttebiblioteket. The objective was to create a web application that allows users to share books within second home areas, as the author several times had been at her second home longing to read a "new" book [1, p. 1].

### **2.8.2 Method**

The specialization project began by defining the requirement specification for the application, before prioritizing them in order of importance regarding the application's functionality. Later a literature search for existing technology and book-sharing solutions was conducted.

Finally, the web application was developed by adding the essential requirements for creating an MVP first and a design based on simplicity for usability.

### 2.8.3 Results

Figure 2.5 shows parts of the implemented design and functionality. The web application makes it possible to create a user, log in, and handle forgotten passwords. The system's authentication was developed by using BaaS, as this was a built-in service.

At the delivery time, the web application contained different functionality such that Hyttebiblioteket could function as a product (see Figure 2.5). The implemented requirements were:

#### Register and login

- A user should be able to register as a system user.
- The system should have authentication with username and password.
- The system should be able to send a new password on mail when "forgotten password".

#### View available books

- The system should show posted books in an area.
- The system should show the book's position in comparison to the user's position.
- The system should show a map with marks where books are.
- The user should be able to filter books based on distance from the current location.
- The system should have a map where the user can see their position in addition to books.

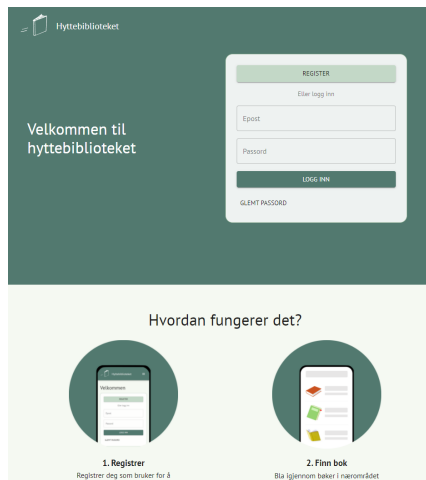
#### Register books

- The user should be able to add a book based on the current location.
- The user should have an overview of all books he/she has posted.
- The user should be able to edit all the books he/she has posted.

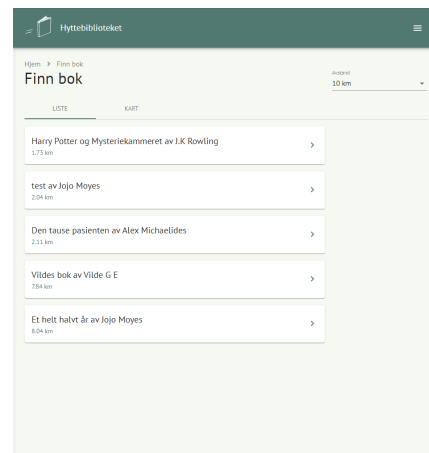
- The user should be able to post a book with another location than the current.

**Collect books**

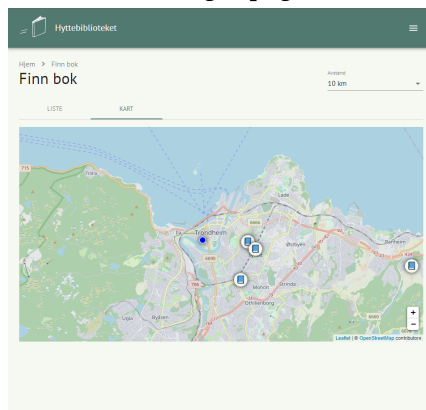
- A user should be able to register that he/she has collected a book.



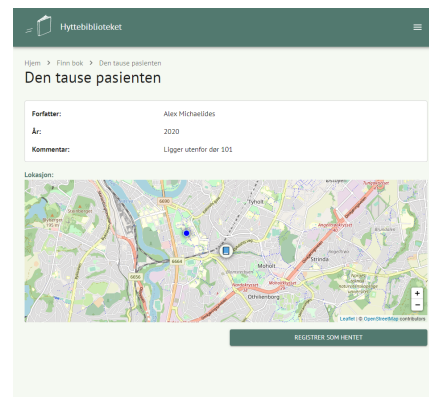
(a) Login page.



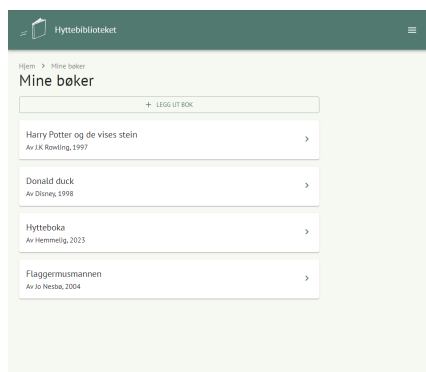
(b) List of available books in the area.



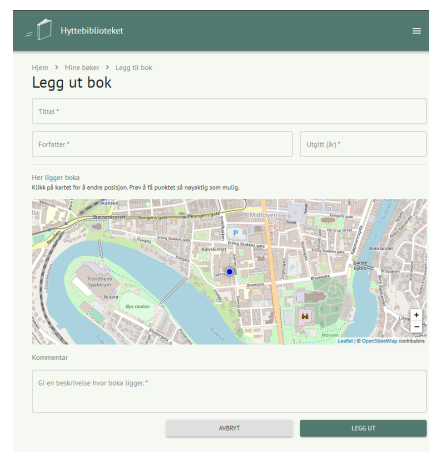
(c) Map with available books in the area.



(d) Details about the available book.



(e) List of the logged-in user's posted books.



(f) Add book

Figure 2.5: Implemented interface in specialization project

# Chapter 3

## Requirement specification

This chapter presents the requirement specification of the application. First, the requirements specification is divided into two main categories; functional requirements and non-functional requirements. These requirements are based on results from the first usability tests, personal ideas and informal talks with friends and family. Later, the functional and the non-functional requirements are numbered and organized into subgroups where each requirement also has a priority telling how important it is to implement it. For example, the requirements to essential to create an app were marked with priority "high".

### 3.1 Functional requirements

Functional requirements define the specification of the behavior and functionality for the application. As the main functionality for the mobile application is the same as the web application's, will many of the functional requirements be similar to the functional requirements in the specialization project [1, p. 7-10].

#### **Register and login**

To keep track of people adding and collecting books, the application requires a user profile. First, the users should be able to register (FR1) with

email and password. Then, the user should confirm the password by writing it twice (FR1.2). After creating a profile, the user should be able to log in with a username and password (FR1.1). By using username and password most people will be able to register in the application, as several elderly do not have a Google account or Facebook.

The system also ensures login for the users by handling "forgotten passwords". The system should handle this by sending an email with instructions to the users (FR1.3).

ID	Description	Priority
FR1	Users should be able to register as a user of the system.	High
FR1.1	The system should have authentication with username and password.	High
FR1.2	The user should confirm the password by entering the password twice.	High
FR1.3	The system should be able to send a new password on email when "forgotten password".	Medium

**Table 3.1:** A table for the system's functional requirements for registering and login.

### View available books

Since the purpose of the application is to share books, it is necessary to show available books for the logged in users. Theoretically, there can be plenty books in the database, therefore, for faster performance the user should be able to see available books in a given area (FR2). Furthermore, from a user perspective, the area should be centered around the user's position making it easy to find relevant books in the neighborhood. By displaying the books in a list, the user can find books based on its title. When finding a book it can also be helpful to see the distance to the book. Therefore, it should be possible to see how far away the book is from the

current location (FR2.1). Additionally, it should be possible to filter the books based on the distance (FR2.3). The idea behind this is that the users can see available books based on the length of their hike.

For simplicity, the users should also see a map with available books centered around the current location (FR2.2). A map can help the users find books based based on its location rather than its title and distance. For example, the position help the users find books located along the hiking route or helps planning a hike based on a specific book's location.

ID	Description	Priority
FR2	The system should show books that are posted in an area.	High
FR2.1	The system should show a list of books posted in the area. The list should contain the distance to the book's location compared to the user's position.	High
FR2.2	The system should show a map with marks where books are.	Medium
FR2.3	The user should be able to filter books based on distance from the current location.	Medium

**Table 3.2:** A table showing functional requirements for viewing available books in the system.

### Register books

The users need to be able to register books they want to share with others for the system to be able to share books. Furthermore, the users should be able to post a book based on the current location (FR3). Thus, the users can quickly post books where they are. Since it is a known fact that GPS can be inaccurate, the users should also be able to post books with a chosen location (FR3.3).

When registering a book, it is natural that the user can administrate it. Therefore, the system should have a list of the logged-in users' posted books (FR3.1) and the possibility to edit them (FR3.2).

By discussion with potential users, some users wanted to keep track of their books. This can be solved by registering ownership when posting a book (FR3.4). With ownership, the publisher can see the book's borrower (FR3.4.1), and only the owner can delete the book from the system.

ID	Description	Priority
FR3	The user should be able to add a book based on the current location.	High
FR3.1	The user should have an overview of all books that he/she has posted.	High
FR3.2	The user should be able to edit all the books he/she has posted.	Medium
FR3.3	The user should be able to post a book with another location than the current.	Medium
FR3.4	The user should be able to register ownership of the book.	Medium
FR3.4.1	The user should be able to see who borrowed the book if it has ownership.	Medium

**Table 3.3:** A table showing functional requirements for registering books in the system.

### Collect books

When a user has found a book that he/she wants to read, it should be possible to register this as borrowed (FR4), and see all borrowed books (FR4.1). When the user is finished with the borrowed book he/she should be able to repost this without adding it as a new book (FR4.2). This will make it easier for the users to post the books, and will increase the possibility for the books to stay in "Hyttebiblioteket".



In the application, it should also be possible to reserve a book (FR5). Reservation is such that the users can reserve a book before a hike and ensure that the book is still there when arriving. If something happens resulting in the user cannot collect the book that he/she has reserved, it should be possible to cancel the reservation (FR5.1).

From the usability tests, a participant also mentioned that it would be helpful to give comments and ratings of a book (FR6) and read others' ratings and comments (FR6.1). However, this was not repetitive feedback and therefore have a very low priority.

ID	Description	Priority
FR4	A user should be able to register that he/she has borrowed a book.	High
FR4.1	Users should be able to see all books they have borrowed.	Medium
FR4.2	A user should be able to repost a borrowed book.	Medium
FR5	Users should be able to reserve a book.	Medium
FR5.1	A user should cancel a reservation for a book.	Medium
FR6	The user should be able to give comments and ratings on a book	Very low
FR6.1	The user should be able to see comments and ratings from other users given on a book.	Very low

**Table 3.4:** A table showing functional requirements for collecting books in the system.

### Deviations

If a book in the application is not to find at a given location, the user should be able to send in a deviation (FR7). First, to simplify the implementation, the deviation will be sent directly to the developer such that she can handle it, either by edit the database and possibly delete the books, or contact the

book's publisher/owner. Later the deviation should be sent directly to the user that posted the book and the owner (FR7.1).

ID	Description	Priority
FR7	The user should be able to notice deviations, for example, books that are not at the posted location.	Low
FR7.1	The system should alert a publisher if there are deviations in his/hers posted books.	Very low

**Table 3.5:** A table showing functional requirements for deviations in the system.

### General

The first usability tests revealed that the users had trouble understanding the concept of Hyttebiblioteket. The application should therefore contain explanatory information about the system (FR9). This information should be easy to find and sufficient for the user to understand.

The system should support dark mode, as this could increase the system's user experience (FR8). This requirement is created as many prefer dark mode on their phones [70]. In addition, can the dark mode preserve the battery on OLED screens and can therefore be helpful for longer hikes.

ID	Description	Priority
FR8	The system should support dark mode.	Low
FR9	The system should show a description of how the system works.	High

**Table 3.6:** General functional requirements for Hyttebiblioteket.

## 3.2 Non-functional requirements

Table 3.7 shows the non-functional requirements for the system. The non-functional requirements are requirements for the application that are not connected to the user's functionality directly.

ID	Description	Priority
NFR1	The application should work on mobile units with iOS and Android.	High
NFR2	The system should have high maintainability; this means that <ul style="list-style-type: none"> <li>• It is a well-documented and clean code with comments where necessary.</li> <li>• Easy to add new features to the system.</li> </ul>	High
NFR3	The application should follow the Norwegian universal design rules for mobile applications [71].	High

**Table 3.7:** Non-functional requirements



# Chapter 4

## Implementation and design

### 4.1 Choice of software development methodology

At the beginning of the development process, an agile methodology was chosen above the traditional waterfall method. Even though the development team consists of one developer and the application has a clear scope, the requirement specification was not clear from the beginning. However, the feedback from the usability test established many of the functionality and requirements for the application. Since the requirements were not clear from start, and the developer had the most experience with agile development before, an agile methodology was suitable for the project.

Some of the most famous agile methodologies are Scrum, Kanban, and Extreme Programming [72]. Scrum has been a popular method since 2002 and is known for its structure, where the process is divided into cycles called *sprints* [73]. The sprints are typically 1-4 weeks in length [74]. The product owner's job is to create a backlog of the product's desired functionality, in addition to its prioritization. This backlog is also the base of the backlog created for each sprint [75]. The developers are responsible for implementing and potentially delivering a product in each sprint. A Scrum board handles the backlog assignments, and every day it is held a daily

stand-up meeting to discuss progress, note issues, and update the board [76]. At the end of each sprint, a review meeting and a sprint retrospective is arranged, where the developers find issues that can improve.

Kanban is another famous methodology. The main difference between Scrum and Kanban is that Kanban does not contain sprints but rather draws tasks from a queue. In addition, Kanban uses a board to visualize the tasks and to limit the work in progress, and releases new finished features continuously [77].

As the development process only has one developer, it was decided to collect inspiration from different methodologies. This is because the methods are adapted to teams with several people and can contain a lot of unnecessary activities and details.

Early in the process, a plan with general milestones for the project was created (see Figure 4.1). At the same time, a board with minor, more concrete activities was planned every second week based on the current status and upcoming milestones (see Figure 4.2). It was believed that this method suited this project well, as planning concrete activities in advantage without sound knowledge about mobile app development is complex.

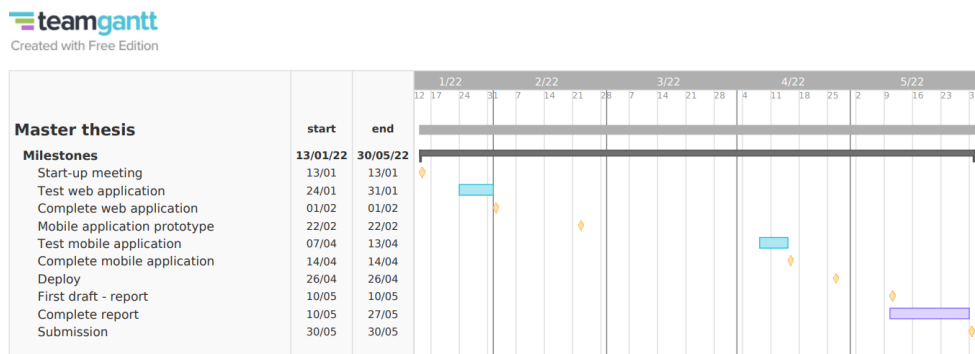


Figure 4.1: General milestones for the project, created in TeamGantt [78].

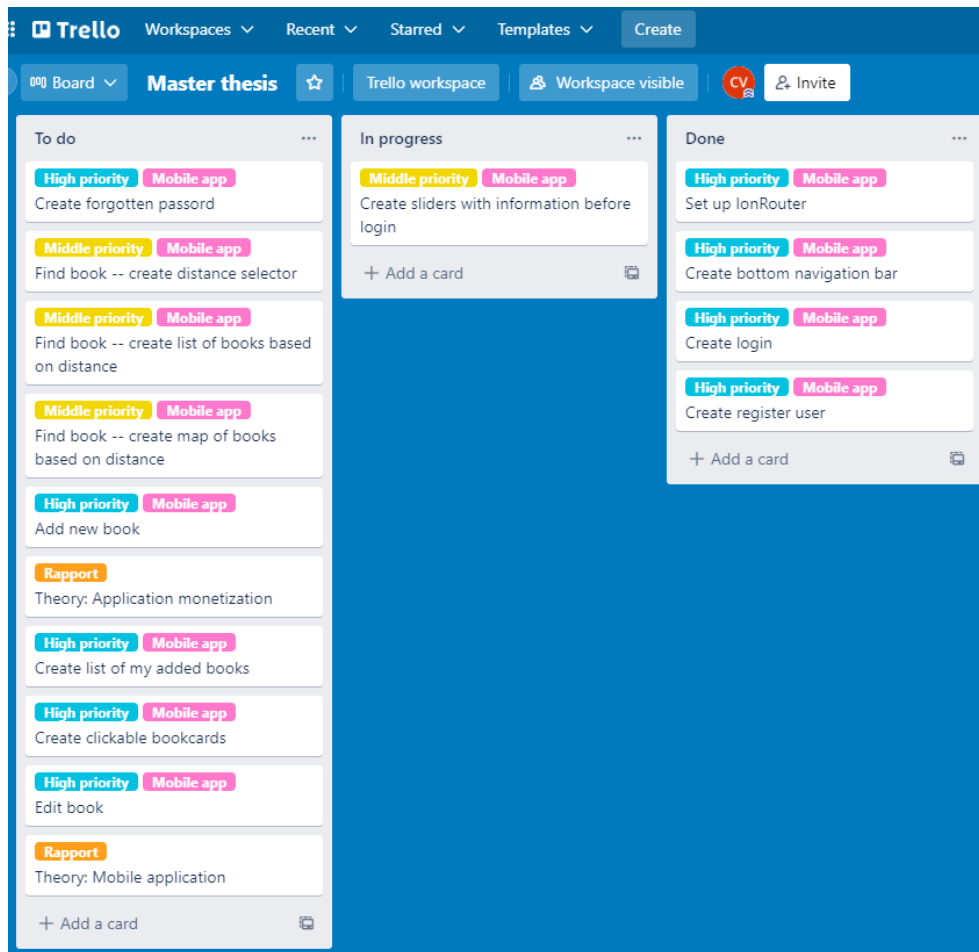


Figure 4.2: Activities for two weeks in Trello [79].

The board can remind of a Scrum or Kanban board. The Scrum and Kanban boards can be physical post-it notes or digital blackboards. In this project Trello [79] was used, an online-based tool that is simple to use. It was necessary with an online tool due to COVID-19, as it was essential to access all tools in case of restrictions and disease.

## 4.2 User interface

### 4.2.1 Color selection

The color theme of the mobile application chooses to use the phone's light or dark mode to decide a light or a dark theme. A light theme refers to a dark-font text on a light background, while dark mode refers to light text on a dark background. It was chosen to implement this as many mobile users has a preference for either light or dark mode. In addition, it might be long-term effects associated with a light mode, and some people with visual impairments will do better with dark mode [80].

The web application created in the specialization project used the tool *Colors* from colors.co [81] to create a color palette (see Figure 4.3). The green color palette was chosen due to its association with nature, and the environment [1]. When choosing a color palette, it is expected to be chosen for the brand, in this case, Hyttebiblioteket. Therefore, it was natural that the color palette of the mobile application was similar to the web application's.

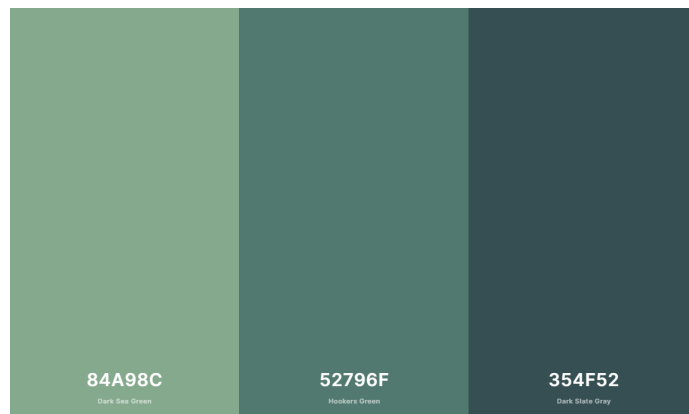


Figure 4.3: Chosen color palette from Colors [81].

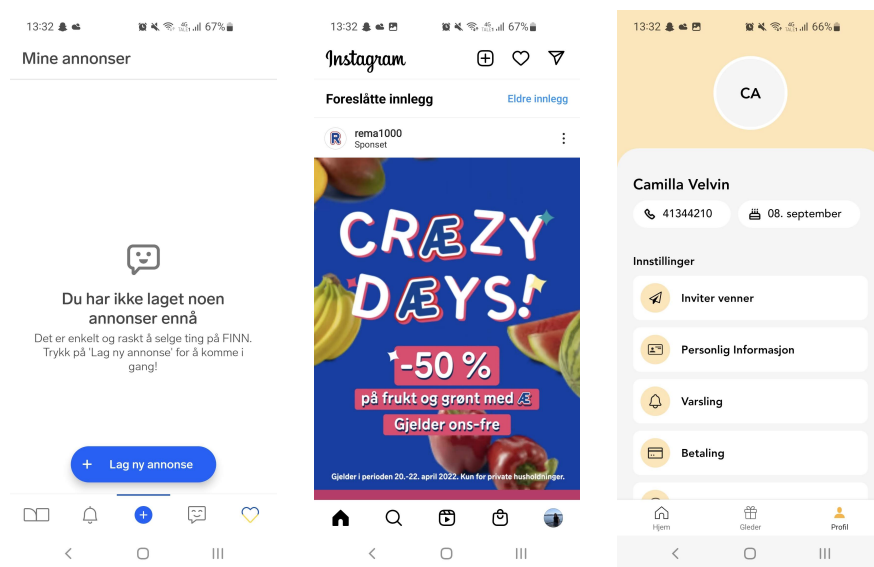
### 4.2.2 Inspiration for the user interface

The requirement specification defined in chapter 3 does not contain specifications on the user interface except that the application should follow the



Norwegian universal design rules (NFR3). Therefore, during the development process, it has been focused on creating an app with a simple user interface with few disturbances and focus on contrasts.

Several existing solutions inspired the user interface. Some of these solutions are; *Finn* [82], *Instagram* [83], and *Glede* [84] (Figure 4.4).



(a) Finn mobile UI [82] (b) Instagram mobile UI [83] (c) Glede mobile UI [84]

Figure 4.4: Inspiration for Hyttebiblioteket user interface

### 4.2.3 Prototyping

A prototype containing all the different pages in the mobile application was created to increase the programming efficiency. There are two different ways to create a prototype; *low-fidelity* and *high-fidelity* [85]. Low-fidelity prototyping is, for example, hand-drawings on paper. These sketches are quicker to create and help visualization in an early phase [86]. High-fidelity prototypes are computer-based and often more realistic with interaction and functionality. However, this model can be more time-consuming and costly than low-fidelity prototyping [87].

Creating a high-fidelity prototype in a design and prototyping tool increased the efficiency of developing the user interface. In this project the prototyping tools Figma [88] and Adobe XD [89] was considered. Figma and Adobe XD has a lot of the same functionality and a free version. However, the free version of Figma offers more functionality than the free version of Adobe XD. In addition, Figma also makes it possible to design complete user interfaces, which can be created as a prototype to test all the basic functionality. This, in addition to previous experiences, simplified the choice of using Figma as the prototyping tool.

### 4.3 Usability tests

Qualitative usability tests made it possible to test the entire system, which was desirable in this master's thesis. Since the system consists of a web- and a mobile application, it was decided to perform two different usability tests, one for each application. First, the web application was tested, and later the mobile application. The goal of the usability tests were to test the interface and the user interaction, and receive feedback on all the features, in addition to input of changes that would improve the system.

Before starting the usability tests, a test plan for execution was designed. The plan ensured that all desired features and designs got feedback from the participants.

#### 4.3.1 Test participants

When using qualitative usability tests it is not required many test participants. The reason is that the more significant errors and problems will be discovered early, such that further testing will not give more insight [50, p. 14][53, 90]. Five different participants are often used to perform a usability test [90, 91], and in this project, five test participants are used for each usability test.

In light of the ongoing COVID-19 pandemic and the rapidly changing restrictions, the test participants should be "close contact" with the developer. Using close contacts results in reusing some of the test participants in both usability tests. As the first test will contain the web application, the reused test participants will know the concept and most of the features available in the mobile application. Reusing the test participants was then seen as an opportunity to check if the reused participants differed from the new participants. Therefore, it was chosen two participants that participated in both usability tests.

### 4.3.2 Preparations and conducting

Before the tests it was created a checklist that should be executed before each new test participant started:

1. Make sure that books are available in the user's area.
2. Remove disturbing notification on the test unit if the participant does not use their own.
3. Disinfect the test unit if the participant does not use their own.

To ensure all participants got the same information about the usability test it was presented a script before starting each test:

*You are now joining a usability test for Hyttbiblioteket. This application is created to share books within second home areas. During this usability test, you will get some tasks, and your job is to solve them. While doing so, tell me what you are thinking. This test is not to test you but a system test, so there are no wrong answers. However, if you feel stuck, just let me know, and I will help you. I will also create notes during the test, but these notes will not contain your name or any information about you. After the tasks, I will ask you some questions about your experience and opinions. Do you have any questions before we start?*

When conducting a usability test, there are three ways to give tasks: *concrete tasks*, *partially open tasks*, and *open tasks* [92]. It was chosen to have

concrete tasks; this is the most used in usability testing and builds on the test participants performing given tasks in a specific order.

The first usability tests tested the web application. There were five tasks with belonging sub-tasks for each test participant to solve:

1. **Create a profile**
  - 1.1. Log in
2. **Post a book in a chosen area**
  - 2.1. See book in overview
  - 2.2. Change position of the posted book
3. **Find a book using the list**
  - 3.1. Find the book's location
  - 3.2. Register that a book is collected.
4. **Find a book using the map**
  - 4.1. Find more information about the book
5. **Find borrowed books**
  - 5.1. Find one of the books that are borrowed
  - 5.2. Repost a borrowed book

The first usability tests were performed early in the project. The results were used to improve the design and prioritize new requirements that should be implemented later in the project, as well as the requirement specification for the mobile application. The second usability tests' tasks differ from the first as new features are implemented at this point. The second usability tests were performed later in the development process and tested the mobile application. These tests consisted of six tasks with belonging sub-tasks:

1. **Create a profile**
  - 1.1. Log in
2. **Post a book in a chosen area**

- 2.1. See book in overview
- 2.2. Change position of the posted book
3. **Find a book using the list**
  - 3.1. Find the book's location
4. **Find a book using the map**
  - 4.1. Find more information about the book
5. **Collect book**
  - 5.1. Register that a book is collected.
  - 5.2. Find one of the books that is borrowed.
  - 5.3. Regret borrowing the book.
  - 5.4. Repost a borrowed book in another location than it was picked up.
6. **Send in deviation**

After the first and second usability tests, a semi-structured interview was performed with the participants. The interview was intended to get feedback from the user that might not have appeared during the tasks. The interview contained five questions, with additional follow-up questions if necessary:

1. How did you experience the application?
2. What were the best and worst with the application?
3. Was it anything that you missed, something that would be useful for you as a user?
4. Is this an application that you would like to use?
5. Any additional feedback?

## 4.4 Technical implementation

### 4.4.1 Development framework

When choosing a development framework, it was essential to consider the non-functional requirement NFR1; the application should work on mobile

units with iOS and Android. To simplify the development due to the time limit it was chosen to develop a cross-platform application.

### **Cross-platform application**

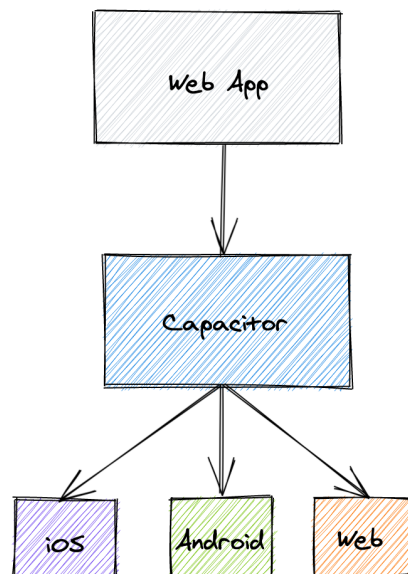
The commonalities between all the considered development frameworks are that they offer iOS and Android development with only one codebase. With only one codebase, the development time will be reduced significantly as the same application only needs to be developed once. In addition, this codebase will also simplify and fasten the maintenance and updates later. The development frameworks considered for this application were NativeScript, an open-source framework for cross-platform development [93], Flutter, a framework by Google [94], React Native, a framework created by Facebook based on the JavaScript framework React [95], and Ionic, a cross-platform framework specialized in mobile units [96]. Even though all these frameworks offer functionality to create an iOS and Android application with only one codebase, problems requiring platform-specific code may occur. How often this is necessary depends on the development framework. Of the considered frameworks, is Ionic the one that requires the least platform-specific code, React Native requires the most, while Flutter and NativeScript are placed in the middle. Ionic is also the framework with the most pre-styled components, making the development process faster as one can use less time on the user interface. On the other hand, React Native has the least pre-style components. Flutter has approximately the same as Ionic, while NativeScript is in the middle [97].

### **Ionic with Capacitor**

Ionic was chosen as the development framework for the mobile application. NativeScript was also a candidate for the application, but the development should be faster with Ionic as Ionic has more pre-styled components than NativeScript. In addition, the codebase should be easier to maintain as Ionic has more reusable code than NativeScript [97].

Ionic is an open-source software development kit for hybrid mobile app

development [98]. Ionic was created in 2012 and enables the creation of cross-platform applications with web technologies like HTML, CSS, and JavaScript. With the integration of the most famous JavaScript frameworks Angular, React, and Vue [99]. To run the code developed with web technologies on mobile units, a link between the web application and hardware is necessary. Through these links, the application has access to the mobile device's features, such as GPS or camera. With Ionic, it is common to use either Capacitor (see Figure 4.5) or Cordova as link [100]. In this project it was chosen to use Capacitor together with Ionic as Capacitor delivers the same cross-platform benefits as Cordova but in a more modern approach [101].



**Figure 4.5:** A simplified model of how a mobile application with Capacitor and web app is built. [102].

## 4.4.2 Libraries

### Frontend library

As mentioned in section 4.4.1, Ionic integrates with different JavaScript frameworks to develop the web application part of the mobile application. The integrated JavaScript frameworks are React, Angular, and Vue,

and in addition is development with HTML, CSS and JavaScript supported in Ionic [99]. When developing larger applications, it is beneficial to use a JavaScript framework instead of standard HTML, CSS, and JavaScript. The reason is that when applications expand, it requires more code. An expansion can be messy with standard JavaScript, HTML, and CSS, while the framework mentioned above is created to encourage standards and reusable components to simplify the code [103].

When selecting a JavaScript framework, Angular an open-source framework developed by Google [104], React, a UI library created by Facebook [105], and Vue, a progressive framework [106] was considered. Since React with TypeScript was used in the specialization project [1], it was decided to use React with TypeScript for the mobile application as some of the code might be reusable.

### **Leaflet**

It was necessary to implement an interactive map in the application to achieve the following functional requirement; a user should be able to see a map with marks where there are available books (FR2.2). From research, there exist different implementations of a map in Ionic. The most popular is Google Maps, but as this has a fee, many developers use Leaflet as this is free [107]. As the web application created in the specialization project [1] implemented Leaflet, using Leaflet in the mobile application was a simple choice as the user interface will have similar looks, and that Leaflet does not require billing.

### **Formik**

Forms are an essential part of web- and mobile applications. Forms collect data for processing from users. For example, the user can send in a form when registering, logging in, and adding a new book in Hyttebiblioteket. To handle the validation, visited fields, and form state, it was decided to use a library. React Hook Form and Formik were the libraries considered for this application. Formik and React Hook Form's main difference is that



Formik uses controlled components while React Hook Form uses uncontrolled components [108]. It was chosen to use Formik as it was desirable to include validation of the controlled components and handle the form submission [109].

### 4.4.3 Backend solution

*Some of following text is copied from the specialization project [1]. The text has been edited for relevance.*

When creating a mobile application, a connection between the user interface and the data is necessary. As the data is only available for logged-in users, the connection needs a form of authentication. This can be solved in several ways, and solutions considered for this project were; a REST API with a SQL database, MongoDB with a Node.js driver [110], or Google's cloud hosted solution Firebase [111]. By comparison, was Firebase the right choice for this application. This is because a REST API with a SQL database or MongoDB with a Node.js driver requires higher requirements for publishing and implementation than Firebase. Another factor that simplified the choice to use Firebase is that Firebase was implemented in Hyttebiblioteket's web application [1].

#### Firestore

Firestore is a Backend-as-a-Service (BaaS) platform developed by Google. This means that Firestore provides pre-written software for server activity, such as user authentication, database management, cloud storage, and hosting [36]. Consequently, instead of creating a server-side, Firestore can be used for all server-related tasks, such as authentication and hosting [112].

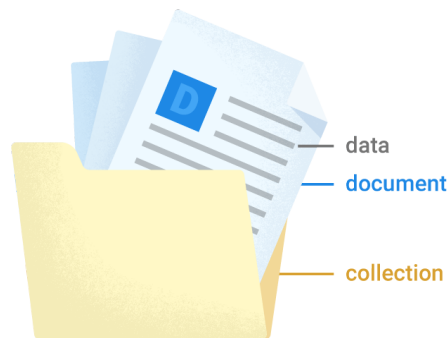
#### Firestore Authentication

For the application, it is crucial to know the user's identity for security and to use of the application on different devices. Firestore Authentication pro-

vides backend services, SDKs, and UI libraries to authenticate users in the application [113]. The Firebase Authentication offers several different authentication methods. For example, the applications' registration and login with email are supported with Firebase Authentication through Firebase Cloud. Firebase also has methods for registration and login with Google, Facebook, or Twitter, which can be implemented later if wanted.

### Cloud Firestore

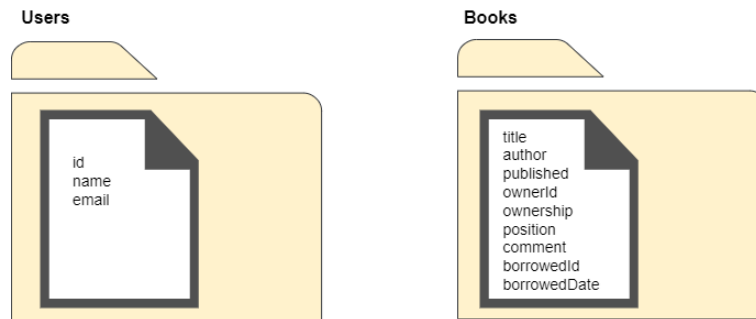
To save book- and user data, the application uses Cloud Firestore. Cloud Firestore is a cloud-hosted NoSQL database from Firebase that stores files and documents in JSON format. Figure 4.6 shows how the data is saved as fields in a document. Documents are stored in a collection, a container used for organizing and building queries for the data [114]. The documents support many data types, making it possible to point to other collections. This helps the structure and logic of the database.



**Figure 4.6:** Structure in Cloud Firestore [114]

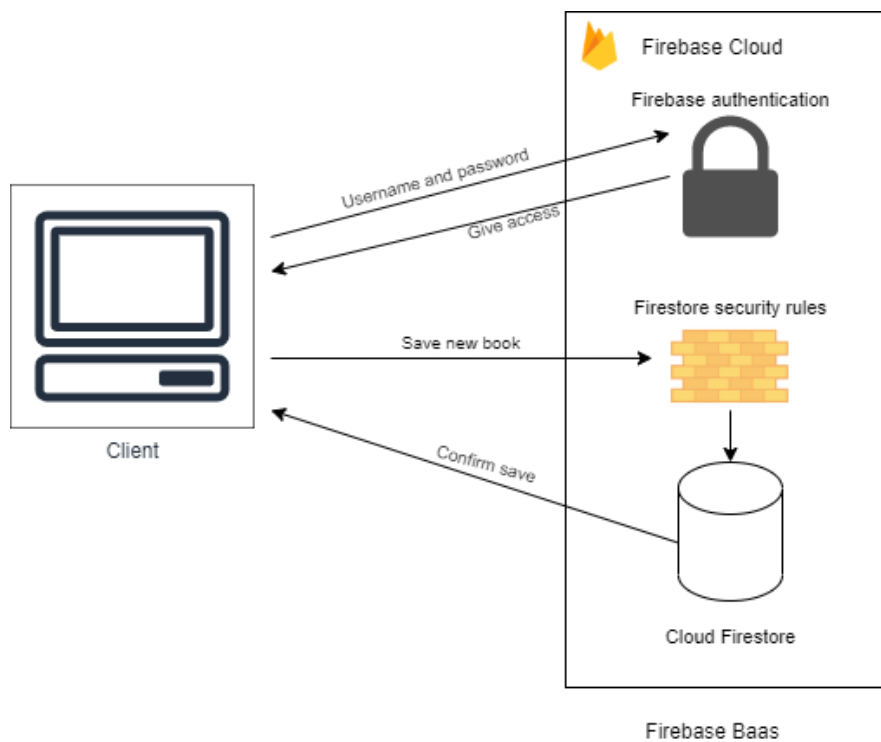
To save data in the application, Firebase Cloud is used. The database consists of two different collections; *users* and *books*. The user collection contains a document for each application user with a user-id connected to Firebase Authentication, name, and email. The book collection contains documents for each book that is registered. A new book should be registered through the application interface by a logged-in user, and contains information about the book, an owner-id, and a borrowed-id connected to

a user-id in the users' collection (Figure 4.7).



**Figure 4.7:** The data structure in Cloud Firestore for Hyttebiblioteket

Cloud Firestore Security Rules and Firebase Authentication are used to secure the data in the application (Figure 4.8). In addition, Firestore Security Rules provide access control that protects the data in Cloud Firestore.



**Figure 4.8:** Diagram for interaction between client and BaaS for login and saving a new book.

## 4.5 Chosen app monetization strategy

When selecting an app monetization strategy, it was essential to consider the target audience. Some audiences are willing to spend money on apps while others are not [115]. As mentioned in section 2.7, an "in-app purchase" model has a free application, but the user can choose to pay for additional features or items. However, as *Hyttebiblioteket* cannot hide several features or items, the in-app model did not fit this application. Therefore, the two monetization strategies considered were "paid download" and "in-app advertisement".

At the beginning of apps, the "paid download" model was the most popular. As apps have evolved, the user expectations have increased, and it can be challenging to convince users to pay for something they have not tried [64]. This, therefore, simplified the choice of "in-app advertisement". In addition, with "in-app advertisement", the application might establish a market faster as users have a lower threshold for downloading free apps. When using "in-app advertising", it is essential to remember that ads that irritate the user can cause frustration such that the user does not want to use the application. It was therefore decided that native ads were the best fit.

Another possible solution to earn money at the application is to sell the idea to investors or companies. By expanding the "*Hyttebiblioteket*" idea to share more than books, for example, to share leisure equipment, a possible investor could be *Røde Kors* [116]. As it is crucial to increase the number of users and introduce new features to attract potential investors [117], it was decided to wait to contact potential investors until the application was deployed.

## 4.6 Deployment

*Deployment* is defined as the procedure of installation, configuration, running, testing, and making necessary changes to release a new software or a new version of software for the customers [118, 119].

To get Hyttbiblioteket ready for launch, it was necessary to test the application thoroughly. Based on the non-functional requirement NFR1 (see section 3.2), the mobile app had to be tested on both Android and iOS. As mentioned in section 4.4.1, the code and mobile units connected with Capacitor. To connect these handles Capacitor a platform sync containing an Ionic build, compilation of web assets, copying the compiled web assets to the Capacitor native platforms, and updating Capacitor native platforms and dependencies [120]. The result of the sync is platform-specific codebases for Android and iOS. As Capacitor does not support building and running the codebases as mobile apps for Android and iOS, it is necessary to use a native IDEs [121]. It was decided to use Android Studio for the Android app and Xcode for the iOS app, as these are the most popular IDEs for mobile app development.

### 4.6.1 Android studio

Android Studio is the official Integrated development environment (IDE) for Google's Android operating system. It is built on IntelliJ, an IDE for developing software, and specifies on developing Android applications [122].

### 4.6.2 Xcode

Xcode is an IDE for macOS, and consists of tools that developers use to build apps for Apple platforms [123]. A necessary tool used for developing the iOS application was the simulator. This can run and test the application in a simulated environment when an actual device is not available.

As the developer for Hyttbiblioteket did not own a computer with ma-

cOS, it was necessary to use *MacinCloud* to build and test the iOS application in Xcode [124]. *MacinCloud* is a Mac cloud platform that allows users to learn application development, develop software, build cross-platform applications, and automate app testing for anywhere with internet access [124].

### 4.6.3 Release of the application

After developing and testing the application, the next step of deployment process is to release the application. A release means that the app will be available for the customers through the digital distribution platforms *Google Play* for Android and *App Store* for iOS.

To build and release an iOS app an Apple developer account is required. Unfortunately, this account's price is relatively high, which resulted in the iOS application being only available to run in a simulator (see Appendix C).

Building and releasing an Android app does not require a developer account, but the release itself has a one-time fee. However, Android devices are able to download the build file and receive the mobile application without releasing the application through Google Play. A guide to downloading the application as a mobile app can be found in Appendix B. Like the iOS app, it is also possible to run the Android app in a simulator (see Appendix C).

# Chapter 5

## Results

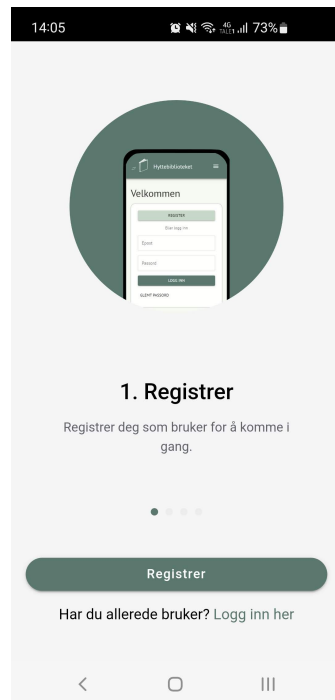
### 5.1 Status of application at delivery time

#### 5.1.1 Design of the application

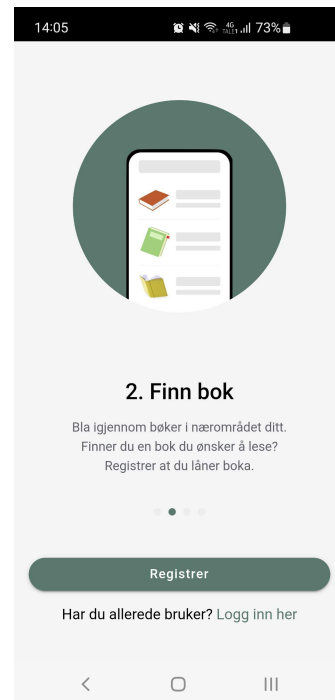
This chapter presents the results of the design of the application. The user interface is based on the prototype created in Figma during the project's design phase (see section 4.2). All figures presented in this chapter are screenshots from the Android version of Hyttebiblioteket on a Samsung Galaxy S10.

#### **Registration and login**

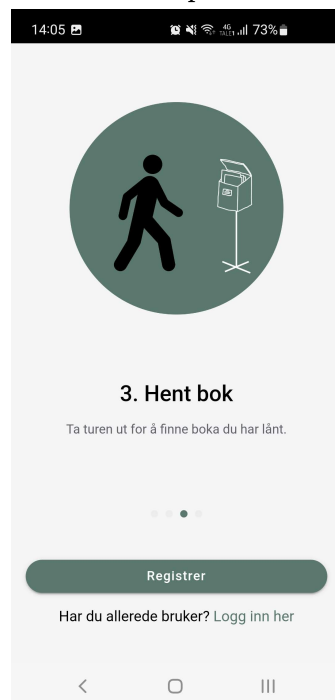
When opening the app, Hyttebiblioteket, the user will enter a page containing information, registration, and log in. The information explains the concept of Hyttebiblioteket (FR9) and is implemented in a multi-section container where each section can be swiped or dragged (see Figure 5.1).



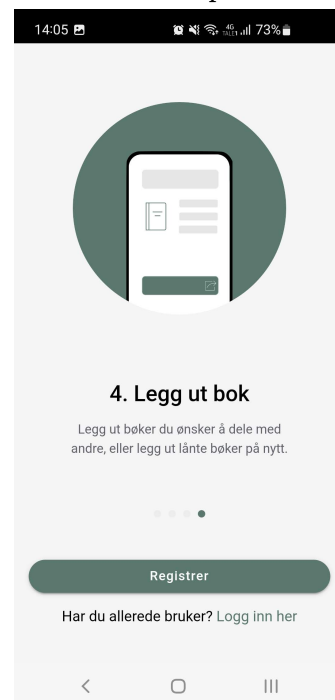
(a) The first step of the slider



(b) The second step of the slider



(c) The third step of the slider

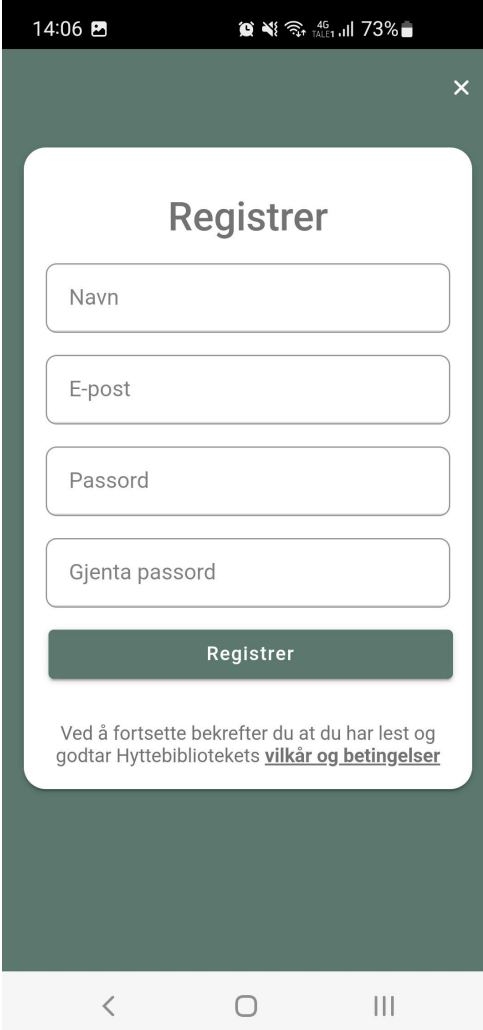


(d) The fourth step of the slider

**Figure 5.1:** Front page of the app. A page containing a slider with information explaining the concept of the application, registration and log in.



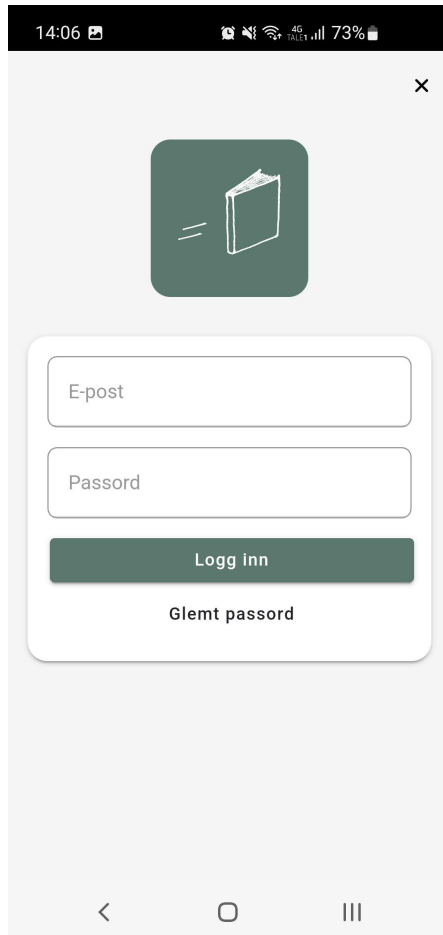
The interface for registering as a user (FR1) (Figure 5.2) is reached by clicking the "Registrer" button on the front page (Figure 5.1). As a result of the first usability tests, the register page contains a repetition of the password (FR1.2) and terms and conditions explaining the terms and conditions, and why the email is necessary for registration.

The image shows a mobile application registration page. At the top, the status bar displays the time 14:06, signal strength, Wi-Fi, 4G LTE, and 73% battery. The page has a dark green background with a white rounded rectangle in the center. The title "Registrer" is centered at the top of the white area. Below the title are four input fields: "Navn", "E-post", "Passord", and "Gjenta passord". A dark green button labeled "Registrer" is positioned below the input fields. At the bottom of the white area, there is a line of text: "Ved å fortsette bekrefter du at du har lest og godtar Hyttebibliotekets [vilkår og betingelser](#)". The bottom of the screen shows the standard Android navigation bar with back, home, and recent apps icons.

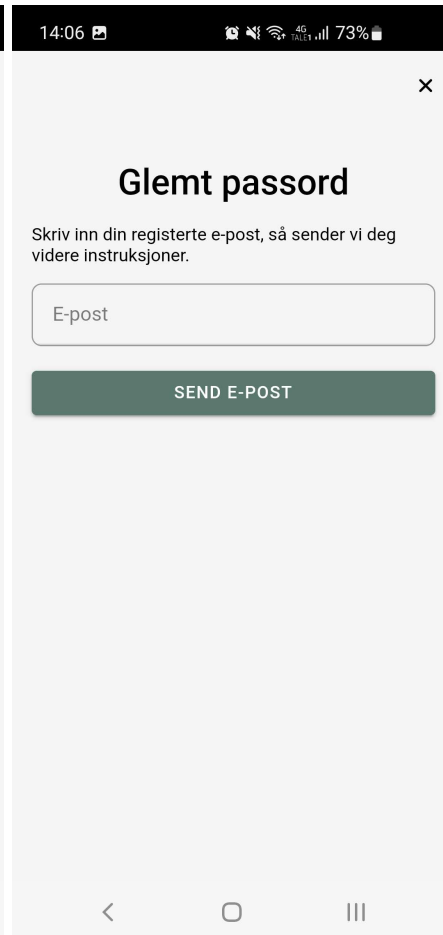
**Figure 5.2:** Registration page for the mobile application.

If a user is already registered in the system, he/she can log in by using email and password (FR1.1) (see Figure 5.3). If the user has forgotten his/her password, this can be handled by clicking the "Glemt passord" button in Figure 5.3 (FR1.3) (see Figure 5.4). The solution for forgotten password is

implemented by using a method from Firebase Authentication, as used in the specialization project [1], and ensures that the user receives an email with a link to reset the password.



**Figure 5.3:** Login page for the mobile application.

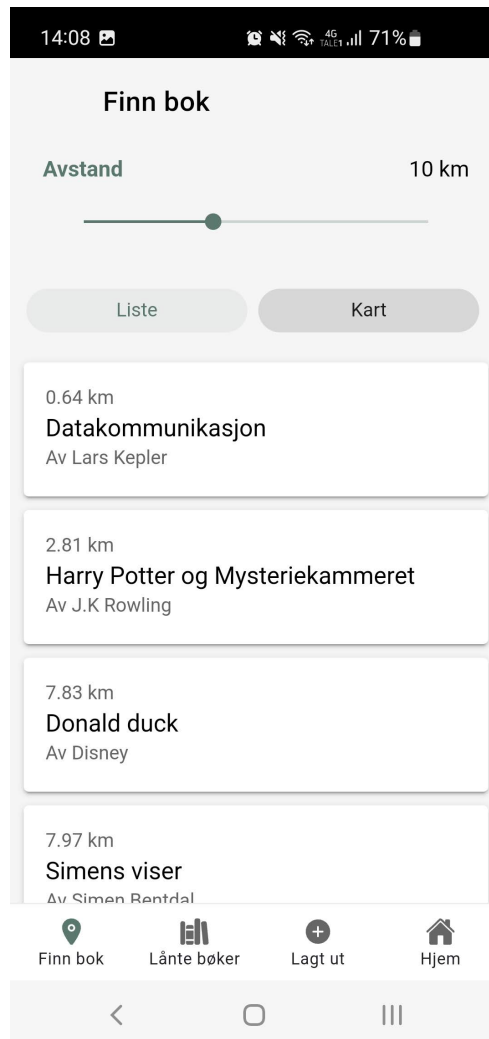


**Figure 5.4:** A page for forgotten passwords in the mobile application.

### View available books

The page showing the available books (FR2) is designed according to the design prototype. A range slider allowing the user to select the maximum distance from the current to the book's position (FR2.3) is placed at the top of the page. The range goes from 1 to 25 km, where the limits are chosen because few people will hike more than 25 km one way to collect a

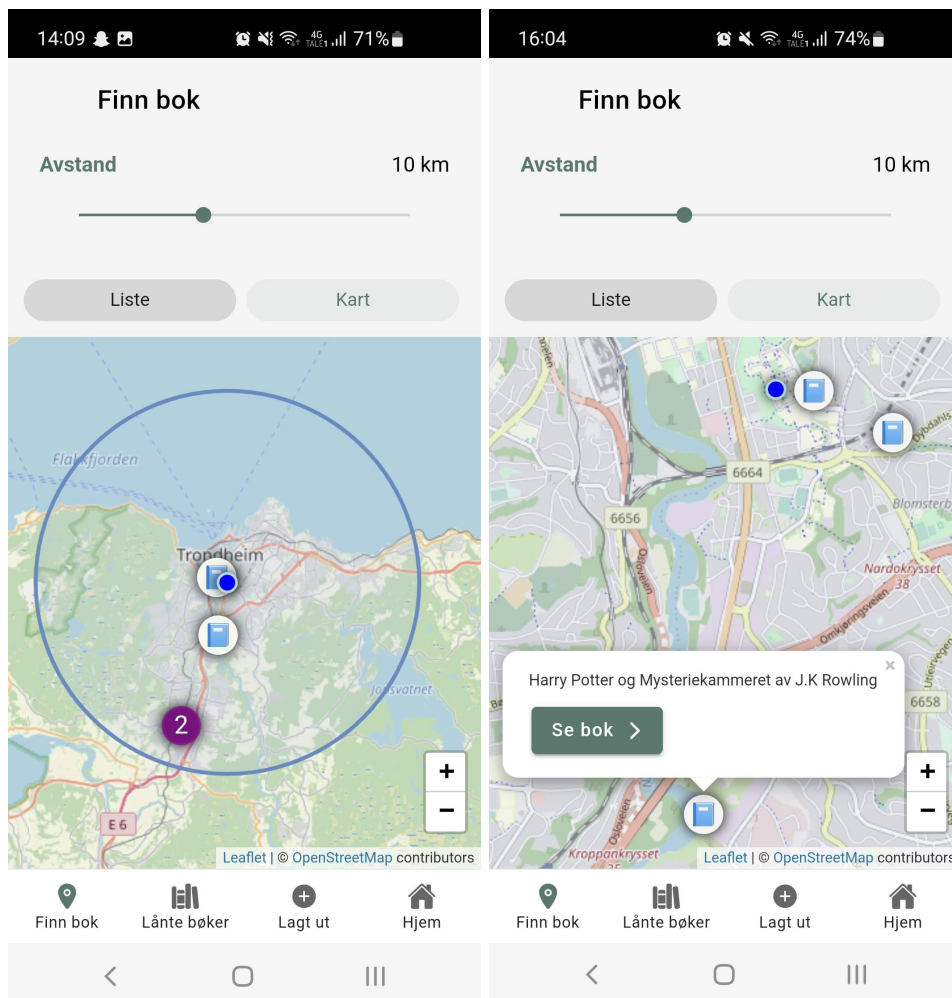
book. Below the range slider are tabs enabling the user to choose between a list and a map. By choosing the list, a list with available books is shown (FR2.1) where each available book is presented as a card containing the book title, author, and distance from the current position to the book (see Figure 5.5).



**Figure 5.5:** List of available books in the mobile application.

By clicking the tab containing "map", a map centered around the current position is shown (see Figure 5.6). The current position is marked with a small blue circle. Around the current position is a larger blue outlined circle that defines the searchable area for books. This circle should help the users

visualize the search and the distance to a specific book. The available books are marked with white circles containing a blue book (FR2.2). If several books are placed in one location, or it is hard to separate the book circles, a purple circle with a number is shown instead. This number indicates how many books that is located at the specific position. By using the map, the user will not receive any information about the book. Therefore, by clicking the bookmarks, more information will appear (see Figure 5.6b).



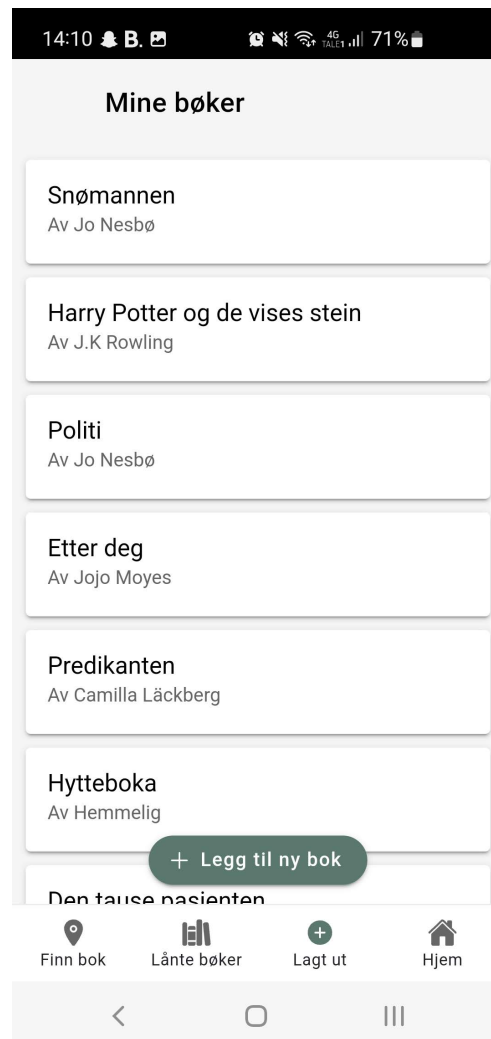
(a) Map with the current position and available books in the area based on distance.

(b) Clickable books on the map.

**Figure 5.6:** Map with available books in the mobile application.

## Register books

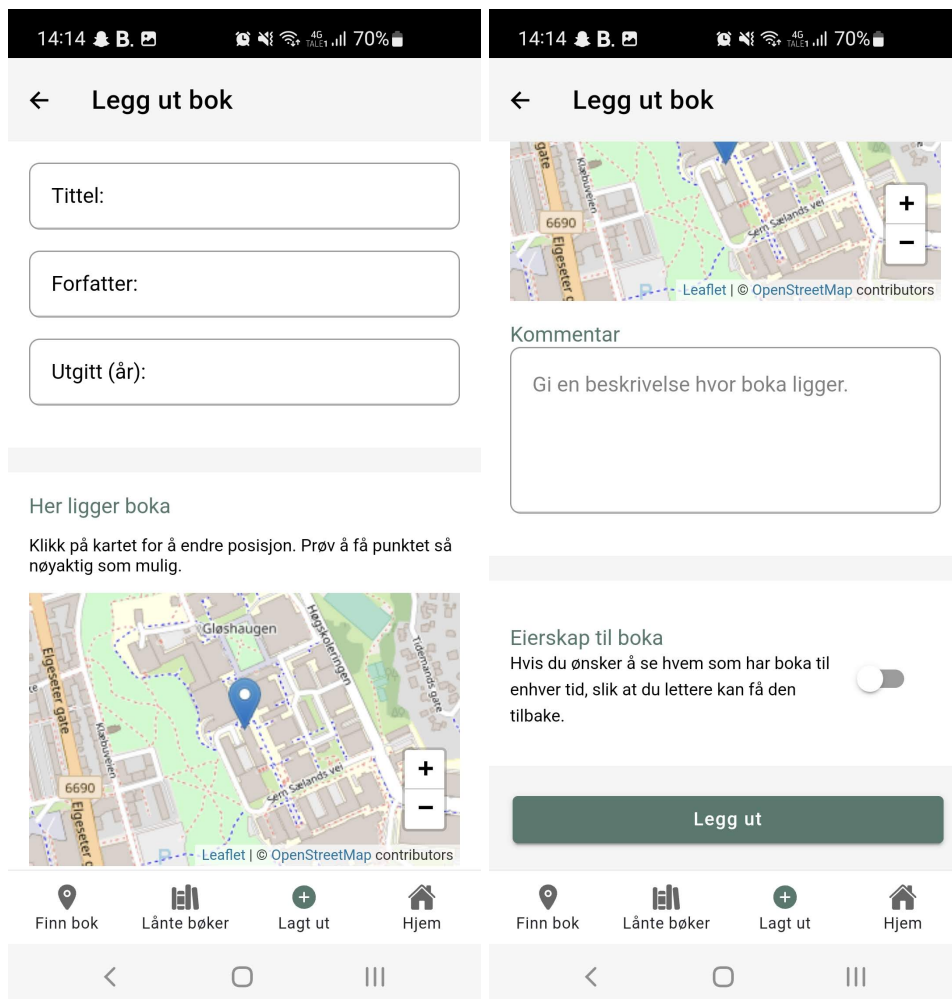
By clicking the button at the bottom navigation bar containing a plus sign, the user will navigate to "Mine bøker". This page contains the logged-in users' registered and posted books (FR3.1) (see Figure 5.7).



**Figure 5.7:** Books registered or posted by the user.

By clicking the green button "+ Legg til ny bok" (see Figure 5.7) the user is able to register a new book in the application. Figure 5.8 shows the user interface for adding a new book. First, the user needs to fill in information about the book. Then, the user needs to locate the book by placing a pin on the map. By default, the pin is located at the user's current position (FR3)

but can be moved by clicking the map (FR3.3). From testing and informal discussion with friends, it was revealed that some users wanted to keep track of their books and be able to get them back at some point. Therefore, users can register ownership of a book they post (FR3.4). An ownership allows the owner to see who borrowed their book at all times (FR3.4.1) and the book's location.

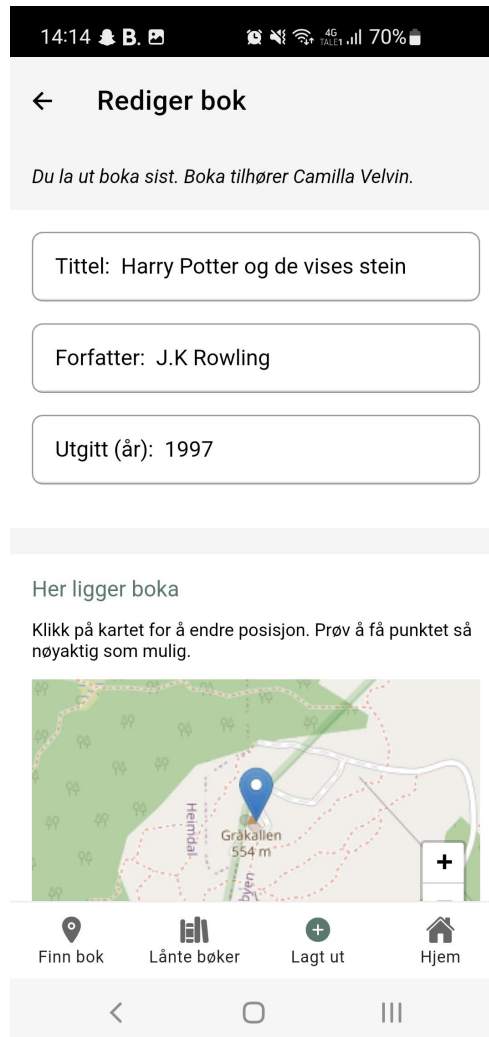


(a) The top of the "add book" page. (b) The bottom of the "add book" page.

**Figure 5.8:** Page to register a new book in the mobile application.

When a user adds a new book, he/she is also able to edit it (FR3.2). By clicking a specific book in the overview of registered and posted books (see Figure 5.7), the user will be able to edit the book, its position, and

ownership (see Figure 5.9).

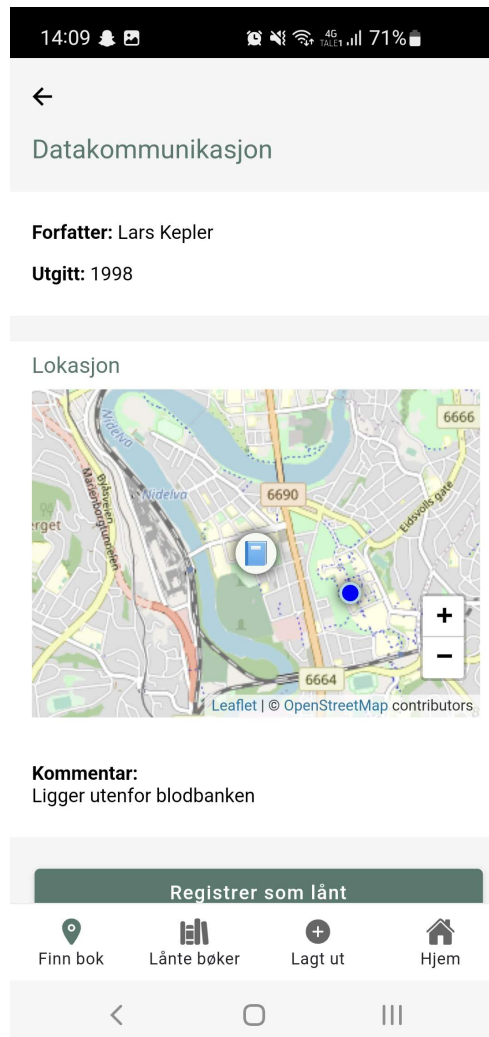


**Figure 5.9:** Page to edit a registered book.

### Collect books

Clicking a specific book in the list of available books (see Figure 5.5) or on the map (see Figure 5.6b), the user will see more information concerning the book (see Figure 5.10). The information contains general information about the book, a map showing the position of the book relative to the user's position, and a comment explaining the location. Further, if this is a book the user wants to read, the user can register the book as borrowed

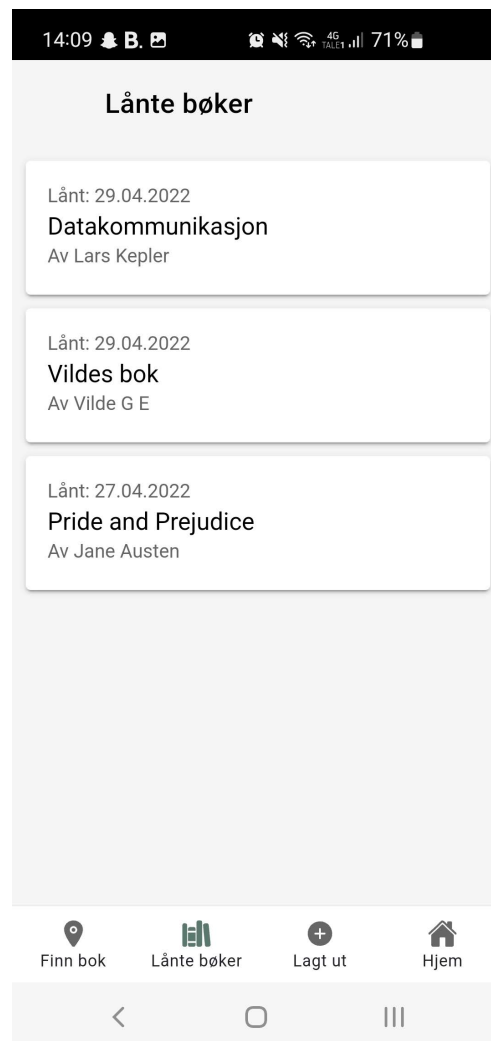
(FR4).



**Figure 5.10:** Information about a specific book.

If a book is registered as borrowed, it will appear in the overview of borrowed books (FR4.1) (see Figure 5.11). The page containing an overview of borrowed books is designed according to the design prototype, and uses the same list view as the page for "my books" and "available books" to create a holistic design.





**Figure 5.11:** Page with an overview of borrowed books.

The interface for a borrowed book is implemented according to the design prototypes. The interface is reached by clicking one of the borrowed books from the list in Figure 5.11, and contains information about the book and its location (see Figure 5.12). By creating this information page, it is possible to borrow a book before collecting it, in other words, reserve a book (FR5). The users can then borrow a book before a hike and be certain that it is still available when he/she arrives at its location. However, if the user is unable to collect a reserved book, he/she can cancel their registration by clicking the "Angre registrering" button (FR5.1) (see Figure 5.12).

If the book is collected, the user can repost the book in a different location (FR4.2). A modal allowing the user to place the book in a new position appears when clicking the filled green button, "Legg ut bok på nytt" (see Figure 5.12). Users can also delete the book from the system, if the book does not have ownership. However, this functionality is mainly created for ruined and lost books, as there is no point for others to collect these books.

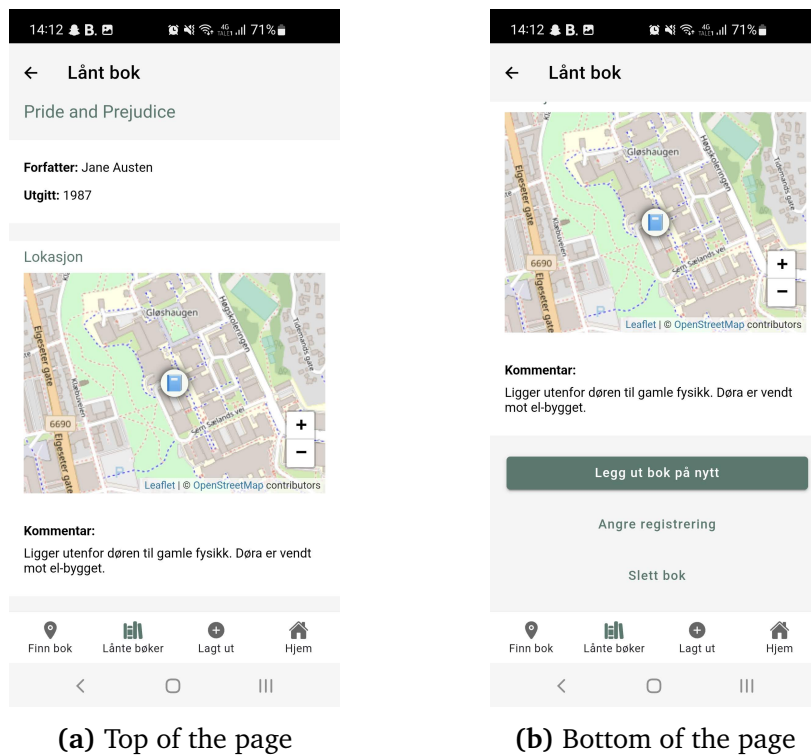
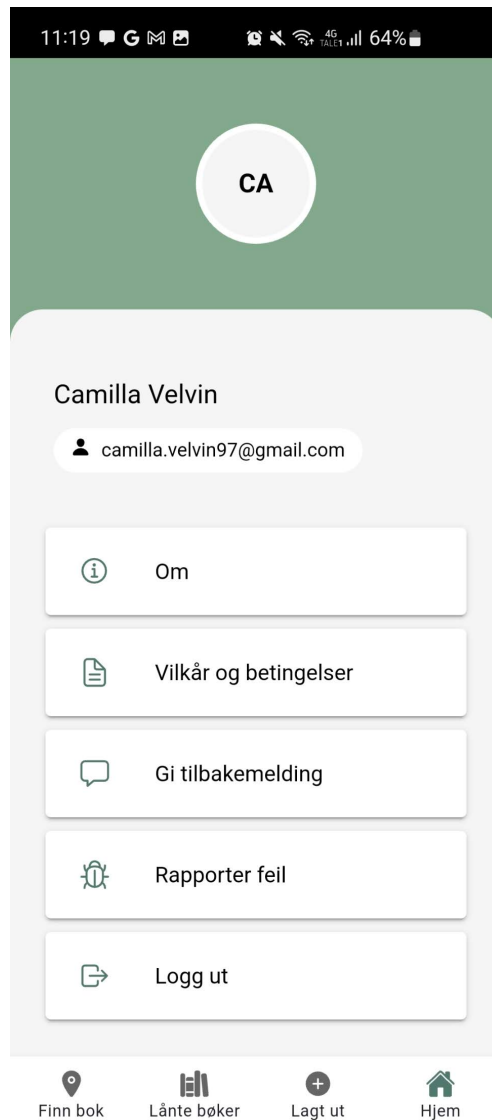


Figure 5.12: Page showing information about a specific borrowed book.

## General

To assemble all information a home page is created (see Figure 5.13). The home page consists of the logged-in users' name and email, a menu with different helpful information for the user, and log-out.

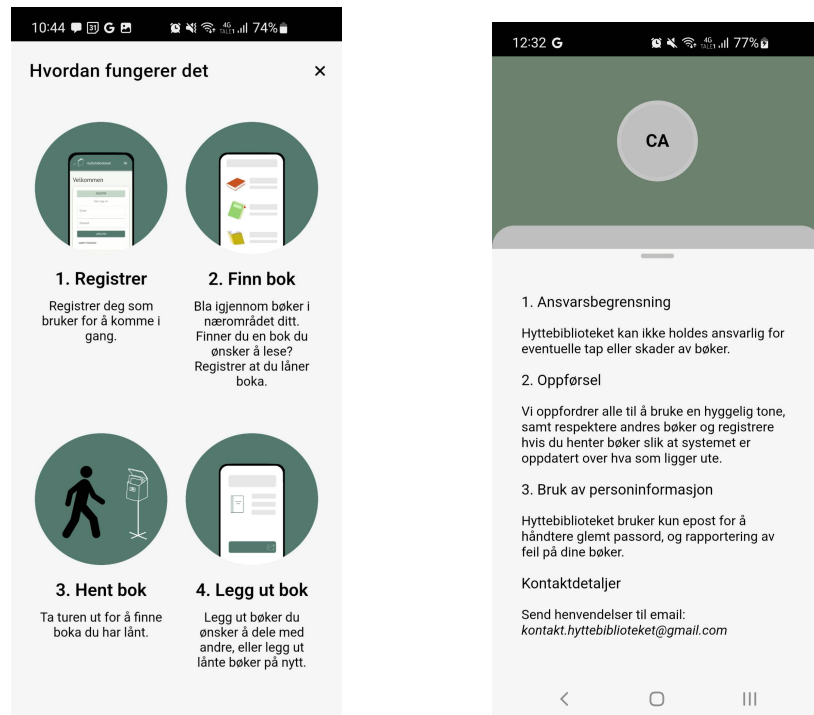


**Figure 5.13:** Home page in the mobile application

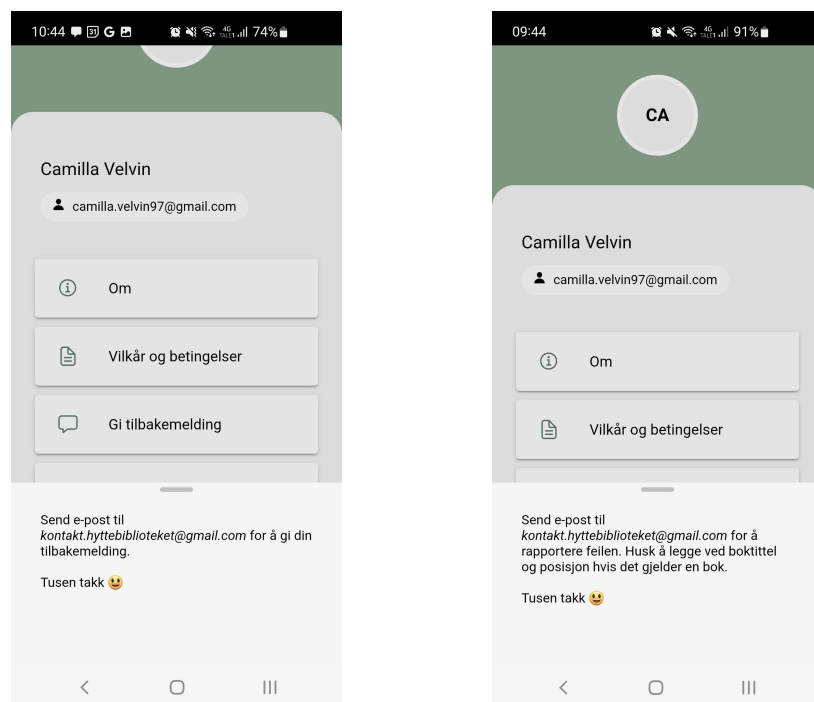
By clicking the first button, "Om", a modal that explains the concept of Hyttebiblioteket will appear (see Figure 5.14a). This modal reuses the text and photos from the slider at the front page (see Figure 5.1), but by creating this modal the user can also read the information after registration and login (FR9).

The second button in the menu opens a modal explaining the terms and conditions of the application (Figure 5.14b). Clicking the third button, the

user receives information on how to give feedback for the application (Figure 5.14c). The fourth button "Rapporter feil", gives the user information on how to send in deviations (FR7) (Figure 5.14d). In the first version of the mobile application the deviations should be sent in by email and handled by the developer. The developer can then choose if she wants to send an email to the book owner or handle the deviation in another way.



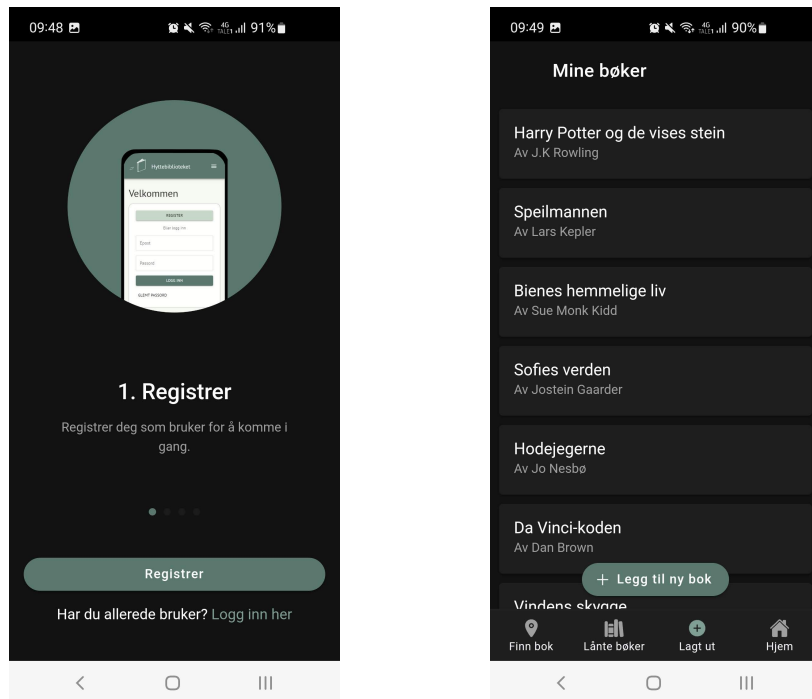
(a) Modal explaining how the app works. (b) Modal explaining terms and conditions.



(c) Information modal on feedback to the system. (d) Information modal for handling deviations.

Figure 5.14: Open modals on the home page.

If the user turns on dark mode on their phone, the application will automatically change its theme to dark mode (FR8) (see Figure 5.15). The application then get a dark background with light text. It was chosen to keep the color palette for the application regardless of dark or light mode as it creates a holistic app.



(a) The first page opening the application. (b) Page with the user's registered and posted books.

Figure 5.15: The application in dark mode.

### 5.1.2 Achievement of functional requirements

This chapter addresses the functional requirement specification defined in section 3.1 and shows whether these requirements are achieved.

- ✓ FR1: Users should be able to register as a user of the system.
- ✓ FR1.1: The system should have authentication with username and password.

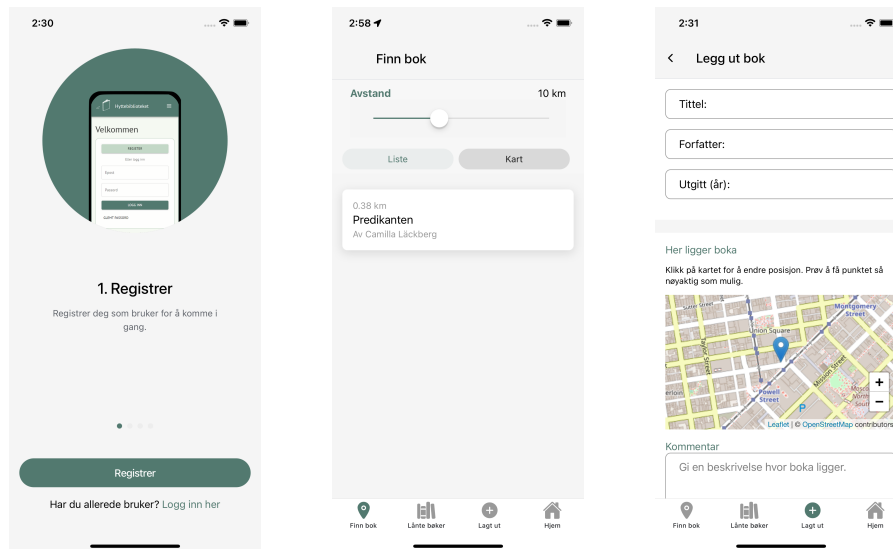
- ✓ FR1.2: The user should confirm the password by entering the password twice.
- ✓ FR1.3: The system should be able to send a new password on email when "forgotten password".
- ✓ FR2: The system should show books posted in an area.
- ✓ FR2.1: The system should show a list of books posted in the area with a distance to where the book is, compared to the user position.
- ✓ FR2.2: The system should show a map with marks where there are books.
- ✓ FR2.3: The user should be able to filter books based on distance from the current location.
- ✓ FR3: The user should be able to add a book based on the current location.
- ✓ FR3.1: The user should have an overview of all books he/she has posted.
- ✓ FR3.2: The user should be able to edit all the books he/she has posted.
- ✓ FR3.3: The user should be able to post a book with another location than the current.
- ✓ FR3.4: The user should be able to register ownership of the book.
- ✓ FR3.4.1: The user should be able to see who borrowed the book if it has ownership.
- ✓ FR4: A user should be able to register that he/she has borrowed a book.
- ✓ FR4.1: Users should be able to see all books they have borrowed.
- ✓ FR4.2: A user should be able to repost a borrowed book.

- FR5: Users should be able to reserve a book.
- FR5.1: A user should cancel a reservation for a book.
- FR6: The user should be able to give comments and ratings on a book.
- FR6.1: The user should be able to see comments and ratings from other users given on a book
- FR7: The user should be able to notice deviations, for example, books that are not at the posted location.
- FR7.1: The system should alert a publisher if there are deviations in his/hers posted books.
- FR8: The system should support dark mode.
- FR9: The system should describe how the system works.

### 5.1.3 Achievement of non-functional requirements

All the non-functional requirements are achieved. All functionality and user interfaces in the mobile application are tested for both Android and iOS (NFR1) (see Figure 5.16).





(a) The first page when opening the application. (b) List of available books. (c) Add a new book.

**Figure 5.16:** Some screenshots from Hyttebiblioteket on an iPhone 12.

The non-functional requirement NFR3 states that the application should follow the Norwegian universal design rules for mobile applications. Therefore, the requirements of usability and availability, in addition to usability test results, controlled implementation of the application.

To achieve a system with high maintainability (NFR2) the code was implemented with a strict folder structure. The folder structure was based on the structure of many popular developers and split the project into parts, causing easy access. The code has also been divided into reusable components with comments explaining the usage.

## 5.2 Usability tests

This chapter represents the results of the ten conducted usability tests. These are the qualitative results from observing the test participants while performing the tasks (stated in section 4.3) and giving feedback regarding the application.

The results presented summarize all the participants' performances and feedback. A table highlighting the problems will be presented at the end of each chapter. The complete reports for the usability tests is found in Appendix D and Appendix E. Finally, the results will be analyzed and evaluated in section 6.1.

### **5.2.1 First usability tests, testing of the web application**

In this chapter the results from the first usability tests, a test of the web application, is presented.

#### **Assignment 1: Create a profile**

All the test participants could perform the first assignment without problems, both registering and logging in. However, it was a moment of uncertainty for two of the participants where they did not understand whether the registration was successful. This uncertainty resulted in the participants clicking around, wondering what happened before they, on their own, figured out that they were logged in. One of the same participants also pondered if the password was correct as he/she did not need to write the password twice.

#### **Assignment 2: Post a book in a chosen area**

The majority of the test participants were able to create and post a book in a chosen area, find the posted book, and change its position. However, one of the participants did not understand why their position had to be allowed to post a book. This user mentioned that a description explaining the need for the position would help. After help from the test leader explaining why the position is necessary, the participant was able to create and post a book in a chosen area. Another test participant encountered a problem when changing the position of the posted book. The test participant understood how to edit the book but only changed the comment, not the position in the map.

**Assignment 3: Find a book using the list**

All the test participants navigated to the list of available books and found a specific book and its location. However, one of the participants tried to navigate by clicking the images on the front page but understood quickly that he/she had to use the navigation bar instead. The test participant then commented that it would be cool if it were possible to click the images at the front page for navigation.

When the participants were asked to register a book as borrowed, one participant got an error, while two other participants did not understand the "Registrer som hentet" button. One assumed that it meant that the book was reserved, while the other assumed that clicking the button would contact the owner so that they could plan a handover of the book.

**Assignment 4: Find a book using the map**

All test participants found the map quickly and naturally clicked the bookmarks for more information about the book. Nevertheless, the second participant would like it if the name of the books were shown on the map without clicking. However, the third participant said; "the map is easy to use and intuitive".

**Assignment 5: Find borrowed books**

All test participants found their borrowed books. A test participant also commented that he/she liked that the system showed all the user's borrowed books when a book was registered as borrowed.

The test participants were then asked to repost a borrowed book. All participants managed to navigate to a borrowed book and understood how they could repost it. However, one of the participants thought the book had to be placed in the same position as it was collected and wanted a "put back" button. After correction from the test leader that the book could be posted wherever the user wanted, the reposting of the book was clear. Although all

the test participants understood how a borrowed book could be reposted the system failed for two of the users.

### **General feedback**

After all the assignments were completed, the test leader performed a semi-structured interview with the test participants regarding their experiences and general feedback on the application. The majority of the test participants commented on how intuitive and straightforward the application was. The participants also commented that the application had a good design with contrasts and colors. Some actions in the app could, however, contain better explanations. For example, two participants misinterpreted what the register as collected button did and wanted a more precise description.

**Table highlighting problems**

Participant No.	Assignment	Problem	Feeling
1	2: Post a book in a chosen area	Do not understand why he/she does not get anything up when not allowing position	Confused
2	1: Create a profile	Do not quite understand if he/she is registered	Confused
2	5.2: Repost a borrowed book	The system failed when clicking repost	Frustrated
3	5.2: Repost a borrowed book	Thinks that the book needs to be reposted at the same location as it was collected	Confused
4	3.2: Register that a book is collected	The system failed when clicking the button	Frustrated
4	5.2: Repost a borrowed book	The system failed when clicking repost	Frustrated
5	1: Create a profile	Wonder if he/she is logged in after creating a profile	Confused
5	3: Find a book using the list	Tries to click on the front page for navigation but use the navigation bar quickly after registering that clicking the front page does not act.	Dejected
5	3.2 Register that a book is collected	Think the book is getting reserved when clicking the button	Confused

**Table 5.1:** A table presenting the problems found in the first usability tests

## 5.2.2 Second usability tests, testing of the mobile application

In this chapter the results from the second usability tests, testing of the mobile application, is presented.

### Assignment 1: Create a profile

All the test participants were able to register as a users in the system without problems. The two participants that participated in the first usability test had already created a profile, and their task was to log in. Unfortunately, the first user did not remember the email that he/she used and had to create a new user. When creating a new user, it was commented that the terms and conditions was hard to see, and should perhaps be placed above the register button. The other participant that created a user during the first usability test did not remember his/her password. This resulted in a test of the forgotten password functionality, and it was commented; "cool that it is so easy to change password".

### Assignment 2: Post a book in a chosen area

When the participants were given the task of posting a book they wanted to give away, several participants took a thinking pause before clicking the correct button in the bottom navigation bar. One participant also commented that it is not entirely evident that posting a new book is underneath the page for posted books.

When registering a new book and editing it, all participants used the map to place the book at a specific location and wrote a comment explaining the book's location. However, one participant used the "next" button on the keyboard and almost skipped the map. Another participant commented that he/she did not understand what ownership meant and suggested that a larger info text could pop up if the user needed an extensive explanation.

**Assignment 3: Find a book using the list**

All participants found the list of available books in the area, and clicked the books naturally to see more information about the book and its location. The last participant mentioned that he/she looked for the book he/she posted, but after selecting another book, the participant commented: "it makes sense that you do not see books that are far away and my own, as one should actually collect it".

**Assignment 4: Find a book using the map**

All participants found the map and dragged the map around to locate books in the area. One of the participants tried to click a specific location without bookmarks, but as nothing happened, the participant understood that he/she had to click the bookmarks instead. Another user tried to find books outside the searched area and did not understand that the search was narrowed due to distance. All participants were able to find a book they wanted to read using the map. A participant also commented that it was clever to have two different views, a list if one wanted to find a book based on the title and the map to find a book based on its position.

**Assignment 5: Collect book**

After locating books on the map and in the list, the participants were asked to register that they borrowed a book. All participants completed this without any problems. After the participants borrowed a book, they were asked to find the borrowed book. This was easy as the system redirected the participants to their borrowed books after they registered a book as borrowed. All participants were also able to regret borrowing the book, even though the first participant was unsure how this was solved as he/she tested a version without the "Angre registrering" button. He/she understood after a while that the book should be posted in the same place as it was collected but wanted a regret button instead.

When the participants were asked to repost the book they borrowed, all

participants used the map to place the book at a new location and wrote a comment describing the book's location. However, some of the participants commented that it is weird that they can delete the book as it is not their book, and mentioned that they would not post their books if others could delete them from the system.

### **Assignment 6: Send in deviation**

All participants eventually found the email for deviations. Some of the participants tried first to find a way to send in a deviation from a specific book. One of the participants even believed that the solution was to delete the book from the system. As he/she tried on a book with ownership, which is impossible to delete, he/she eventually click the home button to find the button to report deviations. Another participant wanted to click the email and be redirected to a blank email.

### **General feedback**

After the assignments were completed, the test leader performed a semi-structured interview. This interview was created to get feedback about the participants' experiences and general feedback for the application. The repetitive feedback from the participants was that the system was responsive, intuitive, and had an excellent design without disturbing elements. One of the participants also comments that he/she likes the concept where the user can know which books that are located in a specific location and that this differs from today's solution. Some participants also wanted new features like; a wish list for books, filtering by genre or title, and contacting the book's publisher to notify deviations.



**A table highlighting the problems**

Participant No.	Assignment	Problem	Feeling	Previous participant
1	2: Post a book in a chosen area	Uses the next button on the keyboard, which navigates to the next input field, resulting in the user not seeing the map.		No
1	5.3: Regret borrowing the book	First thinks that the book should be deleted, but later understands that the book should be reposted at the same location as it was collected.	Confused.	No
2	2: Post a book in a chosen area	Do not understand what ownership of a book means.	Confused	No
2	4: Find a book using the map	Tries to click a specific place on the map to see books, even though there are no marks for books here.	Confused	No
3	4: Find a book using the map	Tries to find a book outside the searched distance, do not understand that the search is narrowed due to distance.	Confused	No
4	2: Post a book in chosen area	Unclear that one posts a new book under "posted books".	Confused	No

5	6: Send in deviation	Tries to delete the book, and tries to find a way to send in deviation under a specific book	Frustrated	Yes
---	----------------------	--	------------	-----

**Table 5.2:** A table presenting the problems found in the second usability tests

# Chapter 6

## Discussion

### 6.1 Analysis of usability test results

This chapter focuses on the usability test results and the problems revealed in the application. It is distinguished between errors and problems. If the system has a software bug, a bug in the code which causes the system to behave in unintended ways, it is defined as an error. A problem relates to choices regarding functions or design that confuses the user, such that the users are unable to use the application as intended. The problems are categorized into [125]:

- Critical: impossible for users to complete tasks
- Serious: frustrating for many users
- Minor: Annoying, but not going to drive users away

The analysis and evaluation are based on the results from the usability tests presented in Table 5.1 and Table 5.2.

#### 6.1.1 Software bugs

During the web application's second usability test it was discovered a software bug. It appeared as the test participant tried to repost a book that he/she had borrowed. The user was able to fill out the necessary information and show that he/she understood the task before the bug occurred. The following usability test was then put on hold to fix the bug. As the bug

occurred at a test participant's profile, the developer tried to recreate the bug at a different profile. Unfortunately, this was not possible, and it was impossible to fix the bug before it occurred again.

At the fourth usability test of the web application, a new software bug occurred. The participant was unable to collect a specific book, but worked when he/she tried another book. Later, during the same usability test, the participant faced the same bug as the second participant. He/she was unable to repost a borrowed book. As the bug had occurred twice, it was possible for the developer to see a pattern. The last usability test of the web application was therefore on hold until these bugs were fixed. The developer logged into her account to fix the bugs. By collecting the same book the participant had trouble with, it was clear that the bug only occurred for books created in the previous usability test. Since the other bug also occurred with books created in other usability tests, it was clear that it was the same issue here. The resolution was then to update the object type for a book created in the code, enforcing a compilation error if the same code error happened again.

The usability tests for the mobile application did not detect any software bugs.

### **6.1.2 Critical problems**

During the tests, some critical problems were discovered which hindered the participants from completing the tasks without help. For example, the first test participant for the web application did not understand why he/she needed to allow location and therefore declined it. However, when the user declined the location, the application showed a blank page. There was no information for the participant explaining that the blank page came as a result from the location being declined, and the test leader had to interfere. Therefore, a text explaining to the user that location is necessary for using the application was implemented.

Another critical problem in the web application was that some participants expected the book to be reserved when they registered it as collected. The participants, therefore, expected that the location of the book would be available after the registration. This is a critical problem as users can register books as collected but not collect them. Therefore, an information page for the loaned books and their position should be implemented. This page would also make it possible for the users to reserve books.

During the mobile application tests, a critical problem occurred when the first participant was asked to regret reserving a book. To solve this assignment, the participant tried to delete the book. This would have resulted in inconsistency with the books posted in the application. Therefore, before executing any more usability tests, it was decided to implement a button that could regret the reservation of a book.

Another critical problem in the mobile application appeared when the last participant was asked to send in a deviation. The participant was sure that this should be done from a specific book. When he/she did not find any buttons to send in a deviation, he/she tried to delete the book. Deleting books can potentially cause problems as books be lost in nature, and can be solved by creating a deviation button at the page for a specific book.

### **6.1.3 Serious problems**

The usability tests also revealed serious problems which hindered the application from being as efficient and straightforward as intended. For example, in the web application, one participant thought he/she had to re-post a borrowed book at the same place as he/she picked it up. This misunderstanding can result in fewer users as the threshold for borrowing and reposting books increases. However, as this was only one participant's mindset, it was decided that this could be solved by creating a better explanation when reposting a book.

In the mobile application, the first participant almost skipped the map

when registering a new book. He/she used the next button on the keyboard, which navigated to the following input field. The next button should, therefore, be disabled at the input before the map, forcing the user to see the map.

#### **6.1.4 Minor problems**

Some minor problems were also detected during the usability tests. These problems can confuse the users, but it is believed that the confusion will disappear after the system is used once. For example, in the web application, two of the users were uncertain if the registration was successful and if they were logged in. They quickly understood that they were, but it might be wise to see the available books in the area as soon as the user is authenticated to avoid confusion.

In the mobile application, several test participants thought it was unclear that posting a new book was found underneath the "Lagt ut" tab. Even though this confusion only appeared the first time the participants used the system, it should be considered to have a more explaining tab name.

In the mobile application, a participant tried to click outside the bookmarks on the map to find books at the location. As this was not possible, the user understood that he/she had to click the bookmarks instead. Even though this was a one-time happening, creating an explanation for the user can prevent other users from making the same mistake.

During the mobile application test, it was also revealed that some of the participants did not fully understand the concept of ownership. To clarify this concept, a more precise information text should be available. This text can, for example, be visible by creating a "read more" button.

Another participant from the mobile application test, also tried to find posted book outside the searched area by using the map. The searched area should therefore become more highlighted for the user. This highlight-

ing can be solved by, for example, making the map outside the searchable area grey or making it impossible for the user to zoom out more than the searchable area.

## 6.2 Evaluation of process

### 6.2.1 Design phase

*Hyttebiblioteket's* design was developed during the design phase and was created in the design and prototyping tool Figma. The design phase has resulted in a holistic and consistent design. Even though the prototype was created from the results of the usability tests on the web application, it can not be ruled out that a usability test on the prototype in Figma could have resulted in better usability. By performing a usability test on the prototype, it would be created two iterations of the prototype in Figma. The first iteration would contain a prototype to find the worst usability mistakes. In the second iteration, a second design prototype based on the results from the usability test on the first iteration would be created. Creating two iterations in Figma, the implemented design could have contained better usability and created better results during the functional usability tests performed at the end of the project.

### 6.2.2 Development phase

The development phase started after the design phase. In this project, was a combination of different methodologies used. As mentioned in section 4.1, general milestones were created early in the project. This resulted from a lack of knowledge about mobile application development and suited the project well. Most of the milestones were reached within their deadline, except for the execution of usability tests for the mobile application. Due to Easter and the test participants' schedules, the usability tests were delayed. As these delays resulted in a delay in the process, it was easy to change the milestones and create new ones.

As mentioned in section 4.1, the development phase also used elements from agile methodology to organize and prioritize tasks during the development process. Due to an agile mindset, has it been easy to implement changes due to changes in the requirement specification and results from the first usability tests. As the development team only consisted of one developer, and the application was simple, another solution than the chosen would have resulted in a significant change of the development phase.

### **6.2.3 The accomplishment of usability tests**

The usability tests were planned in advance and contained information of how the tests should be conducted and which application elements that should be tested. As mentioned in section 4.3, a test plan was created. This test plan was created to be prepared such that the participants got assignments closely resembled to real scenarios. Creating a plan also ensured that all necessary information got conveyed to the participants, resulting in all participants having the same base and assignments.

Before starting the usability tests, the test leader conducted a usability test according to the test plan. This test was to ensure that the test plan was well designed and that errors and critical problems, like software bugs, were uncovered and fixed before the usability test. However, some bugs and critical problems were uncovered during the usability tests (see section 6.1 for examples and elaborating information). This shows the importance of usability tests, as it reveals problems due to usability and functionality in a more realistic scenario. The users may also use the system differently than the developer thinks, which can result in a poor usability. Therefore, as mentioned above in subsection 6.2.1, it might be advantageous to have usability tests earlier in the design phase.



## 6.3 Evaluation of the technical implementation

### 6.3.1 Usage of Ionic and Capacitor

The mobile application is developed using Ionic with React Typescript and Capacitor. Using React with Typescript, a static type checker reduces the type errors and simplifies the debugging. However, as Typescript requires types for all variables, function parameters, and object properties, it has complicated the programming due to type declarations for third-party libraries.

Using Ionic with Capacitor has resulted in one codebase for a mobile application for both Android and iOS. This has minimized the amount of code written, as the need for platform-specific code is minimized. In addition, this makes it easy to fix errors or expand the solution, as changes only need to be implemented once for both Android and iOS. However, it is in some cases required to write platform-specific code. This code can be implemented by creating a Capacitor plugin. Such a plugin is developed in Java for Android and Swift for iOS and is connected with Capacitor.

As Ionic with Capacitor is relatively new, several plugins have not been created yet. For example, the plugin `@capacitor/geolocation` used in the project to track the device's current position using GPS was posted in 2021 [126]. This is, therefore, one of the disadvantages of using Ionic and Capacitor. In addition, both technologies are relatively new, and functionality that other frameworks have good support for might be deficient or non-existing in Ionic and Capacitor.

### 6.3.2 Backend

It was chosen that the backend of the mobile application should be based on the backend created for the web application created in the specialization project [1]. This means that services for authentication and data storage is provided by Firebase as a form of Backend-as-a-Service. As in the spe-

cialization project, the current functionality and specified requirements of the application is provided by services from Firebase. However, if it in the future is desirable with more functionality in the backend, or functionality resulting in a restructuring that exceeds the services of Firebase, it could be necessary to move some of the logic to a dedicated server. As Firebase offers solutions to use another server together with authentication and storage, the main logic for authentication and storage can still use Firebase's services.

## 6.4 Evaluation of the product

In section 5.1 the application design, and requirement results were presented. These results shows that most functional requirements were successfully implemented. Creating a prioritized list of the necessary and wanted functionality ensured that even though all functionality was not implemented, the result would be a functioning product.

The developed mobile application contains a method for sending in a deviation to the developer. This has been one of the main uncertainties during the development process, if sending deviation to the book owner should be prioritized higher. In this version, a dialog telling the user to send an email regarding the problem and potentially the related book was implemented. This solution was a result of the project's time limit. From usability tests and discussion with possible users of the system, it was clear that it is more intuitive to report a problem regarding a specific book at the page for that book, as well as the person reporting the problem can communicate with the book owner. However, if this requirement had been prioritized higher, a different requirement implemented today might not have been fulfilled. As the implemented requirements are more critical for the application to function, it was clear that the prioritizing was correct.

Today, we live in a consumer society, even though the earth breaks global warming records. To help the planet capture more carbon dioxide by reduc-

ing wood pulp, an application for sharing books is created. This application is a small contribution, and uses elements from circular economy. The application is developed as a supplement to the other book-sharing solutions that exists. This application, Hyttebiblioteket, has a lower threshold for collecting books, as one does not need to contact the owner. In addition, it minimizes the possibility of a wasted trip, as the user knows which books are available at a given location. The focus for the application is mainly on second home areas as this is known to be a place where many people read books and where there exist few widespread ways of sharing books. By usability tests, and informal talks it was clear that this application would satisfy a demand that does not exist today.



# Chapter 7

## Conclusion

As presented in section 1.2, the overall goal was to help the planet to capture more carbon dioxide by reducing the use of wood pulp for producing books by using elements from the circular economy. To make this small change for the planet the idea was to create an application, so book readers could share books instead of buying new ones. The application that is made in this project is an improvement of the "Hyttebiblioteket" that evolved in the specialization project in the fall of 2021 [1].

Based on existing solutions, personal experience, and feedback given from the usability tests of the specialization project, the idea was to create a mobile application for Hyttebiblioteket. The application makes it possible for users to post books and collect available books in the area. Furthermore, the application differs from the existing solutions by collecting books without contacting the owner of a specific book, showing which books that are available at a specific location, and the possibility for reserving an available book. The feedback and results from the usability tests and more informal talks with friends and family that have used the application or liked the idea behind the application indicated that the system's functionality most likely would lower the threshold to post books and find books in the area. Even though the usability tests have given positive results it seems that practical use of the application in second home areas can result in needs that are not fulfilled in this project. The application may therefore need

further development and maintenance after release if it should capture all the needs from the users and possible users. However, this application will benefit for users book sharing within second home areas compared to the existing solutions that are in the market as far as I have had possibility to study and find different solution that exist. In the next chapter, I will outline some of the possible improvements and changes that can be further developed in the "Hyttebiblioteket".

# Chapter 8

## Further work

In the last chapter I summarized the project with the concluding marks. In this chapter I'll have a nearer focus on how to improve the app's missing and desired functionality. As this master's project has limitations in terms of both time and scope, there was some feedback from the test groups that it was not possible to implement as it would take too much time and make it impossible to reach the goal within the given time frame. The first extra functionality that should be implemented is the possibility for users to contact the owner of the books regarding deviations by themselves. This lack of functionality was a repeated feedback from several of the test participants during the usability tests. Implementing such a functionality would also minimize the work for the developer as she does not need to distribute to the book owner the problem with the book or actually need to fix the inquiries by herself.

One of the next steps of the process should be to release the application on platforms that most users and future users use for downloading applications. As the mobile application is created for Android and iOS, it needs to be released on Google Play and App Store so that it is easy for people to start using the application.

The mobile application created during this project is based on the idea of the web application created in the specialization project [1]. This project

has resulted in both making the app and implement a lot of more functionality and graphic elements which give users a better experience of the Hyttebiblioteket. The extra functionality developed in the app is based on input from the usability tests of the first version on the web application. Therefore, it would be natural as a further development to implement the new functionality also in the web application. By making this change, the user experience will be almost the same whether one uses the app or the web solution. Improving the user experience in such a way can lead to a more seamless use of the total package with the web application and app solution.



# Bibliography

- [1] C. Velvin, "Hyttebiblioteket," Specialization Project at Norwegian University of Science and Technology, Trondheim., 2021.
- [2] E. Schiro and E. Haraldsrud, *En av fire leser bøker daglig*, Retrieved 11 May 2022, 2019. [Online]. Available: <https://www.ssb.no/kultur-og-fritid/artikler-og-publikasjoner/en-av-fire-leser-boker-daglig>.
- [3] F. in South Africa, *Earth day 2022 - a climate case for wood, pulp and paper*, Retrieved 22 May 2022, 2022. [Online]. Available: <https://forestry.co.za/earth-day-2022-a-climate-case-for-wood-pulp-and-paper/>.
- [4] A. M. of Natural History, *What is biodiversity?* Retrieved 25 May 2022. [Online]. Available: <https://www.amnh.org/research/center-for-biodiversity-conservation/what-is-biodiversity>.
- [5] Y. Wurmser, *Ark bokhandel as*, Retrieved 11 May 2022, 2020. [Online]. Available: <https://www.emarketer.com/content/the-majority-of-americans-mobile-time-spent-takes-place-in-apps>.
- [6] Finn.no, *Finn.no – mulighetenes marked*, Retrieved 14 March 2022. [Online]. Available: [https://www.finn.no/finn/article/finn\\_about\\_us?template=templates/static\\_templat.jsp](https://www.finn.no/finn/article/finn_about_us?template=templates/static_templat.jsp).
- [7] F. AS, *Når bør du legge ut annonsene dine?* Retrieved 14 March 2022, 2020. [Online]. Available: <https://www.finn.no/bedriftskunde/torget/nytt-om-finn-torget/nar-bor-du-legge-ut-annonsene-dine>.

- [8] L. F. Library, *The history of little free library*, Retrieved 17 March 2022. [Online]. Available: <https://littlefreelibrary.org/ourhistory/>.
- [9] L. F. Library, *Little free library's mission and vision*, Retrieved 14 March 2022. [Online]. Available: <https://littlefreelibrary.org/about/>.
- [10] Megan, *Got little free library envy? start here*. Retrieved 17 March 2022. [Online]. Available: <https://littlefreelibrary.org/got-little-free-library-envy-start-here/>.
- [11] *Bookcrossing*, Retrieved 21 March 2022. [Online]. Available: <https://www.bookcrossing.com/>.
- [12] W. Awards, *Community websites and mobile sites*, Retrieved 23 November 2021. [Online]. Available: <https://winners.webbyawards.com/winners/websites-and-mobile-sites/general-websites-and-mobile-sites/community?years=16>.
- [13] *Booksharing.app*, Retrieved 21 March 2022. [Online]. Available: <https://www.bookcrossing.com/>.
- [14] G. Rachiele, *Software development methodologies*, Retrieved 25 January 2022, 2018. [Online]. Available: <https://medium.com/@gianpaul.r/software-development-methodologies-a856883a7630>.
- [15] N. L. Russo, J. L. Wynekoop, and D. B. Walz, *The use and adaptation of system development methodologies*, Retrieved 26 January 2022, 1995. [Online]. Available: <https://www.andrews.edu/~vyhmeisr/papers/sdm.html>.
- [16] S. Shaikh and S. Abro, "Comparison of traditional and agile software development methodology: A short survey," eng, *International Journal of Software Engineering and Computer Systems (Pahang)*, vol. 5, no. 2, pp. 1–14, 2019, ISSN: 2289-8522.
- [17] Airbrake, *Waterfall model: What is it and when should you use it?* Retrieved 01 February 2022. [Online]. Available: <https://airbrake.io/blog/sdlc/waterfall-model>.

- [18] *Software engineering | classical waterfall model*, Retrieved 31 January 2022, 2019. [Online]. Available: <https://www.geeksforgeeks.org/software-engineering-classical-waterfall-model/>.
- [19] M. Alexander, *Agile vs. waterfall: Project methodologies compared*, Retrieved 01 February 2022. [Online]. Available: <https://www.proquest.com/docview/2448481960?pq-origsite=primo>.
- [20] D. Hughey, *The traditional waterfall approach*, Retrieved 13 October 2021. [Online]. Available: <https://www.umsl.edu/~hugheyd/is6840/waterfall.html>.
- [21] A. E. Cloud, *Waterfall methodology*, Retrieved 02 February 2022. [Online]. Available: <https://www.workfront.com/project-management/methodologies/waterfall>.
- [22] Tutorialspoint, *Sdlc - waterfall model*, Retrieved 02 February 2022. [Online]. Available: [https://www.tutorialspoint.com/sdlc/sdlc\\_waterfall\\_model.htm](https://www.tutorialspoint.com/sdlc/sdlc_waterfall_model.htm).
- [23] L. C. Team, *The pros and cons of waterfall methodology*, Retrieved 01 February 2022. [Online]. Available: <https://www.lucidchart.com/blog/pros-and-cons-of-waterfall-methodology>.
- [24] I. Mavuru, *Traditional vs. agile software development methodologies*, Retrieved 02 February 2022, 2018. [Online]. Available: <https://www.lucidchart.com/blog/pros-and-cons-of-waterfall-methodology>.
- [25] P. Rannikko, *User-centered design in agile software development*, Retrieved 02 February 2022, 2011. [Online]. Available: <https://www.lucidchart.com/blog/pros-and-cons-of-waterfall-methodology>.
- [26] A. C. Spataru, *Agile development methods for mobile applications*, Retrieved 03 February 2022, 2010. [Online]. Available: <https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.186.1292&rep=rep1&type=pdf>.

- [27] P. Varhol, *To agility and beyond: The history—and legacy—of agile development*, Retrieved 13 October 2021. [Online]. Available: <https://techbeacon.com/app-dev-testing/agility-beyond-history-legacy-agile-development>.
- [28] K. Beck, M. Beedle, A. van Bennekum, A. Cockburn, W. Cunningham, M. Fowler, J. Grenning, J. Highsmith, A. Hunt, R. Jeffries, J. Kern, B. Marick, R. C. Martin, S. Mellor, K. Schwaber, J. Sutherland, and D. Thomas, *Manifesto for agile software development*, Retrieved 03 February 2022, 2001. [Online]. Available: <https://agilemanifesto.org/>.
- [29] S. Rancic, *What is agile development methodology?* Retrieved 13 October 2021, 2021. [Online]. Available: <https://www.price2spy.com/blog/agile-development-methodology/>.
- [30] E. C. Foster and S. V. Godbole, “Introduction to database systems,” in *Database Systems: A Pragmatic Approach*. Berkeley, CA: Apress, 2014, pp. 3–11, ISBN: 978-1-4842-0877-9. DOI: 10.1007/978-1-4842-0877-9\_1. [Online]. Available: [https://doi.org/10.1007/978-1-4842-0877-9\\_1](https://doi.org/10.1007/978-1-4842-0877-9_1).
- [31] Amazon, *What is a relational database?* Retrieved 25 October 2021. [Online]. Available: <https://aws.amazon.com/relational-database/>.
- [32] E. Codd, “A relational model of data for large shared data banks,” *eng, Communications of the ACM*, vol. 13, no. 6, pp. 377–387, 1970, ISSN: 0001-0782.
- [33] K. T. Hansen and T. Mallaug, *Databaser*. Gyldendal Norsk Forlag AS, 2008, ISBN: 9788205381056.
- [34] Microsoft, *Non-relational data and nosql*, Retrieved 03 November 2021, 2021. [Online]. Available: <https://docs.microsoft.com/en-us/azure/architecture/data-guide/big-data/non-relational-data>.

- [35] G. Batschinski, *Backend as a service list*, Retrieved 04 March 2022, 2019. [Online]. Available: <https://george-51059.medium.com/backend-as-a-service-list-9104bdce845e>.
- [36] Cloudflare, *What is baas? | backend-as-a-service vs. serverless*, Retrieved 04 March 2022. [Online]. Available: <https://www.cloudflare.com/learning/serverless/glossary/backend-as-a-service-baas/>.
- [37] Anurag, *Top 5 cloud platforms and solutions to choose from*, Retrieved 07 March 2022, 2020. [Online]. Available: <https://www.newgenapps.com/blogs/top-5-cloud-platforms-and-solutions-to-choose-from/>.
- [38] K. Taylor, *Top 14 backend-as-a-service providers*, Retrieved 07 March 2022. [Online]. Available: <https://www.hitechnectar.com/blogs/top-14-backend-as-a-service-providers/>.
- [39] B. Fish, *Hva er ui og ux, og hvorfor er det så viktig for nettsiden din?* Retrieved 04 March 2022, 2021. [Online]. Available: <https://bigfish.no/bransjesnakk/hva-er-ui-og-ux-og-hvorfor-er-det-sa-viktig-for-nettsiden-din/>.
- [40] N. Babich, *What is interaction design & how does it compare to ux?* Retrieved 28 February 2022, 2019. [Online]. Available: <https://xd.adobe.com/ideas/principles/human-computer-interaction/what-is-interaction-design/>.
- [41] D. U. C. of Computing & Informatics, *Human-computer interaction (hci) & user experience (ux)*, Retrieved 04 March 2022. [Online]. Available: <https://drexel.edu/cci/academics/graduate-programs/human-computer-interaction-ux/>.
- [42] D. Saffer, *Designing for interaction : creating innovative applications and devices*, eng, 2nd ed., ser. Voices that matter. Berkeley, Calif: New Riders, 2010, ISBN: 9780321643391.
- [43] J. Kolko, *Thoughts on Interaction Design*, eng. Burlington: Elsevier Science & Technology, 2009, ISBN: 9780123786241.

- [44] A. Wrålsen, "Introduksjon til interaksjonsdesign," Artikkel gitt i TDAT2003 Systemutvikling 2 med web-applikasjoner, NTNU, 2016.
- [45] H. Sharp, Y. Rogers, and J. Preece, *Interaction design : beyond human-computer interaction*, eng, Fifth edition. Indianapolis, IN: Wiley, 2019, ISBN: 9781119547259.
- [46] D. Norman, *Signifiers, not affordances*, Retrieved 02 March 2022, 2018. [Online]. Available: [https://jnd.org/signifiers\\_not\\_affordances/](https://jnd.org/signifiers_not_affordances/).
- [47] *Lov om likestilling og forbud mot diskriminering (lov-2017-06-16-51)*, 2017. [Online]. Available: <https://lovdata.no/dokument/NL/lov/2017-06-16-51>.
- [48] uutilsynet, *Universell utforming av apper*, Retrieved 04 March 2022. [Online]. Available: <https://www.uutilsynet.no/regelverk/universell-utforming-av-apper/230>.
- [49] K. Moran, *Usability testing 101*, Retrieved 28 January 2022, 2019. [Online]. Available: <https://www.nngroup.com/articles/usability-testing-101/>.
- [50] S. Krug, *Rocket surgery made easy : the do-it-yourself guide to finding and fixing usability problems*, eng. Berkeley: New Riders, 2010, ISBN: 9780321657299.
- [51] A. Wrålsen, "Interaksjonsdesign som prosess 2," Artikkel gitt i TDAT2003 Systemutvikling 2 med web-applikasjoner, NTNU, 2016.
- [52] D. Grant, *Quantitative usability test*, Retrieved 24 February 2022. [Online]. Available: <https://ux-everything.com/usability-test-quantitative/>.
- [53] R. Budiu, *Quantitative vs. qualitative usability testing*, Retrieved 24 February 2022, 2017. [Online]. Available: <https://www.nngroup.com/articles/quant-vs-qual/>.
- [54] P. Thornton, *How to conduct user interviews*, Retrieved 28 January 2022, 2019. [Online]. Available: <https://uxdesign.cc/how-to-conduct-user-interviews-fe4b8c34b0b7>.

- [55] A. Wrålsen, "Interaksjonsdesign som prosess 1," Artikkel gitt i TDAT2003 Systemutvikling 2 med web-applikasjoner, NTNU, 2016.
- [56] A. Mroczkowska, *What is a mobile app? | app development basics for businesses*, Retrieved 07 March 2022. [Online]. Available: <https://www.thedroidsonroids.com/blog/what-is-a-mobile-app-app-development-basics-for-businesses>.
- [57] E. Stevens, *What is the difference between a mobile app and a web app?* Retrieved 07 March 2022, 2021. [Online]. Available: <https://careerfoundry.com/en/blog/web-development/what-is-the-difference-between-a-mobile-app-and-a-web-app/>.
- [58] StatCounter, *Mobile operating system market share worldwide*, Retrieved 07 March 2022. [Online]. Available: <https://gs.statcounter.com/os-market-share/mobile/worldwide>.
- [59] A. W. Services, *What is mobile application development?* Retrieved 07 March 2022. [Online]. Available: <https://aws.amazon.com/mobile/mobile-application-development/>.
- [60] C. Mobile, *What is native mobile app development?* Retrieved 07 March 2022. [Online]. Available: <https://clearbridgemobile.com/benefits-of-native-mobile-app-development/>.
- [61] A. Marchuk, *Native vs cross-platform development: Pros & cons revealed*, Retrieved 08 March 2022. [Online]. Available: <https://www.uptech.team/blog/native-vs-cross-platform-app-development>.
- [62] I. Naylor, *App monetization: A guide on how to monetize an app*, Retrieved 08 March 2022, 2019. [Online]. Available: <https://www.leanplum.com/blog/free-app-monetization-methods/>.
- [63] Statista, *Most-used mobile app monetization models according to mobile developers worldwide in 2017*, Retrieved 08 March 2022, 2017. [Online]. Available: <https://www.statista.com/statistics/297024/most-popular-mobile-app-monetization-models/#statisticContainer>.

- [64] C. Mobile, *How to choose the right mobile app monetization strategy*, Retrieved 08 March 2022. [Online]. Available: <https://clearbridgemobile.com/choosing-the-right-app-monetization-strategy/>.
- [65] I. E. Team, *What is in-app advertising? (definition and tips)*, Retrieved 08 March 2022. [Online]. Available: <https://www.indeed.com/career-advice/career-development/in-app-advertising>.
- [66] J. Ewen, *Best guide to mobile app monetization 2021 – stats, strategies & insight*, Retrieved 08 March 2022, 2022. [Online]. Available: <https://www.tamoco.com/blog/ultimate-app-monetization-guide/>.
- [67] A. Inc., *In-app purchase*, Retrieved 09 March 2022. [Online]. Available: <https://developer.apple.com/in-app-purchase/>.
- [68] M. Grguric, *In-app purchases guide for successful mobile game monetization*, Retrieved 09 March 2022, 2022. [Online]. Available: <https://www.blog.udonis.co/mobile-marketing/mobile-games/in-app-purchases>.
- [69] Google, *5 monetization strategies for your app*, Retrieved 09 March 2022. [Online]. Available: <https://admob.google.com/home/resources/5-app-monetization-strategies-to-grow-and-monetize-your-app/>.
- [70] FAM, *4 reasons to adopt the dark mode in your app*, Retrieved 21 April 2022, 2021. [Online]. Available: <https://medium.com/geekculture/4-reasons-to-adopt-the-dark-mode-in-your-app-b235632c550c>.
- [71] T. for universell utforming av ikt, *Universell utforming av apper*, Retrieved 23 March 2022. [Online]. Available: <https://www.uutilsynet.no/regelverk/universell-utforming-av-apper/230>.
- [72] D. E. Santo, *Top 5 main agile methodologies: Advantages and disadvantages*, Retrieved 24 March 2022, 2022. [Online]. Available: <https://www.xpand-it.com/blog/top-5-agile-methodologies/>.



- [73] P. Deemer, G. Benefield, C. Larman, and B. Vodde, *The scrum primer*, Artikkel gitt i TDAT2003 Systemutvikling 2 med web-applikasjoner, NTNU, 2009.
- [74] S. M. Nes, *En kort introduksjon til scrum*, Retrieved 25 March 2022, 2019. [Online]. Available: <https://www.visma.no/blogg/en-kort-introduksjon-til-scrum/>.
- [75] D. Radigan, *The product backlog: Your ultimate to-do list*, Retrieved 25 March 2022. [Online]. Available: <https://www.atlassian.com/agile/scrum/backlogs>.
- [76] A. E. Cloud, *Daily stand-up meetings*, Retrieved 25 March 2022. [Online]. Available: <https://www.workfront.com/project-management/methodologies/agile/daily-stand-up>.
- [77] D. Radigan, *What is kanban?* Retrieved 25 May 2022. [Online]. Available: <https://www.atlassian.com/agile/kanban>.
- [78] *Teamgantt: Online gantt chart maker software*, Retrieved 24 March 2022. [Online]. Available: <https://www.teamgantt.com/>.
- [79] *Trello hjelper arbeidsgruppene med å jobbe videre*. Retrieved 25 March 2022. [Online]. Available: <https://trello.com/nb>.
- [80] R. Budiu, *Dark mode vs. light mode: Which is better?* Retrieved 18 March 2022, 2020. [Online]. Available: <https://www.nngroup.com/articles/dark-mode/>.
- [81] F. Bianchi, *Coolors*, Retrieved 18 March 2022. [Online]. Available: <https://coolors.co>.
- [82] Finn.no, *Finn*, Mobile app. [Online]. Available: <https://www.finn.no/>.
- [83] *Instagram*, Mobile app. [Online]. Available: <https://www.instagram.com/>.
- [84] *Glede*, Mobile app. [Online]. Available: <https://www.glede.app/>.
- [85] I. D. Foundation, *Prototyping*, Retrieved 18 March 2022. [Online]. Available: <https://www.interaction-design.org/literature/topics/prototyping>.

- [86] E. Esposito, *Low-fidelity vs. high-fidelity prototyping*, Retrieved 18 March 2022, 2018. [Online]. Available: <https://www.invisionapp.com/inside-design/low-fi-vs-hi-fi-prototyping/>.
- [87] Usability.gov, *Prototyping*, Retrieved 18 March 2022. [Online]. Available: <https://www.usability.gov/how-to-and-tools/methods/prototyping.html>.
- [88] *Figma: The collaborative interface design tool*, Retrieved 21 March 2022. [Online]. Available: <https://www.figma.com/>.
- [89] *Adobe xd - fast & powerful ui/ux design & collaboration tool*, Retrieved 21 March 2022. [Online]. Available: <https://www.adobe.com/no/products/xd.html>.
- [90] J. Nielsen, *Why you only need to test with 5 users*, Retrieved 16 March 2022, 2000. [Online]. Available: <https://www.nngroup.com/articles/why-you-only-need-to-test-with-5-users/>.
- [91] D. Renwick, *How many participants do i need for qualitative research?* Retrieved 16 March 2022, 2019. [Online]. Available: <https://blog.optimalworkshop.com/how-many-participants-do-i-need-for-qualitative-research/>.
- [92] E. Toftøy-Andersen, *Praktisk brukertesting*, nob. Oslo: Cappelen Damm akademisk, 2011, ISBN: 9788202343507.
- [93] *Nativescript*, Retrieved 22 March 2022. [Online]. Available: <https://nativescript.org/>.
- [94] *Flutter - build apps for any screen*, Retrieved 22 March 2022. [Online]. Available: <https://flutter.dev/>.
- [95] *React native - learn once, write anywhere*, Retrieved 22 March 2022. [Online]. Available: <https://reactnative.dev/>.
- [96] *Ionic framework: Cross-platform mobile app development*, Retrieved 22 March 2022. [Online]. Available: <https://ionicframework.com/>.

- [97] Academind, *Which one is best for you? flutter, react native, ionic or nativescript?* Retrieved 22 March 2022, 2019. [Online]. Available: <https://youtube.com/watch?v=PKRXbLnFXXk>.
- [98] Altexsoft, *The good and the bad of ionic mobile development*, Retrieved 22 March 2022, 2019. [Online]. Available: <https://www.altexsoft.com/blog/engineering/the-good-and-the-bad-of-ionic-mobile-development/>.
- [99] Ionic, *Introduction to ionic*, Retrieved 22 March 2022. [Online]. Available: <https://ionicframework.com/docs>.
- [100] Ionic, *What is hybrid app development?* Retrieved 22 March 2022. [Online]. Available: <https://ionic.io/resources/articles/what-is-hybrid-app-development>.
- [101] M. Lynch, *Capacitor vs cordova: Hybrid mobile app development*, Retrieved 22 March 2022. [Online]. Available: <https://ionic.io/resources/articles/capacitor-vs-cordova-modern-hybrid-app-development>.
- [102] M. Lynch, *How capacitor works*, Retrieved 22 March 2022, 2020. [Online]. Available: <https://capacitorjs.com/blog/how-capacitor-works>.
- [103] M. Pekarsky, *Does your web app need a front-end framework?* Retrieved 22 March 2022, 2020. [Online]. Available: <https://stackoverflow.blog/2020/02/03/is-it-time-for-a-front-end-framework/>.
- [104] Google, *Angular*, Retrieved 22 March 2022. [Online]. Available: <https://angular.io/>.
- [105] F. O. Source, *React – a javascript library for building user interfaces*, Retrieved 22 March 2022. [Online]. Available: <https://reactjs.org/>.
- [106] Vue.js, *Vue.js - the progressive javascript framework*, Retrieved 22 March 2022. [Online]. Available: <https://vuejs.org/>.

- [107] V. S. Bisht, *Using leaflet to add maps to your ionic application*, Retrieved 24 March 2022, 2019. [Online]. Available: <https://dev.to/bviveksingh/using-leaflet-to-add-maps-to-your-ionic-application-335g>.
- [108] B. B. Bayhan, *React form libraries comparison: Formik vs react hook form*, Retrieved 24 March 2022, 2022. [Online]. Available: <https://apiumhub.com/tech-blog-barcelona/react-form-libraries-comparison-formik-vs-react-hook-form/>.
- [109] LogRocket, *Building forms with formik in react*, Retrieved 24 March 2022, 2022. [Online]. Available: <https://blog.logrocket.com/building-forms-formik-react/>.
- [110] *Mern stack explained*, Retrieved 23 March 2022. [Online]. Available: <https://www.mongodb.com/mern-stack>.
- [111] *Firebase*, Retrieved 23 March 2022. [Online]. Available: <https://firebase.google.com/>.
- [112] C. Esplin, *What is firebase?* Retrieved 31 October 2021. [Online]. Available: <https://howtofirebase.com/what-is-firebase-fcb8614ba442>.
- [113] *Firebase authentication*, Retrieved 31 October 2021. [Online]. Available: <https://firebase.google.com/docs/auth>.
- [114] *Cloud firestore*, Retrieved 31 October 2021. [Online]. Available: <https://firebase.google.com/docs/firestore>.
- [115] A. Brothers, *Mobile app monetization strategies – which model to choose to make a profit?* Retrieved 06 April 2022, 2021. [Online]. Available: <https://asperbrothers.com/blog/mobile-app-monetization-strategies/>.
- [116] *Røde kors*, Retrieved 21 April 2022. [Online]. Available: <https://www.rodekors.no/>.

- [117] M. A. Team, *How to sell an app idea: 6 easy steps to reach million \$ gain*, Retrieved 21 April 2022, 2019. [Online]. Available: <https://www.mobileaction.co/blog/app-business/how-to-sell-an-app-idea/>.
- [118] GoodFirms, *What is software deployment?* Retrieved 27 April 2022. [Online]. Available: <https://www.goodfirms.co/glossary/software-deployment/>.
- [119] S. Logic, *Software deployment*, Retrieved 05 April 2022. [Online]. Available: <https://www.sumologic.com/glossary/software-deployment/>.
- [120] I. Framework, *Ionic capacitor sync*, Retrieved 27 April 2022. [Online]. Available: <https://ionicframework.com/docs/cli/commands/capacitor-sync>.
- [121] I. Framework, *Ionic capacitor build*, Retrieved 27 April 2022. [Online]. Available: <https://ionicframework.com/docs/cli/commands/capacitor-build>.
- [122] G. Developers, *Meet android studio*, Retrieved 29 April 2022. [Online]. Available: <https://developer.android.com/studio/intro>.
- [123] A. inc., *Xcode*, Retrieved 29 April 2022. [Online]. Available: <https://developer.apple.com/documentation/xcode>.
- [124] MacinCloud, *Macincloud*, Retrieved 29 April 2022. [Online]. Available: <https://www.macincloud.com/>.
- [125] Hotjar, *Usability evaluation and analysis*, Retrieved 10 May 2022, 2022. [Online]. Available: <https://www.hotjar.com/usability-testing/evaluation-analysis/>.
- [126] NPM, *@capacitor/geolocation*, Retrieved 09 May 2022. [Online]. Available: <https://www.npmjs.com/package/@capacitor/geolocation>.



# Appendix A

## Digital attachment

This master's thesis has a digital attachment in form of a zip-file. The zip-file contains:

- *Hyttebiblioteket.pdf* a pdf containing the specialization project conducted by Camilla Velvin fall 2021.
- *hyttebiblioteket.apk* a file containing an apk-file of the developed mobile application, Hyttebiblioteket.





## Appendix B

# User guide for downloading the application on an Android device

To download the Hyttebiblioteket-app, make sure that your device has enabled downloading apps from other sources than Google Play. Go to "Innstillinger", navigate to "Biometri og sikkerhet" and click "Installer ukjente apper".

Download the *hyttebiblioteket.apk* on your Android device from the zip-file attached. Extract the APK file, and Hyttebiblioteket will be downloaded as a mobile app on your mobile device.



# Appendix C

## User guide for running the application in a simulator

### C.1 Environment setup on machine

To setup the necessary environment where the application can run, please follow this guide: <https://capacitorjs.com/docs/getting-started/environment-setup>

### C.2 Clone the repository from Github

The source code can be downloaded from Github. The link to the repository is <https://github.com/camilve/hyttebiblioteket>. Make sure the chosen branch is "master".

To download the repository, you can either use *git* or download it as a zip folder.

If you choose to use *git*, please follow this guide to install at your machine: <https://git-scm.com/book/en/v2/Getting-Started-Installing-Git>. The next step is then to clone the repository. First, navigate to the folder you want your project. Then open your command tool (*Terminal* on MacOS and *PowerShell* for Windows), and type in the command:

```
git clone https://github.com/camilve/hyttebiblioteket.git
```

Then navigate into the folder by entering this command:

```
cd hyttebiblioteket
```

If you choose to download the source code as a zip folder. Enter `https://github.com/camilve/hyttebiblioteket.git`, and press the green "code" button and "Download ZIP". When the zip file is downloaded, unpack it in a folder called "hyttebiblioteket", and navigate into the folder as mentioned above.

### C.3 Run code

When the folder "hyttebiblioteket" is opened in the terminal and NodeJS with NPM is installed. Please enter:

```
npm install
```

```
ionic cap sync
```

To run the application in an Android simulator, make sure that you have Android Studio downloaded on your machine, and enter the following in the terminal:

```
ionic cap open android
```

Select the device and press the green play button when Android Studio is open to start the application.

To run the application in an iOS simulator, please make sure that you have downloaded XCode (only available on macOS) and enter:

```
ionic cap open ios
```

When XCode is open, press the play button to start the application in a simulator.

# Appendix D

## Results of first usability test

This appendix presents the results from the first usability tests, the test of the web application. The test plan was written in English, but as the application and test participants were Norwegian the results are presented in Norwegian.

# Brukertest 1

Brukeren hadde ingen tilknytning eller kjennskap til systemet før testen, og lite dataferdigheter. Bruker Safari på Mac.

## Assignment 1: Create a profile

### 1.1 Log in

Finner lett frem til registrer, og får laget profil.

## Assignment 2: Publish book in a chosen area

### 2.1 See book in overview

### 2.2 Change position of the published book

Bruker skjønner ikke helt hvorfor h\*n må tillatte posisjon; «greit å skrive hvorfor man må tillate posisjon».

Når bruker tillater posisjon, får h\*n fortsatt ikke opp siden. Dette er fordi Mac har skrudd av mulig stedsdeling i innstillinger på pc. Dette skjønner ikke bruker, og det var ønskelig med en feilmelding hvorfor det ikke fungerte.

Når posisjon er satt på så ble bok lagt ut enkelt, og fikk også til å endre posisjonen boka lå.

## Assignment 3: Find a book using the list

### 3.1 Find the book's location

### 3.2 Register that you took the book.

Navigerer seg lett til finn bok, men forstår ikke helt hvorfor h\*n må tillatte lokasjon igjen.

Når dette blir tillatt, finner h\*n enkelt bøkene, og finner en bok h\*n ønsker å lese.

Registrerer enkelt denne som hentet.

## Assignment 4: Find a book using the map

### 4.1 Find more information about the book

Finner fort ut hvordan man får opp kartet og trykker enkelt på bøkene der slik at h\*n får opp mer informasjon.

## Assignment 5: Find borrowed books

5.1 Find one of the books that you have borrowed

5.2 Republish a borrowed book

Finner bøkene som er lånt, og får enkelt til å legge ut boka på nytt.

## Interview after the user test

### *Semi-structured interview*

1. How did you experience the application?

Synes forsiden var fin, bra at den forklarer hvordan ting fungerer!

Jeg synes det fungerte veldig greit, var lett med stegene på forsiden. Greit å vite hvor man skulle gå.

2. What was the best and worst with the application?

Beste; sa seg selv hvor man skulle gå. Godt beskrevet.

Verste; Ikke forståelig at man skulle tillate posisjon.

3. Was it anything that you missed, something that would be useful for you as a user?

Nei, beskrivelser var bra. Kanskje man kunne «ratet» boka og/eller gitt en kommentar på den, og at andre kunne lese disse kommentarene på boka.

4. Is this an application that you would like to use?

Ja, tror det. Lett å bruke, og enkelt for meg som ikke kjøper så mye bøker.

5. Any additional feedback?

Synes det var enkelt og greit. Fint design.

# Brukertest 2

Brukeren hadde ingen tilknytning eller kjennskap til systemet før testen, og noen dataferdigheter. Bruker Safari på Mac.

## **Assignment 1: Create a profile**

### 1.1 Log in

Finner enkelt frem til registrer, og forstår hvordan man får til. Skjønte ikke helt at man var registrert.

## **Assignment 2: Publish book in a chosen area**

### 2.1 See book in overview

### 2.2 Change position of the published book

Gikk greit å legge ut. Enkelt å skjønne hvordan man endret posisjon.

## **Assignment 3: Find a book using the list**

### 3.1 Find the book's location

### 3.2 Register that you took the book

Navigerer enkelt frem til alle bøkene, fikk til å hente bok og registrere at den er hentet.

## **Assignment 4: Find a book using the map**

### 4.1 Find more information about the book

Navigerte til kartet, og trykket fort på bøkene på kartet.

## **Assignment 5: Find borrowed books**

### 5.1 Find one of the books that you have borrowed

### 5.2 Republish a borrowed book

Fant enkelt lånte bøker, men fikk ikke til å legge ut boka på nytt da systemet feilet.



# Interview after the user test

## *Semi-structured interview*

### 1. How did you experience the application?

Jeg synes var fint, ryddig. Likte design og farge, og forside. Lett å registre seg og logge inn. Føler noen ganger når man bruker applikasjoner at man må lete, men det var ikke problem her. Godt helhetsinntrykk.

### 2. What was the best and worst with the application?

Beste; enkel nettside, enkelt å legge ut bøker og enkelt og finne frem. Ikke så hokus pokus og legge ut en bok.

Verste; I kartet kunne man ikke se hva boka het, her måtte man trykke.

### 3. Was it anything that you missed, something that would be useful for you as a user?

Navnet på boka i kartet uten å trykke på den.

Og hvor sikkert er det at du får boka tilbake når du låner den bort, kan man snakke med denne personen? Evt. Skrive hvor lenge boka er til utlån.

### 4. Is this an application that you would like to use?

Ja, hjemme her. Kanskje ikke på hytta, siden det ikke er så mange andre hytter der, hadde ikke gått inn og sjekka.

### 5. Any additional feedback?

Ingen flere tilbakemeldinger.

# Brukertest 3

Gode dataferdigheter. Bruker Chrome på Windows.

## **Assignment 1: Create a profile**

### 1.1 Log in

Gikk veldig bra å registrere seg. Redd man får mye epost når man må registrere seg med epost.

## **Assignment 2: Publish book in a chosen area**

### 2.1 See book in overview

### 2.2 Change position of the published book

Finner ut hvordan man endrer posisjon og legger ut boka enkelt.

## **Assignment 3: Find a book using the list**

### 3.1 Find the book's location

### 3.2 Register that you took the book

Finner bøker, skjønner at man må tillatte lokasjon når h\*n leter på siden.

## **Assignment 4: Find a book using the map**

### 4.1 Find more information about the book

Finner kartet enkelt, og klikker naturlig på bøkene for å lese mer om de.

## **Assignment 5: Find borrowed books**

### 5.1 Find one of the books that you have borrowed

### 5.2 Republish a borrowed book

Finner bøkene som er lånt, men når bruker skal legge den tilbake antar h\*n at den må legges tilbake der h\*n fant den.

# Interview after the user test

## *Semi-structured interview*

### 1. How did you experience the application?

Fint, likte fargespekteret på sida. Ser ganske proft ut på frontend delen. Kul applikasjon. Eneste er når jeg skulle levere tilbake boka til noen andre, hadde det vært chill med en knapp «legg tilbake der du fant den».

Finn bok burde også være på forsiden, siden det er hovedpoenget på sida, men positivt inntrykk.

### 2. What was the best and worst with the application?

Beste; Lett å bruke, med liste og kart – mega intuitivt. Enkelt å bruke kartet.

Verste; bare kunne finne sidene på navbar, tok litt for lang tid å finne, da jeg måtte se etter det.

### 3. Was it anything that you missed, something that would be useful for you as a user?

Ikke som jeg kommer på. Vært dritt nice og ha en søkfunksjon etter forfatter. Eller når man skal legge ut så kan man bare skrive forfatter også velge ferdig utfylt mal med valgt bok.

### 4. Is this an application that you would like to use?

Nei, mest fordi jeg ikke leser bøker. Har enda ikke fullført boka jeg starta på for 3 år siden. Leser fagbøker da.

### 5. Any additional feedback?

Ingen flere tilbakemeldinger.

# Brukertest 4

Gode dataferdigheter. Bruker Chrome på Mac.

## **Assignment 1: Create a profile**

### 1.1 Log in

Finner fort frem til registrering, og får laget profil.

## **Assignment 2: Publish book in a chosen area**

### 2.1 See book in overview

### 2.2 Change position of the published book

Bruker ikke kartet når h\*n endrer posisjon. Endrer kun teksten.

## **Assignment 3: Find a book using the list**

### 3.1 Find the book's location

### 3.2 Register that you took the book

Finner frem til lista, men skjer en feil når h\*n prøver å låne en bok.

## **Assignment 4: Find a book using the map**

### 4.1 Find more information about the book

Finner kartet. Får lånt ved å trykke på bøkene på kartet.

## **Assignment 5: Find borrowed books**

### 5.1 Find one of the books that you have borrowed

### 5.2 Republish a borrowed book

Finner enkelt frem til bøkene som er lånt, men fungerer ikke når h\*n skal legge den ut på nytt.

# Interview after the user test

## *Semi-structured interview*

1. How did you experience the application?

Veldig fint, fint design og ryddig. Enkelt å finne knapper. Liker at det er farger på «naturlige steg» knapper. Rask, kartet kommer fort.

2. What was the best and worst with the application?

Beste; Ryddig design

Verste; forstå hele konseptet

3. Was it anything that you missed, something that would be useful for you as a user?

Kanskje beskrivelse, slik at det er enklere å forstå f.eks finn bok. Kan for eksempel stå velg å låne en bok fra en i nærheten eller noe sånt. Når man trykker på en bok, skjønner ikke helt hva registrer som hentet betyr (tenkte dette betydde at de på hytta legger ut boka for deg).

4. Is this an application that you would like to use?

Ja, f.eks hvis man har vært på hytta en uke og lest ut alt. Veldig greit når det er en time å kjøre til butikken. Og i byen, mer sirkulær økonomi.

5. Any additional feedback?

Vil ha to passord-felt når man registrerer, men likte at det var kjapt å registrere seg.

# Brukertest 5

Gode dataferdigheter. Bruker Chrome på Android mobil.

## **Assignment 1: Create a profile**

### 1.1 Log in

Veldig lett å se registrer da den er øverst. Lurer på om h\*n er inne da det ikke kommer noen verifikasjon; «hadde vært fint med en bekreftelse om hvor man er». Synes også det var rart at man ikke måtte skrive passord 2 ganger.

## **Assignment 2: Publish book in a chosen area**

### 2.1 See book in overview

### 2.2 Change position of the published book

Får lag ut bok enkelt og greit, og klarer bra å endre posisjonen til boka.

## **Assignment 3: Find a book using the list**

### 3.1 Find the book's location

### 3.2 Register that you took the book

Prøver først å trykke på bildene på forsida for å navigere seg frem til «finn bok»; «hadde vært kult om man kunne trykt på fremsida, sånn finn bok, eller start her». Når bok er funnet så er det uklart for brukeren at man skal hente boka når man finner den, tenker at den blir reservert.

## **Assignment 4: Find a book using the map**

### 4.1 Find more information about the book

Likte kartet, klarte å navigere seg frem.

## **Assignment 5: Find borrowed books**

### 5.1 Find one of the books that you have borrowed

### 5.2 Republish a borrowed book

Finner bøkene, synes det er bra man går til siden for lånte bøker når man låner en. Får til å legge ut bok på nytt.

# Interview after the user test

## *Semi-structured interview*

### 1. How did you experience the application?

Fint design, lett å redigere. Mangel på respons tilbake til bruker. Får ikke bekreftelse på at man har lagret etc.

Fint at det er tilbakemelding når man gjør feil, men savner tilbakemelding når man gjør noe riktig, for eksempel lagt ut bok eller liknende.

På fremsiden hadde det vært fint om stegene kunne trykkes på. Legge ut bok fra forsiden.

### 2. What was the best and worst with the application?

Beste; Lett å vite hva man kan gjøre. Skjønner hvordan man kan gjøre ting med en gang. Liker fargevalg.

Verste; mangel på tilbakemeldinger og at man ikke kunne klikke på forsiden.

### 3. Was it anything that you missed, something that would be useful for you as a user?

Samme som ble sagt i stad; mangel på tilbakemeldinger.

### 4. Is this an application that you would like to use?

Absolutt, veldig glad i bøker. Og hvertfall hvis det er så lett som det virker som. Både for å låne og finne.

### 5. Any additional feedback?

Ingen flere. Greit å få med registrer som hentet vs registrert som lånt, det var litt uklart.





# Appendix E

## Results of second usability test

This appendix presents the results from the second usability tests, the test of the mobile application. The test plan was written in English, but as the application and test participants were Norwegian the results are presented in Norwegian.

# Brukertest 1

Bruker Android til vanlig, gode dataferdigheter. Ikke brukt systemet før.

## **Assignment 1: Create a profile**

### 1.1 Log in

Blar igjennom slideren for å se på informasjonen.

Tok feil passord, så ikke dette ettersom tastaturet ikke gikk ned når h\*n trykket gå til. Vil gjerne se hva h\*n har skrevet på passord.

## **Assignment 2: Publish book in a chosen area**

### 2.1 See book in overview

### 2.2 Change position of the published book

Tar litt tid før h\*n skjønner hvor man kan legge ut bok, men trykker riktig etter litt betenkningstid. Legger ut boka, men skjønnte ikke helt at h\*n hadde valgt posisjon i kart når jeg ba om å endre posisjon. Da h\*n trykket neste på tastaturet som førte til at tastaturet bare hoppet til neste tekstfelt.

Redigerer boka, men dobbeltsjekker at endringene er lagret.

## **Assignment 3: Find a book using the list**

### 3.1 Find the book's location

Finner lista enkelt, og klikker seg inn på en spesifikk bok uten problemer.

## **Assignment 4: Find a book using the map**

### 4.1 Find more information about the book

Bruker kartet enkelt, og klikker på bøkene for å se mer informasjon.

## **Assignment 5: Collect book**

### 5.1 Register that you borrow a book.

### 5.2 Find one of the books you have borrowed.

### 5.3 Regret borrowing the book.

### 5.4 Republish a borrowed book in another location than you picked it up.

Skjønner ikke helt hvordan man kan angre at man lånte den, tenker først at man sletter den, men skjønner etter hvert at tanken er at man skal legge ut boka samme sted som h\*n fant den. Synes det var dumt at kommentaren gikk vekk da, og ønsket heller en knapp med «angre».

Når h\*n skal legge ut boka med annen posisjon glemmer h\*n å trykke på legg ut på nytt, men kommer på det når h\*n ikke får trykket på kartet. Får deretter lagt ut i ny posisjon.

### **Assignment 6: Send in deviation**

Tenker at man bør gått inn på boka hvis det gjelder at man ikke finner boka, men finner frem tilslutt.

## **Interview after the user test**

### *Semi-structured interview*

1. How did you experience the application?

Responsiv, liker godt designet, ser veldig ryddig ut. Kan kanskje være litt forvirrende når man redigerer vs legger ut. Mister kanskje informasjon når man kan bruke neste knapp på tastaturet, da jeg mista kartet når jeg la ut bok. Veldig lett oversiktlig i listevisninga, og motsatt på kart. Greit å se, men jeg hadde nok kun brukt liste, iallfall hvis jeg var ute etter en tittel.

2. What was the best and worst with the application?

Beste: Responsivt design, veldig fin å se, og derav bra brukeropplevelse

Verste: ting var skjult, f.eks så ikke kart og feilmelding.

3. Was it anything that you missed, something that would be useful for you as a user?

Kanskje tatt kontakt med den som la ut boka hvis jeg er usikker på lokasjon eller sånne ting.

4. Is this an application that you would like to use?

Ikke der jeg er vant til å hytte, mest fordi det er langt til nærmeste nabo, og jeg ikke er så fan av å lese. Men kunne brukt det her jeg bor hvis mange brukte appen slik at det var mangfold i type bøker.

5. Any additional feedback?

Nei, alt gikk ganske fort inn. Litt usikker på hva man trenger hva epost til. Hva man bruker denne dataen til.

# Brukertest 2

Har god kjennskap til Android, og data. Ikke brukt systemet før.

## **Assignment 1: Create a profile**

### 1.1 Log in

Lager profil enkelt, så igjennom slideren først for å få informasjon om applikasjonen.

## **Assignment 2: Publish book in a chosen area**

### 2.1 See book in overview

### 2.2 Change position of the published book

Trykker på riktig fane med en gang, og finner legg ut. Skjønner ikke helt hva eierskap vil si, men flytter på kartet til å være der h\*n legger ut boka.

Når posisjonen skal endres blir både kart og kommentar endret.

## **Assignment 3: Find a book using the list**

### 3.1 Find the book's location

Finner lett frem og klikker på bøkene for å se mer informasjon om de.

## **Assignment 4: Find a book using the map**

### 4.1 Find more information about the book

Vil låne bok fra et bestemt sted, og trykker da på denne lokasjonen i kartet selv om det ikke er noen merker for bok her. Når ingenting skjer, skjønner h\*n at man må trykke på bøkene i kartet istedenfor.

## **Assignment 5: Collect book**

### 5.1 Register that you borrow a book.

### 5.2 Find one of the books you have borrowed.

### 5.3 Regret borrowing the book.

### 5.4 Republish a borrowed book in another location than you picked it up.

Finner frem, får angret, og får lagt ut boka på nytt med ny posisjon – endrer både kart og kommentar

## Assignment 6: Send in deviation

Trykker seg frem til hjem-skjermen, men synes det hadde vært nice om email hadde vært klikkbar og leda videre til epost-leser.

## Interview after the user test

### *Semi-structured interview*

1. How did you experience the application?

Nice, fin. Enkel. Fine farger. Likte at det ikke var mye støy.

2. What was the best and worst with the application?

Beste; Enkel å bruke

Verste; Det gir ikke så mye mening at det står lagt ut på navbar før man har lagt ut noe, men vet ikke helt hva annet det skulle stått.

3. Was it anything that you missed, something that would be useful for you as a user?

Nei, ikke egentlig som jeg kommer på.

4. Is this an application that you would like to use?

Jeg leser egentlig ikke så mye bøker, men hvis jeg gjorde og iallfall på hytta.

5. Any additional feedback?

Nice hvis man kunne søkt etter bøker eller sjanger, at man f.eks velger kategori når man legger ut. Kanskje jeg ikke hadde hatt lyst til å legge ut mine bøker med tanke på at det er frykt for å miste den. Var litt forvirrende med eierskap, går an å ha infoknapp med mer info.

# Brukertest 3

Har brukt Android før. God kjennskap til data. Ikke brukt systemet før.

## **Assignment 1: Create a profile**

### 1.1 Log in

Laget bruker greit og enkelt. Litt trøbbel når passordet var for kort første gang, men ble ordnet når begge passordet ble oppdatert.

## **Assignment 2: Publish book in a chosen area**

### 2.1 See book in overview

### 2.2 Change position of the published book

Får lagt ut bok, og endrer posisjon enkelt ved å trykke på kartet.

## **Assignment 3: Find a book using the list**

### 3.1 Find the book's location

Skjøpper hvordan man bruker lista. Ser også mer info ved å trykke på boka.

## **Assignment 4: Find a book using the map**

### 4.1 Find more information about the book

Prøvde å se om det lå bøker utenfor rekkevidden som er søkt i. Skjønte ikke helt at det var avgrenset søk basert på distanse.

## **Assignment 5: Collect book**

### 5.1 Register that you borrow a book.

### 5.2 Find one of the books you have borrowed.

### 5.3 Regret borrowing the book.

### 5.4 Republish a borrowed book in another location than you picked it up.

Skjøpper at man skal angre registrering når man ikke trenger boka. Og legger ut boka enkelt på nytt ved å endre kart og kommentar. Synes det er rart med et alternativ om at man kan slette boka.

## Assignment 6: Send in deviation

Gir tilbakemelding. Tenker at rapporter feil er bugs i systemet som skal sendes inn.

## Interview after the user test

### *Semi-structured interview*

1. How did you experience the application?

Synes det var bra. Intuitiv også føles den responsiv ut – rask.

2. What was the best and worst with the application?

Beste; Kanskje bare at den er intuitiv.

Verste; Tenkte at kart var deaktivert på hovedsiden, ønsket mer kontrast i fargene.

3. Was it anything that you missed, something that would be useful for you as a user?

Legg til ønskeliste, jeg vil låne denne boka; er det noen som har den? Også notifikasjon hvis den ble lagt ut.

4. Is this an application that you would like to use?

Nei, for leser egentlig ikke så mye bøker, men hvis den hadde ekspandert til andre ting så kanskje. Litt usikker på hva det kunne vært da.

5. Any additional feedback?

Synes det er bra, leta etter en logo.

# Brukertest 4

Har ingen erfaring med Android, gode dataferdigheter og har testet Hyttebiblioteket før.

## **Assignment 1: Create a profile**

### 1.1 Log in

Husker ikke epost som er logget inn med, og må lage ny profil. Dette går fint.

Kanskje term and conditions bør stå før registrer.

## **Assignment 2: Publish book in a chosen area**

### 2.1 See book in overview

### 2.2 Change position of the published book

Går på lagt ut, men er litt usikker på om dette er riktig. «Ikke helt åpenbart at man legger ut bok under lagt ut, men har ikke noe annet tips til hva det kunne vært.»

Endrer posisjon til boka og skjønner hvordan man navigerer seg rundt på kartet.

## **Assignment 3: Find a book using the list**

### 3.1 Find the book's location

Finner bok, og bruker kartet for å zoome inn og finne boka ut hvor boka ligger.

## **Assignment 4: Find a book using the map**

### 4.1 Find more information about the book

Skjønner at man må trykke på bøkene i kartet, for å se mer av boka.

## **Assignment 5: Collect book**

### 5.1 Register that you borrow a book.

### 5.2 Find one of the books you have borrowed.

### 5.3 Regret borrowing the book.

### 5.4 Republish a borrowed book in another location than you picked it up.

Skjønner at man kan angre at man lånte boka. Når h\*n skal legge ut boka på nytt, byttes posisjon på kartet og kommentar.



## Assignment 6: Send in deviation

Trykker på hjem og rapporter feil.

## Interview after the user test

### *Semi-structured interview*

1. How did you experience the application?

Ryddig, enkel, inneholder akkurat det man trenger – ingen forstyrrende elementer.

2. What was the best and worst with the application?

Beste; Liker ideen, artig konsept. Fine farger og design

Verste; Slet litt med å finne legge ut ny bok, gav ikke helt mening at denne lå under lagt ut. Litt irriterende at pin på kart er der du lånte boka.

3. Was it anything that you missed, something that would be useful for you as a user?

Nei, er såpass enkel app at man forstår konseptet uten så mye mer.

4. Is this an application that you would like to use?

Absolutt.

5. Any additional feedback?

Jeg liker at man får se at man kan låne hva som er hvor. F.eks har vært borti sånne bokdelingsgreier hvor man må dra dit for å se hva som er der. Her får man se hvilke bøker som er og hvor før man drar dit.

# Brukertest 5

Kjennskap til Android. Gode dataferdigheter. Har kjennskap til systemet.

## Assignment 1: Create a profile

### 1.1 Log in

Husker ikke passord, og bruker glemt passord for å bytte. Får deretter logget inn med det nye passordet. «Kult at det var så enkelt å bytte passord»

## Assignment 2: Publish book in a chosen area

### 2.1 See book in overview

### 2.2 Change position of the published book

Prøver å legge ut ved å trykke på kartet under «finn bok». Skjønner at dette ikke går og navigerer seg til riktig plass. Legger ut bok. Trykker på kartet for å endre posisjon. Trykker så på boka for å se den og redigere den.

## Assignment 3: Find a book using the list

### 3.1 Find the book's location

Skjønner ikke helt at bøkene blir filtrert på avstand, da han leter etter bok som han har lagt ut selv, finner en annen bok og klikker på den før h\*n sier; «gir jo kanskje mening at man ikke skal se bøker veldig langt unna, og ikke sine egne da man faktisk skal hente den.»

## Assignment 4: Find a book using the map

### 4.1 Find more information about the book

Klikker rundt på de forskjellige bøkene for å lese tittelen, og navigerer seg videre inn på boka for å låne den.

## Assignment 5: Collect book

5.1 Register that you borrow a book.

5.2 Find one of the books you have borrowed.

5.3 Regret borrowing the book.

5.4 Republish a borrowed book in another location than you picked it up.

Skjønner at man kan angre registrering av bok. Legger ut boka på nytt. Bruker kartet til å posisjonere boka og legger inn kommentar.

### **Assignment 6: Send in deviation**

Prøver å slette boka, men dette går ikke da h\*n har tatt en bok med eierskap. Skjønner tilslutt at h\*n må gå til hjem for å finne ut av dette.

## **Interview after the user test**

### *Semi-structured interview*

1. How did you experience the application?

Fin, ser bra ut. Frontenddelen/designet er fin. Forsiden er kart i nærområde, kasta rett inn i applikasjonen, bedre enn sist hvor man så disse ringene.

2. What was the best and worst with the application?

Beste; Lett når man først hadde lyst til å låne bok i nærheten, både med liste og kart.

Verste; Jeg tror at når jeg mente en bok var ødelagt, så hadde nærmeste metode for å rapportere feil vært under boka. Tok litt tid å finne den faktiske rapporter knappen.

3. Was it anything that you missed, something that would be useful for you as a user?

Kun rapportering av feil under boka.

4. Is this an application that you would like to use?

Eh, ikke egentlig da jeg ikke leser så mye bøker. Men kanskje hvis det var fagbøker hvis det lå der ute.

5. Any additional feedback?

Lett å fikse glemt passord, veldig smud.

