



BookIT

Authors:

Karin Johansson

Lisbet Ersson

Nadja Fadhel

Independent Project

in Sociotechnical Systems Engineering

- IT Systems, 15.0 c

Supervisors:

Dave Clark

Davide Vega D'aurelio

2018-06-03

Uppsala

Abstract

This thesis treats the process of developing an extensive software project conducted by three students attending Uppsala University. The application development included creating a project- and time plan, going from idea to final product as well as learning new technical tools and programming languages. The project outcome was BookIT, a smartphone application created with React Native and Google Firebase as foundation. The idea behind the application was to create a generic solution for booking rooms that could easily be adapted to an organization, company or comparable. The booking system aimed to be easy to access, use and understand. At the end of the project a functioning app had been developed that met the most crucial requirements the team had set up for the application.

Table of content

1	Introduction	5
1.1	RåKod.....	5
1.2	What is BookIT?	6
1.3	BookIT Overview.....	7
1.4	Terminology	8
2	Method.....	9
2.1	Technical Tools	9
2.1.1	Tools for Development	9
2.1.2	Tools for Communication and Documentation.....	13
2.2	Agile methodology	14
2.4	Project Process	16
2.4.1	Initial Idea vs. Final Idea	16
2.4.2	Application Development Process	17
3	Socio-Economic Aspects	19
3.1	Target Users	19
3.1.1	Active Pollutions.....	20
3.2	Business Model	21
4	The User Perspective	23
4.1	Presenting BookIT.....	23
4.2	Graphical Design.....	35
4.2.1	Colors.....	36
4.2.2	Logo and Icons.....	37
4.3	User Interface Design.....	38
4.3.1	The Application Navigation.....	39
4.3.2	User Interface Design Choices.....	40
4.3.3	Mockup and Usability Test.....	45
5	Database Design	48
5.1	ER-Diagram	48
5.2	BookIT Database.....	50
5.2.1	Bookings	51
5.2.2	Users	51
5.2.3	Rooms	52
5.3	Process of the database.....	54
5.4	Security and Privacy.....	54

6	Evaluation.....	56
6.1	Testing Methods.....	56
6.1.1	Database Testing.....	56
6.1.2	Simulator Testing.....	57
6.1.3	Usability Testing.....	57
6.2	Before Launching the Application	58
6.2.1	Create an Administration View	58
6.2.2	Improve User Interface	59
6.2.3	Wanted Features.....	59
6.3	The Application in the Real World	60
7	Conclusions	61
8	References	63
9	Appendix	65
9.1	Individual Assessment and Reflection	65
9.1.1	Karin Johansson.....	65
9.1.2	Nadja Fadhel	67
9.1.3	Lisbet Ersson.....	71
9.2	Usability Assessment	74
9.2.1	Usability Test.....	74
9.2.2	Evaluation Form.....	76
9.3	Mockup.....	77

1 Introduction

As society continues to digitalize, coordination, scheduling and administration that was previously handled manually, are systematically transformed into applications and web services. Despite the digitalization, booking rooms is still a struggle at several institutions. It is common to use a piece of paper, emailing people involved or use different websites. Smaller companies and organizations often lack the money or time to develop a suitable solution for their needs on their own. These companies then end up with either no digital tools to use or a general solution that does not satisfy their specific demands. IT giants, like Google and Apple, have created lots of solutions which are adapted for collaboration. However, these are very extensive to cover the needs of almost everyone and the products include lots of features and functions which can be experienced as superfluous or confusing. The abundance of content can make the applications less user friendly depending on the area of use.

Luckily, modern app development opens up for new ways to meet the needs of everyone. Perhaps it's time to substitute that piece of paper outside of the housing society's laundry room? Or that Google Calendar that's not really the best solution for booking the company's conference room but (almost) does the trick anyway? This is where RåKod comes in.

1.1 RåKod

RåKod is a newly started and imaginary IT-company founded by Nadja Fadhel, Karin Johansson and Lisbet Ersson. All three are currently studying a master program in sociotechnical systems engineering with IT systems profile at Uppsala University. What they all have in common apart from their studies is a great interest in IT and a passion to ease the everyday life for ordinary people. The following report will focus on the company's first project, including everything from ideas and planning to process together with the final product - BookIT.



1.2 What is BookIT?

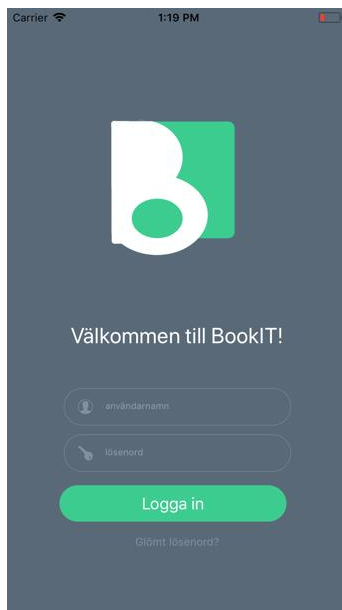
In simple words, BookIT is a mobile application providing a system for booking rooms. The purpose of the application is to create a simple and efficient way for a group of people to coordinate their schedules and spaces. The booking system provided by BookIT enables the users to book and cancel a time and room of their choosing, as well as getting an overview of their own and others' bookings. BookIT aims to reach costumers that are not quite content with the more general solutions that other IT giants provide.

The initial idea of the application was to create a group room booking system for Uppsala University. This was an idea sprung from students complaining about the current insufficient system, the inconvenient steps of creating a booking and the frustration over the fully booked system – an issue corresponding to the difficulty of canceling bookings. After some research on how the system worked and what data needed to be stored, RåKod decided to create a generic version of the application instead, that may later be customized to the university if it is still of interest.

The user of BookIT is a limited group of people that share some communal spatial space between each other and need a system to structure the distribution of this space. This group of people can be a company, an organization, a school or a housing cooperative which share common spaces like conference rooms, laundry rooms or something comparable. The target user of BookIT demands a simple and smooth system to coordinate the bookings of these spaces.

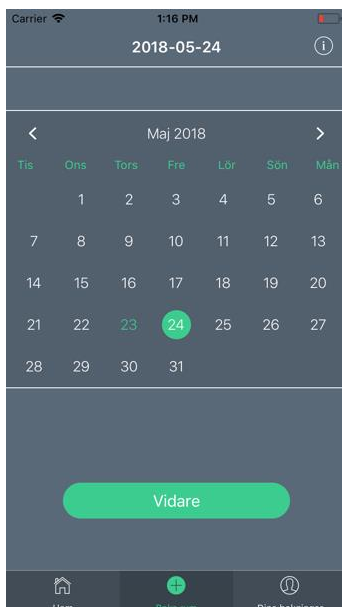
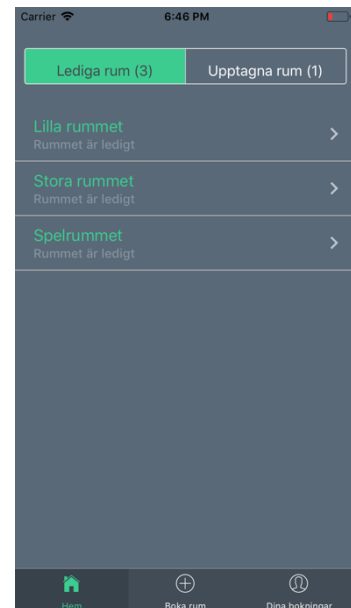
The vision was to make the application suitable for different companies and organizations by just changing certain components between them. The application is therefore structured in a way that enables an easy adaption to different sizes and types of companies and organizations with a different number of common places. The system can with little effort be adapted to a small company with a small amount of rooms as well as a larger company with many users and rooms.

1.3 BookIT Overview



A user signs in with an email address and a chosen password connected to the booking system relevant for that specific user. If a user does not actively sign out it will stay signed in.

BookIT is built around three main views, “tabs”, which are displayed in the tab bar, at the bottom of all pages in the system. The first tab shows which rooms that are available and which that are occupied at the moment. From here a user can find more detailed information and calendar overviews of all rooms.



The second tab is where a user can create a new booking by choosing time, date, booking title and optionally a description of the reservation. Through the booking process a user gets continuous feedback on availability and choices.

The user specific information and features are located in the third tab. There, users can see and manage upcoming bookings, get a calendar overview, etc.



1.4 Terminology

Some words and expressions are used frequently in the thesis and are therefore worth knowing while reading the report and when using BookIT. To simplify the reading and the using of the application this terminology is presented and described in a glossary list below.

“The application” or “the app”

BookIT may often be referred to as just “the application” or “the app” in the text.

“User”

A user is a person who uses the application BookIT. Since BookIT is an application built around a company or an organization the users of the application are employees, members or something comparable within the company or organization in question.

“Booking” or “Reservation”

A booking or a reservation refers to a reservation of the company’s or organization’s common spaces.

“Room”

A room refers to a room or some other common space that could be booked at the company or organization that is using BookIT.

"Navigation"

Navigation refers to the flow between different screens in the application, how the user can move around in the application and switch between screens.

“Component”

Component refers to a small fragment of the application or minor code piece.

2 Method

This project was limited to eleven weeks. During that time, the project went from the first phase, which included coming up with an idea and the first sketch of the application, to a functioning mobile application. This section will cover a description of the development process of BookIT, which includes a presentation of the technical tools that were used, the working method and a description of the time plan.

2.1 Technical Tools

The development of the project required several technical tools; software, editing programs and different web services. Some of the tools were used for the development of the actual application while some were used for the project process and structure like documentation and communication internally in the group as well as externally with the supervisors of the project. These will be described both by functionality and how they were used in this project.

2.1.1 Tools for Development

This section aims to describe the different software and tools that have been used for building BookIT. These tools were used to develop back-end, front-end and building a mockup of the user interface and graphical design. Tools that were used for sharing the project between group members are also described.



Vectr is a free cross-platform vector graphics app, where scalable vector graphics can be created with simple tools (1). RåKod used Vectr to create both the company logo and the BookIT logo. Since the application is easy-to-use there is also some functionality missing, which is the reason it was complemented by Adobe Photoshop to create graphical content for the project.



Marvel is a design tool for building mockups and prototypes of mobile and web applications. Marvel is used in the browser, so there is no need to install any software (2). Marvel was used in the beginning of the application development process to get an overview of what features the team wanted and needed the application to have and how to design the user interface and graphics of the app. Marvel lets the user invite other users to a project, which made it possible for RåKod to review and edit pages in the same mockup simultaneously. With Marvel the team could create a flow between the pages in the mockup by deciding what occurrence should appear when a user clicks on a particular component on the page, which simplified the code writing for building the actual application.



ERDPlus is a web-based database modeling tool that lets the user quickly and easy create Entity Relationship Diagrams (3). This tool was used by RåKod to create an ER-diagram.



The back-end development of the application was handled by Firebase. Firebase is provided by Google, built on their infrastructure and was chosen because it is easy to use, scalable and provides lots of functionality, which enables the team to focus on building the app rather than handling the database. By using Firebase, other relevant services and options became available, like authentication and storage. Firebase offers two cloud-based and client-accessible database solutions, Cloud Firestore and Realtime Database. Both Cloud Firestore and Realtime Database are NoSQL databases. Cloud Firestore is a beta version with a more intuitive data model, richer features, faster queries and scales better than the Realtime Database. (4) RåKod found Cloud Firestore to be simpler to implement to the project and even though it is a beta version, the team considered this database version to be more suitable.

There are no tables or records in a NoSQL database. The data is instead stored in documents, which are organized into collections and subcollections. Each document is identified by a name (document ID) and contains a set of key-value pairs and the collections works like

containers for the documents. The document IDs are set automatically by Firebase, if not specified when created. Cloud Firestore is optimized for storing large collections of small documents. The communication works through data synchronization. Every time data in the database changes, the user receives the updated data. Firebase applications also remain responsive when the user is offline. Once a connection is established the user will get the latest updates on the server. (5)

React Native

React Native is an open-source framework for rendering mobile apps in iOS and Android. React Native was announced by Facebook in 2015 and offers a way to build mobile applications using React and JavaScript. One reason for choosing React Native for building BookIT was that the developers can, to a great extent, use the same code for deployment on both iOS and Android, an advantage that was not used for BookIT (which currently only have an iOS version) but could be implemented later. React Native is using the same fundamental user interface building blocks that are used by iOS and Android. (6) Another reason for choosing React Native is that the app could then be built entirely in JavaScript.



Xcode is an integrated development environment developed by Apple for MacOS. Xcode contains a set of software development tools for building apps for MacOS, iOS, watchOS and tvOS. Xcode has multiple features but for this project it was mainly used for its simulator tool which enabled prototyping and testing builds during the development process of building BookIT. The tool runs on the developers Macbooks and behaves like a standard app while simulating. (7)

Xcode is only adapted to develop environments for Apple's software, including the smartphone environment iOS. By using Xcode during the development of BookIT only the iOS version could be tested. Testing the Android version of BookIT requires software that enables testing Android environments or an Android device. BookIT is intended to work on both iOS devices as well as Android devices but since all members of RåKod have iPhones the project have been focused on first testing the iOS version of the application. Since React

Native support both iOS and Android, it should work on both devices. A few features might have to be added, but the application should basically work on an Android. However since Xcode was used the Android version of BookIT has not yet been tested.



Visual Studio Code is a lightweight open source code editor developed by Microsoft. It is available for Windows, Linux as well as MacOS. Visual Studio Code supports several features, including embedded Git control, debugging and intelligent code completion. (8)



GitHub is a hosting service for version control using Git. Git is a version control system for tracking changes in computer files and coordinating work on those files among multiple people, which makes it possible for a group to store and share code between each other. (9) Git stores data as a series of snapshots. When committing in Git, an object is stored containing a pointer to the snapshot of the content that was staged, with information like the author and message metadata. A branch is a lightweight pointer to one of these commits and the default branch is called the master branch. (10) To connect code from different authors, Github offers a merge function. Merging a pull request means that a request is sent to the upstream branch when the work is completed. If the pull request does not have any merge conflicts it can be merged on Github. (11)

Github Desktop was chosen over Git, since two of the team members had used Github Desktop before and were somewhat familiar with it. However, in the middle of the project, there was a merge conflict with the codes due to a flawed code that was pushed to the master branch. This lead to a lot of problems and was very time consuming. When trying to create new repositories, errors occurred that were connected to the versions of the installed software. The solution was in the end to stop using Github Desktop and just sending the code to each other and merging it manually. This was not an optimal solution but it meant that the application development could continue.

2.1.2 Tools for Communication and Documentation

This section aims to describe the different tools that have been used for the communication between group members and the supervisors of the project as well as the tools used for documentation of different parts of the project.



Slack is a platform for communication and collaboration between people. RåKod has mainly used Slack for communication between the project members but also for communication with the supervisors of the project. Slack provides a solution for basic communication as well as the ability to share documents and different announcements. The user is able to set up communication channels reaching from a dialogue between only two individuals within the team to a group discussion involving several people. Slack is available on the web as well as an app for smartphones, which makes it easily accessible at any time. (12)



Trello is a web-based project management application that allows one, or several users, visually plan and track the development of a project by dividing it into smaller tasks and work items. Trello is working as a to-do-list, allowing the user to add different tasks in a timeline structure as well as marking tasks as ongoing or done. Trello is available both on the web as well as a mobile app for smartphones, which makes it easy to access at any time. (13)



OneDrive is a file hosting service operated by Microsoft, allowing users to create and store documents and files. OneDrive can be used for personal use as well as for sharing and working on documents with other users. (14) Everything related to the project that was not directly included in the actual application was stored in OneDrive in a communal channel which every member in RåKod could access and contribute to.

2.2 Agile methodology

During this project, an agile methodology was used. The core of working agile is to have the team self-organize with collocated team members and to work in a pace that sustains the creativity and productivity. One other principal of agile methodology is to work task oriented and deliver results in small time intervals. The work procedure is intended to encourage flexible and rapid answers to demands on the market. (15)

RåKod had office hours from 09:00-17:00 every day in a common space, accordingly with the agile methodology. If one team member could not make the time schedule that was set up, the member informed the rest of the group. Since the team worked as much together, the members were well informed about what the others worked on. It also enabled team members to demonstrate new components that were created and tests could be conducted together to try the quality of the new features.

Since Firebase was used in this project, the work with the back-end part of the application decreased. Therefore the team chose a task oriented working method instead of splitting the workload between front-end and back-end. The wanted features of the app were written onto the Trello board. When a task was completed and the work was finished, it was erased from the board and the member who had completed the first task started with another one.

2.3 Time Plan

To overview the project structure and the time there was to allocate, a time schedule was created. This schedule is shown in Figure 1 below. The idea of the schedule was for the team to overview the development and to have internal deadlines to work towards.



Figure 1. The time line created for the project. The x-axis represents the time in weeks and the y-axis represents the different phases of the project.

The first step of the project was to decide upon an idea of what application to build and pitch it. The team agreed to do a room booking mobile application and started to work on the rough sketches for the idea with pen and paper. The goal was to create an application that at least met the minimum requirements of the project; an application that enabled making and canceling room reservations for a signed in user. Another minimum requirement was to provide an overview of the availability of rooms.

The next step was to create a mockup and the graphical design. This was done during the second week of the project. During the creation of the mockup, the group also concretized which features should exist and the user interface design of the application. It was during this phase that the team decided to skip the idea of making the application adapted to Uppsala University and instead focus on the generic application system. After the mockup was finished, the development of the app began.

The team had decided to use React Native, Firebase and Xcode to build the app with motivations presented in section *2.1.1 Tools for Development*. The team downloaded the software and installed it on the computers. During this step, the team encountered several issues with the software. Some of the problems were solved almost instantly, while some were quite time consuming. When a member of the team solved its encountered problems, that person continued to the next step of the project, which was to start building the application. Since all languages and tools were new for the team members, the documentation was used as a source of learning. As the project progressed, the knowledge increased which made the development speed up. The progress throughout this period followed the agile methodology.

The timeline was approximately kept, which is a result of it not being too detailed about which specific functionalities should be completed at a certain time. Seen in retrospect, it was a good idea to not make the time line too detailed since it was difficult to estimate how much time each milestone would take. The learning period was integrated in the initial phase of the project. Instead of completing tutorials, the team went through documentation and started developing the application. The team benefited from this by having more time to spend on the actual building of the application and learned the relevant information during the actual

building process. However, it could have been convenient to have learned more basics and gotten introduced to the new languages before starting to use them.

2.4 Project Process

During the project there was a lot to learn and new problems continuously appeared. In the beginning it was difficult to set up the development tools, install and initialize the needed software. This resulted in some team members being able to start coding before others. The team chose to combine the startup phase with learning. Therefore, no online courses or tutorials were taken. Instead the application development started straight away and the documentation of Firebase and React Native was then the main source for learning.

Initially the plan was to divide the team to have some members focusing on the back-end and on integrating Firebase into the application, and others working on the front-end. However, since Firebase handles basically all the back-end by itself and the integration was difficult to implement in the first stages of the app development, it was decided to not divide the team into roles but work task oriented. This method worked better for the team and was kept throughout the project.

2.4.1 Initial Idea vs. Final Idea

The initial idea was to create a mobile application for the group room booking system at Uppsala University. All members of the team are currently studying at this university and the idea was generated by the desire to be able to book group rooms easily from their mobile phones. The current website of the booking system is difficult to access to begin with. It is only found by googling some chosen keywords. Once the website is accessed by the user, the user is forwarded to a separate system for signing in and is then forwarded back to the booking system. The booking system itself lacks a good user interface; it is difficult to both use and understand. Since the processes of finishing a booking and accessing the system is quite time consuming reservations are seldom cancelled, resulting in a misleading booking schedule. This means that rooms appear to be booked even if no one is using them, simply because users do not take the time to cancel the reservations if they cannot make it to them.

All these areas of improvement were used as inspiration for what RåKod aimed to accomplish with BookIT.

By the end of the work with the mockup, the team discussed the wanted design and features of the application. The group started to design the mockup adapted to Uppsala University. During the procedure of making design choices and going through Uppsala University's system more thoroughly; evaluating what data that had to be managed, which features that had to be implemented as well as the scale of the challenge, the team realized the magnitude of work it would take. Some choices would be different depending on if the application would be adapted to the university or a company. For example, the structures of the institutions are different. The university has faculties and several buildings below each faculty that the user would have to choose between while a company likely would just have one office building. It was then decided to make a generic app instead since the workload of data that needed to be handled seemed less when making the app for companies or organizations. Even if the initial object of implementation changed, most of the goals and app features did not. The vision of implementing the app at Uppsala University, or a comparable institution, in the future was still intact. The final product was still an application for booking rooms, or comparable spaces, and aimed to be an easy-to-use, accessible and correct system.

The mockup that was created was accordingly to the initial idea of working with Uppsala University and was not remade to fit into the new idea. The mockup still worked as a guideline for what features would be implemented into the application and how the user interface and graphical design should be constructed.

2.4.2 Application Development Process

A consequence of the mockup being created for the initial idea was that some features turned out to be unnecessary and unsuitable for the second plan. The mockups were designed to fit a larger institution, Uppsala University, with a big database of rooms to search and choose between. Even though the main target of the app changed relatively early in the project, the mockup still worked as a foundation for the app development and the source of inspiration for design.

Throughout the work with the application the team performed constant evaluations whether a certain feature was to be prioritized at the moment or not. Some features were not implemented due to the short amount of time that RåKod had to develop BookIT. In the beginning of the application development of all the initial steps were time consuming since no one in RåKod had any previous experience with either Firebase or React Native. The learning curve was steep and further into the project the time from idea to a working feature was a lot shorter.

A big issue was the difficulty of installing the needed software and get it to function properly for all of the members of RåKod. Two of the group members had MacBooks and one a PC, whilst all three had iPhones. The problems were seldom the same on all computers but differed widely, it was therefore difficult to help each other. A learning experience from this is that it is of great importance to make sure that the chosen tools work on every member's computer and that all members get their software working before starting the coding process of the project. This enables starting with the project together and already in the initial phase of the project get an idea of what can and cannot be done by the different team members.

Other issues were to implement certain functionalities in the code or to come up with the architecture to some features. These were often time consuming and a lot of the time constituted of going through the documentation.

3 Socio-Economic Aspects

In the process of deciding which idea to focus on when developing an application, some socio-economical aspects were taken into consideration. These aspects are important to understand in order to get a picture of what needs the app will fill and to whom the solution is directed to. To examine this, the team of RåKod set up a couple of questions: what users the app would address, how to make a profit of the app and what the user would get out of the app. These aspects will be discussed in this section.

3.1 Target Users

The target user of BookIT is a limited group of people that share some communal spatial space between each other and need a system to structure the distribution of this space. This group of people can be a company, an organization, a school or a housing cooperative which share common spaces, e.g. conference rooms, laundry rooms or something similar. The target user of BookIT demands a simple and smooth system to coordinate the bookings of these communal spaces.

The vision was to make the application suitable for different companies and organizations by just changing certain components between them. The application is therefore structured in a way that enables an easy adaption to different sizes and types of companies and organizations with a different number of communal places. The system can with little effort be adapted to a small company with a small number of rooms as well as a larger company with twice as many users and rooms.

Since BookIT is built around a company or organization the users of the application will follow automatically and will differ depending on the place of implementation. The application is not adapted to any specific age group; it is intended to be suitable for all ages. The application is also not adapted to any specific gender, ethnicity or culture. The only prior knowledge needed for a user of BookIT is to have some experience using a smartphone and know the default user pattern for the phone. However, the user does not have to be experienced in using a lot of different smartphone applications in order to manage the

booking system provided by BookIT. It is also not a requirement that the user should understand the entire system of the application the first time using it. The goal is that the system feels simple and comfortable once the user gets to know it. BookIT cannot be used by a person that is not a part of the group using the application. The users of BookIT are individuals that are allowed to use the common space the booking system applies to.

The goal when building BookIT was to create a generic solution that is suitable for different kinds of companies and organizations that needs a system for making reservations of their space. A company that has this need is Active Pollutions and has therefore been used as a case for the first BookIT edition. A description of Active Pollutions and its connection to BookIT is presented in the section below.

3.1.1 Active Pollutions

Active Pollutions is a made-up company that provides their customers with different digital solutions. The company consists of 20 employees with different areas of knowledge - designers, game developers, business developers, front-enders and back-enders. Active Pollutions develop applications for both iOS and Android, websites and games. The company is located in a building in the center of Uppsala and has four rooms available for reservation. They are currently using a booking system on the web that does not provide the functions the company needs. Several of the employees experience the system as difficult and inconvenient to use and some are therefore avoiding using it, which results in miscommunication among the employees. The current booking system does not provide a simple solution for canceling the bookings which has led to employees not canceling their reservations when they are unable to attend them. This in turn leads to rooms being reserved even though no one is using them.

The current booking system is simply not an optimal solution for Active Pollutions' needs, but the employees of the company do not have the time to develop a solution by themselves. Active Pollutions is therefore demanding a better booking system. The employees demand a system that enables them to book a room, to cancel a reservation and view their current bookings in a simple, efficient and convenient way.

Since Active Pollutions is a company that needs a better system for booking their common spaces this company has been used as a case for the first BookIT edition. The current system of BookIT has been adapted to Active Pollutions.

3.2 Business Model

The generic solution of BookIT is still under development and some changes and adjustments have to be made before the application can be launched. When the application has been launched and developed a secure market position, an economic and profitable aspect could be added to the project.

To profit from BookIT, the companies and organizations should be charged in exchange for them using the application and for RåKod to manage it and provide the customer with support. Since BookIT is intended to be customized based on the specific costumer or organization and its employees or members it is not unreasonable to charge the costumer for using the application, as this customization will require both time and work. There are some different ways in which the charging can be done. One alternative is to simply sell the application with a customized system with a non-recurrent charge, that is, to sell it for a specific sum one time. Another alternative is to charge the customer for the maintenance of the application, e.g. by charging the costumer monthly. A third alternative is to combine the previous alternatives. Since BookIT is built around a group of people, like a company or organization, it will probably be a cost imposed on the company or organization itself and not on every individual within the company or organization, which makes it easier to get financial gain for the application.

However, to make BookIT a business it is first necessary to make sure relevant companies and organizations notice the application. It may be companies and organizations that either have no existing booking system or that use a system that is not as useful and easy-to-use as BookIT. Reaching out to potential users can be done in different ways. One example is to directly introduce BookIT to the relevant company or organization by contacting them, present the application and how they would benefit from using it. This alternative requires that the product owner have identified potential and relevant costumers. Another alternative is to market the application through social media or to invest in advertisement of BookIT.

Since the purpose of marketing BookIT is to reach companies, organizations or comparable social media is probably not the best solution of the mentioned.

4 The User Perspective

This section will focus on the front-end development of BookIT. A presentation of all the features of BookIT will be conducted together with a description and discussion of both the graphical design and the user interface design of the application. This includes a description of how the application logo and the color scheme of BookIT was chosen and why, together with the user interface design choices that were made in order to make the application as usable and user friendly as possible. The development and changes of the graphical and the user interface design will also be presented in this section.

4.1 Presenting BookIT

What a user of BookIT can do and how it is done, including cases of incorrect use, is described in this section. The description is presented together with figures of the different pages of the application system. Note that the figures of BookIT displays the application implemented at the company Active Pollutions, the data that is displayed derives from the database built for that company. Also, the figures of BookIT and its features are in Swedish, since the application is currently only implemented in that language. Most of the Swedish words used in this section to describe the application are translated to English within parentheses.

The first page the user encounters is the page for signing into the application, which is illustrated in Figure 2 below. When a user clicks on either the username field or the password field, a keyboard appears on the screen, enabling the user to fill in the fields, which can be seen in Figure 3. If the user is authorized by the system, i.e. a part of the company or organization for which the system is implemented, the user will be signed in when clicking on the green button labeled with “Logga in” (“Sign in”). If the user does not have the correct username or password it will not be able to sign in to the system. The system will display a message saying that the login process failed or that the user does not have access to use the booking system. The error message is displayed in red and this case is illustrated in Figure 4 below. If the user forgot the password the user can change it by clicking on the text string saying “Glömt lösenord?” (“Forgot password?”). This label is placed below the green button

saying “Logga in” (“Sign in”) in Figure 2-4. When clicking on this text the user is transported to a new screen, illustrated in Figure 5 below, where a question asking the user if it has forgot its password is displayed. If the user has forgot its password the user can write its email address in the field labeled “emailadress” (“email address”) and press the green button saying “Återställ lösenord” (“Reset password”). This will result in a link to be sent to the specified email address which the user can click on to get to a page where the current password can be changed to a new one. If the user does not want to change its password and has clicked on the text for this feature by accident the user can return to the sign in page by clicking on the arrow with the label “Tillbaka till inloggningen” (“Back to the sign in page”), placed at the bottom of the page illustrated in Figure 5.

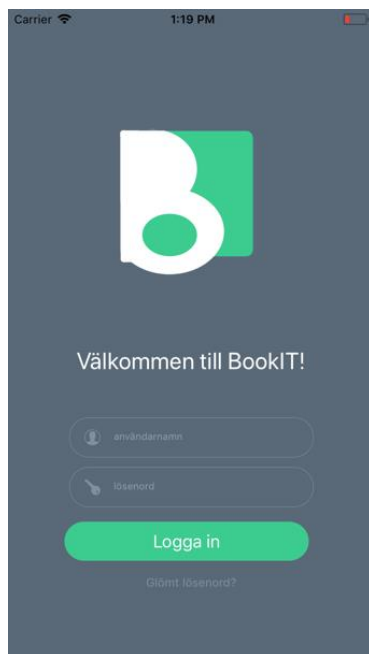


Figure 2. The screen for signing in.

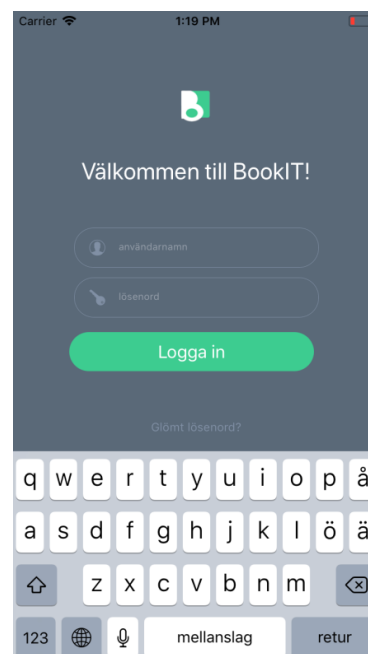


Figure 3. Keyboard appearing when the user clicks on one of the text fields.

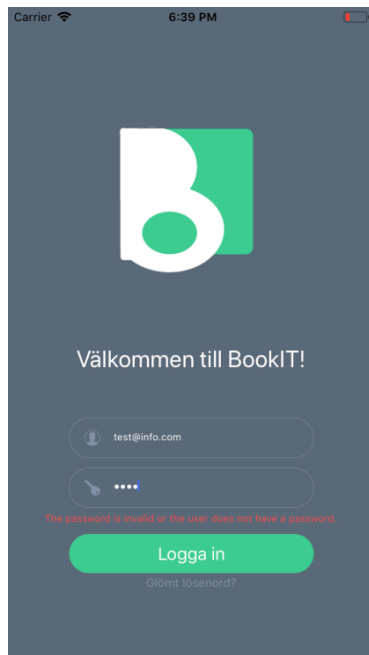


Figure 4. Wrongly typed password makes an error message in red appear on the screen.

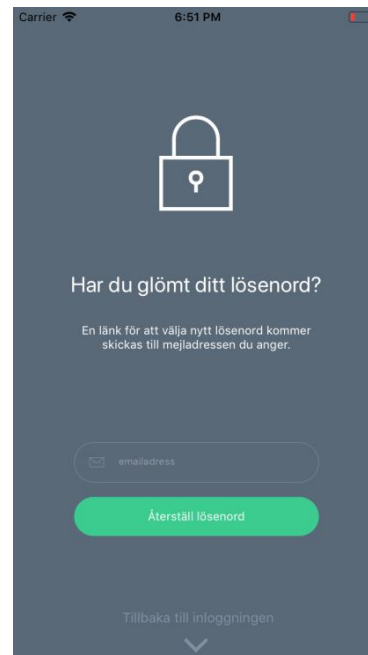


Figure 5. A page of the application where the user can get a link for changing the password.

BookIT has three main screens, “tabs”. These are represented with icons together with a text below each icon and placed in a tab bar at the bottom of each page of the application once the user has signed in. The icons and the following texts below them are intended to give the user a picture of what these pages mean. The three main screens are named “Hem” (“Home”), “Boka rum” (“Book a room”) and “Dina bokningar” (“Your bookings”). The user can access the tab bar wherever in the system the user is. When clicking on one of the tabs the user will directly be forwarded to the main page of the tab in question. This feature will enable the user to quickly switch pages without having to search for or go through several pages to get to one of the three main pages. When a user is located at one of these three tabs the icon belonging to the specific tab will appear in green and the background of the tab will be displayed in a darker shade of gray.

Figure 6-12 below illustrate the pages connected to one of the three main pages, “Hem”. This tab is the default page right after the user has been logged in to the system and gives the user an overview of available and occupied rooms. This main page is divided into two different screens, which is displayed with two tabs at the top of these pages. One of these tabs is labeled “Lediga rum” (“Available rooms”) and is the default page of the main page “Hem” (“Home”). That is, the page that is automatically displayed first when the user logs in to the

system or when the user clicks on the tab in the tab bar that represents this main page, which is why the tab is named “Home”. This page is illustrated in Figure 6. Available rooms are presented in blocks and for each room block, the name of the room and a text saying that the room is available is seen. The user can click on any of the available rooms and will then be transported to a page showing all reservations of that room during the specific day. If the chosen room has no reservations during that day the page will look like the one presented in Figure 7. The figure shows that the user has chosen the room named “Stora rummet” which has no ongoing or upcoming reservations during that day and this information is presented with a text string.

When a user clicks on one of the room blocks displayed in Figure 6, they also get access to a calendar overview of all the current upcoming reservations of the chosen room. This is done by clicking on the component at the top of the page named “Kalenderöversikt” (“Calendar overview”), seen in Figure 7, which makes the component expand and display a calendar. The calendar will show the current month with the current date displayed in green at the front of the page, but the user can see other months by scrolling in the calendar overview. This calendar overview is illustrated in Figure 8. The green dots under the dates in the calendar indicate that there is a reservation of the chosen room during those dates. When clicking on one of the dates in the calendar a list will drop down where the user can see the current bookings of that room. This list is displayed in Figure 9.

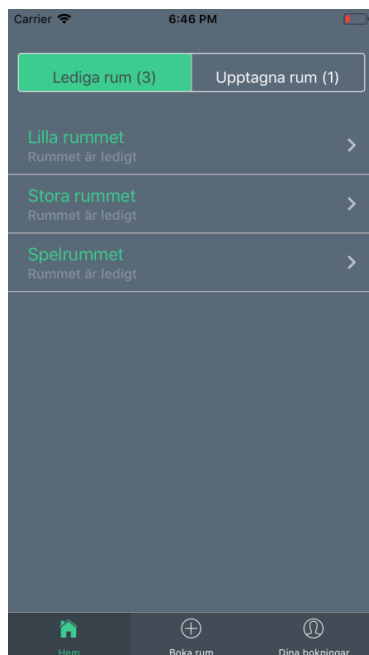


Figure 6. Page showing which rooms are available at the specific moment. This page is the home page of the system.



Figure 7. Screen appearing when clicking on one of the available rooms. This room is named “Stora rummet” and has no bookings during the day.

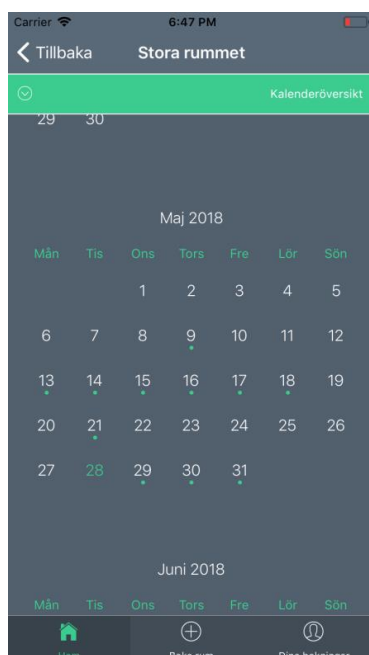


Figure 8. A calendar overview of all the current upcoming reservations for the room named “Stora rummet”.

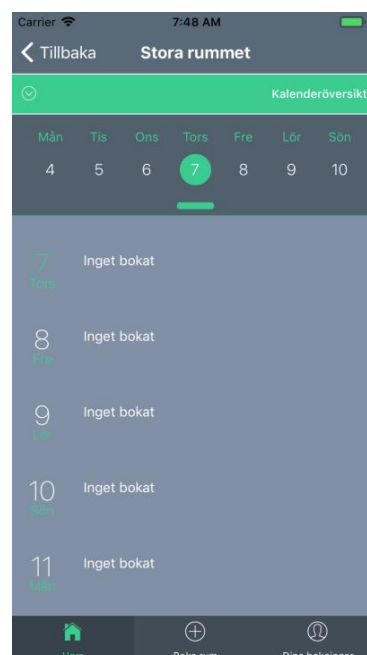


Figure 9. A list of all upcoming reservations drops down when the user clicks on any of the dates in the calendar.

The other tab of the main page “Hem” (“Home”) is labeled “Upptagna rum” (“Reserved rooms”). On this page the user gets an overview of the rooms that are reserved at the moment. This page is illustrated in Figure 10. The reserved rooms are presented in blocks and

for each room block the name of the room and a text saying what time the ongoing reservation will end. The user can click on one of the reserved rooms and will then get to a page where the user can see the bookings of the reserved room during the day. This case is illustrated in Figure 11 below. Since the room is reserved there is an ongoing reservation, which will be displayed in green instead of white, like the upcoming reservations are, to indicate for the user that this reservation is ongoing. Just as with the available rooms the user has access to a calendar overview of the reserved room. This feature looks and works exactly the same as for the available rooms described previously and is for the reserved room in this case displayed in Figure 12.

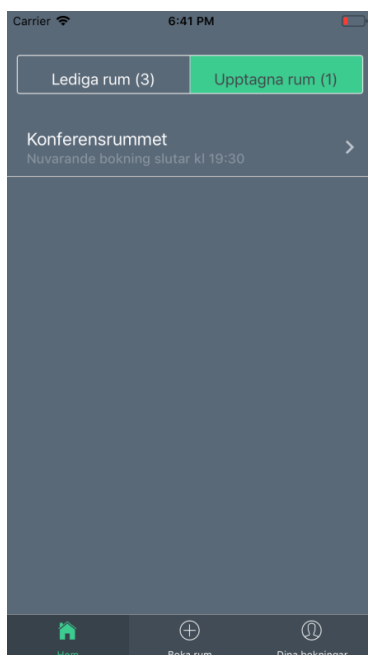


Figure 10. Page showing which rooms are booked at the moment.

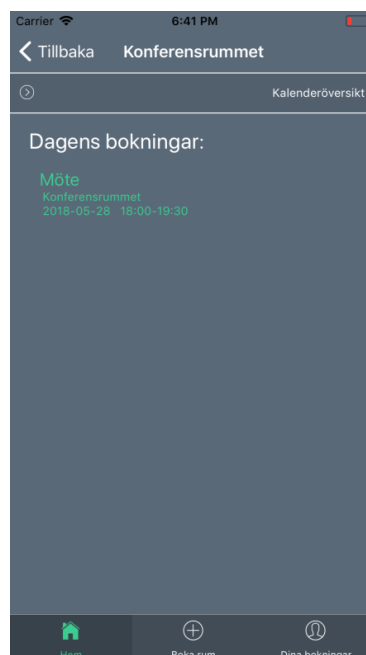


Figure 11. Page showing the bookings of the currently booked room “Konferensrummet”

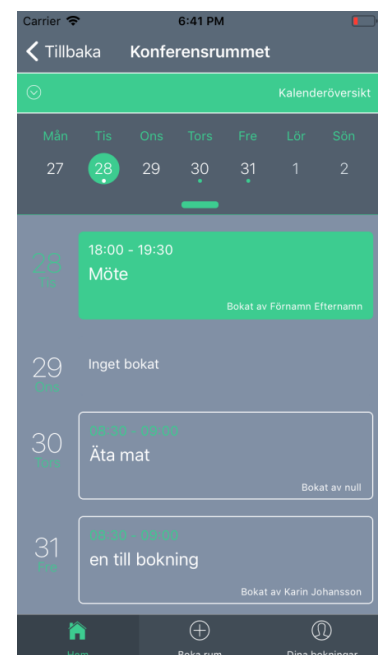


Figure 12. A list of all upcoming reservations drops down when the user clicks on any of the dates.

One of the main functions of the system is to enable a user to book a room at a time and date of the user's choosing. The second main screen, the second tab, of the system provides this function. This tab is named “Boka rum” (“Book a room”). The booking process requires some steps to create a complete booking. These steps are to choose a date, to choose a room, to choose time and to name the reservation. The first screen of this tab is illustrated in Figure 13 below. To book a room the user firstly needs to choose the date of the booking. The current date is the default date and is marked in green while the other dates are marked in white in the calendar shown in Figure 13. When the user clicks on an optional date, the

selected date will be highlighted within a green circle so that the user clearly can see which date has been selected. The selected date also appears at the top of the page, which is illustrated in Figure 14 below. Before a date of the booking has been chosen the button labeled with “Vidare” (“Proceed”) will appear in gray and will not be clickable. The user cannot proceed in the booking process before a date for the booking has been selected. As soon as the user has chosen a date for the booking the button will appear in green and will become clickable so the user can proceed in the process. At the pages shown in Figure 13 and Figure 14, the user is also provided with an information button which is placed at the top right corner. The user can click on this button and will then get information about how the booking system works.

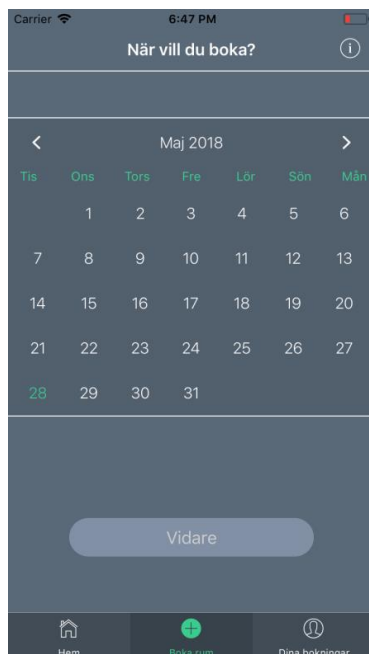


Figure 13. Page showing the first step for booking a room. The user has to select a date for the booking.

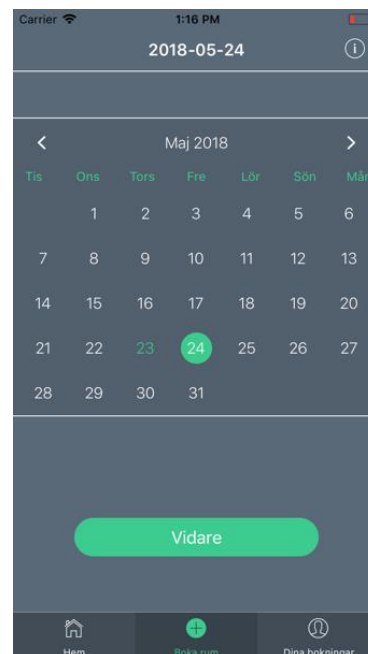


Figure 14. Page showing that the user has selected a date.

When the user has chosen a booking date and clicked on the green button labeled “Vidare” (“Proceed”) the user will get to the page where one of the rooms is to be selected. This page is illustrated in Figure 15 below. In this case the company has four rooms that can be booked. On this page the chosen booking date is displayed at the top of the page to remind the user what date has been selected. The user can always go back and change the date by clicking on the button with the text “Tillbaka” (“Back”) in the navigation bar at the top of the screen. The user will then be transported back to the previous page. The user always has the possibility to

click on the tab bar “Boka rum” (“Book a room”) again, regardless of the user’s current location, which will transport the user to the first page of this tab, the one seen in Figure 13 above. The user selects a room in the lists by clicking on the room. The user will then get to a page where the time for the reservation will be chosen. This page is illustrated in Figure 16 below. The user cannot proceed in the booking process before a time has been selected. To select a time of the user’s choosing the user has to click on the wanted starting time and wanted end time in the list of time intervals. The time intervals between the chosen start and end time will automatically be selected. As soon as the user has clicked at a time interval the green button saying “Vidare” (“Proceed”) will appear. The selected time intervals will be marked as selected and the chosen time will be displayed in the navigation bar at the top of the page, which can be seen in Figure 17. Times that are available for the chosen date and room will have a green clickable circle and a text saying “Ledig” (“Available”) next to it. As soon as a time interval has been chosen the text next to it will change to “VALD” (“CHOSEN”). The already reserved times will be displayed with a red circle together with the title of the reservation and the name of the person who made the reservation under the title. However, the user cannot choose a time that exceeds the already booked time. This case is illustrated in Figure 18 below. Here the user has tried to select the booking time to 17:30-19:30 which is not possible since there already exists a booking of that room at 18:00-19:30. If the user tries to select a time interval that exceeds an existing booked time interval the application will display an error message in red saying that the user has to select coherent times and will automatically select the time closest to the already reserved time. In the case illustrated in Figure 18, the booking time automatically got selected to 17:30-18:00, since the existing reservation starts at 18:00.

As in the previous page the user has the possibility to go back in the booking process by clicking on the back button placed in the navigation bar at the top of the page and will then be transported to the page where a room is to be selected.

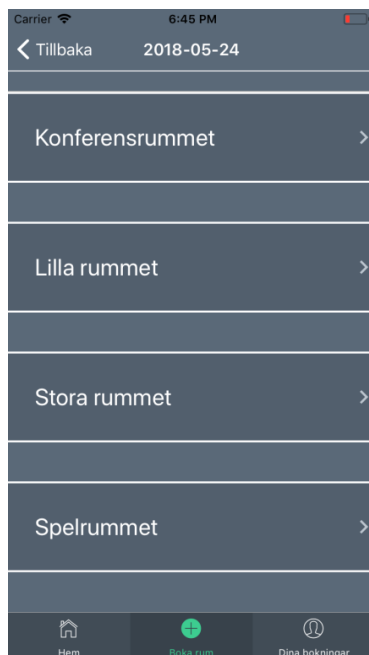


Figure 15. Page showing the second step for booking a room. The user has to select a room.

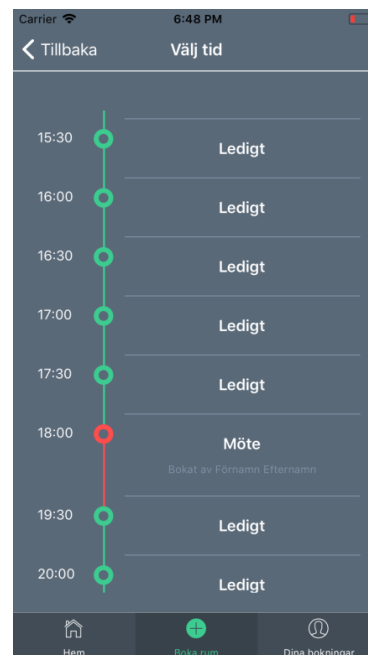


Figure 16. Page showing the third step for booking a room. The user has to select the time.

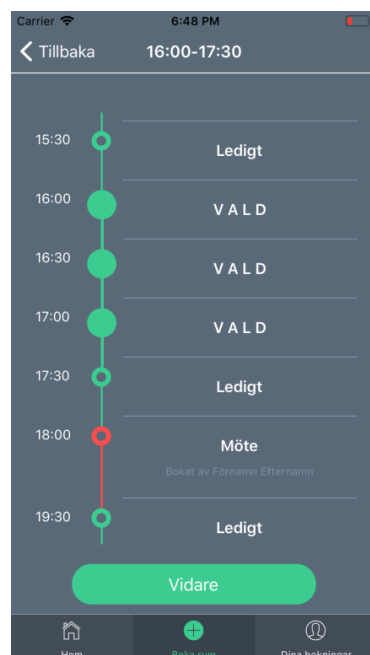


Figure 17. Page showing that the user has selected the time 16:00-17:30

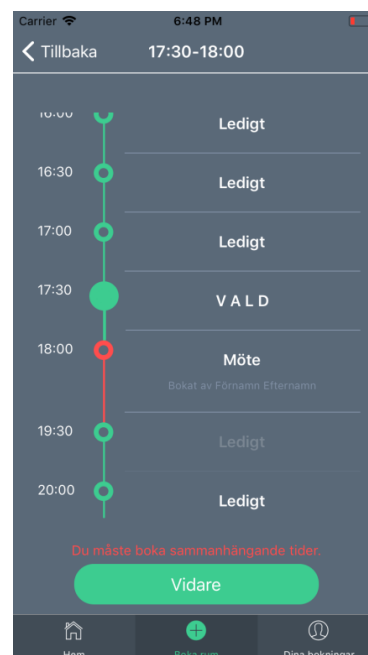


Figure 18. An error message in red appears when the user selects time intervals that are not coherent.

As described above the user can proceed in the booking process by clicking on the green button labeled “Vidare” (“Proceed”). By proceeding from the existing page, shown in Figure 17, the user will be transported to the page seen in Figure 19 below, where the user has to

choose a title and can write a comment for the reservation. This is the last step in making a reservation of a room. The comment is optional while the title is obligatory to be able to place the reservation. This page also displays an overview of the reservation; the chosen room, date and time. As soon as some text has been written in the title field the button labeled with “Boka” (“Book”), will appear in green instead of gray and will become clickable so that the user can complete the booking. When the user clicks on the booking button the reservation will be placed and a page illustrated in Figure 20 will appear, where the user gets a confirmation that the reservation has been placed and also can choose to make a new reservation by clicking on the green button labeled with “Ny bokning” (“New reservation”).



Figure 19. Page showing the fourth step for booking a room. The user has to select a title of the booking.

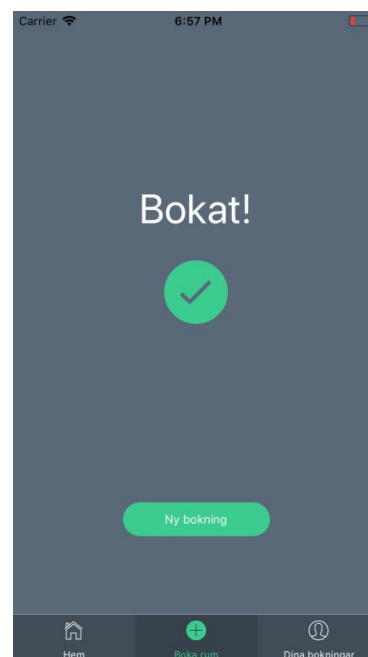


Figure 20. Page showing that the user has made a complete booking.

The third tab in the tab bar, “Dina bokningar” (“Your bookings”), represents the third main screen of the application and is shown in Figure 21 and Figure 22 below. This is the, so called, profile page with the user’s name placed in the header of the screen. Here the user can get an overview of its ongoing and upcoming reservations. If the user has no current reservation this page will display a message saying so. The reservations are displayed in rows sorted chronologically by date and time. Every row of the upcoming reservations contains information about the title of the reservation together with the date and the time of the reservation. The user can click on any of the upcoming bookings which makes a new screen

appear displaying more information about the booking and enabling the user to cancel or change the chosen booking. This case is illustrated in Figure 23. The red button, labeled “Avboka” (“Cancel”) is the button enabling the user to cancel the selected reservation, and the yellow button, labeled “Ändra” (“Change”), is the button for which the user can click if it wants to edit the selected reservation. When the user clicks on the cancel button an alert window pops up asking the user if it is sure about canceling the booking, which is shown in Figure 24. If the user clicks on “Avboka” (“Cancel reservation”) the reservation will be removed from the profile page and from the system. If the user clicks on “Avbryt” (“Exit”) nothing will happen to the reservation. Figure 21 below illustrates a case where the user has no ongoing bookings, only upcoming ones, while Figure 22 illustrates a case where the user has both an ongoing reservation as well as upcoming ones. The ongoing and upcoming bookings are separated in two sections where the upcoming bookings are lined up under the title “Kommande bokningar” (“Upcoming bookings”) and the ongoing booking under the title “Pågående bokningar” (“Ongoing bookings”).

Just like on the pages for the tab “Hem” (“Home”) the user is provided with a calendar overview on the profile page. These calendar overviews look and work exactly the same, except that the one on the profile page presents all the user’s own bookings, and not the general bookings of a specific room. By clicking on the block labeled “Kalenderöversikt” (“Calendar overview”) a list of the user’s ongoing and upcoming bookings drops down.



Figure 21. The third main screen of the application – the profile page.



Figure 22. This page shows that the user has both ongoing and upcoming reservations.

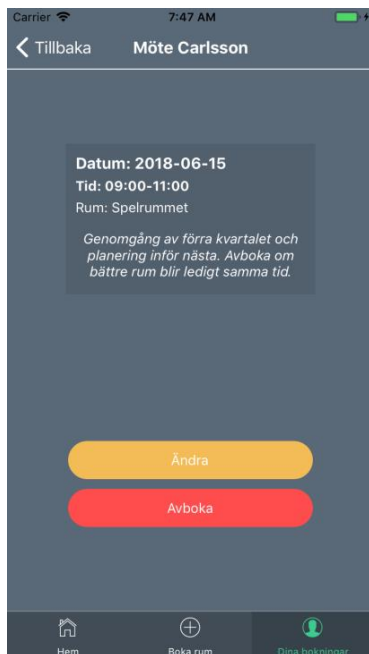


Figure 23. The screen appearing when the user clicks on one of the upcoming bookings.

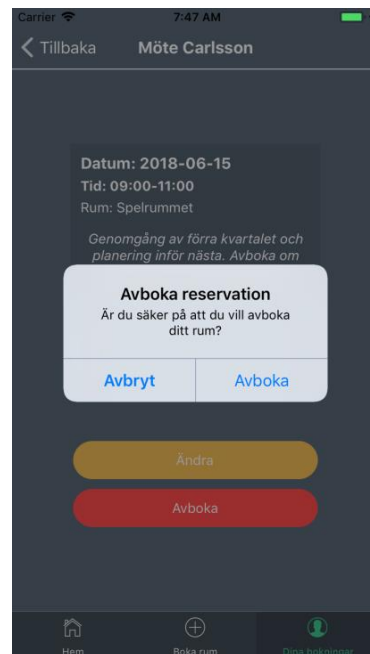


Figure 24. An alert window appears when the user clicks on the button for canceling.

This main page also provides access to the application settings. The button for the settings is represented with an icon in the top right corner of the profile page. Clicking on this icon will make the screen in Figure 25 below appear, where the user can change its password by

clicking on the button labeled “Byt lösenord” (“Change password”) or sign out by clicking on the button labeled “Logga ut” (“Sign out”). When the user clicks on the button for signing out an alert window pops up asking the user if the user is sure about signing out from the application, which is shown in Figure 26. If the user clicks on “Logga ut” (“Sign out”) it will be signed out from the application and be transported to the sign in screen presented in Figure 2 above. If the user clicks on “Avbryt” (“Cancel”) the user will stay signed in and gets back to the usual settings page. To get back to the profile page the user can either click on the button in the navigation bar saying “Tillbaka” (“Back”) or click on the tab representing this page which will make the user be transported to the first page of the tab.

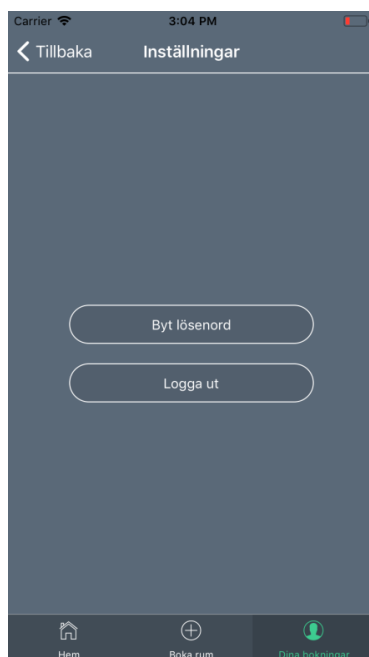


Figure 25. The settings screen.

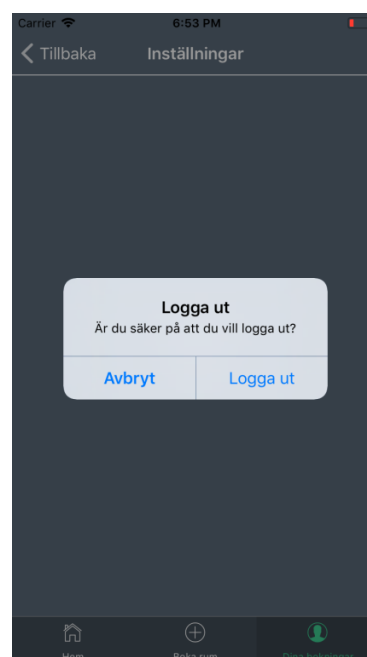


Figure 26. An alert window appears when the user clicks on the button for signing out.

4.2 Graphical Design

The graphical design is a crucial part when developing an application with the goal of being appealing to the user. The graphical design of BookIT has the same theme and style through all pages of the application. This ensures uniformity throughout the app and makes the design clean and comfortable. When deciding how to construct the graphical design of the application the goal was to make it simple but at the same time appealing. The design should not have too many colors or shapes on the same page, but should not be boring or too simple

either. After taking these thoughts and requirements into account, RåKod made decisions of what colors should represent the application's graphical design, how the application's logo should be designed and what icons should be used.

4.2.1 Colors

Right from the start of deciding the graphical design of BookIT, including the logo and the colors of the application, all members of RåKod wanted a clean design and agreed to choose a rather neutral color palette and one more colorful as a highlighter. The colors were chosen with the goal of being able to please as many users as possible and not attract any specific group of people. Since BookIT is supposed to be implemented at different companies and organizations there will be a lot of different users with different preferences, ages, genders and cultures. How you perceive and experience colors will of course differ between people of different ages, genders and cultures but the goal was to make the design appealing to all these different types of users to the greatest extent possible.

After trying out a range of combinations of different colors it was decided to use the colors presented in the color scheme, in Figure 27, below.



Figure 27. Color Scheme for BookIT. The colors presented starting from the left are white, four different shades of gray and a shade of green. The hexadecimal notations of the colors are specified above each color.

As illustrated in the color scheme in Figure 27, RåKod decided to use four different shades of gray together with white and a shade of green. Gray is used as the main color of the graphical design and the different shades are applied for different areas of use in the application. White is used for text and headlines while green is used to highlight active content or clickable buttons.

By mostly using gray shades for the design of the application, a minimalistic style was achieved. It also means that buttons that are green or text that is white get more highlighted and visible to the user. By using mostly gray, the green color is meant to pop out and characterize the application. Green was mostly chosen by RåKod out of personal style and preferences. The color is often associated with calmness and professionalism (16) and the developers of BookIT found the color suitable for an application that will have different companies and organizations as its users. A combination of gray, white and green will hopefully make the application appealing and comfortable for the user.

In addition of the colors above, a shade of red is also used on some areas in the application. This red color has #ff4d4d as its hexadecimal notation and is only used to mark times that are unavailable and for the text in error messages that appear if something is done wrong or is not possible to do in the application system.

4.2.2 Logo and Icons

The logo of BookIT was designed after the color scheme of the application was set, which is the reason that the logo is designed in green and white. RåKod wanted the colors of the logo to come from the color scheme. The shade of green used in the logo is the same shade of green used in the application's graphical design. RåKod has used the same theme and design through the entire application, including everything connected to it, to create uniformity.



Figure 28. The logo of BookIT.

After some sketches of potential designs for the logo were made, RåKod discussed them and finally decided to design the logo as presented in Figure 28. The logo has the letter B, extended from one of the edges, in white with a background in the green shade presented in the color scheme above. B symbolizes the first letter in the name of the application.

Some icons are visible on every page of the system, except the page where the user signs in, since they are placed in the tab bar that is available on all pages through the application. The

icons that are used in the design of BookIT are displayed in Figure 29 below. The icons presented from the left represents the home button, that is the page where the user can see which rooms are and are not available, the profile button, the button for booking a room and the settings button. The first three icons are the ones placed in the tab bar, and thereby visible on all pages of the system. Some of these icons are commonly used in other applications and may thereby be self-explanatory. However, to make it clear for the user what these buttons represent in the system a describing text has been added under the icons in the tab bar.



Figure 29. Icons used in BookIT. The icons presented from the left represents the home page, the booking page, the profile page and the settings. The three first icons are located in the tab bar.

4.3 User Interface Design

A good user interface design is of big importance when developing an application that is intended to be both easy to use and efficient while providing the user with excellent functionality. A poorly designed user interface may result in the user feeling uncomfortable and can cause the application to feel more complicated than it actually is. The main idea when deciding the user interface design has therefore been to make the application as easy to use as possible while providing all the required functionality at the same time. The users, and not the app, should be in control. The application should suggest courses of action or warn about consequences, but it should not take over the decision-making. The idea was to make the application help the user further by telling the user what it can and cannot do. This simply by using messages and buttons of different sizes and colors highlighting different types of content and signaling the user what is to be done.

The user interface design follows the same theme throughout the entire application to create uniformity and make the use of the application comfortable. Since BookIT has users of different ages, cultures and genders the user interface has to be appealing to all these different types of people. The goal is to make the user feel comfortable when using BookIT. This section will illustrate how the user interface design of BookIT has been developed, what design choices that has been made and why.

4.3.1 The Application Navigation

The application is structured in a combination of two different navigation styles, namely flat and hierarchical navigation. A flat navigation is a structure where the user can switch between multiple content categories (17). It is a navigation pattern in which all pages reside at the same hierarchical level (18). This structure is used throughout the entire application with the tab bar at the bottom of the application which makes the three main screens to be located at the same hierarchical level. The user is able to quickly switch between these main screens. Hierarchical navigation is a navigation pattern in which the screens of the application are grouped into multiple sections and levels, which means that the user makes one choice per screen until it reaches a destination (17) (18). To go to another destination the user has to retrace its steps or start over from the beginning and make different choices. The content within each tab of the tab bar is implemented with hierarchical navigation. For example, the process of creating a booking of a room is structured hieratically. If the user wants to e.g. change the date or time of the booking the user has to go back among the previous screens in the process to get to the screen showing the calendar or click on the tab representing these screens to get to the first page and start over.

When creating the navigation among the different pages of BookIT the goal was to make the path through the content logical and predictable as well as easy to follow. In order to achieve this, a tab bar was implemented at the bottom of each screen of the application as soon as the user has signed in. By having a tab bar throughout all pages of the application the user gets access to the main pages connected to the tabs wherever in the system the user is located. Users that have used different applications before are probably familiar with tab bars and should not have any bigger problems knowing how to get around the application. The tab bar should make it easier for the users to navigate themselves in the application. The user always has the possibility to click on one of the tabs and will then be transported to the first page associated with the specific tab, regardless of the user's current location. The tab bar lets the user quickly switch between the different main sections of the application. The active tab, that is, the tab where the user is located at the moment is displayed with its icon in green and its background in a darker shade of gray, to clearly show the user's current location in the application. The tab bar is displayed in three different screens in Figure 30 below, where the bar on each screen is marked with a red circle and the different screens show each active tab of the three tabs.

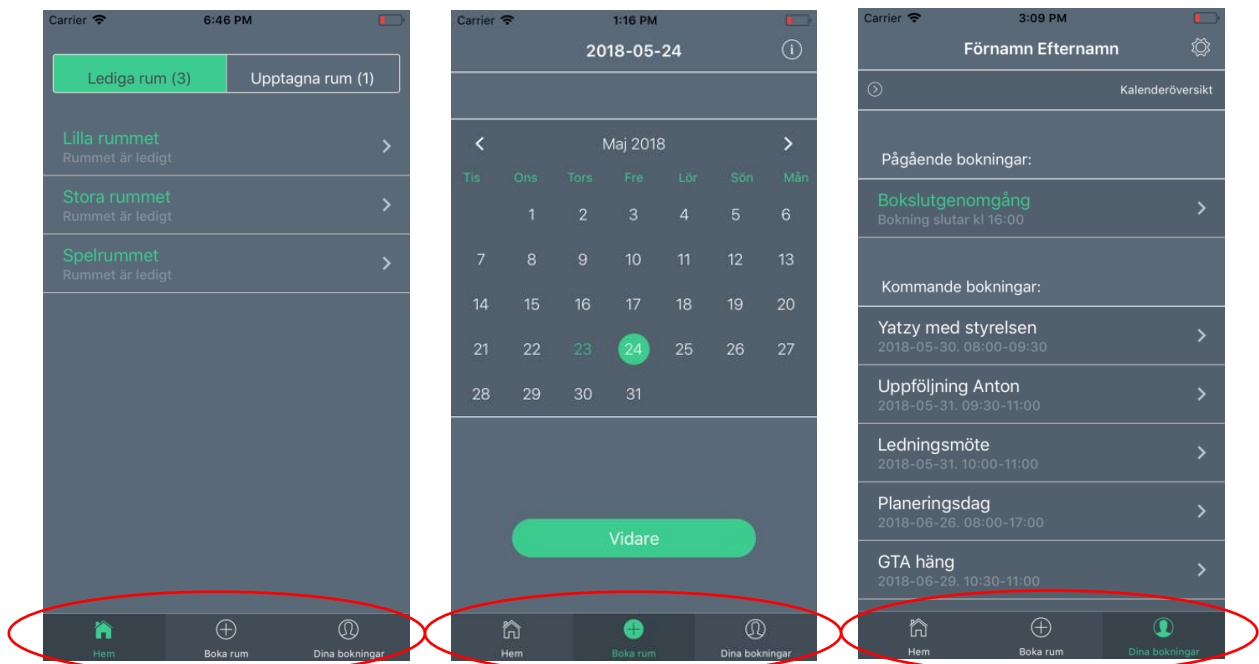


Figure 30. Three different screens in BookIT which illustrates the tab bar, marked with red circles. Each screen shows each active tab in the tab bar.

Another choice that were made in order to make the application efficient and user friendly, was that the navigation through the screens of the booking process and the steps required for booking a room was made to as low as possible. Too many steps in the process of booking a room can contribute to a sense of complexity.

4.3.2 User Interface Design Choices

As mentioned before BookIT is intended to provide a system that is simple and easy to both use and understand. The system should be efficient while providing all the required functionality.

In order to achieve a simple application system, it was built so that it would not have too many features and buttons that would confuse rather than help the user. BookIT has a minimal system, that is, no unnecessary additions but the features that are considered to be needed and helpful for the user. The goal with the application is that it will gently guide the user to the right direction, and not overwhelm with choices.

In order to distinguish between different kinds of features and components, like optional features or features that are required in order to fulfill some actions, different colors, texts and sizes of elements associated with these features has been used. By using different colors and sizes for content of different type the user gets a more clear view of what is and what is not possible to do when using the application and what different content imply. Additional features are presented with a small sized text label or icon in white or gray. Examples of such features are the information button and the button for changing the password marked with red circles in Figure 31 and Figure 32 below.

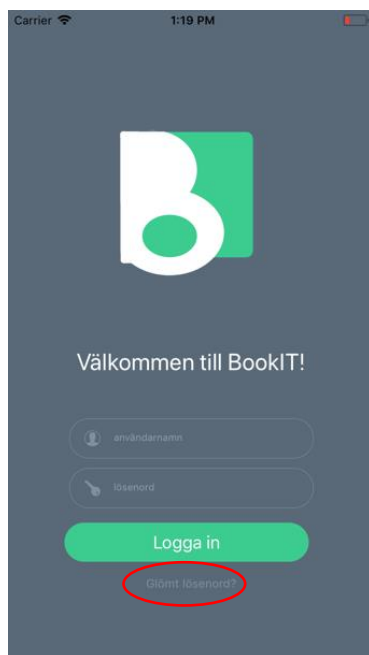


Figure 31. The button for changing the password marked with a red circle.

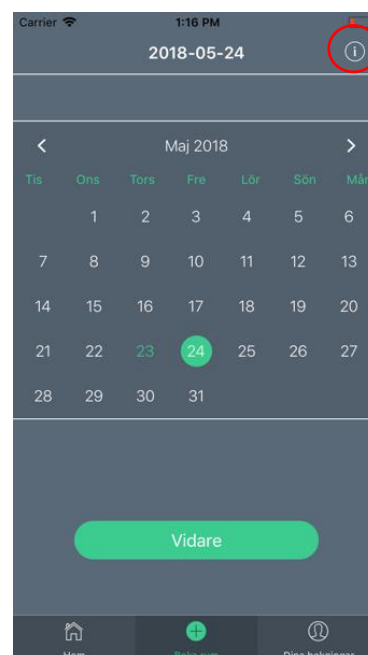


Figure 32. The Information button marked with a red circle.

Components and content that are active or that have to be clicked in order for the user to proceed on some places in the application are highlighted in green. Examples of such content or components are the button for signing in and the ongoing reservations, marked with red in Figure 33 and Figure 34. Beyond dividing up a user's ongoing and upcoming bookings in different sections with different headings on the profile page, the ongoing bookings are presented with green text and the upcoming with white text. This is to clearly distinguish between the two different types of reservations, to make sure the user does not get confused and to show that one of the booking types is currently underway. This case is illustrated in Figure 35 below.

The buttons that are considered to be important were also made green and big, in order to make sure that the user clearly sees them and understands that they are clickable. To show that some actions have to be conducted before the user can proceed in the process some of these buttons are displayed in gray and are not clickable before the actions have been implemented. As soon as the required action has been taken, these buttons will change to green and become clickable. This case is illustrated in Figure 36 below, where the picture to the left shows the button for proceeding in the booking process before the user has selected a date and the picture to the right shows the same button when the user has selected a date.

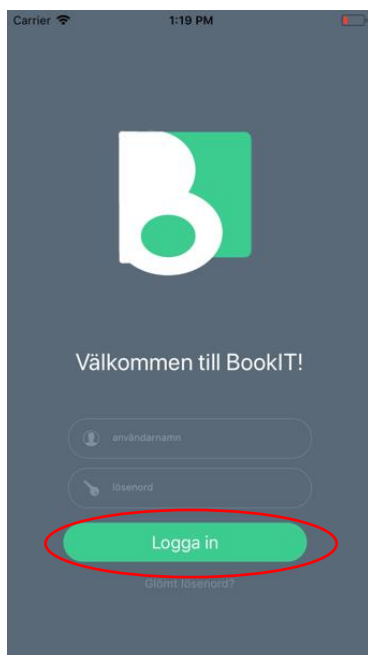


Figure 33. The button for signing in marked with a red circle.

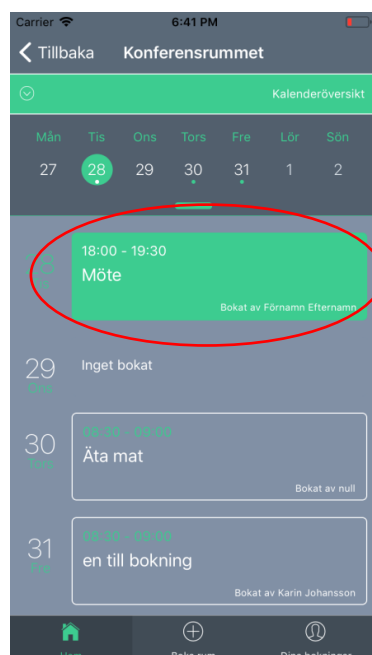


Figure 34. An ongoing reservation marked with a red circle.

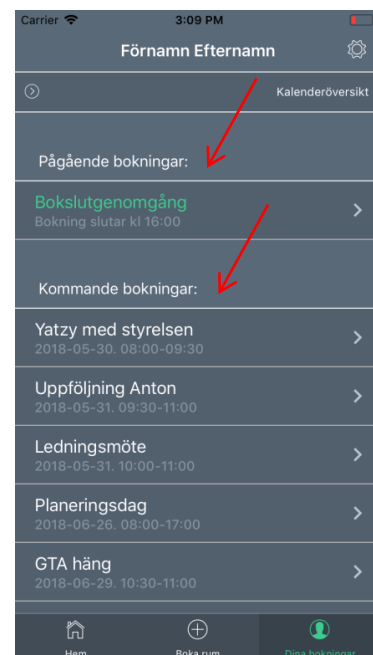


Figure 35. The different appearance of the upcoming and ongoing reservations.

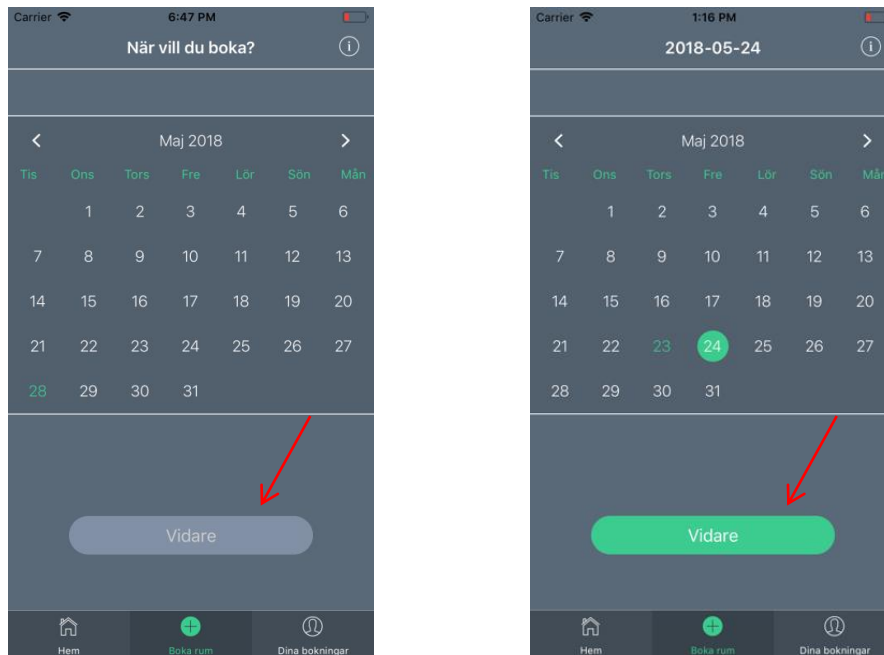


Figure 36. Two screens presenting the button for proceeding. The screen to the left shows the button when the user has not selected a date for the booking yet – gray and not clickable. The screen to the right shows the button when the user has selected a date for the booking - green and clickable.

To show that something is done wrong or is not possible to do an error message in red appears on the active screen. Two different cases when an error message can appear are presented in Figure 37, where the error messages are marked in red. A more detailed description of what colors are used in BookIT and the different features of the application are presented in section 4.2.1 *Colors* and 4.1 *Presenting BookIT* respectively.

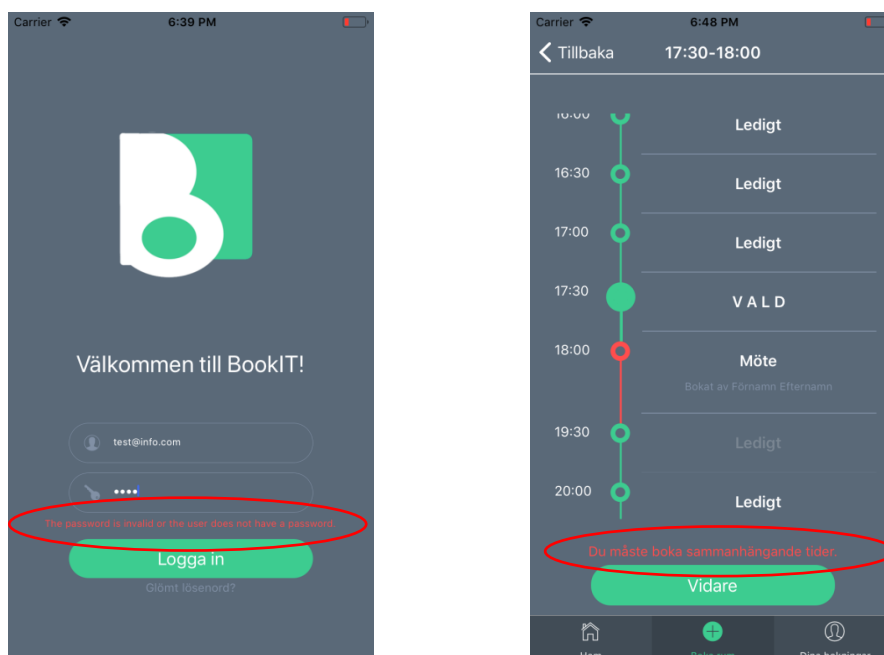


Figure 37. Two different screens of the application where error messages in red appeared. The error messages are marked with a red circle.

The system warns the user when the user is about to perform an action which creates changes that cannot be undone by using alert windows. The infrequency of alerts helps to ensure that the users take them seriously and avoids results performed by mistake. The alerts are only used in situations where it is considered important to ask the user if it is sure about its current performed actions and notify the user of the consequences. Cases where an alert pops up in BookIT are when a user clicks on the button for signing out and the button for canceling a reservation. Both these features are presented and described in section 4.1 *Presenting BookIT* above. The alert appearing when a user tries to sign out is presented in Figure 38 below. Each alert offers two different choices. The choice buttons of each alert are labeled with one or two words that are intended to describe the result of selecting the button.

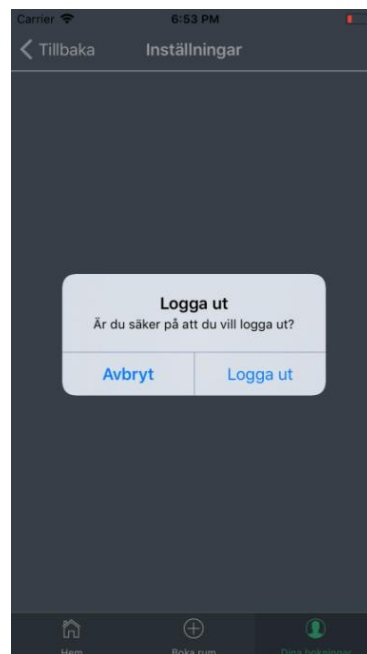


Figure 38. An alert window popping up when the user clicks on the button for signing out.

To make it easy for the user to get an overview of upcoming bookings, the user is provided with calendar overviews of both the bookings of a specific room and the user's own bookings. The different calendars are placed on pages within different tabs. By having these calendars the user can easily and fast get an overview of the upcoming bookings days, weeks or even months ahead.

To avoid confusion or frustration it has been made clear when content in the application is loading with an activity indicator in the form of a spinner. Without denoting that something is happening when the content is loading the application will display a blank or static screen,

which appears as if the app is frozen. This may in turn result in confusion or frustration. The spinner is moving while the content is loading to clarify that the page is loading and not stalled. This case is illustrated in Figure 39.

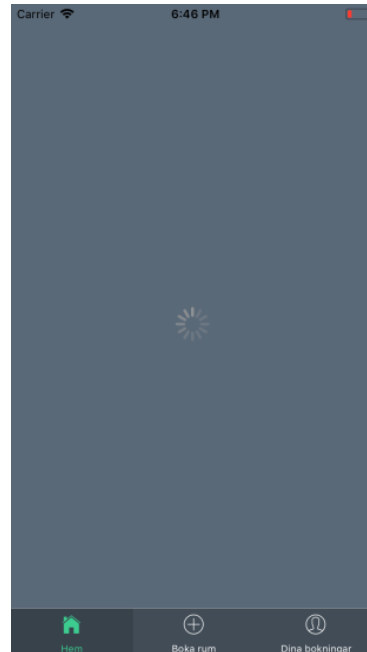


Figure 39. A spinner which appears when content in the application is loading.

4.3.3 Mockup and Usability Test

The front-end development began with creating a mockup of BookIT during the initial phase of the development process. The mockup consisted of different screens RåKod wanted BookIT to have, which included the wanted components, functionality and design. As described in section 2.4.1 *Initial Idea vs. Final Idea* and 2.4.2 *Application Development Process*, the mockup was created based on the initial idea of BookIT, which was to build the application specifically for Uppsala University. Even though the final idea differs from the initial idea, the mockup still worked as a guideline for how the application should be built during the entire building process. The final product thus differs from the mockup in some aspects regarding both the graphical design as well as the user interface design. As mentioned the difference is partly due to the change of the idea, but also due to improvements that RåKod realized had to be made after performing usability tests on both the mockup and the actual application. The test tasks along with the evaluation form that were used in the usability testing of BookIT can be found in the Appendix, section 9.2 *Usability Assessment*.

Some things that were learned through the usability tests of the former version of BookIT were changed accordingly. Among other things, the improvements were changing the color of the tab bar to a darker shade of gray since it turned out to be difficult to distinguish between the different tabs in the tab bar. At the same time the background of the active tab was changed to an even darker shade of gray instead of only displaying the icon of the tab in green. The icons in the tab bar were replaced with clearer icons that became more visible both when the tab was active as well as inactive. Some button labels seemed to be confusing or misleading and were therefore changed to more clear and describing labels. RåKod also realized that one of the tabs in the tab bar was unnecessary, since it only contained two features. The settings function was instead put on the user profile page, where it is displayed with an icon at the top of the page and not a tab in the tab bar. Figure 40 shows a picture of the mockup and a picture of the corresponding page in the final application, where some of the mentioned differences can be seen.

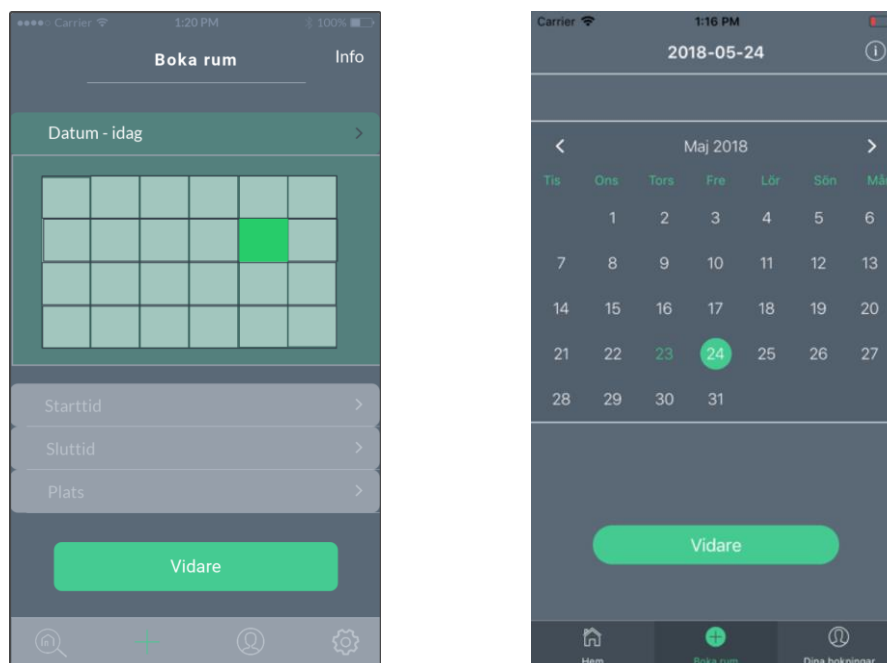


Figure 40. The left picture illustrates a page from the mockup and the picture to the right illustrates the corresponding page of the final application.

A component that was a part of the mockup but got removed during the building of the final application was the search bar at the page “Hem”, where the user gets an overview of which rooms are and are not available at the moment. The search bar was initially included since the mockup was created based on Uppsala University which has faculties and several buildings under each faculty and these building in turn have a lot of bookable rooms. This means that a

lot of rooms would be available at the same time. A search bar would enable the user to search for a specific room or building and thereby avoid searching for that room or building by scrolling in the long list. A company or organization is likely to have bookable rooms in only one building and thereby does not need the search bar. Figure 41 shows a picture of the described page in the mockup and a picture of the corresponding page in the final application, where the removal of the search bar can be seen.

All pictures of the mockup can be found in Appendix, section 9.3 *Mockup*.

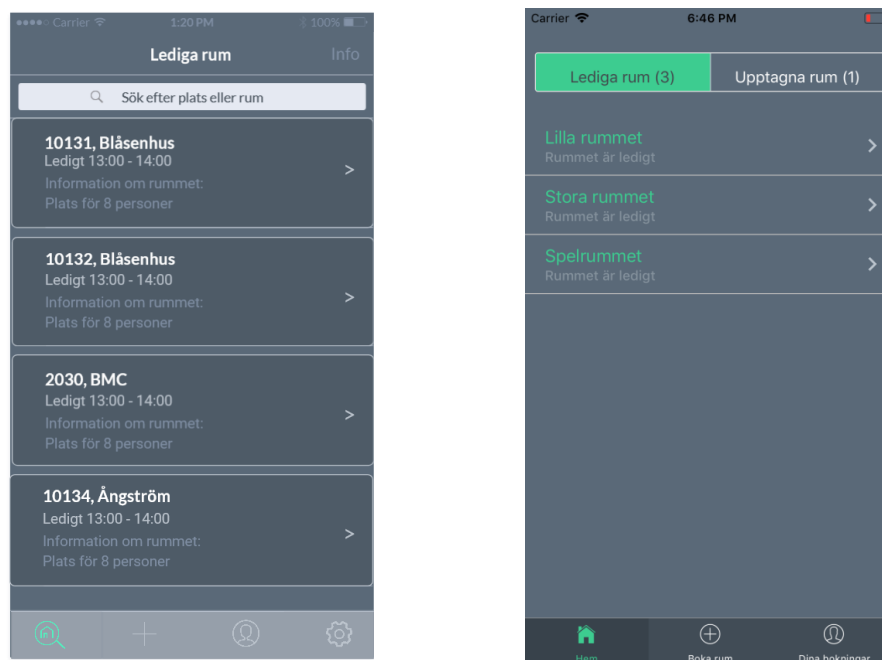


Figure 41. The left picture illustrates a page from the mockup and the picture to the right illustrates the corresponding page of the final application.

5 Database Design

The previous section covered the front-end, including the user interface and graphical design of the application. This section will focus on the back-end and database of BookIT. The database is an important part of the application since it is where the traffic of the application is handled and where the information to keep the app going is stored. The database is the place where all the user information, and all other valuable information, is stored, which makes it crucial to keep the quality of the database high and make sure that the storage is secure. The database structure can also affect the performance of the application. A well-structured and thought-out database will result in a faster application. The requirements for the database for BookIT were to have an easily handled back-end that still has a high standard in security and performance.

5.1 ER-Diagram

Before creating a database, it is relevant to map out the data that needs to be stored to get an idea of how the database should be structured. To get a graphical understanding of what data needed to be stored, an ER-diagram of the system was made. An ER-diagram is used to set up a SQL database and since Firebase does not offer relational databases an ER-diagram does not describe the structure of the database required for BookIT. The ER-diagram was only used in the initial phase of the project to understand what data that had to be stored and get an idea of how the database needed to be constructed.

The ER-diagram of BookIT has three entities: Users, Bookings and Rooms. Data that is directly connected to the entities has to be stored. This data is represented by attributes and in the ER-diagram they are illustrated as ellipses, which can be seen in Figure 42 below.

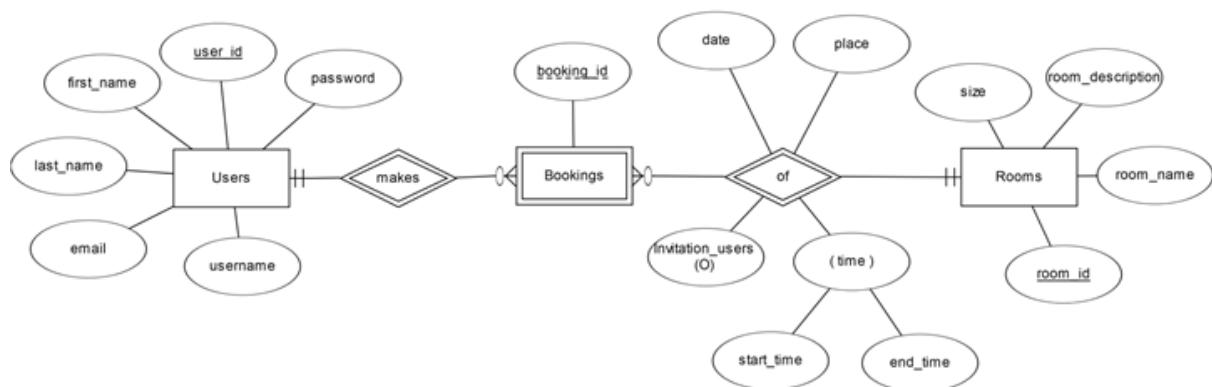


Figure 42. An ER-diagram of the room booking system. Entities are represented by rectangles, attributes by ellipses and relations by rhombs.

In the ER-diagram Users, Bookings and Rooms are connected by relations. Each booking from Bookings must be related to one user from Users and to one room from Rooms, but a user or a room does not have to be related to a booking in order to exist. Both Users and Rooms will exist regardless of whether bookings have been made or not. A booking from Bookings on the other hand does not exist until a user actively creates a reservation of a room, which makes Bookings a weak entity. A weak entity is an entity that cannot exist without an owner entity and is graphically represented in the ER-diagram with a double border around the entity rectangle. In this case Bookings cannot exist without Users and Rooms, which therefore will represent the owner entities. The relationship between a weak entity and its owner is called identifying relationship and is represented in the ER-diagram by a double border around the relationship rhomb, as can be seen in Figure 42. The weak entity type needs a partial key which in this case is `booking_id`. A partial key is an attribute which together with the owner's key can identify the weak entity and is underlined with a dashed line in the ER-diagram shown in Figure 42.

A reservation does not exist if it is not related to one user and one room. A user can have many different bookings, but one booking can only belong to one user. The invite feature enables the user to invite other users to a booking, but the owner of the booking is still the only user that can remove the booking from the database. The invited users can only remove their invitation to the booking, thus not appearing in their own profile. A room can belong to several bookings, provided that the bookings cover different dates or time, but for a booking only one room applies.

5.2 BookIT Database

Even though the ER-diagram could not be converted into a NoSQL database, there are still a few similarities between the structures. The three entities in the ER-Diagram correspond to the three main collections in the database built with Cloud Firestore. The database is illustrated in Figure 43 below, where the collections are marked with red. All entities have values connected to it, that have to be stored, which in the ER-diagram above is represented as attributes and in the final database as fields in documents. The ER-diagram with its attributes made it possible to overview what data needed to be stored in order to get a functional booking system with all the features of BookIT.

In contrary to SQL databases, the NoSQL database can have duplicates of data if it improves the performance of common queries. As a result, some actions like adding, removing or changing data has to be performed in several places in the database, which must be specified in the code. The cost of this improvement of performance is more resources in the form of space, since there is more data that has to be stored. The basic structure of the database is built around the three main collections: Users, Bookings and Rooms. These are presented and described in the sections below.

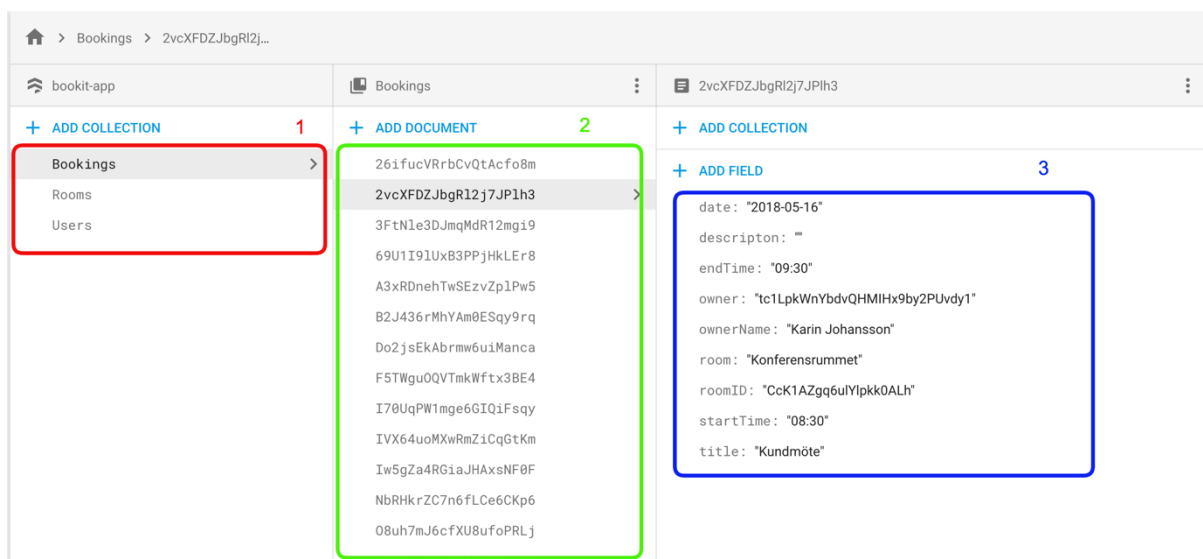


Figure 43. Screenshot of the database of BookIT. The main collections are marked with red. The fields marked with green are the documents under *Bookings*. Fields marked with blue is the data, stored in one of the bookings documents, that is, the data that one booking contains.

5.2.1 Bookings

The main collection *Bookings* contains all bookings that have been created by all users. These are marked with green and shown in Figure 43 above. All fields have its own ID, which consists of numbers and letters. These ID:s are set automatically by Firebase in the main collection *Bookings*. In the subcollections *bookings* of *Users* and *Rooms* the ID:s are set manually in the code where they are copied from the automatically generated ID from the main collection *Bookings*.

5.2.2 Users

When building the database of the application, the usage of the application had to be considered. The application has three different main features which are located in the screens “Hem” (“Home”), “Boka rum” (“Book a room”) and “Dina bokningar” (“Your bookings”). These are presented with a more detailed description in section 4.1 *Presenting BookIT* above. To make the fetching of data more efficient, some data was duplicated to speed up the process. For example, in the screen for the profile page, “Dina bokningar”, the user wants to have its user information as well as its own bookings. Instead of using the user ID and going through all bookings that all users have created and compare the user ID with the owner ID of the booking, the user has a subcollection of its own bookings. The User collection and the subcollection of a user’s bookings are shown in Figure 44 and Figure 45 respectively below. When a user goes to its profile page, the data can easily be fetched since there is no filtering needed, which speeds up the process. The data that is stored in *Users* is the name of the user, the user’s email address and the bookings that the user has created. There is also an item named “invitations”, which can be seen in both Figure 44 and Figure 45 below. This item has been created in the database but the feature has not been implemented in the application yet.

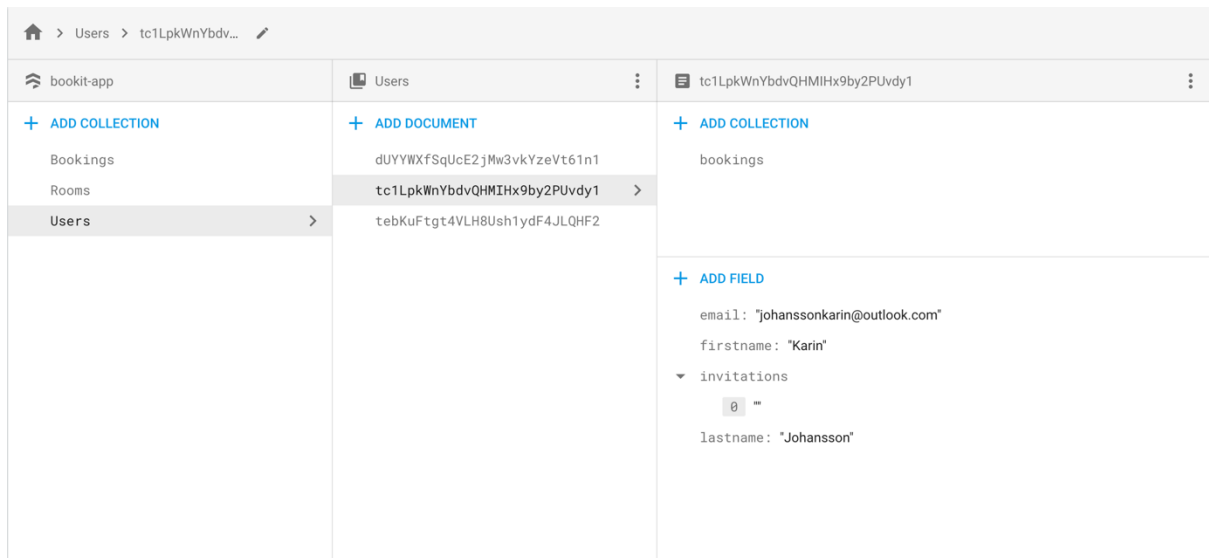


Figure 44. A screenshot of the content of the main collection *Users*.

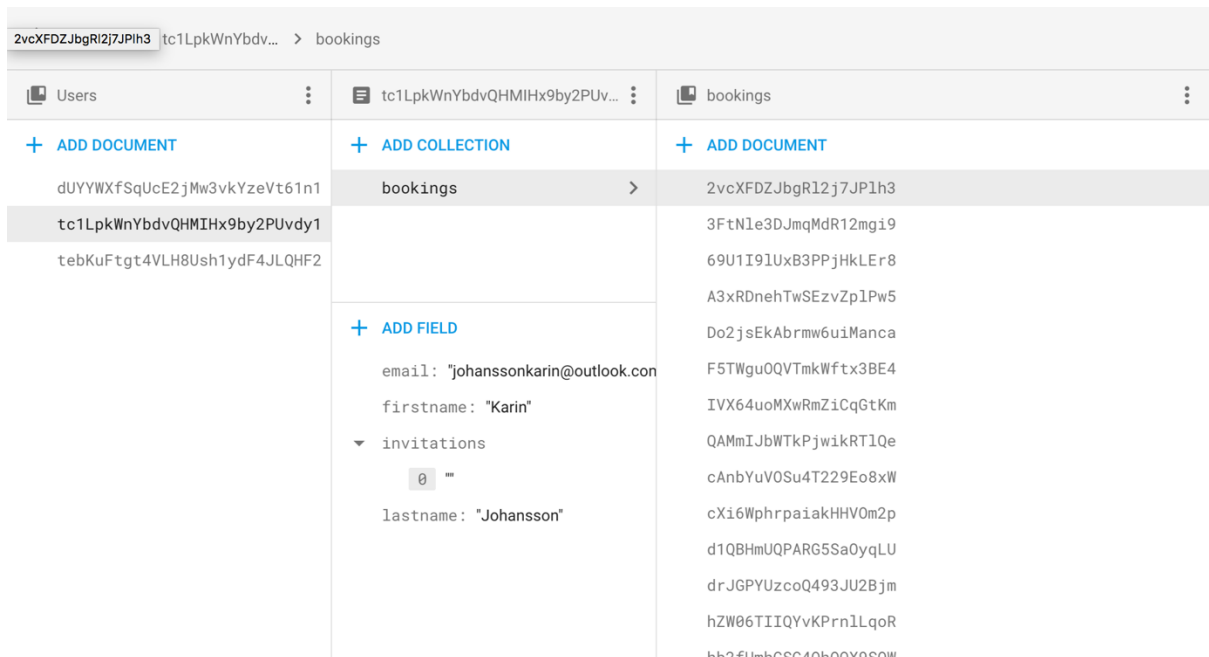


Figure 45. A screenshot of the content of the subcollection *bookings* which is located under main collection *Users*.

5.2.3 Rooms

The collection *Rooms* contains all rooms at the company Active Pollutions. The data that is stored is the name of the room, the room ID, information about the room, the location of the room and all the bookings that are placed in that room. The data structure for *Rooms* and *Users* is similar to the structure of *Bookings*. The main difference is that both *Rooms* and

Users have subcollections which contain copies of the bookings for that specific room respective user. This can be seen in Figure 46 and Figure 47 below. Initially, the bookings were arranged into arrays and not subcollections. However, when tested, it became clear that updating, removing and fetching data as well as performing queries, became severely more complicated by this structure. It was therefore decided to create subcollections instead, which solved the issue of managing the data and also made the overview of the database considerably easier.

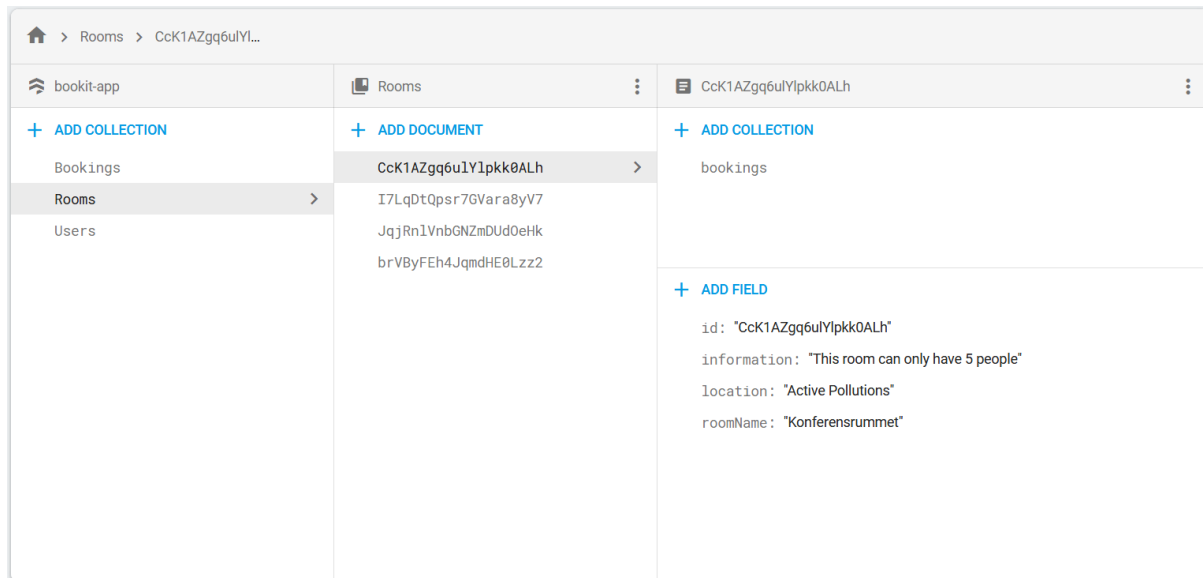


Figure 46. A screenshot of the content of the main collection *Rooms*.

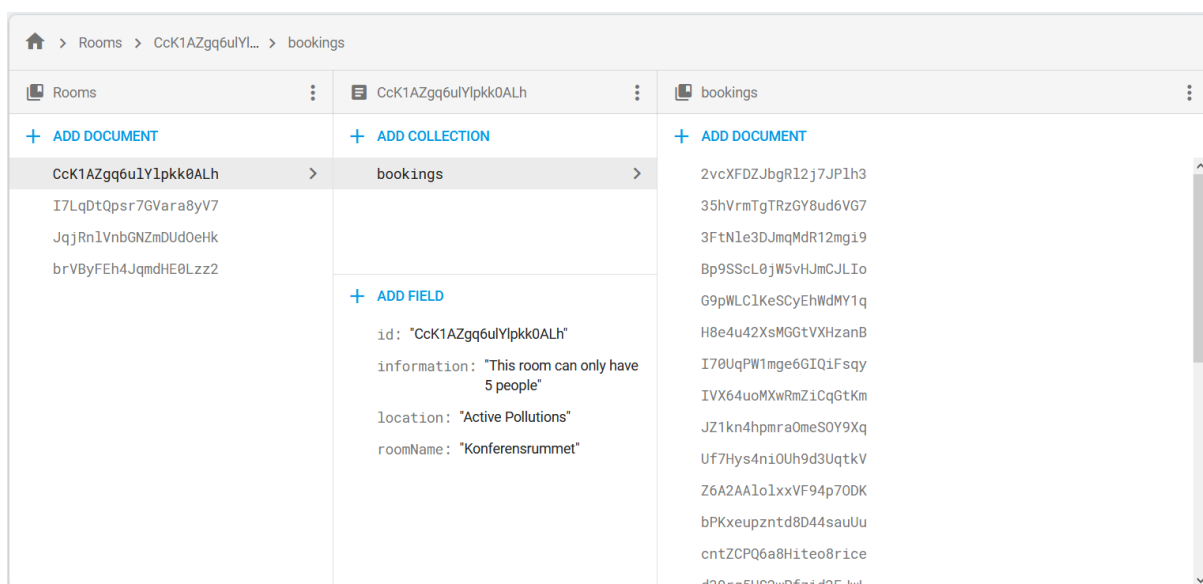


Figure 47. A screenshot of the content of the subcollection *bookings*, which is located under the main collection *Rooms*.

5.3 Process of the database

Since a NoSQL database was a new concept for the developers of BookIT, the initial approach was to construct an ER-diagram based on the SQL database structure. The thought was to afterwards “convert” the diagram into the corresponding NoSQL database. As the team learned more about how to construct and handle a NoSQL database, they learned that the two types of databases are significantly different. The ER-diagram was therefore not of much help with the storage structure of the database but worked as an overview of what data that had to be stored. The ER-diagram is a way of mapping out the data and the structure of the application.

5.4 Security and Privacy

Since BookIT is built around a specific company/organization, the users are protected by the fact that only they are the ones that can access the system implemented for their specific company or organization. No external person or third party can sign into or access the system if they are not a part of the company and may use the company’s communal spaces. Firebase Authentication provides back-end services, Software Development Kit and User Interface libraries to authenticate users to the app. For BookIT, the authentication was handled with a password, one of the possible verification systems that Firebase provides. (19) This function means that no other user of the system can make reservations in another user’s name unless the user knows the other user’s password.

Firebase security applies Google's internal expertise of managing one of the largest account databases in the world (20). It is built on Google’s infrastructure and scales automatically (21). Services of Firebase encrypt data in transit using HTTPS and logically isolate customer data. Several Firebase services also encrypt their data at rest such as Cloud Firestore and Firebase Authentication (22). By using Firebase, BookIT gets the security functions that Cloud Firestore provides.

Since there was no administration site for BookIT there are no users with special privileges to the data, no user can read the data connected to other users. If an administrative page were to

be made, the security for that page would have to be structured in a way that keeps the integrity intact for the users.

6 Evaluation

By the end of the project the minimum goals that were set up in the beginning were reached. Although there were several areas of improvement and some wanted features that, due to the lack of time, did not get implemented. There was a lot to learn during the time period and lots of problems to tackle. To evaluate the application throughout the project a lot of tests were made by the developers. This section will discuss the testing part during the app development as well as an evaluation of the final product.

6.1 Testing Methods

During the development of the application, tests were continuously performed. All functionality tests were completed by RåKod and the usability tests were created by the developers but performed by test persons. Since the application was built for iOS, the tests were only made on iPhone simulators, and not in any Android environment.

Before beginning creating a new component during the building of the application, the members of the team investigated how other applications had designed the same or a similar feature. If there is some sort of standard for implementing this kind of feature or if some application had a relevant solution, it could work as inspiration for how it could be implemented in BookIT. The apps that were explored were Spotify, iOS calendar, Acast, Facebook, SL together with some other applications.

6.1.1 Database Testing

This application is built to support different databases. The idea is that a company, organization or some other institution with users such as employees, students or members, that have common rooms they want to share can use this service. To be able to test the application during the building phase, a testing database was created with the company Active Pollutions as a reference. The test database was used both to test the application but

also to test the structure of the database and how a company database would have to be constructed.

6.1.2 Simulator Testing

One of the tools used when building the application was Xcode, where one of the offered features is a simulator. Xcode is described more detailed in section *2.1.1 Tools for Development*. Most of the testing was performed through this simulator. Tests were performed both during and after each built component. The simulator enables the testing of both the design and the functionality of the component. Since the simulator looks like the device which is simulated, the proportions that is shown are correct compared to a mobile device. Xcode also helps with debugging and sends out warnings when there are errors or inconsistencies in the code. These errors and warnings also include a stack trace and information to make it easier to solve the issues.

The Xcode simulator supports both unit testing and system testing. Unit testing was achieved mostly by alerting which is similar to logging in the console, but differs in the way that alerts shows a pop up window with a wanted message or text string. By placing these alerts in different parts or units of the code, these were tested and debugged. The system was also continuously tested by running the app on the simulator and navigating through the application while testing the functionality and integration of features.

6.1.3 Usability Testing

The usability tests were performed both when the initial mockups were put together and when the final product was almost completed. The mockup testing was done with a computer in Marvel, which is a program described in section *2.1.1 Tools for Development*. The testing of the final product was done on an iPhone 7. The test persons were between 22 to 68 years of age and fluent in Swedish. They had never tested or seen the application before. The test persons had different level of technical skills but all of them had their own iPhone and where used to the system specific standards of navigation and layout. The test tasks along with the

evaluation form that were used in the usability testing of BookIT can be found in the Appendix, section 9.2 *Usability Assessment*.

As previously stated, the test of the mockup was the main reasons behind that the initial project idea shifted. The first tests mostly helped figure out logical flaws in the navigation and mayor layout choices. The second round of testing resulted in rethinking some earlier design choices. This is described in more detail in section 4.3.3 *Mockup and Usability Test* above. The user interface tests of the application were performed by the end of the development process. One reason why the testing was not performed earlier in the process was that the booking feature was finalized in the latter part of the project and the already finished features did not get tested until the booking feature was completed. The user tests were performed when there was an actual application with all the minimum required features. However, it could have been of interest to test even small components in the application earlier in the development process, something that was learned through this experience.

6.2 Before Launching the Application

Even though the app reached the minimum requirements there are some things that should be altered before it is launched on the market. BookIT is a generic application but still need some adjustments to work properly for all costumers. This section will present features that are wanted but could not be implemented due to time limitations of the project.

6.2.1 Create an Administration View

The idea of this application was to sell it cheap but then also sell the support as a subscription service. Since the application does not have an administration site, the only way to set up the database is for the developers to manually do it. The maintenance of the system also has to be performed by the developers, for example if some user should be added or removed. One solution could be to build an administration screen, with one or several users with access to it. Since it would severely impact the flexibility for the users to be able to update the database of users and rooms, an administrative page would greatly improve the quality and usability of the application. It might even be necessary to be implemented before it would be usable. The

administrative page could also include some possibilities for the administrator to change the design. For example, changing the colors of the app dependent on the company's graphical profile or own wishes.

6.2.2 Improve User Interface

One of the main things that the usability testing showed was that the user interface lacked a lot of intuition and would have to be improved to make the application more sufficient. A lot of work could be done with just an upgrade of the design of the application, but the main thing would be to make it more generic in relation to other apps; buttons with same appearance has same function and that the buttons with a certain functionality are located in the same places as in other apps.

6.2.3 Wanted Features

Some features that were high on the priority list but were not reached by the end of the project were

- A language button.

Since there are several foreigners working on companies or organizations in Sweden it would be preferable to make it possible for them to use the application by implementing a language button that supports Swedish and English, at the very least. Pressing the button would switch all language of the coded names into English, the data and names of the room would stay the same.

- Making the app Android compatible.

Since the application is written in React Native all code should in theory be working for an Android device as well. There are some small fixes like the app icon, launch screen and performance tests that need to be done before being ready to release the app for Android. The app could probably not be used in a company if it was not compatible with both Android and iPhone since a demand for iPhone is unlikely.

- Invitations.

Enabling a feature to make it possible to invite other users to the booking, since users would want to overview all bookings they are to attend and not just the ones they have created themselves.

- Integration into other apps.

The integration would enable useful features such as loading bookings into the calendar in the mobile device. Often people want one unit to collect all information in and this is something that would fulfil that need.

- Push notifications.

The possibility to add a reminder that you have a booking coming up or to be notified whenever someone else has invited you to a booking. This would improve the user experience and possibly result in more people remembering to cancel bookings they cannot make.

6.3 The Application in the Real World

The scalability of the system is handled by Firebase which scales completely automatically, meaning that the data do not have to be shared across multiple instances. The performance of the system would have to be tested on a bigger database. The structure of the database, especially the duplication of data, would have to be tested in order to make sure that the performance of the application would not be significantly affected by the larger amount of data that the structure demands.

7 Conclusions

In the beginning of the project the team had big ideas of what to construct, that systematically got more narrow and precise until it finalized in the generic version of BookIT. The initial plan about creating a group room booking app for Uppsala University was as desirable by the end of the project as it was in the beginning. However, the members of RåKod are still pleased with the decision of creating the generic version instead of concentrating on adapting it to Uppsala University. Sticking to the initial plan would have meant spending a lot of time on understanding and using an existing system instead of spending time on learning, planning and finishing a new smartphone application from scratch.

There were a lot of features of BookIT that was planned on being implemented but was not, due to the lack of time. One of the major components that was not implemented and that probably would have enhanced the experience of the app was the invitation feature. Other features were to send bookings to the calendar on the phone, create push notifications when a booking was to start and creating a more intuitive and graphical dependent way to choose the start and end time of the booking. Creating a new and more intuitive way of choosing start and end time was something that the team believed was possible, but that turned out to be more complicated than imagined. The features and tools that React Native offers might not be sufficient enough for this purpose, at least at the developers of RåKod's skill level.

The biggest learning experience from this project was probably to not spend too much time on some of the issues that arose. Sometimes it was more efficient to choose a solution that adequate and worked, instead of trying to succeed with the best one that consumed lots of time to implement. The first steps of the project were the most crucial and it would have been better to wait and assure that all software worked for all members in the group before beginning building the application. It also would have been much easier if everyone in the group had the same hard ware because then all members would probably have the same errors and problems and would have been able to help each other.

The entire team learned a lot during the project, including coding languages, NoSQL databases, app development, working in a team together with planning and structuring a

project. Lastly, RåKod also learned a great deal about the process of developing a software project and how to better structure and plan future projects.

8 References

- (1) Vectr App. *Free Vector Graphics Software*. [Online] Vectr Labs Inc., 2018. [Cited: 2018-05-23] <https://vectr.com/>.
- (2) Marvel. *Why Marvel App*. [Online]. [Cited: 2018-05-23] <https://marvelapp.com/why-marvel>.
- (3) ERDPlus. *Welcome to ERDPlus*. [Online] [Cited: 2018-05-23] <https://erdplus.com/#/>.
- (4) Google Firebase. *Cloud Firestore Data Model*. [Online] [Cited: 2018-05-23] <https://firebase.google.com/docs/firestore/data-model>.
- (5) Google Firebase. *Cloud Firestore*. [Online]. [Cited: 2018-05-23] <https://firebase.google.com/docs/firestore/>.
- (6) React Native. *React Native*. [Online]. Facebook Inc., 2018. [Cited: 2018-05-23] <https://facebook.github.io/react-native/>.
- (7) Apple. *Simulator Overview*. [Online]. Apple Inc., 2018. [Cited: 2018-05-23] <https://help.apple.com/simulator/mac/current/#/deve44b57b2a>.
- (8) Visual Studio Code. *Code editing. Redefined*. [Online]. Microsoft, 2018. [Cited: 2018-05-23] <https://code.visualstudio.com/>.
- (9) GitHub. *About*. [Online]. GitHub Inc., 2018. [Cited: 2018-05-23] <https://github.com/>.
- (10) Git. *.1 Git Branching - What a Branch Is*. [Online]. [Cited: 2018-05-23] <https://git-scm.com/book/en/v1/Git-Branching-What-a-Branch-Is>.
- (11) Github Help. *Merging a pull request*. [Online]. GitHub Inc., 2018 [Cited: 2018-05-28] <https://help.github.com/articles/merging-a-pull-request/>.

- (12) Slack. *Where work happens*. [Online]. [Cited: 2018-05-28]
<https://slack.com/>.
- (13) Trello. *About*. [Online]. Atlassian, 2018. [Cited: 2018-05-28]
<https://trello.com/about>.
- (14) Microsoft Onedrive. *Onedrive*. [Online]. Microsoft, 2018. [Cited: 2018-05-23]
<https://onedrive.live.com/about>.
- (15) Dingsøyr, Torgeir; Brede Moe, Nils; Nerur, Sridhar; Balijepal, VenuGopall. 2012. *A decade of agile methodologies: Towards explaining agile software development*. Journal of Systems and Software. Vol. 85, 6. Pages 1213-1221. [Cited: 2018-05-21]
- (16) Smelink.se. *Konsten att välja färger till din hemsida*. [Online]. [Cited: 2018-05-29] <https://www.smelink.se/valja-farger-hemsida.html>
- (17) Apple Developer. *Human Interface Guidelines – App Architecture – Navigation*. [Online]. Apple Inc., 2018. [Cited: 2018-05-30]
<https://developer.apple.com/ios/human-interface-guidelines/app-architecture/navigation/>
- (18) Novák, István. 2012. *Beginning Windows 8 application development*. Indianapolis, IN: Wiley. Page 164. [Cited: 2018-06-03]
- (19) Firebase. *Firebase Authentication*. [Online]. [Cited: 2018-05-23]
<https://firebase.google.com/docs/auth/>.
- (20) Firebase. *Firebase Authentication*. [Online]. [Cited: 2018-05-28]
<https://firebase.google.com/products/auth/>
- (21) Firebase. *Firebase helps mobile app teams succeed*. [Online]. [Cited: 2018-05-30] <https://firebase.google.com>
- (22) Firebase. *Privacy and Security in Firebase*. [Online]. [Cited: 2018-05-30]
<https://firebase.google.com/support/privacy/>

9 Appendix

9.1 Individual Assessment and Reflection

This section will present the reflections of the members of the project team RåKod. Each member will describe what roles they had during in the project, how they contributed to the project and what they have learned from the project.

9.1.1 Karin Johansson

Two years ago I had not written a single line of code in my life and could, most likely, not imagine myself doing so. I could probably even less see myself aiming towards a career within the IT sector or developing a smartphone application. The first programming course I attended really was an eyeopener for me and got me hooked on the possibilities within IT. This course was very different but nevertheless confirmed that programming is surprisingly fun. It is almost overwhelming looking back at the past two years, and especially the at last two months, and thinking about how much I've developed, both on a professional and on a personal level.

Role in the project

The instructions that my team and I got when starting up the project were stressing that each team member should take on a number of different roles. Since we were only three people working together, we all agreed that it felt both difficult and a bit odd to put fancy labels on who should do what. We talked a lot about what we all wanted with the project and what parts that each person wanted to focus extra on. I have in previous projects had the tendency to take on the role of managing and structuring but felt more motivated to try out a different side of a project group this time. This was much due to the project context and that I wanted to work on my programming skills. I would like to say that is also what I did and the fancy label for it would have been 'technical manager'.

Since RåKod decided to use Firebase there was not really much of a point dividing responsibilities with front-end and back-end, because they turned out to more or less the

same. So all of the team developers, including myself, could be seen as both front-end and back-end. I think it was good to be able to get a better overview and learn to think of the whole picture, how the data flows from the database to the user. Especially on a project of this size which I was involved in creating from scratch.

One of the bigger struggles we had was to keep up the team spirit throughout the project, especially towards the end when the deadlines were closing in. This led to me kind of taking on the role as ‘team counselor’ towards the end of the project. This came in a natural way but is perhaps a role that is just as important as any other if we would have taken on roles from the beginning. I do not think that having more specific roles or responsibilities would have affected either the final product or the team spirit so much but that is just guessing.

Contribution to the project

As mentioned above my role in the project was foremost toward the technical side. Throughout the project I am content that I really contributed a lot to the product itself. I did most of the work behind the graphical design, i.e. company and application logos and was a driving force when it came to the application development. For example I constructed a code skeleton, all navigation, app icon, launch screen and a lot of major component like the booking calendar, agendas and so on. This was probably the best and most fun part of the project in my opinion, to be able to learn and create so much in such a short amount of time.

My many contributions to the code and final product led to me being a bit more passive when it came to writing the report. I initially wrote the introduction, parts of the evaluation as well as the conclusion. These text parts were however improved after receiving feedback from another project group by other members of RåKod. All members of the team were involved in the proofreading which makes it harder to distinguish exactly what text parts I wrote.

Learning experiences

Firstly, the process of developing BookIT was a great learning experience for me. I had never worked with either React Native or Google Firebase before so the startup phase of the project was quite long and stressful. I think the stress was mostly generated by seeing the time pass and not seeing any result or development on the actual product. Since this was the first time I

did a project like this I had no idea that the learning curve would be steep and that the component that took a long time to build in the beginning would be implemented in no-time towards the end of the project. I think it was a good choice to try to scale down the project to the amount of time we had to finish the product. I also think it was wise of us to choose tools and frameworks such as React Native and Firebase. This enables us to basically write the entire application in JavaScript. I really feel like I had the time actually learn the React library rather than just beginning to learn the basics as in all the other courses we have had until now. However, it was still profitable that I to some extent was familiar with JavaScript and the different choices of editors since earlier. I think what I learned through this was not to try to plan too much when you are going in to a project blind. If you have no idea what you are doing there is simply no point in trying to make a super strict time plan. I also learn not to stress the learning process simply because it is so important and what I value the most in retrospect. Secondly, I learned different aspects of working intensely together with your close friends. This is truly something that has its pros and cons but in the end something that I am really glad that we did. It was a good feeling that the group managed to finish a real application with the minimum features that we set out as goals. Especially, doing so together.

To sum up, It is hard to point out all the things I have learned throughout this project. One thing I know for sure is that in two years from now when looking back I think I really would have had use for everything this project gave me.

9.1.2 Nadja Fadhel

This project has really been rewarding. It has been fun and educational as well as challenging and frustrating at times. I have learned a lot, both regarding technical and non-technical aspects.

I am glad that I did this project with the other members of the team. I am happy all of us learned this much together and I am proud of what we managed to contribute in such a short amount of time.

Role in the project

Since our team, RåKod, consisted of only three members, we decided to not divide the team in different project roles. Partly because we felt that it would be difficult to decide which one should take which role and also because it felt a bit odd to divide the members into roles when we were such a small group. The project roles kind of evolved naturally during the working process. In the bigger picture my responsibility and project role became to manage most of the administrative and theoretical parts of the project, which includes producing the bigger part of the thesis and other parts of the project that are not directly connected to the actual code writing of the application. This responsibility evolved due to me having technical issues with the needed software preventing me from being able to live view the result of the code writing, which I will discuss further below. I do not think that having more specific roles or responsibilities among the team members would have affected the final result of the project significantly, since everybody contributed to the extent they could within the limited time frames. What, on the other hand, probably affected the outcome of the project and could have been avoided was to spend too much time trying to solve problems that in the end did not get resolved. The time spent on trying to solve these problems could have been put on other parts of the project, which would lead to a better result of those parts. However, it is difficult to determine how much time a specific problem will take to solve and you do not know beforehand if you will be able to solve the problem on your own or if you will need help from others, which makes it difficult to control how much time to spend on solving different problems before it is time to leave them and move on.

Contribution to the project

My vision before beginning with the project was to contribute on all parts of the project, which includes the directly technical parts, as building the actual application as well as the administrative and theoretical parts, as writing the thesis.

In the initial phase of the project I contributed to the development of the project idea and to the creation of the mockup together with the other team members. It was in the initial phase of the project we together decided what features and colors we wanted the application to have. During this state we also decided to begin to develop an application for iOS, since all three of us have iPhones. When we decided to use React Native we realized that we could

develop the application as compatible for both iOS and Android, by just changing some platform specific features and include the environment specific packages. Both Karin and Lisbet have Macbooks and therefore had the possibility to install Xcode that offers a simulator tool enabling prototyping and to get a live view of the application while developing it. I worked on a PC and could not use Xcode and therefore searched for other options that provided the same feature. I tried both Expo XDE and Android Studios to get the simulator tool. I spent a lot of time trying to open up the current project for BookIT with Expo. I googled a lot and tried many different solutions. Some of the problems were resolved but as one error got resolved another error appeared. I turned to the supervisors of the project to get help but none of them could solve the problems. Eventually I dropped the idea of trying to get it to work with Expo and installed Android Studios instead. But the wanted feature of this software did not work for me either and I spent a lot of time solving different error messages that appeared when I tried to open up and simulate our project. These issues were very time consuming and caused a lot of frustration and I did not really know where to turn for help. My team members were of course aware of the issues but I did not ask them for help since I did not want to disturb their work with the application and I saw these issues as my own. In retrospect I should have communicated that I needed help and even though my team members probably would not have been able to solve the problems, since I had tried everything, we could have discussed how to proceed in the project a lot earlier. At last I dropped the attempts of trying to get the software with the simulator feature to work and proceeded directly to write and hand in the next deliverable. I then decided to take responsibility for the administrative and theoretical parts of the project, that is, the remaining required parts that did not consist of writing code.

Since I was not able to contribute to the code writing of the project I decided to take responsibility for writing most of the thesis. I wrote the *1.4 Terminology* section, and the section *2.1 Technical Tools*, which includes two sections with a description of all the tools that were used during the project. I wrote *3 Socio-Economic Aspects*, which includes a section about the target users of BookIT together with a section describing the company Active Pollutions, and lastly a section about the business model of BookIT. I also wrote the entire *4 The User Perspective* that includes a section where the entire application with all its screens and features is presented, a section about the graphical design as well as a section about the user interface design of the application. Further, I created the ER-diagram of BookIT and wrote *5.1 ER-Diagram*, which is a description of the initial database design. This

section was initially a section describing both the initial database design and some information about the final database design but was eventually divided into two different sections by another member of the team where the second section was added with a lot more text. This means that I wrote some parts of *5.2 BookIT Database* as well. I have also written *5.4 Security and Privacy*. Beyond the sections I wrote myself I have continuously been reading through the thesis, giving feedback on other sections as well as changing some sections myself by adding or removing content.

Another part of the project that I took the responsibility of was creating the usability test form, which consists of a description of how the usability test is to be performed, how the results of the tests will be used, the rights the test persons have and the tasks that will be performed during the test. Beyond the usability test form I created an evaluation form which the test persons could fill in after they had performed the tasks of the usability test.

Learning experience

I have never done a project like this before and even though I unfortunately could not contribute to the code writing, which I really had looked forward to, I still have learned a lot.

I have learned how a NoSQL database works, especially Firebase, as well as the difference between a non-relational and a relational database. I have learned the importance of reading error messages and how to better search for solutions. I have learned how crucial it is to ask for help even if you feel that it is your own problem that you think you should manage to solve by yourself. Asking for help from others can save a lot of time even if the problems do not get solved. However, I turned for help a couple of times but it did unfortunately not solve any of the issues I encountered.

I have learned which parts are required when developing an application. I have gained experience in the importance of letting people from outside of the project test the application. The more tests performed the better will the final result become. I gained experience in structuring a project of this size as well as in writing and structuring a report as extensive as this one. Further I learned the importance of continuous communication within the team. It is important to communicate what you feel and think regarding everything within the project to improve things.

To summarize, it is difficult to point out all the things I have learned throughout the project since there are a lot, but I am really glad we as a team managed to create an application and write a well written report as well as being able to learn things together.

9.1.3 Lisbet Ersson

Overall I would say I have learned a lot from this project. I am proud of how much we learned as a team through this experience and I am glad we managed to create a working application with most of the important features we wanted. The biggest difficulty we faced, apart from technical problems, was the project process. Something that I believed would have improved the project over all is the communication within the group.

Role in the project

Since we were only three people in the group we did not divide into separate project roles. In retrospect, this is something that could have helped with the structure of the project. Although it was everybody's responsibility to ensure the progress of the project, it might have helped to have assigned some responsibilities in the beginning. Overall, the roles in the group evolved naturally through different phases of the project where the members contributed in different amounts on different tasks. I had some project managing role in some of the phases of the project. In the initial part of the project, making the first pitch, mockup and planning the project I took part in trying to structure the workload and plan the project. While in the application development phase I was not managing the process, but accepting tasks to do and working on them.

Contribution to the project

Working with the application

Before we started coding on the application we worked on the design of the app and decided on which features we wanted the app to possess. We used the mockup tools a lot to visualize out thoughts, at this step everybody in the team contributed equally and we discussed within the group how the app should look like and which functions it should have.

In the development of the application, I struggled a bit in the beginning and some in the intermediate parts of the project. In the beginning of the development we thought to divide the work between front-end and back-end a bit. Since we used Firebase, the back-end was not a heavy workload but our intention was that I should integrate our Firebase database into our project. This changed when we realized that the implementation was trivial and from then on, I worked alongside with the rest of the team and focused on building different components. We reused a lot of each other's codes since some code parts were useable in several different components. This fact made it a bit difficult to say exactly which parts of the application I contributed to. My major component contributions were in the screen named "Dina bokningar" and the screen named "Home". But I did not complete them on my own but got help from the other team mates.

I learned a lot about the code languages and about NoSQL databases but I believe my biggest contribution was to contribute with different features and solutions in the code that the team mates also used in different parts of the code. Lots connected to the fetching, updating and removal of documents and fields in the database. I believe I could have done a lot more if I would have known where to turn to get help and did not have to spend as much time with issues that appeared along the way.

Writing the report

The team chose to not hand in all the deliverables which resulted in a bigger workload by the end of the project when writing the report. I would say that I contributed a great deal in writing the report, both in writing different sections as well as reading through the report and giving feedback to other team members and deciding how to structure the work. I contributed mostly to the method section and the other half of the report. I have never written a report this extensive, it was educational to learn how to write a paper with as much content and information in an eloquently and structural manner.

Learning experiences

I learned a great deal in the process of creating the application. I learned to use React Native as well as Firebase, that were both new to me in the beginning of the course. This is something that I am very pleased with. The NoSQL database was new to me and I am glad that I learned how to use it as well as the difference between non-relational databases and relational ones. Besides the languages, I learned more of how to handle errors and finding

solutions. Sometimes it was too time consuming to come up with the best solution that it was better to settle with a working one but with less quality.

I have learned a great deal from this project, in the programming parts of the project but mostly the social aspects of a project. As it turned out, the biggest issue that the group had were the project structure and collaboration part of it. The team struggled with lots of technical issues that affected the project structure and team balance. This experience thought me the importance of communication within the team, to be able to confront each other in order to help each other with the issues that different members faced, as well as the advantage of ongoing evaluation and overview of the project. In this project, all members of the group were eager to contribute and perform and when issues were met it was hard for the teammates to ask for help. Partly because it was hard to admit the issues and partly because the other teammates seldom had the knowledge to help.

When reaching out for help to supervisors, teammates and other helping sources did not help, it was hard to know what to do. This led to frustration that was hard on the team spirit. I believe that some of this team related issues could have been solved by the problems being resolved when contacting people for help. It would have been good to have some kind of workshop in the beginning of the project for everybody in the course to start up all software and have some kind of startup session. Furthermore, I believe the team should have arranged continuous evaluations, although I understand that some of the problems were hard to notice while being in the middle of the project, to detect the project structure weaknesses and being able to talk about the frustration within the team. Lastly, I think the team would have benefited from moving back to the office with the rest of the class. The classmates could have helped with the technical difficulties and it might have helped the structure of the project to be inspired by how the other teams structured themselves. This was something that the team discussed by the end of the project and agreed to do in the last part of the project.

In the end, the team managed to create an application, learn about a new kind of database and implement it to the app and write a, according to me, well written report. The team faced a lot of problems and it was difficult to handle them which put a lot of pressure on the group. This was a big learning experience for me and for the entire team. We learned from our mistakes and in the future we will hopefully improve on how to construct a project.

9.2 Usability Assessment

This section presents the usability test form and the evaluation form that was used in the usability tests of BookIT. Both forms are written in Swedish since both BookIT is in Swedish and all test persons speak Swedish.

9.2.1 Usability Test

RåKod


Karin Johansson, Lisbet Ersson & Nadja Fadhel

Usability Test

Nedan följer information om denna användbarhetsutvärdering. Informationen syftar till att beskriva vad testet innebär och ge dig underlag för beslutet ifall du vill delta i testet eller inte.

I denna utvärdering kommer vi att be dig att testa på vår app BookIT. Testet görs i syftet att utvärdera användbarheten och användarvänligheten av appen. Resultatet kommer senare att användas för att avgöra hur appen kan förbättras och vidareutvecklas. Du kommer att få ett antal uppgifter som täcker appens olika funktioner. Under tiden du utför uppgifterna får du gärna berätta högt vad du tänker och tycker samt ställa frågor om det är något som är otydligt i uppgifterna. För att på bästa sätt uppfylla syftet med testet kommer vi inte att hjälpa till med själva utförandet av uppgifterna.

Den informationen som samlas in under detta test och som eventuellt senare används i vårt arbete kommer inte att associeras med ditt namn. Du är helt anonym och all information om ditt deltagande kommer att behandlas konfidentiellt. Om du väljer att delta har du all rätt att när som helst avbryta utvärderingen utan att detta innebär några som helst konsekvenser för dig.



BookIT

BookIT är en applikation för smartphones som tillhandahåller ett smidigt och enkelt system för bokning och avbokning av ett företags eller organisations gemensamma utrymmen. Dessa utrymmen kan vara i form av exempelvis rum, lokaler eller en tvättstuga. BookIT är under utveckling och är för tillfället implementerat på ett företag som har 4 rum. En användare av BookIT får en egen profil och har möjlighet att göra bokningar av rum, avboka sina bokningar och se vilka rum som är lediga för tillfället. Användaren har även möjlighet att göra vissa ändringar på sina befintliga bokningar och är försedd med information om applikationen och företaget bakom den.

Vänligen skriv under nedan om du förstår dina rättigheter i denna utvärdering och vill delta.

Ort och datum

Namnteckning

Namnfortydligande



Karin Johansson, Lisbet Ersson & Nadja Fadhel

Nedan följer uppgifterna som vi önskar att du genomför under detta test.

Uppgift 1:

- Logga in (med det befintliga användarnamnet och lösenordet du fått)
- Utforska appen i några minuter
- Logga ut

Uppgift 2:

- Logga in igen (med samma användarnamn och lösenord du fick i tidigare uppgift)
- Gå till din profil för att se befintliga bokningar
- Gå till sidan där du kan boka rum
- Boka Konferensrummet den 31:a maj klockan 11:00-12:00
- Döp bokningen till "möte"

Uppgift 3:

- Utan att ha ändrat något från Uppgift 2, gå till sidan som visar vilka rum som är lediga just nu
- Tryck på ett av rummen som lediga och titta om det är ledigt imorgon också.

Uppgift 4:

- Utan att ha ändrat något från Uppgift 3, gå till din profil
- Du ska nu se två befintliga bokningar. Redigera en av bokningarna genom att byta titel på den och gå tillbaka.
- Avboka en av bokningarna.
- Logga ut.

Uppgift 5:

- Du har glömt ditt lösenord och ska nu byta det. Byt ditt lösenord.

9.2.2 Evaluation Form

Rå K o d

Karin Johansson, Lisbet Ersson & Nadja Fadhel

Här följer ett utvärderingsformulär kring hur du upplevde BookIT och dess funktioner. Formuläret innehåller ett antal påståenden som du svarar på genom att välja en siffra på en skala 1-5, där 1 = stämmer inte alls och 5 = stämmer helt och hållet.

- | | | | | | | | | | | | |
|--|---|---|---|---|--|--|---|---|---|---|---|
| 1. Jag tycker att appen var lätt att använda redan från början. | <table border="1"><tr><td></td><td></td><td></td><td></td><td></td></tr><tr><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td></tr></table> | | | | | | 1 | 2 | 3 | 4 | 5 |
| | | | | | | | | | | | |
| 1 | 2 | 3 | 4 | 5 | | | | | | | |
| 2. Jag tycker att appen var svår att använda i början men att det blev enklare ju mer jag använde den. | <table border="1"><tr><td></td><td></td><td></td><td></td><td></td></tr><tr><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td></tr></table> | | | | | | 1 | 2 | 3 | 4 | 5 |
| | | | | | | | | | | | |
| 1 | 2 | 3 | 4 | 5 | | | | | | | |
| 3. Jag tycker att appen var komplex och svår att hantera. | <table border="1"><tr><td></td><td></td><td></td><td></td><td></td></tr><tr><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td></tr></table> | | | | | | 1 | 2 | 3 | 4 | 5 |
| | | | | | | | | | | | |
| 1 | 2 | 3 | 4 | 5 | | | | | | | |
| 4. Jag tror jag behöver en genomgång av appen innan jag kan känna mig säker i användningen. | <table border="1"><tr><td></td><td></td><td></td><td></td><td></td></tr><tr><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td></tr></table> | | | | | | 1 | 2 | 3 | 4 | 5 |
| | | | | | | | | | | | |
| 1 | 2 | 3 | 4 | 5 | | | | | | | |
| 5. Jag tycker att de olika funktionerna i appen var väl integrerade. | <table border="1"><tr><td></td><td></td><td></td><td></td><td></td></tr><tr><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td></tr></table> | | | | | | 1 | 2 | 3 | 4 | 5 |
| | | | | | | | | | | | |
| 1 | 2 | 3 | 4 | 5 | | | | | | | |
| 6. Jag upplever appens system som intuitivt. | <table border="1"><tr><td></td><td></td><td></td><td></td><td></td></tr><tr><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td></tr></table> | | | | | | 1 | 2 | 3 | 4 | 5 |
| | | | | | | | | | | | |
| 1 | 2 | 3 | 4 | 5 | | | | | | | |
| 7. Jag upplever appens utseende som behagligt. | <table border="1"><tr><td></td><td></td><td></td><td></td><td></td></tr><tr><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td></tr></table> | | | | | | 1 | 2 | 3 | 4 | 5 |
| | | | | | | | | | | | |
| 1 | 2 | 3 | 4 | 5 | | | | | | | |
| 8. Jag tycker att appens utseende är för enkelt för att upplevas som behagligt. | <table border="1"><tr><td></td><td></td><td></td><td></td><td></td></tr><tr><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td></tr></table> | | | | | | 1 | 2 | 3 | 4 | 5 |
| | | | | | | | | | | | |
| 1 | 2 | 3 | 4 | 5 | | | | | | | |

9.3 Mockup

In this section all pages of the mockup can be found.

