Bachelor's project

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

Espen Kalleberg
Håkon Harnes
Svein Jakob Høie

Influential factors when implementing authentication in web-applications and how software development has been affected by COVID-19 restrictions.

Bachelor's project in Computer Engineering
Supervisor: Ali Alsam (IDI NTNU), Arne Pukstad Juliussen (Sonans)
May 2021

NTNU
Kunnskap for en bedre verden

Espen Kalleberg
Håkon Harnes
Svein Jakob Høie

Influential factors when implementing authentication in web-applications and how software development has been affected by COVID-19 restrictions.

Bachelor's project in Computer Engineering
Supervisor: Ali Alsam (IDI NTNU), Arne Pukstad Juliussen (Sonans)
May 2021

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

NTNU
Norwegian University of
Science and Technology

Preface

This bachelor thesis is written as a part of the computer engineering study program at the Norwegian University of Science and Technology. The project has been carried out over the spring semester of 2021 (11.01.21 - 20.05.21). It was developed under contract with MI & MA Holding AS.

We are assuming the reader of this thesis has a general understanding of system development and basic computer science concepts.

Trondheim, May 20, 2021

Espen Kalleberg

Håkon Harnes

Svein Jakob Høie

Acknowledgements

We would like to thank our supervisor, *Ali Alsam*, for constructive feedback and guidance. In addition to this, we would also like to thank our employer, *Arne Pukstad Juliussen*, for continuous cooperation.

Problem description

The task was to develop a system that could be used by high school students for (A) room booking and (B) classroom attendance registration. The authentication system had to be robust, but also user-friendly. In addition to this, the design had to be responsive and accommodate smart phones.

(A) Room booking consisted of booking a room by either scanning a QR-code on the door, or by searching for the room in the web-application. When selecting a room, the user should be able to see when it is available for the next two weeks. The system also required a separate user interface for administrators and customers. This would be used for managing user accounts and booking policies.

(B) Classroom attendance registration consisted of scanning a QR-code on the classroom door to register attendance. Due to COVID-19, Sonans Trondheim had already acquired a system for this. Therefore, this part of the task was no longer relevant.

For further details, see the Vision Document and the Software Requirements Specification.

Abstract

The security aspect of modern web-applications is more important than ever. As we put more of our information, assets and tools on the internet, we have to protect it. Our task was to develop a room booking application, and we wanted to gain insight into which authentication methods are available, as well as which method is most suited for a system like SmartBooking.

The web-application we developed, SmartBooking, has three types of accounts: User, customer and administrator. Each type has different security requirements. For example, a user is not able perform potentially destructive actions like deleting other user's bookings or information. Therefore, user accounts do not require the strictest and least user-friendly authentication method, but administrator and customer accounts require more security.

Based on the findings in regards to security, we have concluded that using passwords as the primary authentication method is best suited. We have introduced separate requirements for password strength based on the type of user account you have. Two-factor authentication has also been implemented as an additional authentication mechanism for customers and administrators.

At the time of writing this thesis, COVID-19 is an ongoing pandemic that has posed several global challenges. Digital system development has brought a number of challenges to the table. The collaboration between us as team members was impacted in regards to how well we could cooperate when developing the web-application. We set ourselves the goal of exploring the challenges related to developing the web-application. We conclude that it is not the lack of tools that can assist in digital development that is the problem, but the difference between how we are used to communicate, and how we have been forced to communicate. Losing the human interaction and presence is more demanding than what we thought beforehand.

Abstrakt

Sikkerhetsaspektet ved moderne web-applikasjoner er viktigere enn noensinne. Når vi legger ut mer og mer av vår informasjon, ressurser og verktøy på internett, må vi beskytte det. Vår oppgave var å utivkle en applikasjon for reservering av rom, og vi ønsket å få innsikt i hvilke autentiseringsmetoder som er tilgjengelige, og hvilken metode som er best egnet for et system som SmartBooking.

Web-applikasjonen som vi utviklet, SmartBooking, har tre ulike typer kontoer: Bruker, kunde og administrator. Hver type har forskjellige sikkerhetskrav. For eksempel har ikke en bruker rettigheter til å utføre potensielt destruktive handlinger som å slette andres reservasjoner eller informasjon. Brukerkontoer trenger derfor ikke den strengeste og minst brukervennlige formen for autentisering, men administrator- og kundekontoer krever mer sikkerhet.

Basert på funnene våre innenfor sikkerhet, har vi konkludert med at å bruke passord som hovedformen for autentisering er best egnet. Vi har innført separate krav til passordstyrke basert på typen brukerkonto. To-faktor autentisering har også blitt implementert som et ekstra sikkerhetstiltak spesielt tiltenkt kunder og administatorer.

I skrivende stund er COVID-19 en pågående pandemi som utgjør flere globale utfordringer. Digital systemutvikling brakte en rekke utfordringer. Samarbeidet mellom gruppemedlemmene ble påvirket i forhold til hvor bra vi kunne samarbeide med utviklingen av web-applikasjonen. Vi satt oss et mål om å utforske de utfordringene som er relatert til utviklingen av web-applikasjonen. Vi konkluderer med at det ikke er mangelen på verktøy som kan bistå i en heldigitalt utvikling som er problemet, men forskjellen mellom hvordan vi er vant til å kommunisere, og hvordan vi har blitt tvunget til å kommunisere. Å miste menneskelig interaksjon og tilstedeværelse er mer krevende enn vi hadde forutsett.

Contents

List of Figures

Acronyms and definitions

2FA	Two-Factor Authentication
2SV	Two-Step Verification
CD	Continuous Delivery/Continuous Deployment
CI	Continuous Integration
CSRF	Cross-Site Request Forgery
JWT	JSON Web Token
MFA	Multi-Factor Authentication
NTNU	Norwegian University of Science and Technology
NIST	National Institute of Standards and Technology
OTP	One-Time Password
SFA	Single-Factor Authentication
XSS	Cross-Site Scripting

1 Introduction

This chapter describes the problems identified in this thesis, as well as the research questions and report structure.

1.1 Problem description

The 950 students attending Sonans Trondheim have 17 rooms available for group work. They also have the option of booking one of the 11 classrooms, on the off-chance that there are no lectures at that time.

The room booking system at Sonans was based on students writing their name on the door to book a room. Such a booking system undoubtedly has its problems. When students want to book a room, they have to walk around the school in search of an available room. This is an impractical and time consuming process. Say a student books a room two days in advance. That student might forget when they booked the room for, or even which room they booked. To check this, the student would have to walk around to find the room and read through the booking lists on the doors. If the student then decides to cancel the reservation, they would have to bring correction ink to remove themselves from the booking list.

A digital room booking system would provide a list of available rooms. Students would be able to easily search the list and filter it by the room size and when it is available. If a student forgets the booking details, he can simply check the system. The system would also provide a way for the school to enforce their booking policy. Perhaps, due to the pandemic, students are only allowed to make three reservations per week. Enforcing this rule is simple with a digital room booking system, but impossible with the system as it was.

The digital system which we developed will be offered to other Sonans schools. As of writing this thesis, that would result in a user base of over 6000 students. In addition to this, the system may be used in similar organizations where a room booking system is needed. This means that the system by default needed to be scalable.

1.2 Research questions

1.2.1 Research question 1

In the initial documentation for the case, four-digit PIN-code authentication was suggested. The PIN-codes would be generated from the user's phone number using a secret algorithm. In the event that a student forgot their PIN-code, the administration could simply calculate it by using the algorithm. This PIN-code method was chosen by Sonans because it made the authentication system simple to implement and use.

The suggested authentication system, as described in the initial documentation, was implied to be secure. After reading literature and studies about authentication security, it became evident to us that an authentication system is only as secure as the weakest link. We looked at the core concept of the suggested authentication system: Sonans' algorithm for generating the PIN-codes. By graphing the algorithm as a function of the user's phone number, we observed that it was a linear function. This meant that an attacker using knowledge of linear regression could find the algorithm. Further investigation revealed that PIN-code authentication is also susceptible to brute-force attacks. This type of attack does not require linear algebra knowledge, only sufficient computational power.

Sonans' seemingly secure authentication system was vulnerable to multiple types of attacks. This raised the question: If PIN-code authentication is not suited for SmartBooking, which authentication method is? How do we ensure security, while maintaining the originally required user-friendliness? What needs to be considered when implementing the authentication system? This has led to the following research question:

Research question 1
For SmartBooking:
(a) Which authentication method is suited?
(b) What are the influential factors when implementing the authentication system?

1.2.2 Research question 2

On March 11th 2020 The World Health Organization declared COVID-19 a pandemic. [4] At the time of writing this thesis, in spring 2021, the pandemic is still raging. The effect of this is that this project has been done digitally in its entirety. This has undoubtedly affected the development process, and forced us to think differently. This has led to the following research question:

Research question 2
What challenges has the lack of physical presence posed on our collaboration as a team, and how have these challenges been handled?

1.3 Structure of the report

The research questions cover both authentication and system development. Therefore, each chapter in this thesis will first cover authentication, then system development.

- Chapter 2 - Theory: This chapter first describes authentication concepts, methods and implementation strategies. Then, system development methods and collaboration strategies are presented.

- Chapter 3 - Method and choice of technology: This chapter first describes the process of selecting and evaluating authentication methods. Then, the authentication implementation is described in detail. Finally, the system development methods, strategies and tools that have been used throughout the project are presented.

- Chapter 4 - Results: This chapter presents the authentication, engineering and administrative results.

- Chapter 5 - Discussion: This chapter discusses the implementation of the authentication system. Then, it describes how we have experienced remote collaboration. Finally, the strengths and weaknesses of the system is accessed.

- Chapter 6 - Conclusion: This chapter presents the conclusion, as well as further work and improvements.

2 | Theory

This chapter first describes authentication concepts, methods and implementation strategies. Then, system development methods and collaboration strategies are presented.

2.1 Authentication

Authentication is the process of ascertaining that somebody is who they claim to be. When a user enters their email-address or username, they are providing their identity. When the correct password is provided, they have successfully proven their identity and are thus authenticated.

"Authentication is the act of proving an assertion, such as the identity of a computer system user. In contrast with identification, the act of indicating a person or thing's identity, authentication is the process of verifying that identity." [5]

— *The Economic Times*

2.1.1 Authorization

This thesis is about authentication, not authorization. However, since they are related and both abbreviated to *auth*, they are often confused with each other. To distinguish between them, the definition of authorization follows:

Authorization is the process of giving users permission to access resources. If you have a private Facebook profile, only your friends can view your posts, i.e. they are authorized to view your posts. Everbody else are not able to view your posts, i.e. they are unauthorized to view your posts.

2.1.2 Authentication factors

An authentication factor is a category of credentials that is used for authentication purposes. Here, the three factors of authentication are presented.

Something you know

The concept is that you know a secret that distinguishes you from all other individuals. There are two prerequisites:

1. You need to remember the secret
2. The secret needs to be kept a secret

The secret can be a password, a passphrase or an answer to a security question only you know.

Something you have

This refers to some physical object in the possession of the user. There is only one prerequisite: The user needs to be in possession of the object. The physical object could be your smart phone, or a separate code generator.

Something you are

This authentication factor is based on distinct characteristics like your fingerprint, face, retina or voice. It is something you are, information that is in *you*. This is called biometrics. For instance, to access a web-application on a smart phone, the user can provide their fingerprint to authenticate themselves.

2.1.3 Multi-factor authentication

Multi-Factor Authentication (MFA) depends on several authentication factors. This is more secure, because the user has to provide additional information (credentials) to prove their identity. For instance, the authentication system for a web-application can require users to provide:

1. A password, i.e. something the user knows
2. A code from a physical device, i.e. something the user has

The advantage of this is that if an attacker knows your password, but is not in possession of the physical code generator, he will be unable to authenticate himself. A 2019 Google study showed that 100% of automated bot attacks and 76% of targeted attacks are stopped with device-based MFA. [6]

2.1.4 Two-factor authentication

Two-Factor Authentication (2FA), a type of MFA, depends on two authentication factors. The example in the previous section, 2.1.3, was 2FA.

2.1.5 Single-factor authentication

Single-Factor Authentication (SFA) depends on one authentication factor. A web-application that uses only password-based authentication is a form of SFA. In situations where additional security is required, MFA is recommended.

2.1.6 Two-step verification

Two-Step Verification (2SV), a type of SFA, only depends one authentication factor. It is called *two-step* verification because it requires two login credentials. These login credentials for are the same type of authentication factor, for example something the user knows. If the login credentials are different types of authentication factors, for example something the user knows, and something the user has, it is 2FA.

2.2 Authentication methods

2.2.1 Password authentication

A password is a secret word or phrase that can contain upper- and lowercase characters, numbers and special characters. The idea is that the password is something only the user knows.

Strength factors

1. Length: Longer passwords taker longer to crack.
2. Character set: The more characters that can be used in a password, the greater number of possible combinations of characters. This leads to a greater password space, making it computationally harder to crack.

3. Randomness: The more random the password is, the harder it is to crack. Attackers will often use dictionary attacks where they attempt to log in with commonly known passwords. If the user chooses a random set of characters, rather than an English word, it will be harder to crack.

Vulnerabilities

The issue with password authentication is that:

- Passwords that are easy to remember are easy to guess
- Passwords that are hard to remember are hard to guess

Memorizing passwords is not recommended, because the user would likely pick a password that is easy to remember, and easy to guess. However, 53% of people rely on their memory to manage their passwords, [7] and only 24% use password managers. [8]

Passwords, even strong ones, are susceptible to credential stealing. For instance, shoulder surfing is a social engineering technique where someone spies over the victim's shoulder to see what they are typing. Phishing is another attack where an attacker impersonates themselves as a trustworthy entity. The attacker lures the victim to enter personal information into a fake web-application which matches the look and feel of a legitimate one.

Once the credentials are stolen, they can be used to gain access to other web-applications. This is called credential stuffing, and is made possible because 81% (of Americans) use the same password for multiple online accounts. [9]

Recommendations

The National Institute of Standards and Technology (NIST) have developed a list of password guidelines: [10]

1. Length over complexity: The password should be minimum eight characters long. There are no password-complexity requirements. The reason for this is that requiring complexity can lead to *less* secure passwords. Users tend to add complexity by capitalizing the first character of their password and appending numbers and special characters to the end of the password. This is a pattern that attackers can exploit.

2. Breached password protection: Passwords should be checked against a blacklist con-

taining breached passwords.

3. Two-factor authentication: Require another authentication method for added security.

PIN-codes

PIN-codes are numeric four to six-digit passwords. They are as such weak passwords, and not suited for web-application authentication. Four-digit PIN-codes only have 10.000 unique combinations and are therefore susceptible to brute-force attacks. Furthermore, a study showed that by attempting the five most used PIN-codes, one would have a 20% chance of guessing right. [11]

Usability

A study comparing six authentication methods found passwords to be the second most user-friendly authentication method. [12]

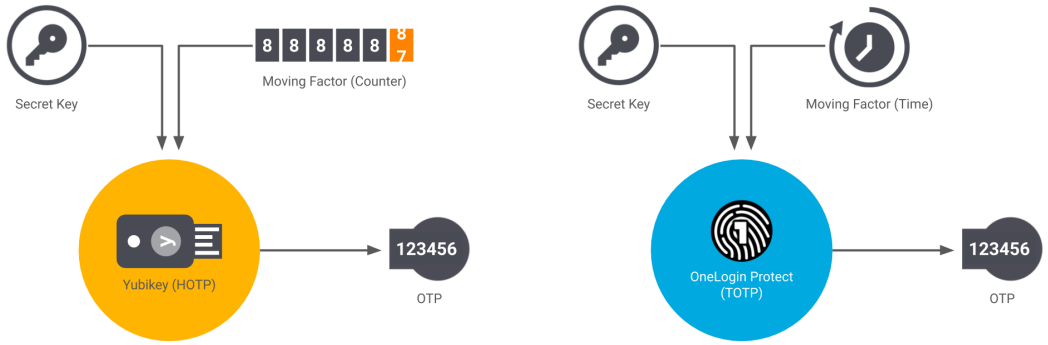
2.2.2 One-time password authentication

A One-Time Password (OTP) consists of randomly generated characters or numbers. It is a *one-time* password, meaning it can only be used once. It is often used in combination with a traditional password as an additional authentication method.

There are two types of OTPs:

1. HOTP: HMAC-based OTP. It is event-based, meaning it is based on a counter that is incremented every time a new OTP is generated.

2. TOTP: Time-based OTP. It is time-based, meaning it is based on time for generating OTPs. They are only valid for a certain time period, say 30 seconds. This makes it harder for attackers to abuse them.



(a) HOTP (b) TOTP

Figure 2.1: One-time passwords [1]

Vulnerabilities

OTPs need to be transmitted securely. SMS is a particularly insecure transmission channel. [13] NIST does not recommend sending OTPs over email or SMS because it *"does not generally prove the possession of a specific device"* [10]. This is the reason why companies like Google [14] and Facebook [15] consider OTPs over email or SMS as 2SV, not 2FA, when paired with password authentication.

Recommendations

By generating the OTPs on-the-fly using an authenticator application, there are no transmission vulnerabilities for the attacker to exploit. Authenticator applications require the user to have physical access to their phone. When paired with another knowledge-based authentication method like passwords, it is considered 2FA.

Usability

A 2019 study compared five 2FA methods and found that passwords authentication with TOTP using Google Authenticator to be the most user-friendly. [16]

2.2.3 Push notification authentication

Push notification authentication works by sending a push notification to a secure medium that the user has access to. This can for instance be a secure application on the user's phone,

or an email-address. The user is then notified that an authentication attempt is taking place, and can choose to accept or deny it. This approach is usually used in conjunction with password authentication as an additional authentication method.

The main advantage to this approach is that users do not need to memorize passwords or provide additional credentials. This makes it a user-friendly and seamless experience. Since the user does not have to enter anything, it is not prone to user errors.

Vulnerabilities

Users often get into the habit of automatically pressing approve when receiving push notifications. This can lead to accidentally approving a fraudulent request. A study from the university of Pittsburgh and Google showed that the human brain drops in attention when shown just two security warnings in a short time span. [17]

Usability

A 2019 study found that push notifications was the fastest 2FA authentication method out of five others. [16] However, it only scored third best for user-friendliness due to requiring possession of a smart phone and bugs in the push notification application.

2.2.4 Biometric authentication

Biometric authentication is based on distinct personal characteristics like your fingerprint, face, retina or voice. The authentication process works by comparing biometric information provided by a scanner, e.g. a fingerprint scanner, with the biometric information stored in the database.

Vulnerabilities

Biometric authentication is subject to presentation attacks, often called spoofing. This works by using an artifact to mimic a person's distinct characteristic. For instance, simple face recognition can be spoofed by presenting an image of the person's face. This is hard to prevent if it occurs, because the victim can easily change their password, but they can't (easily) change their face. However, modern biometric authentication techniques like FaceID is hard to spoof. [18]

The biometric information is stored in a database. This information can be leaked if the

database is compromised. Biometric information, like other sensitive information, should be encrypted before it is stored.

2.2.5 Usability

Fingerprint authentication was perceived by users as the most user-friendly authentication method in a study by Roar S. Sollie. [12] A 2019 study comparing fingerprint, iris and facial recognition systems found that fingerprint authentication was the most user-friendly. [19]

2.3 Authentication implementation

2.3.1 Stateful authentication

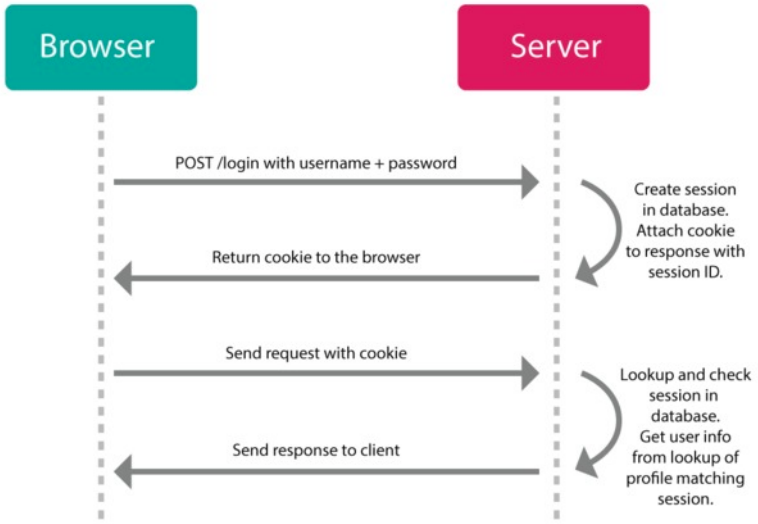


Figure 2.2: Stateful Authentication [2]

In stateful authentication, user sessions are stored on the server-side. When a user successfully authenticates themselves, the server creates a new session with a corresponding session ID. This session ID is stored a cookie in the user's browser. For every subsequent request, the cookie with the session ID is automatically attached. The server can then find the corresponding session in memory, check its validity, and process the request accordingly. The process is illustrated in figure 2.2.

This approach has long been the tried-and-true method for authentication. Despite this, it has several disadvantages, mainly to do with scalability. As the amount of users increase, so does the amount of active sessions stored on the server. The amount of active users is then limited by the server's storage capacity. Also, the more active sessions, the more server resources are occupied.

Sessions distributed over several servers can also cause issues. Since the user session is stored on the server, each subsequent user request has to be made to the same server. This can be handled by synchronizing all the user sessions across the servers by using a centralized database. The drawback to this is that it introduces complexity and increases server overhead.

2.3.2 Stateless authentication

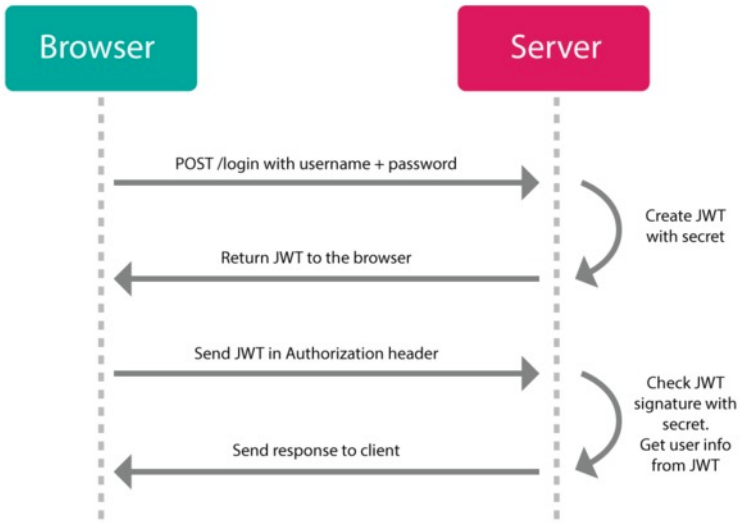


Figure 2.3: Stateless Authentication [2]

In stateless authentication, user sessions are stored on the client-side. The session data is signed with a secret and stored in a token, usually as a JSON Web Token (JWT). The client stores the token, and for every subsequent request, the token is attached. The server can then decode and validate the token by using the secret. If the token is invalid, it has either expired, or the session data has been modified. The process is illustrated in figure 4.15.

Since the sessions are stored on the client-side, there are no scalability issues. On the contrary, since session data is stored in tokens, the token can become large, causing additional load on the network. Also, since the token is encoded, and not encrypted, anyone can decode it to reveal its contents. For these reasons it is important to limit the amount of data stored in the token, and to avoid storing sensitive data in the token.

Stateless authentication is generally considered the more flexible and modern approach. The remaining theory of section 2.3 is therefore in the context of stateless authentication.

2.3.3 JSON Web Token

JSON Web Token (JWT) is an open standard for securely transmitting information as a JSON object. The JWT can be signed to ensure the integrity of the payload and header contained within the token. This means that if an attacker alters the payload, the signature is invalidated.

For authentication purposes, the payload can contain user-specific details like the user's email-address. The JWT can then be decoded to retrieve the email-address. The server can leverage this by using the email-address to query the database for user-specific data.

Structure

Header	`{`
	`  "alg":  "HS256",`
	`  "typ":  "JWT"`
	`}`
		Specifies the signing algorithm and token type. In this case, HMAC-SHA256 and JWT-token.

Payload	`{`
	`  "email":  "user@smartbooking.no",`
	`  "iat":  1422779638`
	`}`
		Contains user data like an email-address or username. Also contains standard fields like the Issued At Time (iat) field.

Signature	`HMAC-SHA256(`
	`  base64UrlEncode(header) + "." +`
	`  base64UrlEncode(payload),`
	`  your-256-bit-secret`
	`  )`
		Verifies the signature.

2.3.4 Access tokens

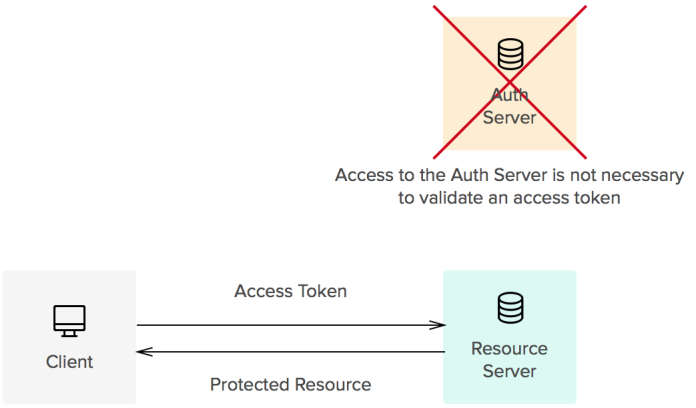


Figure 2.4: Access token flow [3]

Access tokens are used to access protected resources. When the server receives an access token, it decodes and validates the token. Thereafter, it uses the information stored in the token for authorization purposes. Whenever a user successfully authenticates themselves, for example by entering the correct email-address and password, an access token is issued. Access tokens are usually short-lived, meaning they have a short expiration date. Figure 2.4 shows the access token flow.

2.3.5 Refresh tokens

Refresh tokens are used to obtain new access tokens. Whenever an access token expires, the client sends a refresh token to server. If the refresh token is valid, the server issues a new access token. If the refresh token is invalid, the user has to re-authenticate to obtain new access- and refresh tokens. Refresh tokens are long-lived, meaning they have a long expiration date. This way, users do not have to authenticate themselves too often. The flow is illustrated in figure 2.5.

Since refresh tokens are long-lived, they must be stored and transmitted securely to keep them from being abused by attackers. Since access token are short-lived, the security considerations are less strict. Refresh tokens are only transmitted when they have to, i.e. when the access token expires. Access tokens are sent with every request. The implication of this

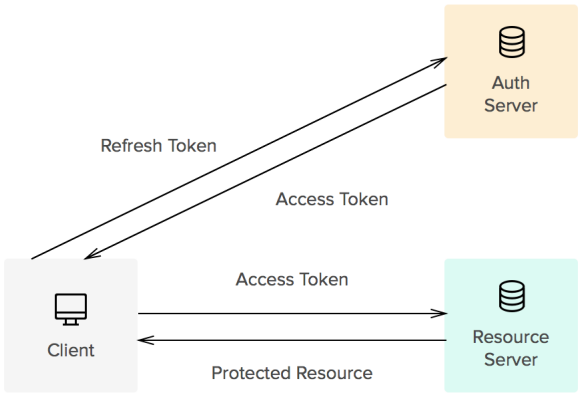


Figure 2.5: Refresh token flow [3]

is that refresh tokens are sent less frequently, and are therefore less likely to be exposed to an attacker.

2.3.6 Verification tokens

When 2FA is enabled, the authentication process consists of several authentication steps. For instance, a user might have to provide a password and an OTP from their email. First, when the user correctly provides the email-address and password, a verification token is issued. Then, when providing the server with the OTP from the email, the verification token is also sent in the same request. This way, the server can first validate the verification token to ensure the correct email-address and password was provided initially. Then, the server can validate the OTP and send an access token if it is valid.

2.3.7 Password reset tokens

Password reset tokens are used for resetting passwords. If a user has forgotten their password, they can enter their email-address or phone number to ask for a password reset link. When the user clicks the password reset link and enters a new password, a password reset token (from the password reset link url) is also sent to the server. The server first validates the password reset token to ensure it is valid, before updating the user's password.

2.3.8 Token vulnerabilities

Tokens are vulnerable to the following attacks:

1. Cross-site scripting (XSS): Allows an attacker to execute JavaScript code in the user's browser.
2. Cross-site forgery request (CSRF): Allows an attacker to make a user perform malicious requests that they do not intend to.

Storing the tokens securely mitigates the risks of XSS and CSRF attacks. Tokens are often stored in local storage or cookies, but they can also be stored in memory.

Token transmission

If the token is stored in a cookie, it is sent *automatically* with every request. If the token is stored in local storage or memory, the token has to be included in the authorization header of the request. The implications of this is described in the following sections.

Local storage

Local storage is susceptible to XSS, but not CSRF. Say the web-application has an XSS-vulnerability and the attacker is able to execute arbitrary JavaScript code. To obtain the token, the attacker could simply dump the contents of local storage with:

```
JSON.stringify(localStorage)
```

Since the contents of local storage is revealed, the token is exposed. The attacker can then make authenticated requests to the server by including the token in the authorization header of the malicious request.

OWASP concludes *"It's recommended not to store any sensitive information in local storage."* [20]. Similarly, Auth0 recommends developers *"Don't store tokens in local storage."* [21].

Memory

Tokens stored in memory, e.g. in a JavaScript variable, are still susceptible to XSS. However, *"Auth0 recommends storing tokens in browser memory as the most secure option."* [21].

The reason for this is *"to avoid a brute force XSS attack from being able to dump your localStorage somewhere else that could allow it to be used for nefarious purposes. Storing in memory just makes it so someone has to go through the extra effort of targeting your app specifically."* [22]

The in-memory method does not provide persistence across page refreshes and browser tabs. This means that the user would be logged out on page refreshes. This can be solved with a combination of refresh and access tokens, but it makes the in-memory method more complicated.

Cookies

Cookies are safe from XSS if the HTTP-only flag is set. That way, JavaScript code can not read the contents of the cookie. However, cookies are still susceptible to CSRF.

Suppose a vulnerable web-application contains a function that lets a user delete their account. This is done by sending a POST request to the /delete end-point. The attacker constructs a malicious web-application that sends a POST-request to the vulnerable web-application's /delete end-point. When a user visits the malicious web-application, the following will happen:

1. The malicious web-application will trigger a POST request to the /delete end-point of the vulnerable web-application.
2. If the user is logged in to the vulnerable web-application, the cookie will be *automatically* sent with the request.
3. The vulnerable web-application will process the malicious request as usual, and delete the user's account.

To prevent this, the SameSite flag can be set on the cookie. This ensures only requests from the host domain are processed, disregarding requests from other domains. Another approach is to use anti-CSRF tokens. These are randomly-generated tokens that are included and sent with every form on the web-application. This guarantees that every request is tied to the authenticated user, as the malicious web-applications do not know the value of the anti-CSRF tokens. However, this adds complexity to the implementation.

2.4 System development

2.4.1 Digital workspace

A digital workspace is a virtual equivalent of a physical workplace allowing users to work from anywhere. The team has been unable to meet physically, which has caused the workflow to be affected. This applies to all phases of the development process. We have therefore tried to implement a form of digital workspace.

In software development, geographical distribution of the team is not necessarily a new phenomenon, but in some phases of the development process it is advantageous to be physically present. This could not be done for this project. Through use of digital tools and technologies the team has adapted to the situation.

The remainder of this chapter reviews technologies and methods that have been used under the development process.

2.4.2 Scrum

Scrum is a development methodology used by individuals and teams in the development of complex products. The theory of scrum is deliberately incomplete and defines only what is considered the core elements required to implement the scrum workflow. Other processes, techniques and methods can be implemented in addition to scrum to accommodate different situations.

Scrum generates an environment where: [23]

1. A product owner orders the work for a complex problem into a product backlog.
2. The scrum team turns a portion of the work into an increment of value during a sprint.
3. The scrum team and stakeholders inspect the results and adjusts accordingly.
4. Repeat.

Scrum consists of combining these four events into sprints. The sprints are an essential aspect in the scrum methodology. With this development approach an attempt is made to optimize predictability and control risk. This works due to the three pillars of scrum: Transparency, inspection, and adaptation.

Transparency means that the process and work must be easily accessible to both the contractor and the client. This transparency enables inspection. Scrum artifacts and the progression towards the defined goals must be inspected often and carefully. This is to find potential errors, omissions and problems. These inspections then make it possible to adapt. If any of the work process moves beyond the accepted limits, or if the result is not acceptable, it must be adjusted. These adjustments should take place as soon as possible to minimize further deviations.

The scrum artifacts represent value and work. In line with the three pillars of scrum, they are designed with transparency in mind to optimize the exchange of critical information about the project.

- For the product backlog it is the product goal.
- For the sprint backlog it is the sprint goal.
- For the increment it is the definition of done.

2.4.3 Collaborative writing

Documentation is a crucial part of any software development project. The process of collaborative writing can be divided into three: Planning, drafting and revising. The planning phase includes what happens before the actual writing starts. This is an important phase since it is critical that everyone involved agrees on core aspects of the project.

Creating a draft as a team involves a lot of communication and can take more time than with individual writing. This, in turn, often creates interesting and instructive discussions. It can also be reassuring if you write about complex topics. In the same way as drafting, the review of the project before submission can create interesting conversations. Perhaps even more detailed then before as the project is now nearing its end and the team has acquired more knowledge during the work.

A common mistake with collaborative writing is that the team immediately starts distributing the work. This leads to more work later in the project when the different sections are to be assembled. [24] The sections must have a logical flow without any repetition. In addition, differences in writing style must be removed, and inconsistent text must be merged. Good preparation can save a lot of work towards the end of the project.

It is often tempting to leave all merging, editing, and revising to one person to ensure a consistent text. The danger with this is that the person may not have a full understanding of every concept, phrase or word used by the co-authors. This can lead to errors in the final text. Furthermore, editing takes a long time. It is easy to underestimate the amount of time and effort required. When the deadline approaches and one editor is under pressure, it is easier to make mistakes or take shortcuts. [24]

2.4.4 Collaborative software development

Software development requires good communication. Awareness of individual and group activities is critical for a successful collaboration. When a team is geographically distributed without physical contact, it creates a social- and information barriers. It becomes more challenging to collaborate without the passive information the team acquires by physically working together. [25]

The initial planning phase is a creative phase where the framework of the project is established. Ideas are suggested through brainstorming sessions. There is a lot of information in a short time span, and it is unlikely that everyone leaves with the same impressions.

This is problematic because it is critical that the foundation for the development is established. If everyone understands the structure, it opens up a opportunity for members to work more individually. The team members can acquire segments and develop these with regards to the agreed structure.

The development phase is often the most productive phase. It still depends on good communication, but not as much as in the planning phase. Each member is free to take independent creative solutions within their segment of code. It is at the end of an increment that a much closer collaboration starts up again to connect the various "isolated" parts that have been developed.

2.4.5 Version control

Version control is a technology to keep track of changes made to source code. Version control allows for long-term history that shows all the changes that have been made. These changelogs also include information about the author, date and a note about the changes.

Version control also provides the ability to branch and merge code. This means that several

developers can work simultaneously, but in isolation with the same code. One can look at each branch as an information flow. These flows of information can be merged so that the isolated versions of the code can benefit from each others contributions to the code.

2.4.6 CI / CD

Continuous Integration (CI) is a technique that can be used during development projects to ensure that the code uploaded to the repository is working. When a change is made and uploaded to the repository, the code will be automatically tested. If a test fails, the developers will be notified. This way, bugs can be discovered and fixed immediately.

CD is an abbreviation for both "Continuous Delivery" and "Continuous Deployment". These are often used interchangeably, but there is a clear difference. Delivery means that the code is uploaded to a remote repository if all the tests are successful. This means that the code has a working version that is ready to be used in some form if desired. Deployment is a technique where the new code is automatically put into production. This requires that all involved developers have discussed what new functionality is expected.

Once the team has developed an MVP version of the application, the workflow can be streamlined using CI / CD. The team can create tests that everyone works towards and agrees upon. CI not only ensures that the code works, but also ensures that the team works with the same project structure during development.

2.4.7 Impact of COVID-19 in software development

The COVID-19 pandemic is relatively new at the time of writing. There is not much research regarding the pandemic's impact on software development specifically. The information here is therefore largely based on the research from *"A Deep Dive on the Impact of COVID-19 in Software Development"*. [26]

The survey shows that most developers do not feel that the pandemic has had a negative impact on the quantity or quality of produced code. Although data mining from popular version control platforms shows that productivity in the form of commits has declined, this survey contradicts this.

It turns out that many are able to maintain a healthy working life. This is affected by social and personal characteristics, but also by the nature of the work. Software development can

be developed remotely due to technologies like version control.

A significant difference between this survey and the situation we find ourselves in, is that the subjects in the survey are professional system developers. They come from a well-established work environment and have a fair amount of experience. We, as students, are less experienced. Our situation is therefore not directly comparable the results of the survey.

3 | Method and choice of technology

This chapter first describes the process of selecting and evaluating authentication methods. Then, the authentication implementation is described in detail. Finally, the system development methods, strategies and tools that have been used throughout the project are presented.

3.1 Authentication method evaluation

The authentication methods were evaluated by reading literature and studies. The following factors were considered:

1. Security. How secure is it?
2. Usability. How user-friendly is it?
3. Complexity. How complex is it to implement?

The result of the authentication method evaluation is presented in chapter 4.

3.2 Authentication implementation

The authentication system for SmartBooking has been implemented as stateless authentication with JWTs. This section describes the implementation details.

3.2.1 Tokens

Token	Issued when	Expiration
Access token	- The user successfully logs in - The system refreshes the access token w/ refresh token	15 minutes
Refresh token	- The user successfully logs in	User: 365 days Customer: 7 days Admin: 7 days
Verification token	- The user provides the correct email-address and password, and 2FA is enabled	5 minutes
Password reset token	- The user requests a password reset link	5 minutes

When the refresh token expires, the user has to re-authenticate. Users only have to re-authenticate once a year, while admins and customers have to re-authenticate once a week. This has been implemented as an additional security mechanism.

Payload

The tokens contain the following payload:

```
{
  "email": "email-address",
  "tokenVersion": 0,
  "iat": 000000000,
  "exp": 000000000
}
```

The `email` field is used as a user identifier to request user-specific data from the database. The `tokenVersion` field is used for invalidating old tokens. Each user in the database has a token version. If the token version from the token payload does not match the token version in the database, the token is invalidated. This way, if a user updates their password, the old access tokens can be revoked by simply incrementing the token version in the database.

The `iat` and `exp` are used for invalidating expired tokens. This is done automatically on the server-side when verifying the token. On the client-side, these fields are used for checking if the access-token is about to expire. If it is, it attempts to refresh the access token before

making the intended request.

Storage

The access token is stored in memory, while refresh token is stored in a cookie. Since the access token is stored in memory, it is not persistent across page refreshes. Whenever the web-application refreshes, the system attempts to acquire a new access token with the refresh token stored in the cookie. This way, the user does not have to log back in on refreshes. The refresh strategy is explained in section 3.2.2.

The refresh token is stored in a cookie and can only be used to request new access tokens. This makes it safe from CSRF attacks, because the attacker can not read the response. The refresh token alone can not access protected resources and is useless in a CSRF attack.

Attack prevention

For preventing XSS, the server uses an XSS data sanitizer, namely the node package xss-clean. In the event that an XSS is possible, the cookie is set to HTTP-only. This means an attacker can not read the refresh token value.

With only the *refresh token* being stored in a cookie, the system is safe from CSRF. Additionally, the SameSite flag is set on the cookie and therefore not included in requests made from other websites. Also, it has its path property set to /refresh. This means the cookie is only sent to the /refresh end-point on the server.

3.2.2 Authentication flow

Login with SFA

The login flow with SFA is as follows:

1. The clients sends an email-address and password.
2. The server queries the database for the user with the corresponding email-address.
3. The server validates the password. This is done by hashing the password provided by the client and comparing it with the hashed password from the database.
4. The server sends an access and refresh token if the hashed passwords match.

The flow is illustrated in figure 3.1.

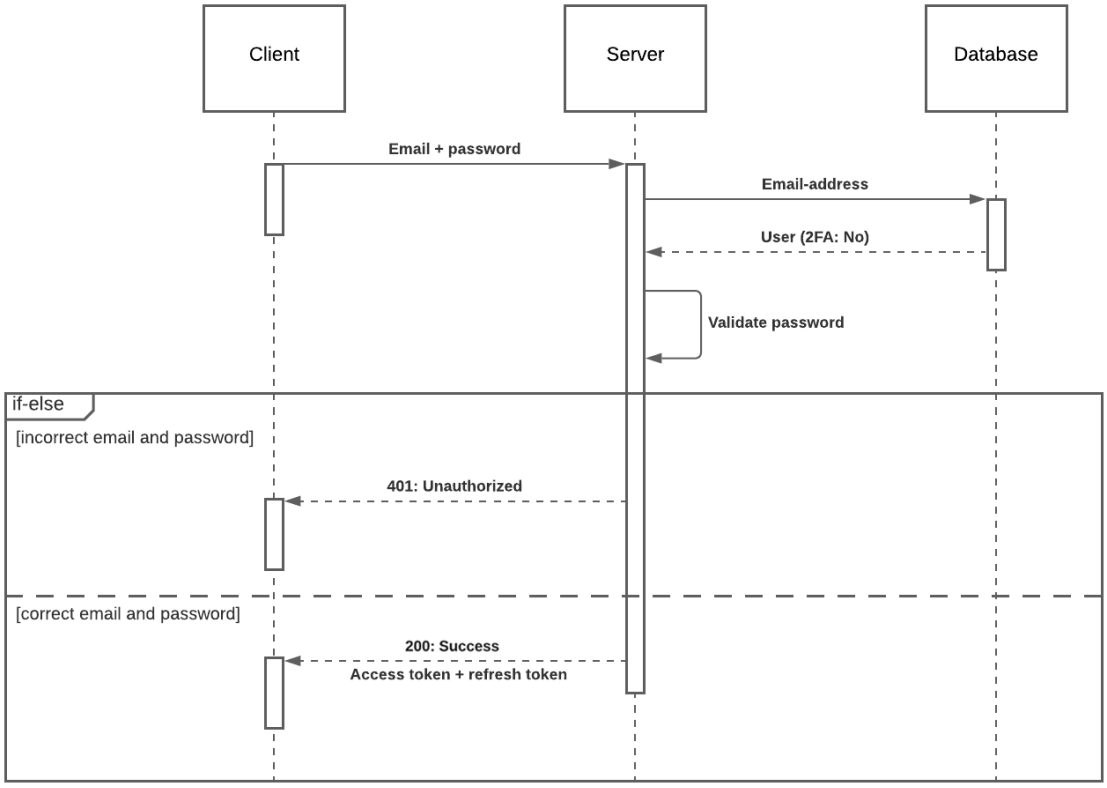


Figure 3.1: Login with SFA

Login with 2FA

The login flow with 2FA is as follows:

1. The clients sends an email-address and password.
2. The server queries the database for the user with the corresponding email-address.
3. The server validates the password. This is done by hashing the password provided by the client and comparing it to the hashed password from the database.
4. The server sends a verification token if the hashed passwords match. If the 2FA-method is email, the server sends an email with the verification code.
5. The client gets the verification code from email or authenticator application. The verification token and code is sent to the server.
6. The server verifies the verification token and code.

7. The server sends an access and refresh token if the verification code and token is valid.

The flow is illustrated in figure 3.2.

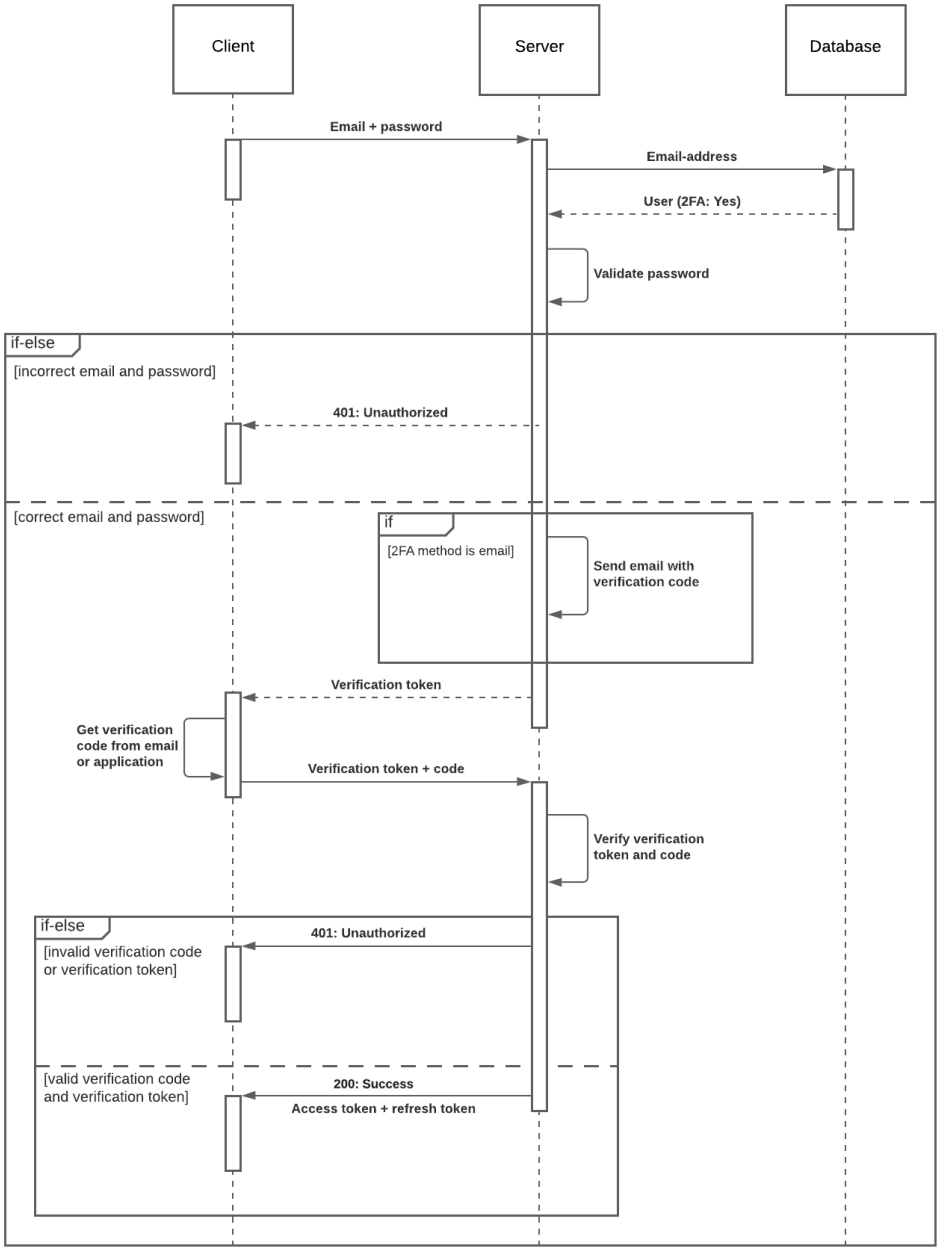


Figure 3.2: Login with 2FA

Refresh token

The refresh flow is as follows:

1. The clients sends the refresh token.
2. The server validates the refresh token.
3. The server sends an access token if the refresh token is valid.

The flow is illustrated in figure 3.3.

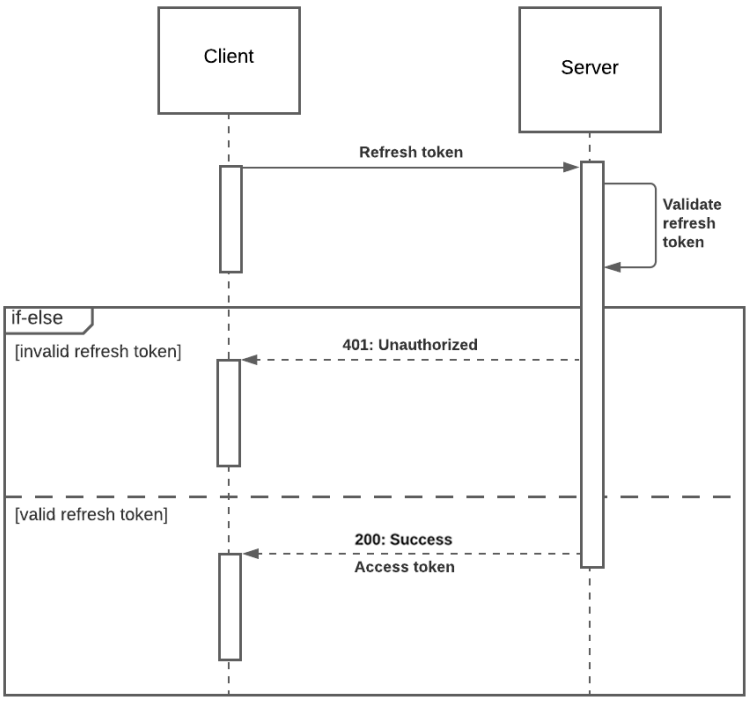


Figure 3.3: Refresh flow

Reset password

The reset password flow is as follows:

1. The clients sends the email-address.
2. The server sends an email with a password reset link.
3. The client gets the password reset link from the email. Then, the client then sends

the new password along with the password reset token. The password reset token is extracted from the password reset link.

4. The server validates the password reset token.
5. The server queries the database to update the users password if the password reset token is valid.
6. The server sends the response to the client.

The flow is illustrated in figure 3.4.

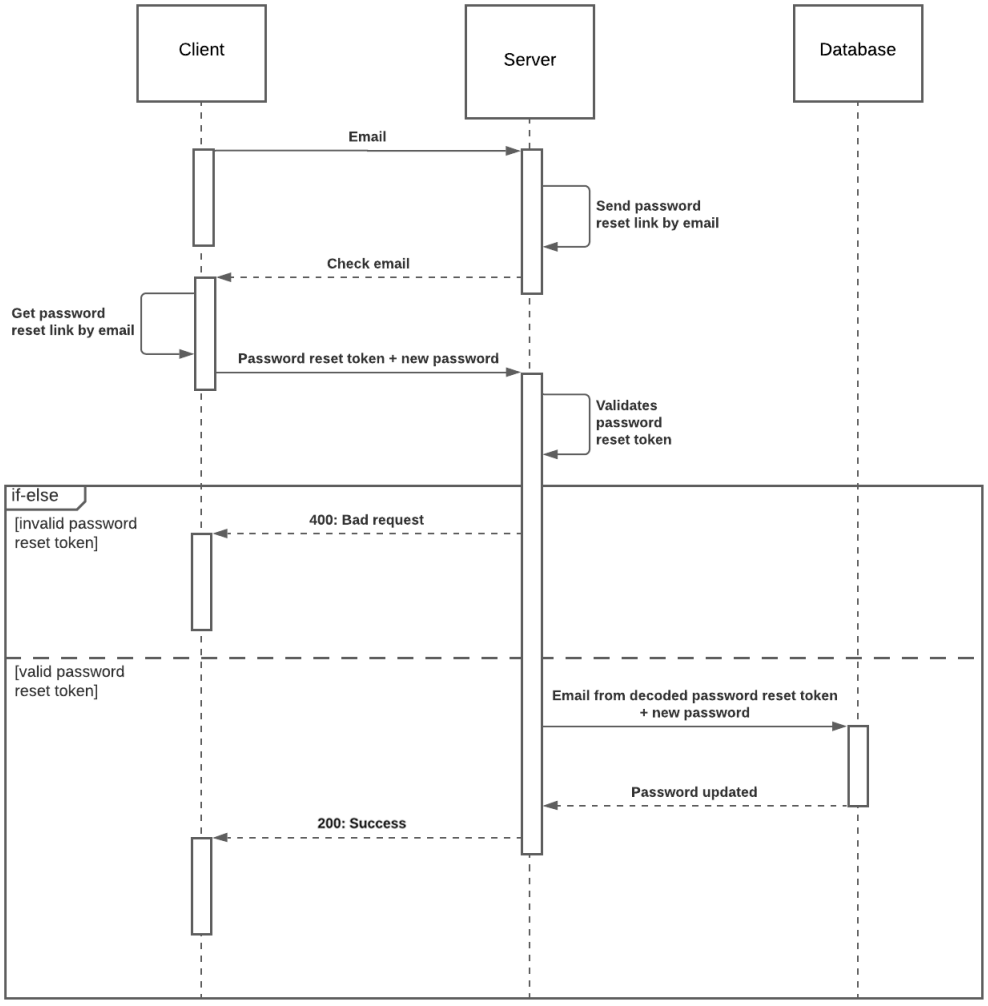


Figure 3.4: Reset password flow

3.3 System development

3.3.1 Agile software development

For this project, we used an Agile Software Development method that we adapted to our needs. Scrum is largely based on The Agile Manifesto and fits very well with the development of this application. The problem description had clear and well-defined requirements and we knew that communication along the way would be critical to ensure that the client would be satisfied with the finished product.

There are some elements of scrum we omitted as the team is relatively small, and we are already well acquainted. We chose to include key concepts for development and to a lesser extent concepts that deal with the team's chemistry. Although it is said in the Scrum Guide that *"changing the core design or ideas of Scrum, leaving out elements, or not following the rules of Scrum, covers up problems and limits the benefits of Scrum, potentially even rendering it useless."* [23], we did not find it necessary to implement the role of a "Scrum Master". The next sections will deal with the concepts we implemented and how these were used.

Sprint

Sprints were implemented. The development was split into two sprints. These lasted approximately one month each. As mentioned above, it was very important for us and the client that we always had the same picture of how the application should look and function.

We therefore chose to host the client side of the application online during the development so that the client could always see the progress and development. By doing this communication became even more continuous.

Sprint planning

At the start of each sprint, a backlog was set up to distribute and plan the various tasks that should be completed before the next sprint. It was important that we did this for each sprint so that we could take into account the feedback we had received.

Sprint review

At the end of a sprint, although the applications progress was available to the client at all times, we organized meetings to be able to more easily argue for changes and choices we had made, and to get clear and constructive feedback.

3.3.2 Interaction tools

One aspect that has had extra weight during this project is digital interaction tools. Communication within the team is critical during any project, but due to the COVID-19 pandemic restrictions throughout the development, we were forced to use digital interaction tools to an even greater extent than before. There are several tools that can be used for this. Some we already had good knowledge of, while others we had to learn to get maximum effect.

Google drive

Google Drive has been a core part of the communication and cooperation within the team. It offers writing in real-time, in both documents and spreadsheets. This has been essential for us, and we therefore chose this as our main platform for all documents.

Discord

Discord is a platform comparable to Microsoft Teams and Zoom. Voice chat, video chat and screen sharing is offered by Discord, and we used it frequently. Discord was chosen over the other alternatives because of how easy it is to organize and adapt to different contexts. Internally in the team, this platform was used daily.

Trello

Trello is a platform that offers us a digital sprint backlog. This has been used diligently during sprints. Similar to Google Documents, Trello offers a co-writing feature. This is convenient for us, because it allows us to have a platform with updated information that is always available.

Overleaf

Overleaf was used to produce the larger and more critical documents in LaTeX. Co-writing is also offered here. We chose Overleaf over Google Docs for this as Overleaf is better for

displaying different types of information such as graphs and figures in a visually pleasing way.

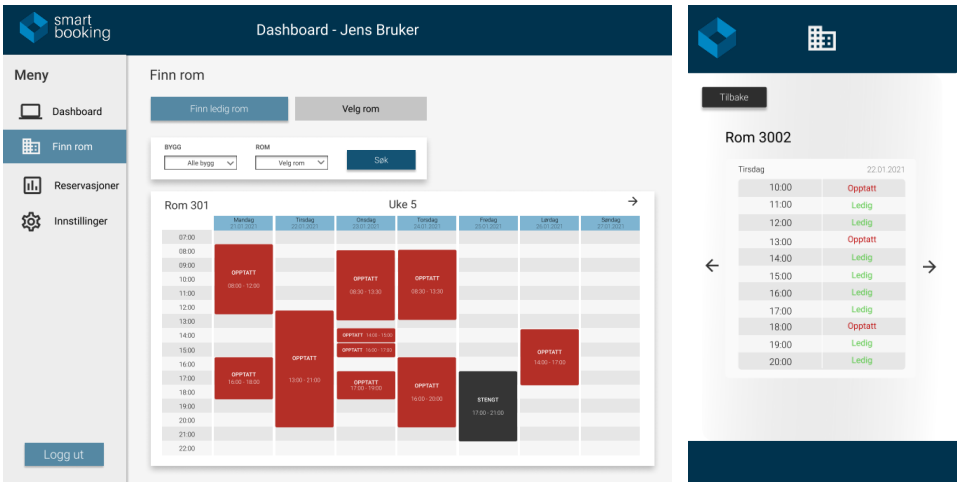
Figma

Figma is a wireframe and drawing tool that was used in the initial phase of the project, as well as to develop user tests. Co-writing is also offered here. Real-time co-writing is even more important in this program than any other. This is because here, it is not words and text that are to be discussed, but images and design.

3.3.3 Prototyping

Wireframes

When we received the task description from the client, we were positively surprised at how well thought out the design of the application was. Of course, there were some elements we thought could be improved, but the core design was there. The team sat down and used Figma to improve the design we had received. Since Figma offers live cooperation, it was easy to suggest changes to each other and create alternatives to design choices that were made. Figure 3.5 shows an example wireframe.



(a) Desktop (b) Mobile

Figure 3.5: Wireframe of a room schedule

After the team had agreed on the design, several alternative wireframes were added so that

we could create a working prototype that would be used in user tests.

3.3.4 Testing

User test

We performed two instances of user testing during this development. The first round of user tests were performed directly in Figma. Since the application had limited functionality and scope, and that it had already been well thought out by the client, the tests were carried out without any big surprises or problems.

Mål	Oppgave	Svar	Notater
Finne ut om det er klart hvordan man kan finne et spesifikt rom	Finn rommet "2LB"	1. Trykk på "Finn rom" 2. Trykk på "Velg rom" 3. Velg "Område" 4. Finn rommet i liste"	
Finne ut om det er klart hvordan man kan reservere et spesifikt rom	Book rommet fra intervallet "20.03.2021-09:00" til "20.03.2021-11:00"	1. Trykk på starttidspunkt i kalender 2. Velg slutttidspunkt 3. Trykk reserver	
Finne ut om det er klart hvordan man kan få foreslått et rom av appen	Finn ledige rom for tidsintervallet "20.03.2021-13:00" til "20.03.2021-14:00"	1. Trykk på "Finn rom" 2. Velg tidsintervallet og evt. velg område	
Finne ut om det er klart hvordan man kan reservere et foreslått rom	Reserver et rom som applikasjonen foreslår	1. Trykk på "+" ikonet 2. Bekreft ved å trykke reserver	
Finne ut om oversikten over reservasjoner er brukervennlig	Slett reservasjonen for rommet "2LB"	1. Trykk på "Reservasjoner" 2. Trykk på "Søppel" ikonet 3. Bekreft ved å trykke "ja"	

Table 3.1: User test schema

This was the only time we used Figma for user testing. It worked well, but as we made changes, the tests could be performed directly in the web-application using our CI/CD solution. As it only involved small changes, we did not consider it time-saving to create new wireframes compared to just changing small parts of the code.

CI/CD

JEST is a framework for unit testing, and it was used to test the API. After an MVP version of the application was developed, a CI/CD solution was implemented for the remainder of the development. This was done using GitHub Actions and JEST. The Git repository would be tested automatically with every commit. This was a great help for the team as it led to everyone being able to work independently. The members of the team were also automatically notified if an error was detected so that it could be corrected immediately.

3.4 Technology choices

3.4.1 Version control

When it came to the choice of version control, the choice fell on Git. Git is the de facto standard for version control and the team considered this an easy choice. GitHub was chosen as the git service. This is because GitHub has additional functionality that makes it simple to connect to other services like Netlify and Heroku. We were also eager to try out GitHub Actions. Actions enables a CI/CD solution for development and we knew this would be useful.

3.4.2 Front-end

Language

For the development of the front-end application, we chose JavaScript. JavaScript is the largest and most widely used language to develop websites. The framework Blazor was discussed as an option, but Blazor is still in an early stage and does not have as broad coverage and support as JavaScript does. As we are in our first years of developing applications, we prioritized having forums to look up any questions we might have, where the chance of someone having been through exactly the same, was largest.

Framework

For the development of the front-end client, we decided to use Vue. The main competitor here was React. After doing research on which framework to use, we discovered that it was a common opinion that Vue was easier to learn and understand, and because other developers may work on this project in the future, this was a deciding factor. Vue is a framework with focus on versatility and performance, and this would help us to develop a dynamic and

stylish application in an efficient way. Nobody in the team had experience with Vue, and therefore everyone attended web-based courses on Udemy [27]. Vuex was chosen to handle application-wide state, and Axios to perform HTTP-requests to the server.

3.4.3 Back-end

Language

JavaScript was also used for the back-end development. This is because when we first had chosen a JavaScript-based framework on the front-end client, it would be much easier to develop the server in the same language. The team also had knowledge of JavaScript from previous projects. As mentioned, JavaScript has broad community support if we needed advice to implement challenging functionality. Another aspect that was important to the client was that we used a language that is relatively popular. This was due to the further development of the product at the end of this project, possibly from a new team.

Runtime engine

Node.js was chosen for the back-end development. As we already had chosen JavaScript, Node.js was a natural choice. Node.js used in conjunction with Express.js is the staple when it comes to servers developed in JavaScript. Node.js also allowed us to use the NPM packet-manager, and we were able to set up a back-end server that could be expanded as the application grew. Other options like Java and Django were considered, but the fact that Node.js allowed us to write in the same language and with very similar syntax as the front-end client, made the choice fall on this runtime engine.

Database

Early in the planning phase, we had to make a choice regarding which database solution we should use. A non-relational database solution in the form of MongoDB was considered. Especially one of the team members had experience with this from private projects. The team, however, ended up with the choice of using a relational database and MySQL. The client requested a SQL solution, and we saw no reason for not using SQL. MySQL can be easily installed and run locally. There are also convenient user interfaces such as PhPMyAdmin that make it efficient to work with such a database.

3.4.4 Deployment

Once the MVP was completed, it was deployed online at all times. This allowed the client to check the application's progress whenever he wanted to. This solution also simplified the process of user testing. Due to the pandemic and restrictions, travelling to our test subjects was not an option, and it would have been impractical for them to download all the software in order to test the application. It was practical to be able to send the users a link to the website, where the application was already deployed.

Netlify

The front-end client of the application was hosted on Netlify. Netlify is a cloud-based hosting solution which is intuitive and easy to work with. Netlify supports continuous deployment, which means that it automatically detects whenever a push to the master branch on Git is done. It then compiles and deploys the updated code. By having a hosting solution like Netlify, we could focus more on the development and not spend much time on manually keeping the deployed project updated.

Heroku

The back-end of the application was hosted on Heroku. Because Netlify only allows server-less applications, we had to choose another platform. Heroku is also a cloud-based platform which supports continuous deployment. By using Heroku as our hosting service for back-end, we could send requests and receive responses from the server anytime, anywhere, without having to run the server on our own machines. It was necessary to host the back-end in order for the front-end of the application to work properly when tested outside of our home network.

4 | Results

This chapter presents the authentication, engineering and administrative results.

4.1 Authentication results

This section first presents the results of the authentication method evaluation. Then, the authentication system that has been implemented in SmartBooking is described.

4.1.1 Authentication method evaluation

The authentication methods have been evaluated by their security, usability and complexity. The evaluation is based on theory and studies from chapter 2.

Password authentication

By following the NIST password guidelines, as outlined in section 2.2.1, it is ensured that users create strong passwords. Also, since there are no password complexity requirements in the password guidelines, the passwords are easier to remember. This makes password authentication both secure and user-friendly. Password authentication does not rely on users receiving information on their email or phone, which also increases usability.

However, password authentication is still susceptible to credential stealing through social engineering strategies. The passwords also need to be stored in the database. This means the password can be leaked, so they need to be salted and hashed. This increases the complexity of implementing passwords, and it poses a potential security threat.

One-time password authentication

OTP authentication is secure if an authenticator application, like Google Authenticator, is used. OTPs transmitted over SMS or email is less secure.

The user only needs to enter a code from their smart phone or email to authenticate themselves. This makes the method user-friendly. However, usability is decreased if an authenticator application is used, because the user has to be in possession of their smart phone.

A token secret needs to be stored in the database to be able to verify and generate OTPs. This is increases the complexity of implementation and introduces a sensitive data that may be leaked.

Push notification authentication

Push notification authentication is secure if a push notification application, like Authy, is used. Like OTPs, push notification transmitted over SMS and email are vulnerable.

Push notifications are user-friendly, as the user only needs to click a button to either accept or deny the login attempt. However, this poses a security threat, since people have a tendency to automatically accept any login attempts. Also, if the push notifications are received with the application, the user has to be in possession of their smart phone.

Push notifications can be implemented with tokens and does not require storing any sensitive data in the database. This makes it easier to implement than OTPs and password authentication.

Biometric authentication

Biometric authentication is secure and user-friendly. However, it requires the user to be in possession of a smart phone with biometric authentication sensors. Furthermore, the implementation is complex and would have to be handled by a third-party service.

Evaluation result

Password authentication with OTPs received over email or a separate authenticator application has been implemented in SmartBooking. A discussion of this result can be found in chapter 6.

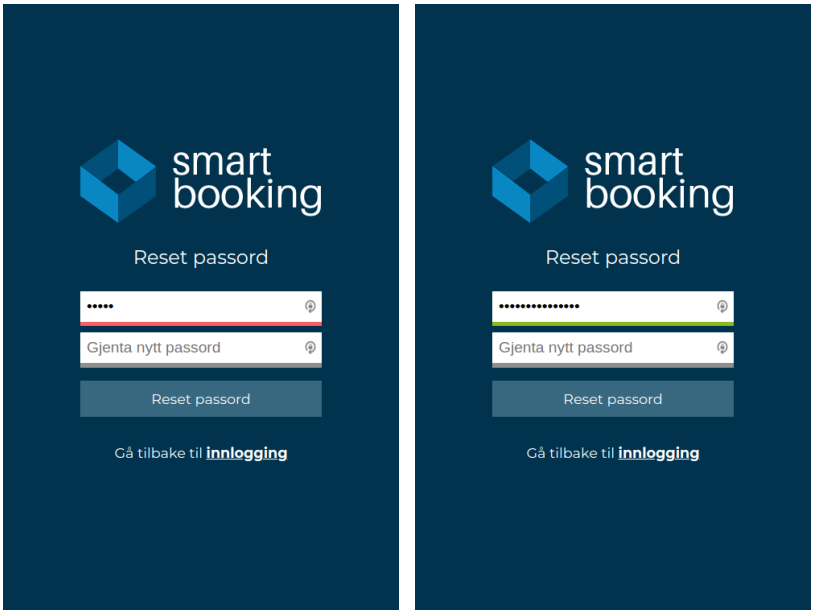
4.1.2 Authentication implementation

This subsection describes the authentication system that has been implemented in Smart-Booking.

Password authentication

The passwords need to be minimum eight characters long for student accounts, and twelve characters long for administrators and customers. This way, it is harder to crack administrator and customer passwords, as they require more security.

To encourage users to pick a strong password, a password strength meter has been implemented. The strength meter uses the zxcvbn algorithm [28] to calculate the password strength.



(a) Weak password (b) Strong password

Figure 4.1: Password strength meter

The 100.000 most common passwords [29] has been filtered into two lists:

- Top 1000 common passwords of minimum 8 characters
- Top 1000 common passwords of minimum 12 characters

When a user updates their password, the appropriate list is checked to see if it is a common password. If a user enters a common password, they are prompted to pick another one, as depicted in figure 4.2.

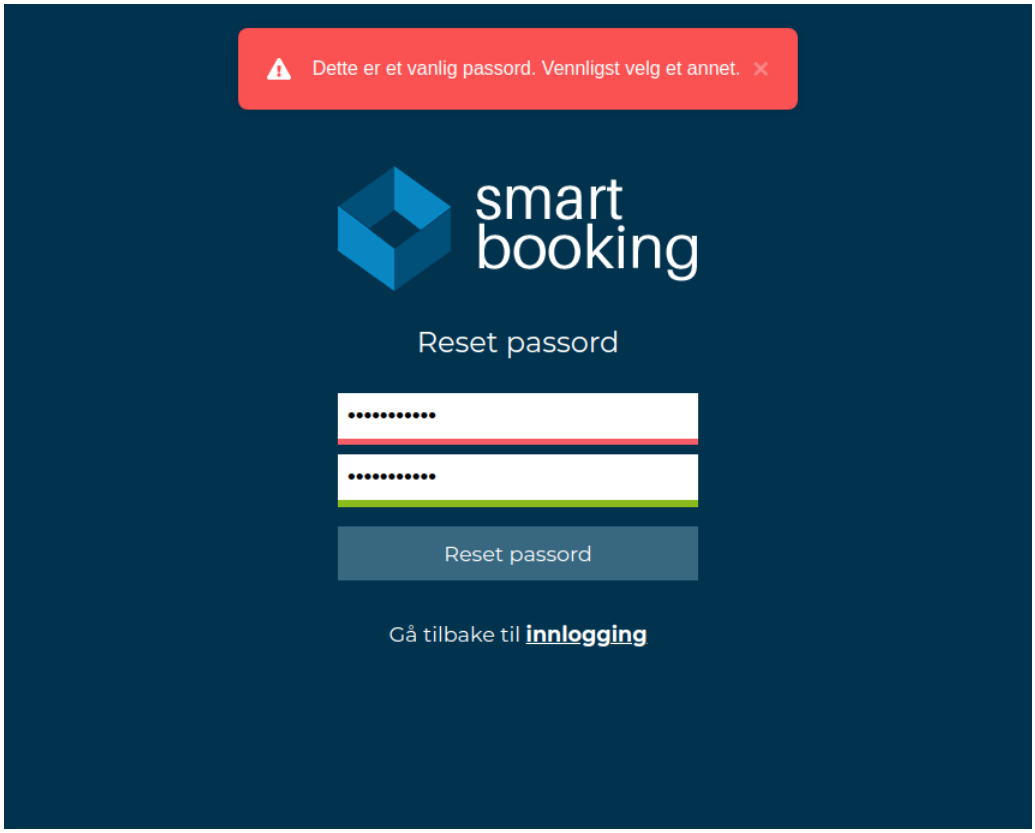


Figure 4.2: Common password prompt

One-time password authentication

The OTPs are time-based. When they are received by email, they are valid for 5 minutes. When they are generated by Google Authenticator, they are valid for 30 seconds, before a new one is generated. OTPs are received by email or generated in an authenticator application like Google Authenticator. See figure 4.3.

(a) Email

(b) Google Authenticator

Figure 4.3: Receiving OTPs

When the user has provided the correct email-address and password, they are redirected to the validation page. Here, they are prompted to enter the OTP, as shown in figure 4.4.

Figure 4.4: Entering a OTP

4.2 Engineering results

At the end of the project, the final product was delivered. This product contained a fully functional web application with associated server and database setup.

4.2.1 User stories

Here, some examples of user stories is presented. For further details, see the Software Requirements Specification.

As an administrator
I want to add a new organization
So that new organizations can start using the system

As an administrator
I want to add new customers connected to an organization
So that they can start using the system

As a customer
I want to add new rooms
So that my users can use all the available places

As a customer
I want to maintain a booking policy that sets rules for reservations
So that users can't exploit the reservation system.

As a customer
I want to set opening times for buildings/places
So that users can't book rooms in places that are not open.

As a user
I want to book a room
So that I can be sure that I will have a place to sit

As a user
I want an overview of my reservations
So that I can see when and where I have booked a room

As a user
I want to scan a QR-code on the door of a room
So that I can see its calendar and book it when available

As an administrator/customer/user
I want to enable and use two-factor authentication
So that I can keep my account secure

4.2.2 Application

The application contains three user groups. These have different access rights, and functionality available to them. Functionality for the user groups is as follows:

Administrator functionality

- Administrator can add new administrators to the system
- Administrators can add new organization to the system
- Administrators can create users with the role of customer and connect them to an existing organization
- Administrators can change and update customer information

Customer functionality

- Customers can add users to their organization. Either manually, or by importing from CSV files
- Customers can edit the information of users connected to their organization
- Customers can delete users connected to their organization
- Customers can register room either manually or by importing from CSV files
- Customers can edit existing rooms
- Customers can register and edit their organization's booking policy

- Customers can edit opening hours for areas.
- Customers has the opportunity to review all upcoming and previous reservations in the "reservations" tab

User functionality

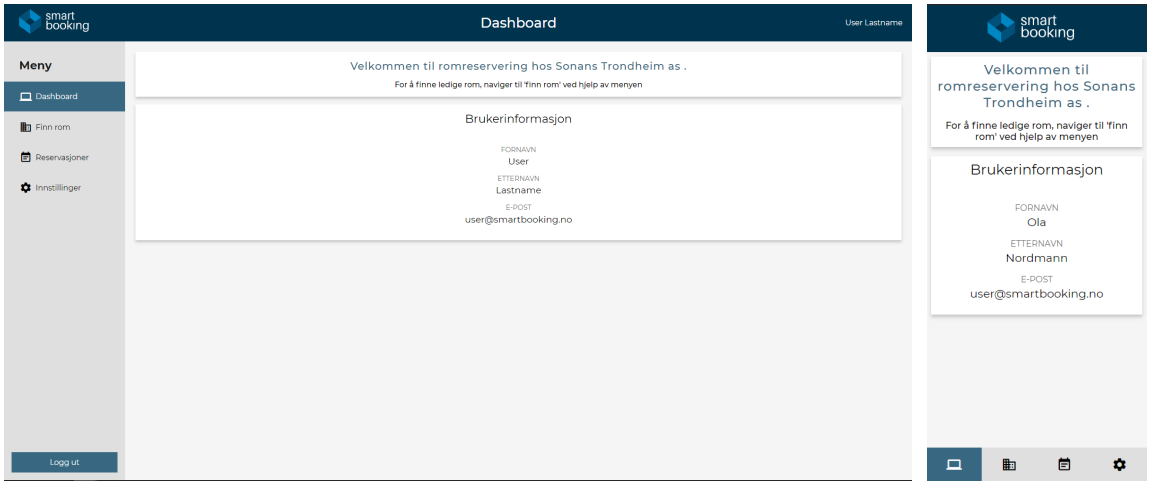
- Users can book available rooms. Either by choosing a specific room, or searching for an available room given date and time.
- Users can see an overview of their reservations in the "reservations" tab
- Users can delete their reservations in the "reservations" tab
- Users can use their mobile phone camera to scan a QR code outside a room. The user will then automatically be redirected to the room's calendar
  - Users can change their password by requesting a password reset link. This will be delivered by email
  - Users can change their password by going to the settings when logged in. He can then type in the old and the new password to change it
- All users have the ability to turn on two-factor authentication in the settings tab

Design

The design is simple and is relatively similar for all three user groups. Here we will briefly show relevant functionalities for the three different groups with screenshots from the application.

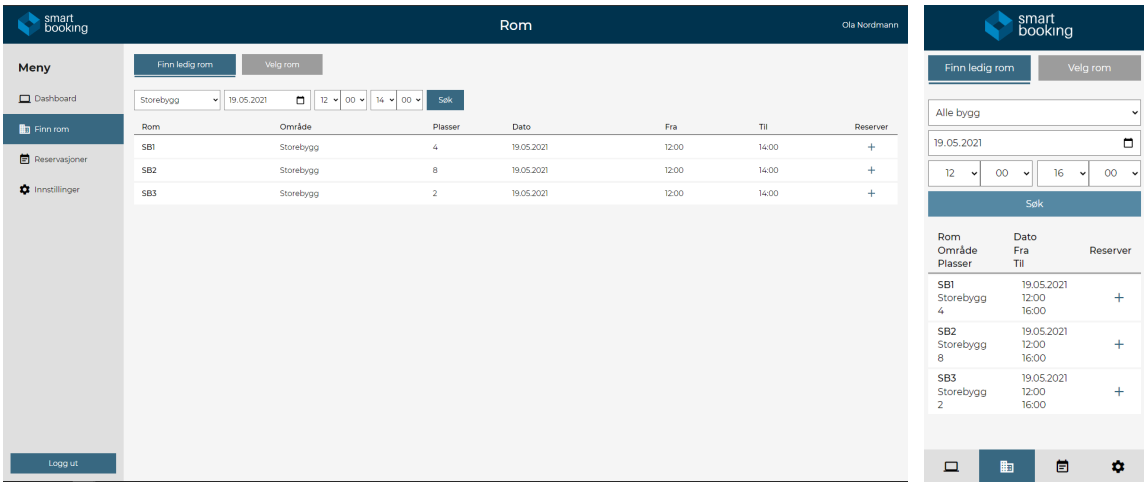
- **User**

  The figures that follow depict the room booking flow.



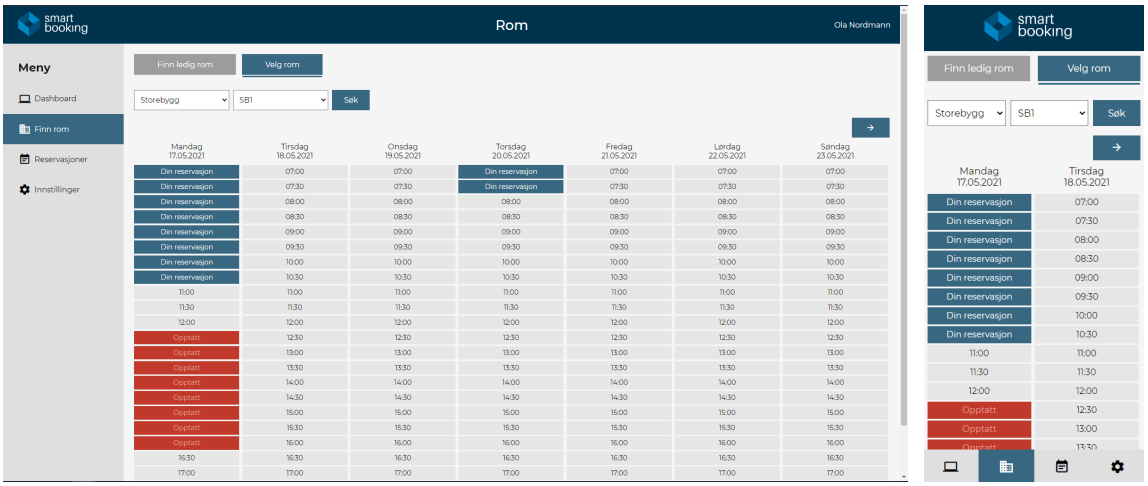
(a) Desktop (b) Mobile

Figure 4.5: Dashboard



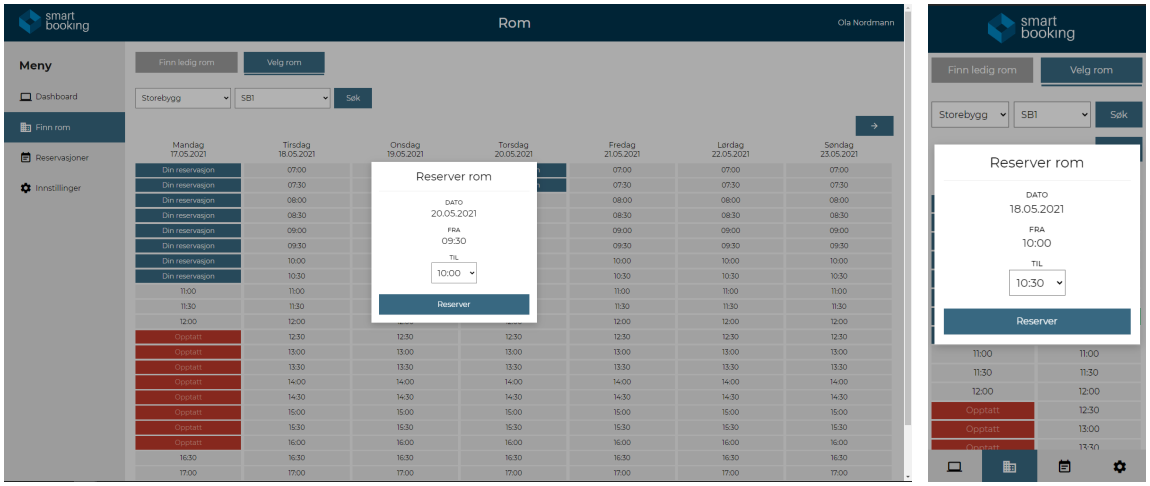
(a) Desktop (b) Mobile

Figure 4.6: Find room page



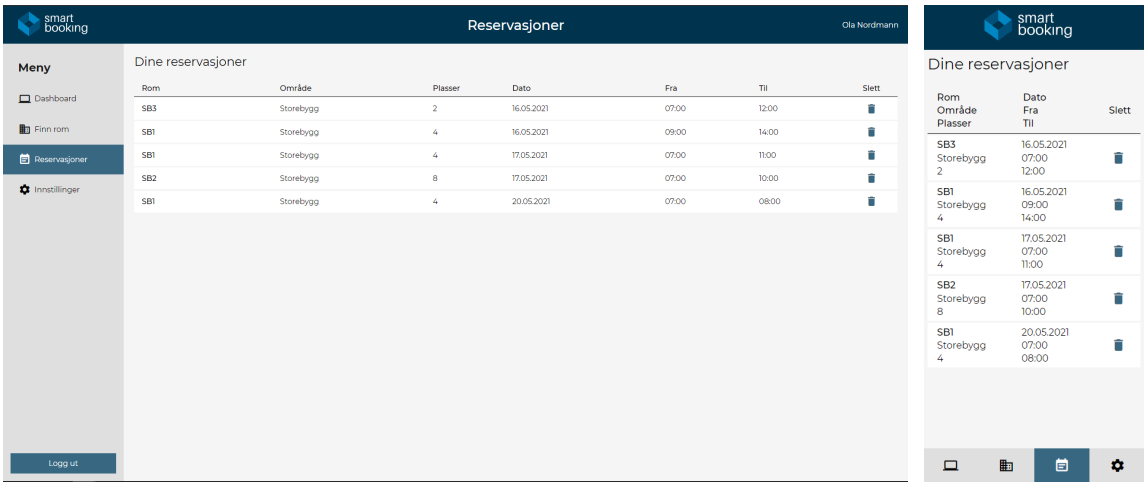
(a) Desktop (b) Mobile

Figure 4.7: Room schedule page



(a) Desktop (b) Mobile

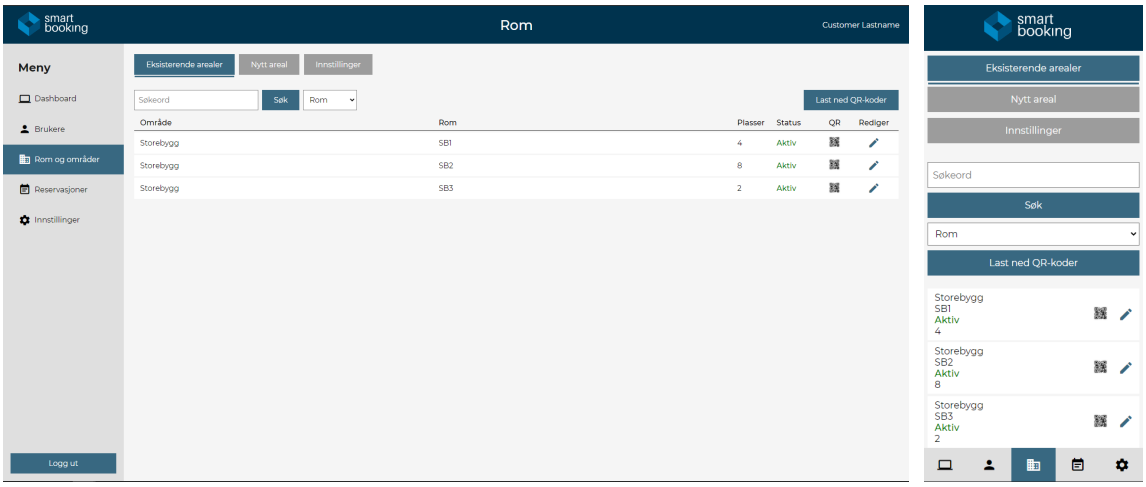
Figure 4.8: User confirm reservation page



(a) Desktop (b) Mobile

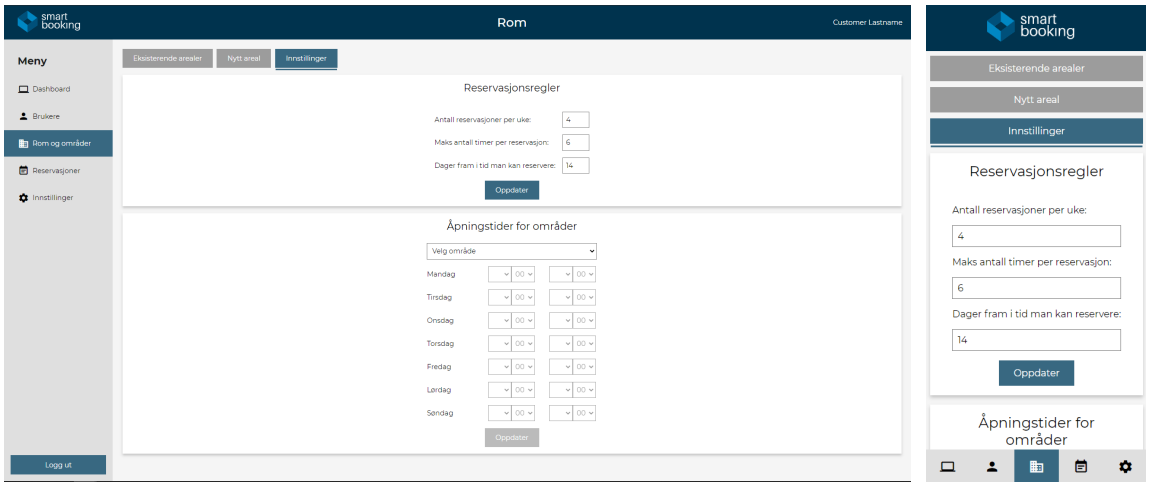
Figure 4.9: User reservation overview page

- **Customer**
  For a customer, it is important to have an overview of their rooms and areas. It must be easy to set and change their reservation guidelines.



(a) Desktop (b) Mobile

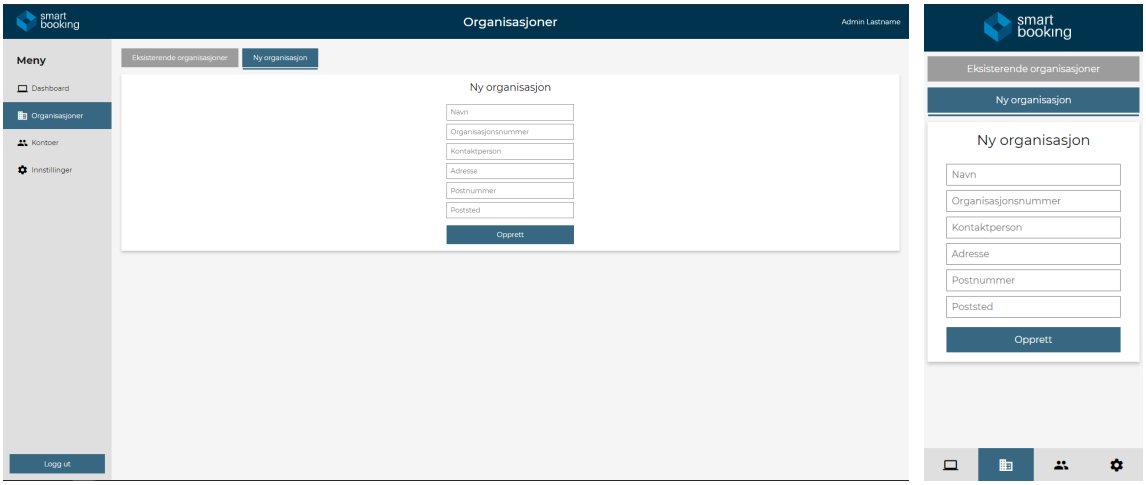
Figure 4.10: Customer area overview



(a) Desktop (b) Mobile

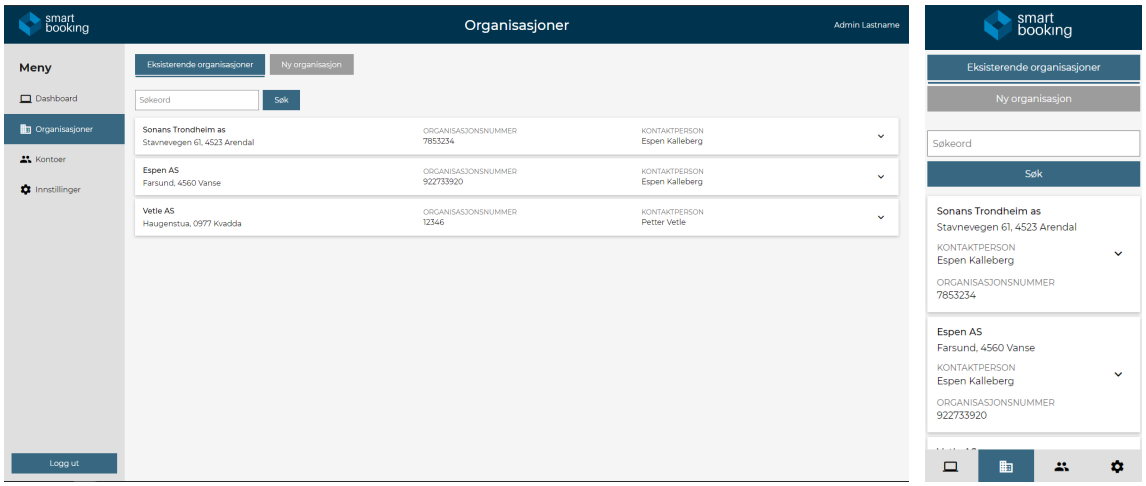
Figure 4.11: Customer policy settings

- **Administrator**
  An administrator's main task is to manage organizations that will use the system.



(a) Desktop (b) Mobile

Figure 4.12: Administrator register organization



(a) Desktop (b) Mobile

Figure 4.13: Administrator organization overview

4.2.3 Database architecture

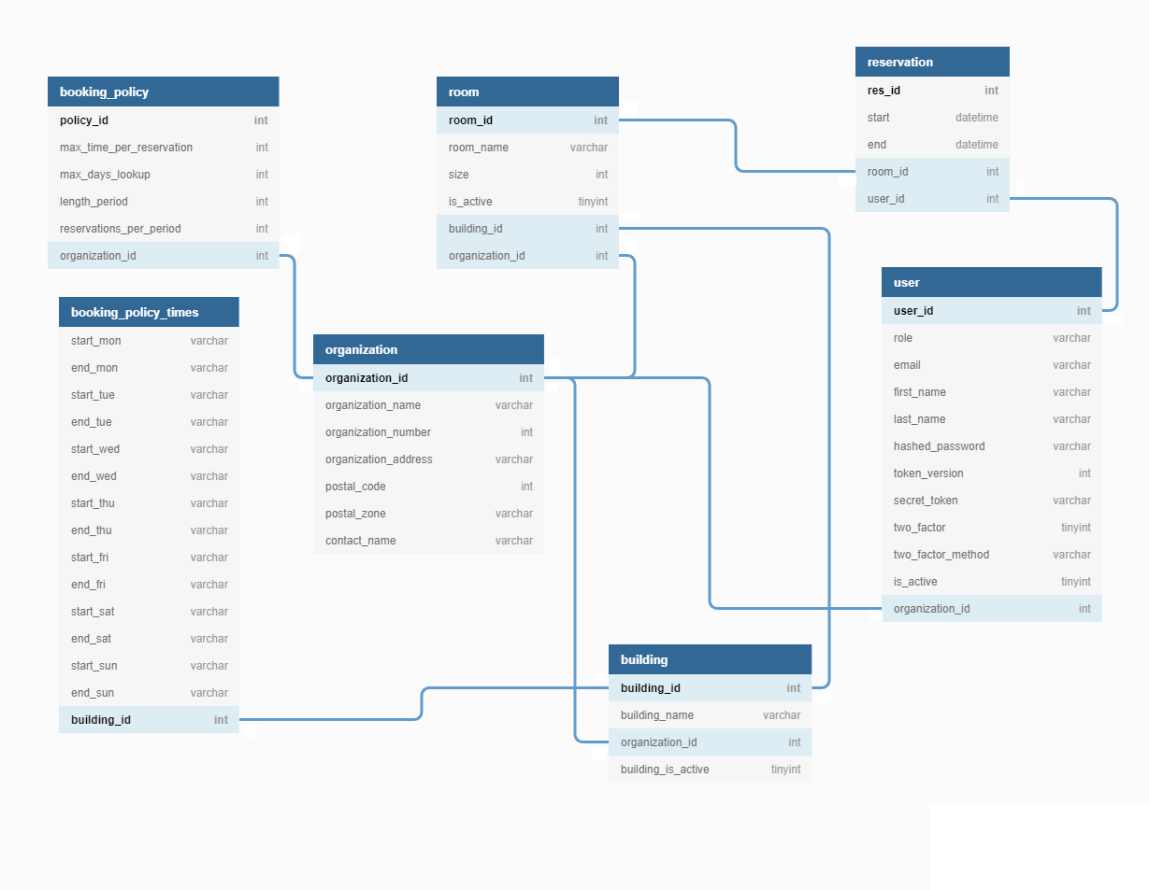


Figure 4.14: Database schema

4.3 Administrative results

4.3.1 Progress plan

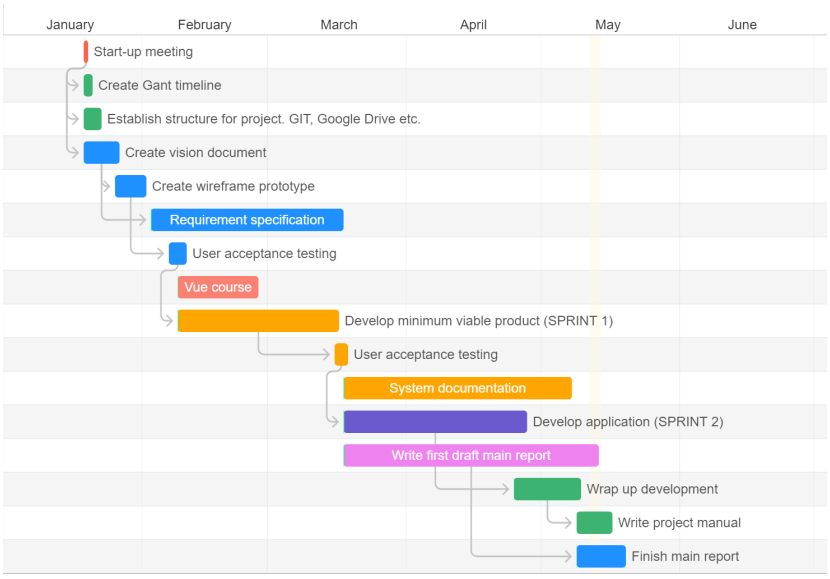


Figure 4.15: Gantt chart showing our expected timeline

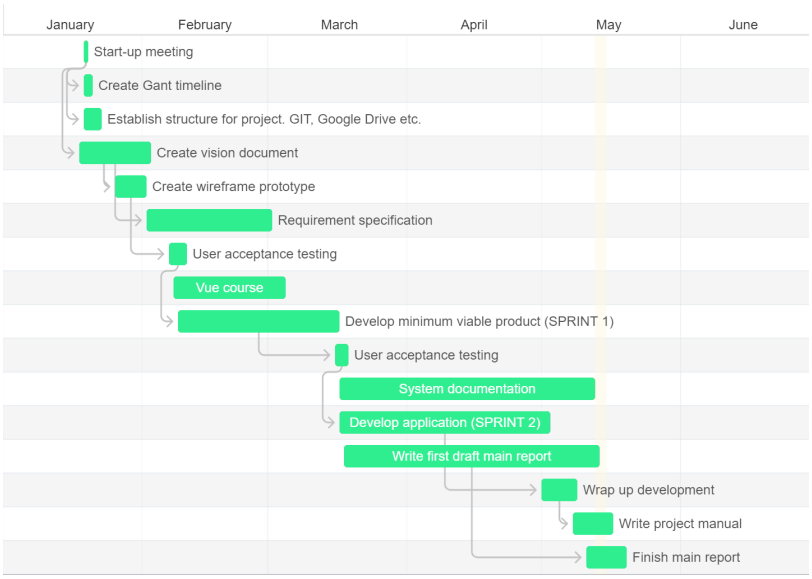


Figure 4.16: Gantt chart showing the actual timeline

Sprints

Scrum boards

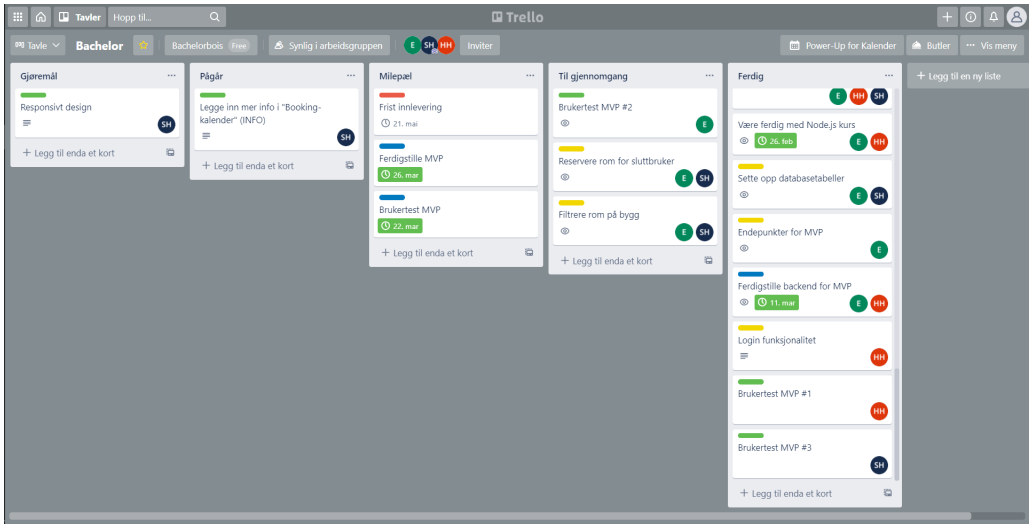


Figure 4.17: Scrum board at the end of sprint 1

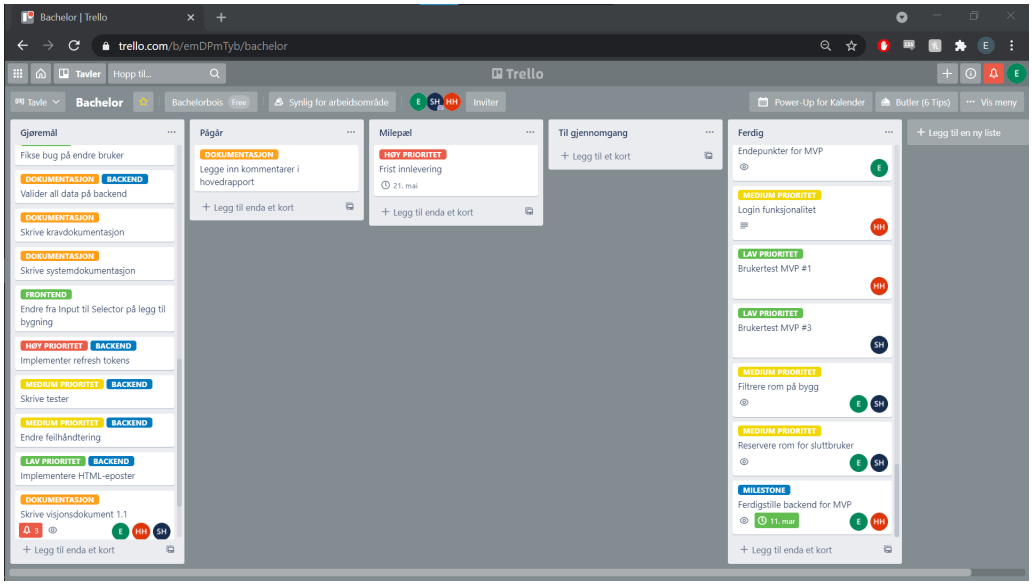


Figure 4.18: Scrum board at start of sprint 2

4.3.2 Working hours

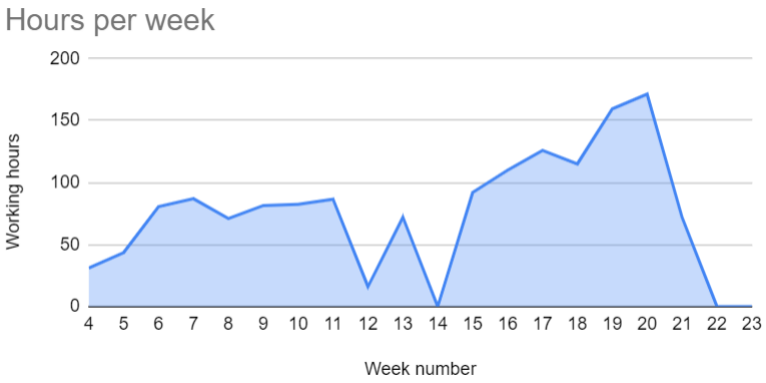


Figure 4.19: Team hours per week

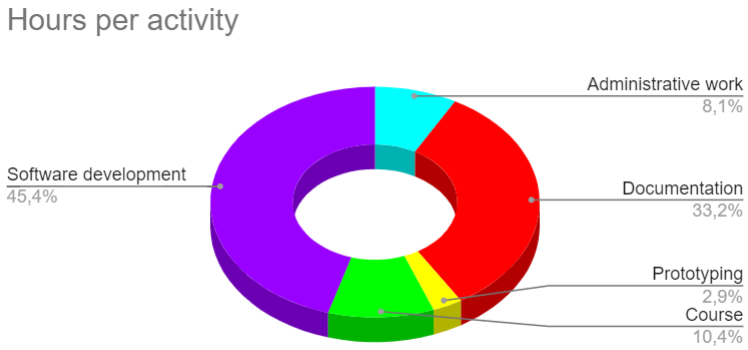


Figure 4.20: The team's distribution of time on different activities

Justification of time use

As the spring semester has an intensive period with courses that last until march and the rest of the time is intended for the bachelor thesis, our graph matches the expectations. Regarding the distribution of time per activity, it may seem that prototyping has been given little time. This is true but it is due to a very detailed requirements specification from the client. This meant that we did not have to develop the whole design from scratch.

5 | Discussion

This chapter discusses the implementation of the authentication system. Then, it describes how we have experienced remote collaboration. Finally, the strengths and weaknesses of the system is accessed.

5.1 Authentication

Passwords have been implemented as the primary authentication method. This is because students should be able to log in to the system without the hassle of checking their email or an application on their phone. This requires the authentication method to be knowledge-based, i.e. something the user knows. Any of the other proposed methods would require the user to check additional resources.

Passwords are not the most secure authentication method, but the risks with user accounts is also low. If an attacker gains control over a user account, the worst he can do is create and delete reservations. The victim can regain control by resetting the password through their email-address. Then, they can delete the undesired reservations.

In addition to passwords, magic links, a type of push notifications, could have been implemented. The user would then have the option of either entering their password, or receiving a magic login link by email. This is not as secure as passwords, but in some cases more user-friendly, as the user does not have to remember their password.

For administrator and customer accounts, the security risks were higher. An attacker with access to these types of accounts could delete users, or leak sensitive information. More security was needed, and that is why OTPs have been implemented as an additional authentication mechanism. It is used in addition to password authentication, either with

Google Authenticator (2FA), or over email (2SV).

When 2FA is enabled, the user has to be in possession of their phone to retrieve the OTP. This is not decreases the usability, and is why OTPs can be sent over email (2SV) instead. This is less secure, but more user-friendly, and certainly more secure than password authentication alone.

Due to security considerations, OTPs were implemented instead of push notifications. If an administrator receives a push notification for a login attempt on their email, the might unintentionally accept it, as studies have shown.2

The access and refresh token strategy makes the tokens safe from XSS and CSRF attacks. To further improve security, refresh token rotation and automatic reuse detection could be implemented. See section 6.2.

The refresh tokens expire after one year for students, and one week for customers and administrators. This way, customers and administrators have to re-authenticate once a week, adding an additional layer of security.

The authentication system is implemented as stateless authentication. Since the session data is saved on the client the system can accommodate a large amount of users and scales well as the user base grows.

5.2 System Development

5.2.1 Remote Collaboration evaluation

Although the project took place during a pandemic, the project was largely carried out in line with agile development. The core elements from scrum were also used without any special adaptations to the situation. It worked well, but there are some elements that are natural to implement digitally.

We discovered during the development that the stand-up meetings were not as beneficial when done digitally compared to physically. The purpose of stand-up meetings is a short and effective update for everyone in the team. In a physical work environment, it is easier to get an overview of what is being worked on, what needs to be worked on, and what has been done. When a member explains what is being worked on, you can easily relate the task

to a note on the board. In a digital environment, it is easier for these meetings to slip into irrelevant conversations because you sit and watch a screen as you would otherwise. Then you lose the clear signal that the working day has started, which you would otherwise get with physical attendance.

Scrum boards also looses some of its benefits with a digital environment. When you do not have a physical reminder to update the board, it can be forgotten easily. This has a snowballing effect, as there will be extra work when updating the next day. If forgotten for more than a couple of days, problems arise. The board may have become outdated to such an extent that it is perceived as extra work to set up the board correctly again. It is then perceived as a chore rather than a useful tool.

An annoyance during the development was how limited pair programming and generally helping each other was. It required more time to set up the process digitally than it would be to just take the laptop to another team member. This was handled by either screen sharing and voice chatting, or using the Live Share extension in Visual Studio Code.

When sharing a screen, you spend more time presenting the code. Often you have to zoom in to show the code clearly on the screen, and then zoom back out when you are going to code yourself. People have different settings and limitations on their own computers that may cause collisions when they are supposed to work together. If the person that is viewing the shared screen is coming with suggestions, it can be difficult to explain using only words. Physically, one would correct the problem by writing the code and explaining along the way. If there is a major change, the code change can then be sent in chat and pasted in. If it does not work as intended, the process is repeated. These small additions of wasted time build up so that small problems takes a long time fix.

The Live Share option in Visual Studio Code is good when writing code together, but it a hassle to set up, and takes more time than it should. When helping with a problem, it is not easy to predict whether the problem is significant enough to go through that hassle to enable Live Share. Therefore, Live Share was only used when it was known beforehand that the problem was large enough to warrant the extra time it took to set up.

What worked

- **Online and available at any time**
  Thanks to Discord's server functionality, we were always available to each other throughout the entire working day. The server was divided into different voice channels, so that we could easily move between to talk to those who were needed.

- **Distribution of tasks**
  Due to the communication on Discord, we were able to distribute tasks easily, even though the scrum board was not utilized to the greatest extent.

- **CI/CD**
  The CI/CD solution has many advantages for collaborative projects. We felt we were able to take advantage of this to a great extent in this project. We used CI during the development, and it worked very well. CD was implemented so that the client could see the applications progression continuously. It was also useful in terms of user testing.

What did not work

- **Stand-up Meetings**
  As mentioned above, stand-up meetings were not used effectively. This is probably due to lack the formality where you actually stand up in front of the others with the sole purpose of communicating what to do that day. Maybe it would have worked better if screen sharing or webcams had been used during the meetings. In our case, these "meetings" eventually died out.

- **Scrum board**
  Early in the project, we selected Trello to create scrum boards. The idea was good but it became cumbersome to update the scrum board on a third party website. It was often forgotten and ended up with us having to make adjustments to how we used Trello.

- **Time estimates**
  Throughout the project, time estimation has not always been accurate. When setting deadlines for when work is to be completed, overly optimistic deadlines have often been set. This lead to postponement of tasks, and a feeling of being behind on schedule.

- **Consistent database**
  As we only had one deployed database, once in a while, changes to database structure were done by a member of the team that made the application break for other versions of the application. However, this was not a very big problem as it only happened a couple of times, and when it did, the team was quick to sort it out.

Adjustments

- **Stricter start time**
  Since the stand-up meetings stopped after a while, we had to make stricter demands to each other about when we should actually start working.

- **More general scrum tasks**
  Since we failed to keep Trello updated, we made a generalization of the tasks on the scrum board. For example, we took three detailed goals of the design type and merged them into a more general task. By doing this, Trello required less maintenance, and it was easier to keep it up to date.

Improvements

- **Better review of code before pushing to dev/master**
  In the future we would want to establish stricter routines for how to update the main branches of the remote repositories. This way, we would be certain that everyone agrees on the structure of the code, and that it looks and works to an acceptable extent. By using GitHub's Pull Request feature, we could comment and give constructive feedback to the team members if there were improvements to be made.

- **Person responsible for scrum board**
  A person should be responsible for updating the scrum board. This person does not necessarily have to update the board themselves, but to remind the other team members. The reason why Trello did not work for us was not laziness, but rather forgetfulness. So if a person was responsible and could get into the habit of checking Trello on a regular basis and remind others, it would likely help.

- **Overall status of the project**
  The combination of inaccurate time estimates and lack of scrum board updates meant that we sometimes lost track of which tasks we should prioritize for the product as a

whole. Towards the end of the project, we had implemented all the desired functionality. If there had been an earlier deadline, and we had to prioritize tasks, this would have required some work to find out what could actually have been implemented with the right priorities.

5.2.2 Strengths and weaknesses of our product

This product is delivered as a complete and implementation-ready platform to the client. Here we will look at the product in its entirety and clarify what we consider to be strengths and weaknesses.

Strengths

- **Security**
  With this assignment, we have used the opportunity to study modern authentication methods. This is reflected in the application, and we consider the product to be at least as secure as today's similar applications.

- **Scalability**
  The application is designed in such a way that administrators can easily add new organizations to the system. Customer accounts can be added to the organizations. Customers can then add users to their organization, who can register with a password. No prior work is required before adding new user accounts to the system.

- **User friendly**
  Since the start of the project, there has been great focus on creating an application that both looks good, and is easy to use. We feel confident that we have achieved this, especially after hearing feedback from user tests.

Weaknesses

- **Lack of administrator functionality**
  The administrator has all the necessary functionality to control the application. It still feels like the administrator role has some shortcomings when it comes to the relationship between the application and its customers. The business aspect of operating a platform such as this was not part of the assignment, but it is perceived that the product owner needs an increased functionality for this in order for the platform to function optimally in production.

- **General application**
  This depends somewhat on perspective. A general area of use means that this application can be used by several different organizations. But if the application had been developed specifically for a type of organization, it could have been optimized better with additional information. For instance, for classrooms, additional information could have been if the room had a blackboard or a projector

6 | Conclusion

6.1 Conclusion

In the following chapter, we outline the conclusions as they relate to the research questions, as well as further work and improvements.

6.1.1 Research question 1

Research question 1
For SmartBooking:
(a) Which authentication method is suited?
(b) What are the influential factors when implementing the authentication system?

(a) Password authentication is suited as the primary authentication method. For additional security, one-time passwords have been implemented as a secondary authentication method.

(b) The authentication system is influenced by security, usability, complexity and scalability considerations. The authentication method has to be secure, but also user-friendly and simple to implement. In addition to this, it needs to be implemented in such a way that it is not vulnerable to attacks like XSS and CSRF. The authentication system also needs to be scalable so it can accommodate more users in the future.

6.1.2 Research question 2

Research question 2
What challenges has the lack of physical presence posed on our collaboration as a team, and how have these challenges been handled?

The observations we have made correspond to the current research in the field. It is not the lack of technical solutions that makes this working method more challenging, it is rather the difference in how we work day to day compared to before the pandemic. Without the physical presence of each other, we lose a substantial part of how we are used to communicate, and how we gather information.

A digital workspace results in separation of information. Instead of having the relevant information about tasks available in the same room, where you only need to move your gaze or quickly ask someone, in a digital work environment, one must seek out the information on different platforms. This means that concepts such as scrum boards and stand-up meetings lose some of their effect.

6.2 Further work

This section presents our thoughts on further work.

6.2.1 Authentication

- **Refresh token rotation**
  Currently, every time a client requests a new access token with a valid refresh token, a new access token is issued. With refresh token rotation, the server issues a new access token and a new refresh token. This way, users will not have to re-authenticate every time the refresh token expires.

- **Automatic reuse detection**
  When refresh token rotation is implemented, old refresh tokens should be invalidated. This helps safeguard the application from replay attacks.

- **Notification when unknown device is authenticated**
  When an unknown device is authenticated, the user should be notified. If an unknown

device is able to enter the correct password, but not the OTP, the user should also be notified and encouraged to change their password.

- **Push notifications**
  Push notifications can be implemented as an additional authentication mechanism. It can either replace, or be implemented in addition to passwords.

- **Magic Links**
  Students can request a login link be sent to their email-address instead of entering their password.

6.2.2 System Development

- **Front-end testing**
  The application we developed did not have front-end testing, which is an important piece that we would focus on implementing properly given the time to do it. As the deadline of the project approached, we had to select the most critical elements of the system to develop, and front-end testing was one of the aspects that were left out.

- **Proposition algorithms**
  It could be appropriate to implement an algorithm that could suggest both rooms and reservations to a user. This could have been that if a room turned out to be closed or inaccessible, the application could have suggested a similar room. Or if a pattern of reservations arises, a proposal to create this reservation for the next period could come as a notification on the dashboard.

- **WCAG 2.0**
  Although the application is user-friendly, it could have been done further work to make it more accessible to people with disabilities. Sound functionality, or an option to set the text size larger could have been implemented to help the visually impaired.

- **Statistics and historical data**
  The solution in our application is relatively simple. For a customer, it is possible to find out who has reserved a room and when. In a future version, it would be useful for a customer to get more insights, for example; which rooms are most popular, what time of day is busiest, who are most active and other statistics that could contribute to improving the overall experience for the end users.

Bibliography

[1] OneLogin. What's the difference between otp, totp and hotp? Accessed: 18.02.2021. [Online]. Available: https://www.onelogin.com/learn/otp-totp-hotp

[2] R. Raghuwanshi. (2017, Apr) Jwt (json web tokens) are better than session cookies. Accessed: 26.03.2021. [Online]. Available: https://dzone.com/articles/jwtjson-web-tokens-are-better-than-session-cookies

[3] Auth0. (2020, Feb) Refresh tokens: When to use them and how they interact with jwts. Accessed: 26.02.2021. [Online]. Available: https://auth0.com/blog/refresh-tokens-what-are-they-and-when-to-use-them/

[4] AJMC. A timeline of covid-19 developments in 2020. Accessed: 21.04.2021. [Online]. Available: https://www.ajmc.com/view/a-timeline-of-covid19-developments-in-2020

[5] T. E. Times. (2020, Nov) Definition of 'authentication'. Accessed: 26.03.2021. [Online]. Available: https://en.wikipedia.org/wiki/Authentication#cite_note-1

[6] Google. (2019) New research: How effective is basic account hygiene at preventing hijacking. Accessed: 02.03.2021. [Online]. Available: https://security.googleblog.com/2019/05/new-research-how-effective-is-basic.html

[7] DataProd. (2021, Feb) Save your data with these empowering password statistics. Accessed: 27.02.2021. [Online]. Available: https://dataprot.net/statistics/password-statistics/

[8] G. . H. Poll. (2019, Feb) Online security survey. Accessed: 27.02.2021. [Online].

Available: https://services.google.com/fh/files/blogs/google_security_infographic.pdf

[9] SecureAuth. (2017, July) Wake-up call on users' poor password habits. (PDF) Accessed: 22.03.2021. [Online]. Available: https://1radn12ycpxi3po60v21nhf7-wpengine.netdna-ssl.com/wp-content/uploads/2018/09/180926-CIAM-Infographic.pdf

[10] N. I. of Standards and Technology. (2017, June) Digital identity guidelines. Accessed: 22.02.2021. [Online]. Available: https://pages.nist.gov/800-63-3/sp800-63b.html

[11] Pin analysis. Accessed: 14.04.2021. [Online]. Available: https://www.datagenetics.com/blog/september32012/index.html

[12] R. S. Sollie. Security and usability assessment of several authentication technologies. Accessed: 20.02.2021. [Online]. Available: https://ntnuopen.ntnu.no/ntnu-xmlui/bitstream/handle/11250/143896/Sollie%20-%20Security%20and%20usability%20assessment%20of%20several%20authen.pdf?sequence=1&isAllowed=y

[13] C. Mulliner, R. Borgaonkar, P. Stewin, and J.-P. Seifert. (2020, Mar) Sms-based one-time passwords: Attacks and defense. (PDF) Accessed: 22.03.2021. [Online]. Available: http://www.mulliner.org/collin/academic/publications/mulliner_dimva2013.pdf

[14] Google. (2021) Bedre sikkerhet for google-kontoen din. Accessed: 20.03.2021. [Online]. Available: https://www.google.com/landing/2step/

[15] Facebook. (2021) Hva er totrinnsverifisering, og hvordan fungerer det på facebook? Accessed: 02.03.2021. [Online]. Available: https://www.facebook.com/help/148233965247823

[16] Usenix. A usability study of five two-factor authentication methods. Accessed: 05.03.2021. [Online]. Available: https://www.usenix.org/system/files/soups2019-reese.pdf

[17] D. Goodin. (2015) Mris show our brains shutting down when we see security prompts. Accessed: 20.04.2021. [Online]. Available: https://arstechnica.com/information-technology/2015/03/mris-show-our-brains-shutting-down-when-we-see-security-prompts/

[18] Wired. We tried really hard to beat face id—and failed (so far). Accessed: 05.03.2021. [Online]. Available: https://www.wired.com/story/tried-to-beat-face-id-and-failed-so-far/

[19] W. Hindawi. Usability evaluation model for biometric system considering privacy concern based on mcdm model. Accessed: 05.03.2021. [Online]. Available: https://downloads.hindawi.com/journals/scn/2019/8715264.pdf

[20] OWASP. (2020, Jan) Html5 security cheat sheet. Accessed: 03.03.2021. [Online]. Available: https://cheatsheetseries.owasp.org/cheatsheets/HTML5_Security_Cheat_Sheet.html

[21] Auth0. (2020) Token storage. Accessed: 20.02.2021. [Online]. Available: https://auth0.com/docs/security/data-security/token-storage#don-t-store-tokens-in-local-storage

[22] A. E. N. Sabena. (2018, Nov) Why is storing tokens in-memory recommended? Accessed: 22.02.2021. [Online]. Available: https://community.auth0.com/t/why-is-storing-tokens-in-memory-recommended/17742

[23] K. Schwaber and J. Sutherland. (2017, Nov) The scrum guide. the definitive guide to scrum: The rules of the game. Accessed: 10.03.2021. [Online]. Available: https://scrumguides.org/docs/scrumguide/v2017/2017-Scrum-Guide-US.pdf

[24] A. A. Lunsford and L. Ede. Writing together: Collaboration in theory and practice. Accessed: 14.03.2021. [Online]. Available: https://books.google.no/books/about/Writing_Together.html?id=QWbEuAAACAAJ&redir_esc=y

[25] K. Constantino, S. Zhou, M. Souza, E. Figueiredo, and C. Kästner. Understanding collaborative software development: An interview study. Accessed: 14.03.2021. [Online]. Available: https://dl.acm.org/doi/pdf/10.1145/3372787.3390442?casa_token=lUBwzikpsQcAAAAA:UPuAwOu9z2skI5PE5fapl-39asIk4OORUQIlcDD4cMBMNVdjz5rzQm-0QUVxj-_IHb2U3VahV9kIog

[26] P. A. da Mota Silveira Neto, U. A. Mannan, E. S. de Almeida, N. Nagappan, D. Lo, P. S. Kochhar, C. Gao, and I. Ahmed. (2020, Aug) A deep dive on the impact of covid-19 in software development. Accessed: 02.04.2021. [Online]. Available:

https://arxiv.org/pdf/2008.07048.pdf

[27] Udemy. (2021) Udemy. Accessed: 20.02.2021. [Online]. Available: https://www.udemy.com/

[28] D. I. Daniel Lowe Wheeler. zxcvbn: Low-budget password strength estimation. Accessed: 20.02.2021. [Online]. Available: https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/wheeler

[29] GitHub. (2020, May) 10k-most-common.txt. Accessed: 02.03.2021. [Online]. Available: https://github.com/danielmiessler/SecLists/blob/master/Passwords/Common-Credentials/100k-most-used-passwords-NCSC.txt

A | Appendix

A.1 Vision Document

NTNU | Norwegian University of Science and Technology

BACHELOR THESIS

VISION DOCUMENT

Espen Kalleberg Håkon Harnes Svein Jakob Høie

May 19, 2021

Revision History

Date	Version	Description	Authors
27.01.2021	1.0	First draft	Espen Kalleberg, Håkon Harnes og Svein Jakob Høie
09.04.2021	1.1	Rewritten so that there is a clearer distinction between employer, customer and user.	Espen Kalleberg, Håkon Harnes og Svein Jakob Høie

Contents

1 Introduction

The goal of this document is to describe visions related to our bachelor project at the Norwegian University of Science and Technology (NTNU) Trondheim. The task is to create a system used to manage reservations and bookings for customers of the product owner. The system will make use of QR-codes and modern technologies in order to make the process of booking rooms more efficient and user-friendly.

2 Problem and product

2.1 Problem

The problem	is that today's system relies on physical presence by using pen and paper in order to book rooms.
This affects	users that want to book rooms and customers that have to maintain the system.
As a result of this	the users are forced to book rooms on the premises by pen and paper.
A satisfying solution will	allow users to book rooms digitally.

2.2 Product

For	MI OG MA HOLDING AS.
That need	a digital system for booking rooms.
The product	SmartBooking.
Allows for	digital room booking and administration.
Contrary to	booking rooms by pen and paper.
Our product	allows for booking rooms digitally.

3 Stakeholders and users

3.1 Stakeholders

Name	Description	Role under development
Employer	MI OG MA HOLDING AS / Arne Pukstad Juliussen	Make requirements. Approve the product.
Customer	Customers of MI OG MA HOLDING AS	Testing
User	User of the system	Testing
Supervisor	Ali Alsam	Guidance

3.2 Users

Name	Description	Role under development
Employer	MI OG MA HOLDING AS	Testing
Customer	Customer of MI OG MA HOLDING AS	Testing
User	User of the system	Testing

3.3 User environment

This system will be used in environments with 10 - 50 different rooms available for booking. For the user it should be as fast and simple as possible with minimal amount of overhead to find and book a room. A user can scan a QR-code outside of a room to quickly get a overview over the next two weeks schedule or manually search for a room.

3.4 Summary of employer needs

Need	Priority	Today's solution	Suggested solution
Add Admins	High	No solution	Admins can add other admins by filling a form
Change admin passwords	Medium	No solution	Admins can change other admins passwords
Add customer	High	No solution	Admins should be able to add new customers by typing their information into forms and submitting them
Manage customers	High	No solution	Admins should be able to manage existing customers, set their accounts inactive or delete them
Manage admins	Medium	No solution	Admins should be able to manage other admin accounts, pausing or deleting them
Search customers	High	No solution	Admins should be able to search for existing customers
Import CSV data	Medium	No solution	Import data about new customer from CSV file

3.5 Summary of customer needs

Need	Priority	Today's solution	Suggested solution
Add users	High	No solution	Let the customer add new users
Manage users	High	No solution	Let the customer manage existing users
Manage rooms	High	No solution	Let the customer manage and add rooms
Manage booking policy	High	No solution	The customer can set parameters regarding the room booking process
Import CSV data	Medium	No solution	Import rooms and users from CSV files
Import Excel data	Medium	No solution	Import rooms and users from Excel files
Search for user	High	No solution	Functionality to search for users
Edit PIN for user	High	No solution	In case of lost PIN, a customer should be able to edit it
Find reservations on rooms	High	No solution	Find existing reservations on rooms, downloading a file of the reservations should be possible
List all rooms	High	No solution	List existing rooms

3.6 Summary of user needs

Need	Priority	Today's solution	Suggested solution
Register account	High	No solution	User can register a new user contact information
Log in to account	High	No solution	User can log in to account
View a rooms schedule	High	Paper form on-site	View a rooms schedule for the next 14 days
Get room schedule by QR-code	High	Paper form on-site	View a rooms schedule by scanning a QR-code on the door
Get room schedule by search	High	No solution	View a rooms schedule by searching for it
Book room from schedule	High	Paper form on-site	Book a room by selecting desired time interval from the calendar
Change user display information on calendar	Low	No solution	A user should be able to choose which personal information is displayed on booked rooms on calendar
Retrieve PIN	High	No solution	Get your PIN code by email or text
Find available rooms	High	Paper form on-site	Search functionality with filters for availability

3.7 Alternatives to our product

Alternatives to our solution is the current solution, which uses paper form on-site, or other third-party digital solutions like TimeEdit [1], LettStyrt [2] or Cenera [3].

4 Product overview

4.1 The products role in the user environment

The system will be a independent and complete solution that replaces the current system. Implementation of the system, with QR-codes, should require minimal advance work and maintenance.

4.2 Prerequisites and dependencies

1. The system requires a internet connection to access room schedules and book a reservation.
2. To use QR-code functionality the system need to acquire camera on the users device to scan the room.

5 Functional requirements

Functionality	Description
Add and manage accounts	Admins should be able to add new admins and register new customers. Customers should be able to add users to their plan with their details.
Change and reset PIN-code	Changing PIN-code should be easy and possible for all accounts by getting a reset email, whether the account is an admin, a customer or a user.
Add and manage new rooms	Customers should be able to add new rooms to their plan, and also manage existing ones. Setting a room inactive for a time period needs to be possible.
Make a room reservation	A user is able to book a room from the rooms schedule if its available during the given time frame
Manage reservations	One should be able to manage your existing reservations, e.g. delete a reservation you do not need.
Check room schedule	A user is able to check a rooms schedule to get information about the availability of the given room
Search for room	A user should be able to find a room in the system either through search functionality or with the QR-code
Import data from external files	Where information needs to be input, it should also allow for external files to be imported. For example when adding rooms, one should be able to import a list of all rooms and relevant information about them.
List all rooms	A user should be able to list all rooms and select a room from the list to view its schedule.
Log into account	A user should be able to log into account with phone number and PIN-code.
Scan QR-codes for room schedule	One should be able to scan a QR-code on a room and be redirected to the rooms schedule.
Generate QR-codes for rooms	The system shall be able to generate new and unique QR-codes to be placed on the entrance of rooms.
Set a booking policy	Customers should be able to set a booking policy for their users. This should include parameters that decide how many reservations a user can make per week, when booking opens and closes and more.

6 Non-functional requirements

Keyword	Description
Availability	System must be available from anywhere with an internet connection
Compatibility	System must operate effectively for both computers and mobile phones
Maintainability	System must be easy to maintain, and require only few hours of maintenance
Reliability	System must be reliable and have as few failures as possible
Scalability	Admins should be able to add new customers. Customers should be able to add rooms and user to their needs
Security	Only admins should be able to add and delete customers. Customers should be able to add and delete users for their part of the system

References

[1] (2021, jan) Timeedit. [Online]. Available: https://www.timeedit.com/

[2] G. Skari. (2021, jan) Lettstyrt. [Online]. Available: https://lettstyrt.no/bedrift/booking-moterom/

[3] (2021, jan) Cenera. [Online]. Available: https://cenera.no/tjenester/moterom/

A.2 Software Requirements Specification

NTNU | Norwegian University of Science and Technology

BACHELOR THESIS

SOFTWARE REQUIREMENTS SPECIFICATION

Espen Kalleberg Håkon Harnes Svein Jakob Høie

May 20, 2021

Revision History

Date	Version	Description	Authors
27.01.2021	1.0	First draft	Espen Kalleberg, Håkon Harnes og Svein Jakob Høie
10.03.2021	1.1	Added user stories	Espen Kalleberg, Håkon Harnes og Svein Jakob Høie
15.03.2021	1.2	Added wireframes	Espen Kalleberg, Håkon Harnes og Svein Jakob Høie
18.05.2021	1.3	Changed the appearance of the table so that it matches other attachments.	Espen Kalleberg, Håkon Harnes og Svein Jakob Høie

Contents

1 Introduction

This document is written in correlation with the subject TDAT3001 - Bachelor Thesis in Computer Engineering. The document is the specification of the software requirements related to the system SmartBooking which is part of the project. In this document, user stories will be used in order to cover all user functionality for the system. For each of the user stories, acceptance criterias which need to be met in order for the requirement to be fulfilled, are presented. A domain model which shows how parts of the system are connected, as well as prototypes of the system are also presented.

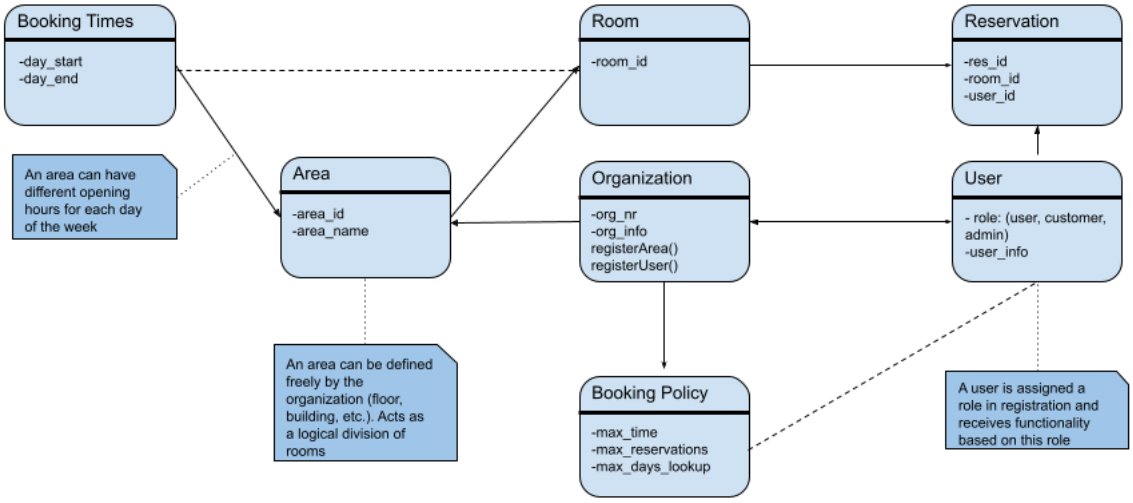
2 User stories

Num	Description	Acceptance criteria
1	As an administrator, I want to add new administrators, so that they can help maintaining the system	Administrator can add new administrators to the system
2	As an administrator, I want to add a new organization, so that new organizations can start using the system	Administrators can add new organization to the system
3	As an administrator, I want to add new customers connected to an organization, so that they can start using the system	Administrators can create users with the role of customer and connect them to an existing organization
4	As an administrator, I want to edit existing customers/administrators, so that their information is correct and up to date	Administrators can change and update customer information
5	As a customer, I want to add new users so that they can start using the system	Customers can add users to their organization. Either manually, or by importing from CSV files
6	As a customer, I want to edit existing users, so that their information is correct and up to date	Customers can edit the information of users connected to their organization

7	As a customer, I want to delete existing users, so that the user list is up to date	Customers can delete users connected to their organization
8	As a customer, I want to add new rooms so that my users can use all the available places	Customers can register room either manually or by importing from CSV files
9	As a customer, I want to edit existing rooms, so that they are always up to date	Customers can edit existing rooms
10	As a customer, I want to maintain a booking policy that sets rules for reservations, so that users can't exploit the reservation system.	Customers can register and edit their organization's booking policy
11	As a customer, I want to set opening times for buildings/places, so that users can't book rooms in places that are not open.	Customers can edit opening hours for areas.
12	As a customer, I want to have a overview over reservations so that I can see which rooms are used and have the opportunity to see who has used the room	Customers has the opportunity to review all upcoming and previous reservations in the "reservations" tab
13	As a user, I want to book a room so that I can be sure that I will have a place to sit	Users can book available rooms. Either by choosing a specific room, or searching for an available room given date and time.
14	As a user, I want an overview of my reservations so that I can see when and where I have booked a room	Users can see an overview of their reservations in the "reservations" tab
15	As a user, I want to delete my reservations so that I don't occupate a room when I won't be able to use them	Users can delete their reservations in the "reservations" tab
16	As a user, I want to scan a QR-code on the door of a room, so that I can see its calendar and book it when available	Users can use their mobile phone camera to scan a QR code outside a room. The user will then automatically be redirected to the room's calendar

17	As an administrator/customer/user, I want to change my password so that I make sure my account is secure	(a) Users can change their password by requesting a password reset link. This will be delivered by email (b) Users can change their password by going to the settings when logged in. He can then type in the old and the new password to change it
18	As an administrator/customer/user, I want to enable and use two-factor authentication, so that I can keep my account secure	All users have the ability to turn on two-factor authentication in the settings tab.

3 Domain model



4 Prototypes

For this application, wireframes were designed with Figma. They were then connected together with buttons in order to create a clickable prototype. The clickable prototype was used for user testing.

4.1 Wireframes

Figures 1, 2 and 3 show examples of wireframes from the prototype. The entire prototype is linked in the references. [1]

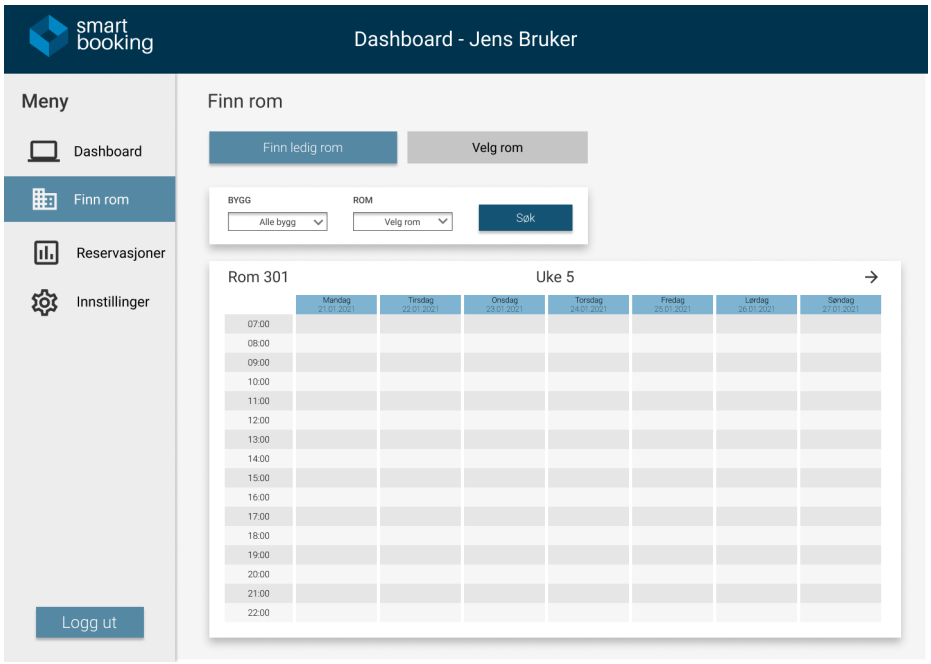


Figure 1: Calendar wireframe

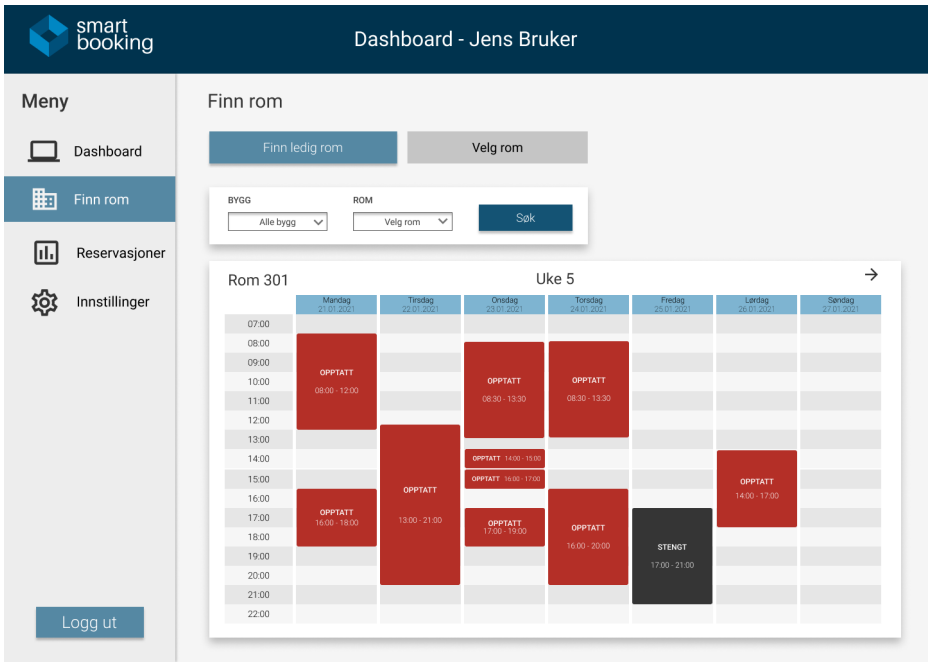


Figure 2: Calendar wireframe

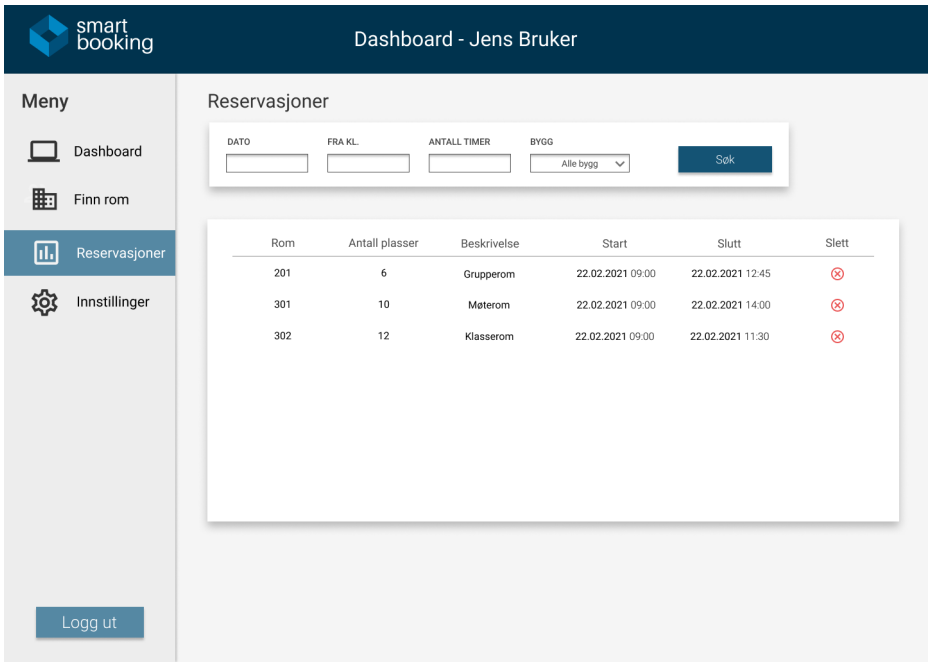


Figure 3: Reservations wireframe

References

[1] E. K. Svein Jakob Høie and H. Harnes. (2021, Mar) Figma. [Online]. Available: https://www.figma.com/file/jm0ua7NhmeoFMqZEn4c6nt/Wireframes?node-id=1%3A2

A.3 System Documentation

NTNU | Norwegian University of Science and Technology

BACHELOR THESIS

SYSTEM DOCUMENTATION

Espen Kalleberg Håkon Harnes Svein Jakob Høie

May 20, 2021

Revision History

Date	Version	Description	Authors
27.01.2021	1.0	First draft	Espen Kalleberg, Håkon Harnes og Svein Jakob Høie
13.05.2021	1.1	Written remaining chapters before submission	Espen Kalleberg, Håkon Harnes og Svein Jakob Høie

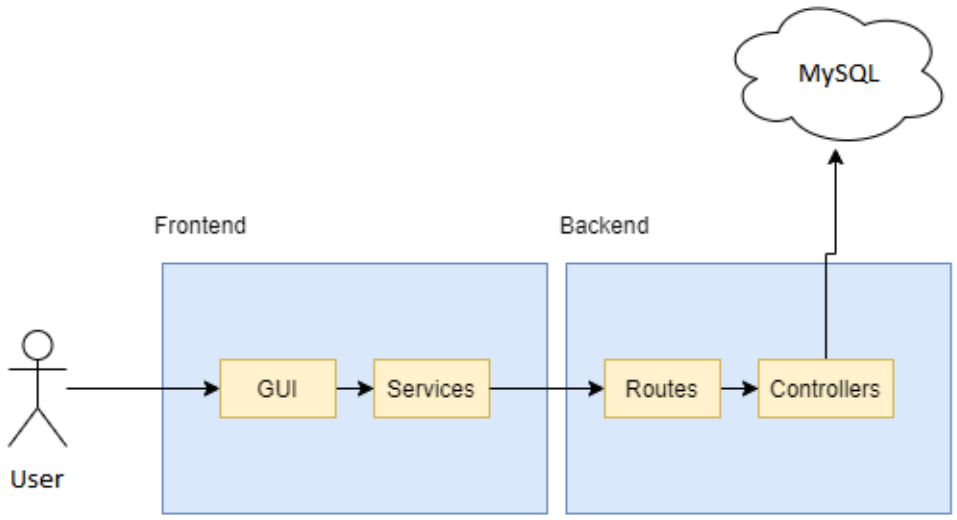
Contents

1 Introduction

This document was written in correlation with the topic TDAT3001, Bachelor Thesis in Computer Engineering, during the spring semester of 2021. The document describes the system that has been developed, as well as an installation guide. The task was to create a room booking application intended for use by students at Sonans Trondheim, with the possibility that the product could be offered to other Sonans schools in Norway.

2 Architecture

Below is the main architecture of the system.



A user uses a Graphical User Interface (GUI) that communicates with service classes. These service classes request information from the back-end server on behalf of the user. For authorization purposes, an access token is sent in the Authorization header of the request. Based on the endpoint, what type of request it is and what parameters the user has attached, a controller will perform relevant queries to the database and return the result.

3 Project structure

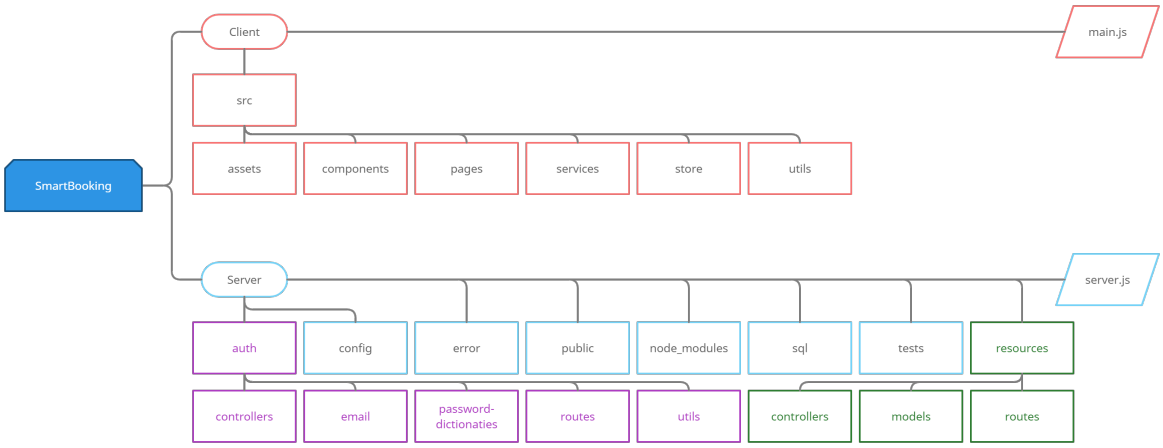


Figure 1: Project structure

The system follows a traditional folder structure for both front-end and back-end. Figure 1 shows the project structure.

4 Database model

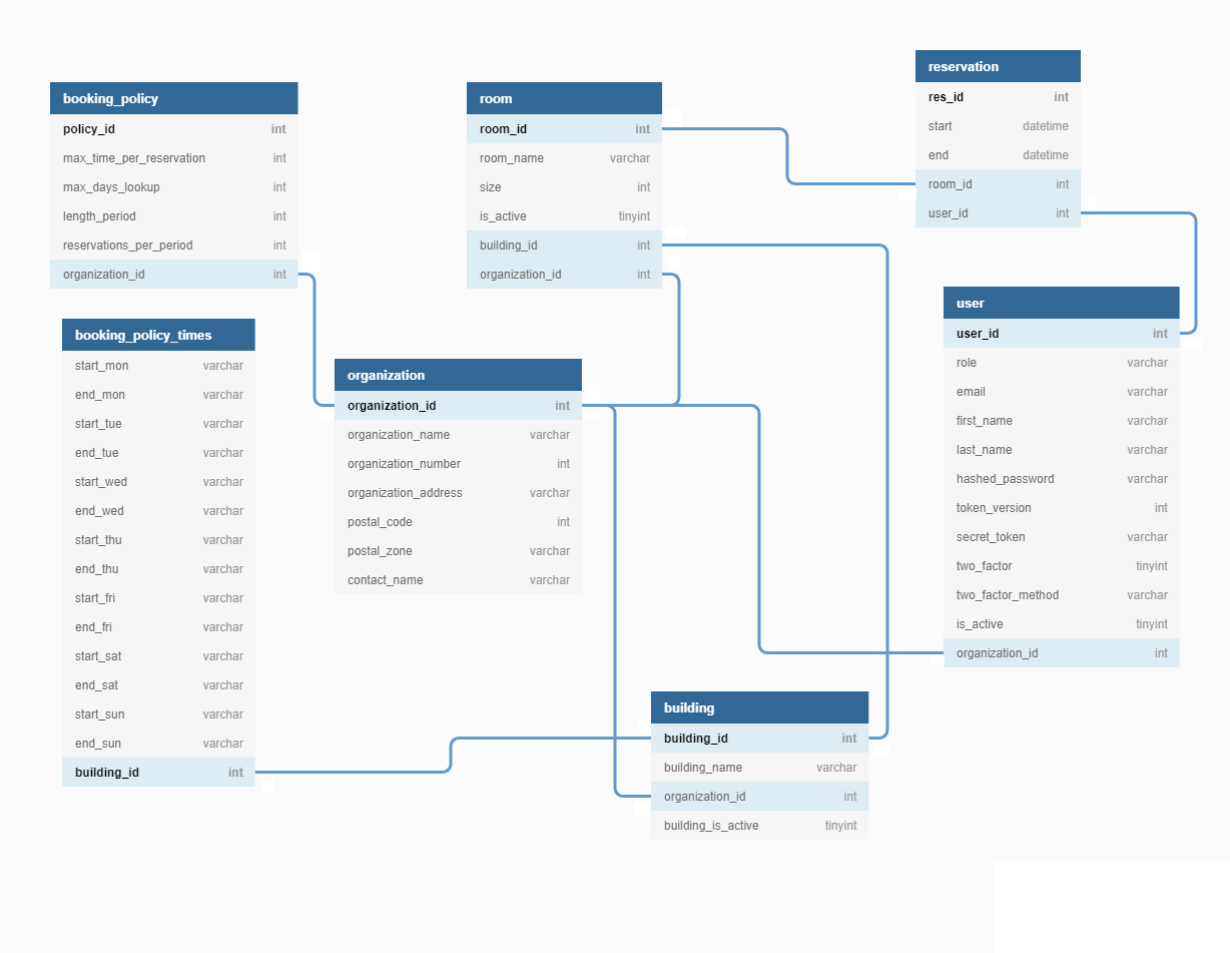


Figure 2: Database model

This database model shows how the tables are structured. A core part of the system is the organization table. Organization_id is essential and has many connections. This is because it is used to structure the various “domains” in the system and separate institutes from each other. It is also critical to understand that the “role” variable in the user table is where the rights and functionality of a user are determined. It can be set to one of the following: User, customer or admin. Figure 2 shows the database model.

5 API

An overview of available endpoints is presented below.

End-point	HTTP-method	Description
/api/login	POST	Login a user with email and password
/api/logout	GET	Logs a user out. Sends an empty cookie with short expiration date to remove refresh token cookie from client.
/api/register-admin	POST	Registers a new admin
/api/register-customer	POST	Registers a new customer
/api/register-user	POST	Registers a new user
/api/refresh	POST	Refreshes the access token with a refresh token
/api/forgot-password	POST	Sends email with forgot password link
/api/update-password	PATCH	Updates a user's password
/api/reset-password/reset-token	PATCH	Resets a user's password from reset token
/api/verify	POST	Verifies one-time-password
/api/disable-two-factor	PATCH	Disables two-factor authentication
/api/enable-two-factor	PATCH	Enables two-factor authentication
/api/users?organization_id=?	GET	Get users by organization
/api/users/:user_id	GET, PUT, DELETE	Get, update, delete a user
/api/rooms/	POST	Create a new room
/api/rooms?building_id=?	GET	Get all rooms in a building/area
/api/rooms?organization_id=?	GET	Get all rooms for a organization
/api/rooms/search?building_id =?start=?end=?	GET	Get all available rooms in a building given a start and end time
/api/rooms/search?organization_id =?start=?end=?	GET	Get all available rooms for a organization given a start and end time
/api/rooms/:room_id	GET, PUT, DELETE	Get, update or delete a room
/api/policies/	POST	Create a policy
/api/policies/:id	DELETE, PUT	Delete or update a policy
/api/policies?organization_id=?	GET	Get policy for a organization
/api/buildings/	POST	Create new building
/api/buildings?organization_id=?	GET	Get all buildings for a organization

End-point	HTTP-method	Description
/api/buildings/:building_id	PUT, DELETE	Update or delete building
/api/reservations/	POST	Create reservation
/api/reservations?organization_id=?	GET	Get all reservations in an organization
/api/reservations?user_id=?	GET	Get all reservations for a user
/api/reservations?room_id=?	GET	Get all reservations for a room
/api/reservations?room_id=? start=?end=?	GET	Get all reservations for a room given start and end time
/api/reservations/:reservations_id	GET, PUT, DELETE	Get, update or delete a reservation
/api/policy-times/	POST, PUT	Create or update Policy times
/api/policy-times?building_id=?	GET	Get policy times for a building
/api/policy-times/:id	DELETE	Delete a policy time
/api/organizations/	POST, GET	Get all organizations or create organization
/api/organizations/:organization_id	GET, PUT	Get or update a organization

These end-points are protected with a middleware functions to ensure only authorized users are able to access the protected resources. Also, some resources are restricted to administrators or customers only, like deleting users. This is detailed in section 6.

6 Security

Some of the end-points are protected. This has been achieved with JSON Web Tokens (JWT). For authorization proposes, there are four tokens:

1. Access tokens
2. Refresh tokens
3. Verification tokens
4. Password reset tokens

Access tokens are used to access protected resources. Whenever a user successfully authenticates themselves, for example by entering the correct email-address and password, an access token is issued. The access token is sent in the authorization header of all HTTP-requests after a successful login. When the server receives an access token, it decodes and validates the token. Thereafter, it uses the information stored in the token for authorization purposes. The server can then decode the token to reveal the payload. The payload contains the user's role and is used to check if it they are authorized to perform the request. For example, a user is not able to delete other user's reservations, but administrators are.

The authorization functionality is implemented with middleware functions. They are `requiresAccessToken`, `requiresRefreshToken`, `requiresPasswordResetToken` and `requiresVerificationToken`. Additionally, a `restricTo` middleware function can be applied to the end-point to for example restrict it to administrators.

The refresh token is used to request new access tokens. It is stored in a secure HTTP-only cookie to make it safe from XSS attacks. The SameSite flag is also set to protect against CSRF attacks. The access token is stored in memory to protect it from CSRF and XSS attacks. With the refresh and access token strategy, together with data sanitizers against XSS and SQL injection on the server-side, the system is safe from most common attacks.

Verification tokens are used for two-factor authentication purposes. A user might have to provide a password, and a one-time password from their email. When the user correctly provides the email-address and password, a verification token is issued. When providing the server with the one-time password from the email, the verification token is also sent. This way, the server can first validate the verification token to ensure the correct email-address

and password was initially provided. Then, the server can validate the one-time password and send an access token if it is valid.

Password reset tokens are used for resetting passwords. They are received by email and is embedded in the password reset link.

Password authentication was chosen as the primary authentication method. This is sufficient for user accounts, because the security risk is low. If an attacker gains control over a user account, the worst he can do is create and delete reservations. The victim can regain control by resetting the password through their email-address. Then, they can delete the undesired reservations. The passwords are hashed and salted using *bcrypt.js*.

For administrator and customer accounts, the risks are higher. An attacker with access to these types of accounts could delete users and leak sensitive information. More security is needed, and that is why one-time passwords have been implemented as an additional authentication mechanism. It is used in conjunction with password authentication, either with Google Authenticator (2FA), or over email (2SV).

7 Installation

In order to run the system, there are a couple instructions that need to be followed.

1. First you need to have the correct ZIP-file that contains the project. The contents of the ZIP needs to be extracted into a fitting folder.
2. The folder named *smartbooking* is the main folder. It contains two folders; one for the *front-end* part and one for the *back-end* part of the application. In each of these two folders, one has to go to their location on the command line and type the command *npm install*, in order to install necessary dependencies. Note that Node.js, which comes with the package manager *npm*, has to be installed on the system for the command to be recognised.
3. The system requires environment variables on both the client and server.
    - CLIENT: Create a file named .env, placed on the highest level in the folder structure. See table 1.
    - SERVER: Create a new file, .env, places on the highest level in the folder structure. See table 2.

When the .env files are made and contains the correct variables, the client and server can be run by typing *npm start* in the command line, when inside the respective folders. The application can then be accessed on the IP-address/domain where the client is running.

Variable	**Description**	**Example value**
`VUE_APP_LINK`	Client production link	https://smartbooking.no

Table 1: Client .env file

Variable	Description	Example value
NODE_ENV	The current environment. Can be set to development for additional logging.	production
CLIENT_PRODUCTION_LINK	Client production link	https://smartbooking.no
CLIENT_DEVELOPMENT_LINK	Client development link	http://localhost:8080
API_PRODUCTION_LINK	API production link	https://smartbooking.no
API_DEVELOPMENT_LINK	API development link	http://localhost:3000
PORT	Server port	3000
DATABASE_HOST	Database host	cpanel.proisp.no
DATABASE_NAME	Database name	SmartBooking
DATABASE_USER	Database username	smartbooking
DATABASE_PASSWORD	Database password	lkasDcdedå
ACCESS_TOKEN_SECRET	Access token secret. Must not be shared.	32-character-long-secret-string!
REFRESH_TOKEN_SECRET	Refresh token secret. Must not be shared.	a-different-super-secret-string!
VERIFICATION_TOKEN_SECRET	Verification token secret. Must not be shared.	another-super-secret-token-value
PASSWORD_RESET_TOKEN_SECRET	Password reset token secret. Must not be shared.	the-last-secret-value-dont-share
EMAIL_FROM	Email account to send emails from from	donotreply@smartbooking.no
EMAIL_USERNAME	Email username	smartbooking
EMAIL_PASSWORD	Email password	!asDDsadååÅØW
EMAIL_HOST	Email Host	smtp.mailtrap.io
EMAIL_PORT	Email port	2525

Table 2: Server .env file

8 Source code documentation

8.1 Front-end

Documentation of service classes is provided below.

AuthService			
Method	**Parameters**	**Type**	**Description**
disableTwoFactorAuth			
enableTwoFactorAuth	twoFactorMethod	String	2FA Method
forgotPassword	email	String	User email
login	email password	String String	User email User password
logout			
refresh			
updatePassword	oldPassword newPassword	String String	Old password New password
verify	code	String	Verification code

BuildingService			
Method	**Parameters**	**Type**	**Description**
createBuilding	building_name organization_id	String Number	Name of building Organization id
deleteBuilding	id	Number	Building id
getBuildings	id	Number	Organization id
getBuildingPolicy	id	Number	Building id
updateBuilding	building_name building_is_active organization_id building_id	String Number Number Number	Name of building 0 for inactive, 1 for active Organization id Building id
updateBuildingPolicy	id times	Number Object	Building id Object with updated times

OrganizationService			
Method	**Parameters**	**Type**	**Description**
createOrganization	organization_name	String	Name of organization
	organization_number	String	Organization number
	organization_address	String	Organization address
	postal_code	Number	4-digit postal code
	postal_zone	String	Postal zone
	contact_name	String	Name of contact person
delete	id	Number	Organization id
getOrganizations			
updateOrganization	organization_name	String	Name of organization
	organization_number	String	Organization number
	organization_address	String	Organization address
	postal_code	Number	4-digit postal code
	postal_zone	String	Postal zone
	contact_name	String	Name of contact person

ReservationService			
Method	**Parameters**	**Type**	**Description**
createReservation	start	Date	Start of reservation
	end	Date	End of reservation
	room_id	Number	Room id
	user_id	Number	User id
	organization_id	Number	Organization id
deleteReservation	id	Number	Reservation id
getReservationsByRoomAndTime	id	Number	Room id
	start	Date	Start of reservation
	end	Date	End of reservation
getReservationsByUserId	id	Number	User id
getReservationsAndUsers	id	Number	Organization id

RoomService			
Method	**Parameters**	**Type**	**Description**
createRoom	room_name	String	Room name
	size	Number	Number of seats
	is_active	Number	0 for inactive, 1 for active
	organization_id	Number	Organization id
	building_id	Number	Building id
createRooms	rooms	Array[room]	Array of room objects
deleteRoom	id	Number	Room id
getAvailableRooms	organization_id	Number	Organization id
	start	Date	Starttime of search
	end	Date	Endtime of search
	building_id	Number	Building id
getRoom	id	Number	Room id
getRooms	id	Number	Organization id
updateRoom	room_id	Number	Room id
	room_name	String	Room name
	size	Number	Number of seats
	is_active	Number	0 for inactive, 1 for active
	building_id	Number	Building id

UserService			
Method	**Parameters**	**Type**	**Description**
deleteUser	id	Number	User id
forgotPassword	email	String	User email
getAccounts			
getLoggedInUser			
getUserFromResetToken	resetToken	String	Reset token
getUsersByOrganization	id	Number	Organization id
login	email	String	User email
	password	String	User password
logout			
register	first_name	String	User firstname
	last_name	String	User lastname
	email	String	User email
	role	String	User role
	organization_id	Number	Organization id
registerUsers	users	Array[user]	Array of user objects
resetPassword	password	String	New user password
	token	String	Reset token
sendVerificationToken			
updateUser	user_id	Number	User id
	first_name	String	User firstname
	last_name	String	User lastname
	email	String	User email
	is_active	Number	0 for inactive, 1 for active
verifyVerificationToken	token	String	Verification token

8.2 Back-end

Documentation for the server API has been generated. The folder named *smartbooking/back-end/docs* contains the API docs. In this folder you will find the documentation for the API. It can be opened by opening the *index.html* file in a web browser.

9 Continuous integration and testing

The API repository has a CI/CD solution in GitHub with the help of GitHub Actions and JEST. The tests in the tests folder are performed automatically when a push is performed. These are performed in the latest version of Ubuntu. A virtual database is set up for the tests using the setup.sql file. The test environment is controlled by the node.js.yml file in the .github/workflow folder.

Tests have been made for the most essential parts of the application. This involves tests for retrieval and editing of rooms and organizations. Tests have also been made for functionality that deals with reservations. It is important to make sure that different date and time variables are in the same format throughout the application, so this is also tested here.

A.4 Process Documentation

NTNU | Norwegian University of Science and Technology

BACHELOR THESIS

PROCESS DOCUMENTATION

Espen Kalleberg Håkon Harnes Svein Jakob Høie

May 19, 2021

Revision History

Date	Version	Description	Authors
27.01.2021	1.0	First draft	Espen Kalleberg, Håkon Harnes og Svein Jakob Høie
09.04.2021	1.1	Added meeting notices and minutes from the meetings with the supervisor and client	Espen Kalleberg, Håkon Harnes og Svein Jakob Høie
19.05.2021	1.2	Added remaining meeting documentation and timesheets	Espen Kalleberg, Håkon Harnes og Svein Jakob Høie

Contents

2 Progress plan

2.1 Gantt timeline

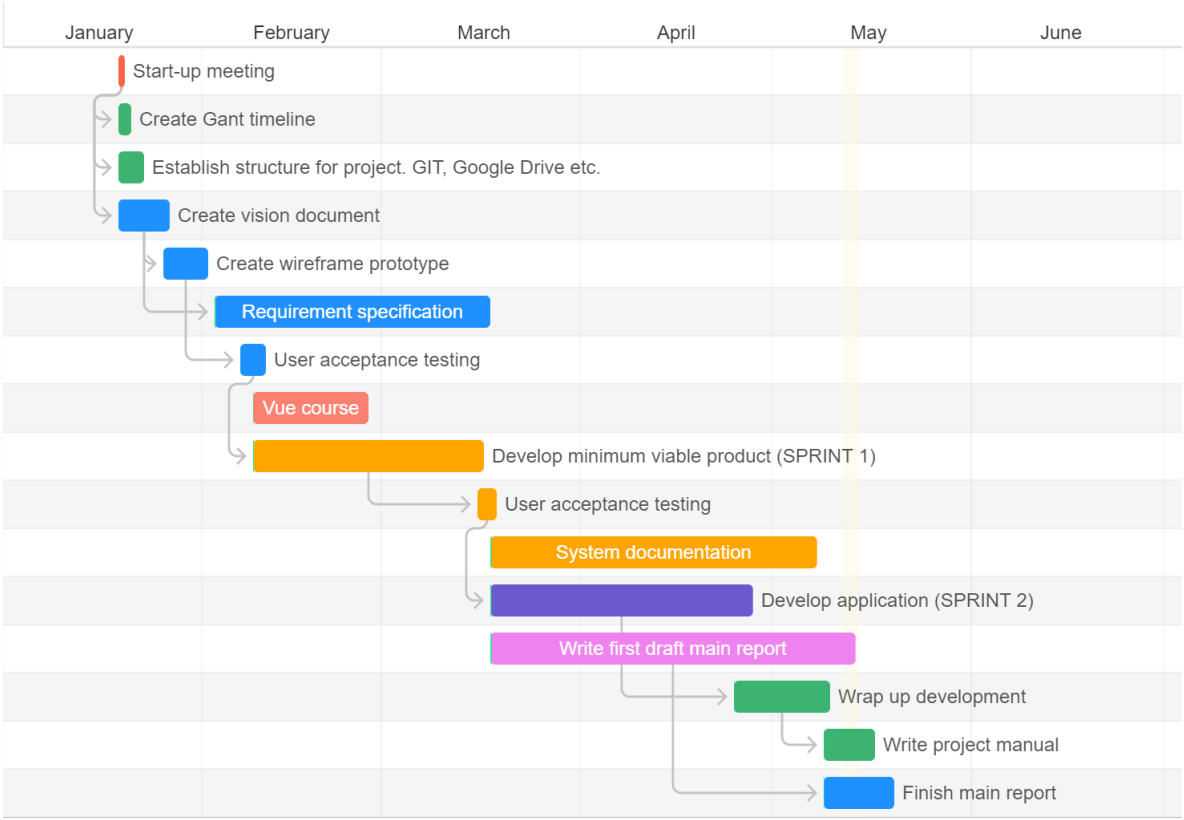


Figure 1: Progress plan

2.2 Time estimations

Activity	Estimation (hours)
Start-up meeting	5
Create Gantt timeline	15
Establish structure for project	15
Create vision document	75
Create wireframe prototypes	35
Requirement specification	85
User acceptance testing	20
Vue course	120
Develop Minimum Viable Product	270
User acceptance testing 2	15
System documentation	75
Develop application	325
Write first draft main report	275
Wrap up development	85
Write project manual	45
Finish main report	40
Sum:	1500

Table 1: Time estimations are combined for all three members of the team.

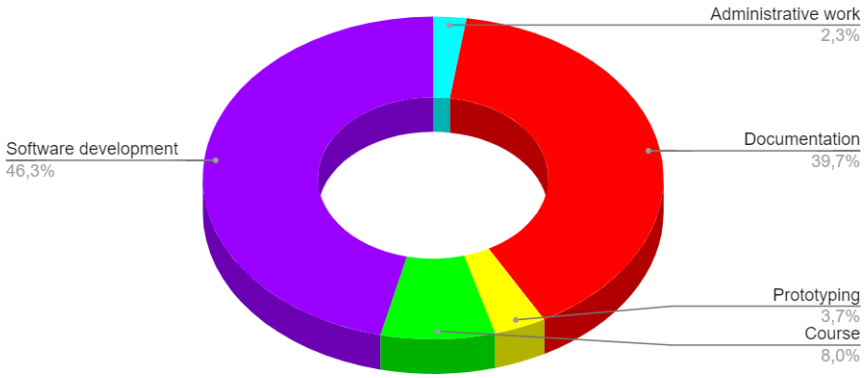


Figure 2: Pie chart showing our estimated work distribution on activities

3 Meetings

At the client's request, the meetings with him have been held in Norwegian. The meetings with our supervisor has been held in English.

3.1 Meeting 1

NTNU

Møteinnkalling
Bacheloroppgave 31

Tid: 11:00-12:30
Dato: 19.01.2021
Sted: Rom 201, Kjøpmannsgt 65

Invitert:

1. Arne Pukstad Juliussen (Oppdragsgiver)
2. Ali Alsam (Veileder)
3. Svein Jakob Høie
4. Espen Kalleberg
5. Håkon Harnes

---

§01/2021 Innledning
§02/2021 Kommentarer til møteinnkalling
§03/2021 Produktdefinisjon
Hovedpunkter som bør gjennomgåes:

- Teknologivalg
- Design
- Brukerdata
- Bekreftelse av oppmøte v/ scanning av QR-kode
- Prioritering av del A og del B
- Forhindring av juksing med oppmøte
- Annen funksjonalitet

§04/2021 Brukertester
§05/2021 Fremtidig kommunikasjon
§06/2021 Signering av avtale
§07/2021 Eventuelt
§08/2021 Gjennomgang av møtereferat

NTNU

Møtereferat
Bacheloroppgave 31

Tid: 11:00-11:50
Dato: 19.01.2021
Sted: Rom 201, Kjøpmannsgt 65

Deltagere:

1. Arne Pukstad Juliussen (Oppdragsgiver)
2. Ali Alsam (Veileder)
3. Svein Jakob Høie ‡
4. Espen Kalleberg
5. Håkon Harnes

---

§01/2021 Innledning

Møteleder, Svein Jakob Høie, åpnet møtet. Deretter presenterte oppdragsgiver oppgaven.

§02/2021 Kommentarer til møteinnkalling

Møteinnkallingen ble godkjent.

§03/2021 Produktdefinisjon

Teknologivalg:
Oppdragsgiver ønsker web-grensesnitt. PHP og MySQL foretrekkes, men oppdragsgiver presiserer at han er åpen for andre programmeringsspråk for front-end såfremt domeneshop støtter dette. På back-end brukes MySQL-database.

Design:
Det er åpent for eget design, så lenge det er oversiktlig. Oppdragsgiver ønsker at kunden skal kunne laste opp sin egen logo.

Brukerdata:
Det skal lagres navn, mobilnummer, epost-adresse og pin-kode. Pin-koden regnes utifra mobilnummeret.

3.2 Meeting 2

NTNU

Meeting Notice
Bachelor Task 31

Tid: 10:00-11:00
Dato: 29.01.2021
Sted: Digital, Zoom

Invited:

1. Ali Alsam (Supervisor)
2. Svein Jakob Høie
3. Espen Kalleberg
4. Håkon Harnes

---

§01/2021 Introduction
§02/2021 Meeting notice feedback
§03/2021 Vision document feedback
§04/2021 Research question feedback
§05/2021 Other matters

Please contact us at our student email if you are unable to attend.

Sincerely,

Svein Jakob Høie
Espen Kalleberg
Håkon Harnes

Trondheim, 25.03.2021

NTNU

Meeting Summary
Bachelor Task 31

Tid: 10:00-10:45
Dato: 29.01.2021
Sted: Digital, Zoom

Invited:

1. Ali Alsam (Supervisor)
2. Svein Jakob Høie
3. Espen Kalleberg
4. Håkon Harnes

---

§01/2021 Introduction

Meeting leader, Håkon Harnes opened the meeting. He started with presenting different research questions the team thought was interesting.

§02/2021 Meeting notice feedback

The meeting notice was approved.

§03/2021 Vision document feedback

No further comments were made concerning the vision document.

§04/2021 Research question feedback

The questions the team had, were very similar, and hard to measure.

Ali was focused on the fact that we had to find relevant litterature on relevant research. The team needs to define a research question that easily can be measured, and ensure that the process in measuring it is correct and predictable.

One approach is to involve the user. For example, invite 2-3 students and interview them about their thoughts, what they would want in a system like this.

§05/2021 Other matters

Regarding programming language, one way is to ask experienced developers that know multiple

3.3 Meeting 3

NTNU

Meeting Notice
Bachelor Task 31

Time: 11:00-12:00
Date: 03.02.2021
Place: Digital, Zoom

Invited:

1. Ali Alsam (Supervisor)
2. Svein Jakob Høie
3. Espen Kalleberg
4. Håkon Harnes

---

§01/2021 Introduction
§02/2021 Research question
§03/2021 Other matters

Please contact us at our student email if you are unable to attend.

Sincerely,

Svein Jakob Høie
Espen Kalleberg
Håkon Harnes

Trondheim, 01.02.2021

NTNU

Meeting Summary
Bachelor Task 31

Tid: 11:00-11:50
Dato: 03.02.2021
Sted: Digital, Zoom

Attendees:

1. Ali Alsam (Supervisor)
2. Svein Jakob Høie
3. Espen Kalleberg
4. Håkon Harnes

---

§01/2021 Introduction

Svein Jakob Høie opens the meeting.

§02/2021 Research question

The supervisor says that all proposed categories are interesting and relevant.

Technology:
The suggestion of learning new programming language is not relevant. The supervisor says the research question needs to be broader. The supervisor suggests that we instead investigate how and why a specific programming language is chosen for a project.

User:
Design is interessenting. Suggestion 2 and 3 are the most relevant.

Security:
This is an interesting topic, with many research questions. The supervisor suggests we do a security evaluation with the task giver (Arne Pukstad).

§03/2021 Other matters

There were no other matters.

3.4 Meeting 4

NTNU

Møteinnkalling
Bacheloroppgave 31

Tid: 11:00-12:00
Dato: 09.02.2021
Sted: Rom 201, Kjøpmannsgt 65

Invitert:

1. Arne Pukstad Juliussen (Oppdragsgiver)
2. Svein Jakob Høie
3. Espen Kalleberg
4. Håkon Harnes

---

§9/2021 Innledning
§10/2021 Kommentarer til møteinnkalling
§11/2021 Sikkerhet

1. Behovet
2. Algoritmen
3. Autentisering

§12/2021 Design
§13/2021 Funksjonalitet
§14/2021 Eventuelt

Ta kontakt med undertegnede på epost dersom du ikke har anledning til å komme.

Med vennlig hilsen,

Svein Jakob Høie
Espen Kalleberg
Håkon Harnes

Trondheim, 04.02.2021

NTNU

Møtereferat
Bacheloroppgave 31

Tid: 11:00-11:45
Dato: 09.02.2021
Sted: Rom 201, Kjøpmannsgt 65

Deltagere:

1. Arne Pukstad Juliussen (Oppdragsgiver)
2. Svein Jakob Høie
3. Espen Kalleberg
4. Håkon Harnes

---

§9/2021 Innledning

Møtelederen, Håkon Harnes, åpnet møtet ved å presentere dagens agenda.

§10/2021 Kommentarer til møteinnkalling

Møteinnkallingen ble godkjent.

§11/2021 Sikkerhet

1. Behovet
   Håkon presenterte sikkerhetsrisikoer ved systemet. Oppdragsgiver ønsket i utgangspunktet lav sikkerhet, men ønsker etter dette høyere sikkerhet.

2. Algoritmen
   Håkon viste hvordan den foreslåtte algoritmen for å regne ut PIN koder kan bli utnyttet. Oppdragsgiveren var enig i at PIN koder ikke være egnet til autentisering.

3. Autentisering
   Håkon foreslo autentisering med email og passord, framfor telefonnummer og PIN kode. Oppdragsgiveren var enig, og foreslo høyere sikkerhet for admin-brukere.

3.5 Meeting 5

NTNU

Meeting Notice
Bachelor Task 31

Time: 11:00-12:00
Date: 10.02.2021
Place: Digital, Zoom

Invited:

1. Ali Alsam (Supervisor)
2. Svein Jakob Høie
3. Espen Kalleberg
4. Håkon Harnes

---

§01/2021 Introduction
§02/2021 Meeting notice feedback
§03/2021 Research question
§04/2021 Other matters

Please contact us at our student email if you are unable to attend.

Sincerely,

Svein Jakob Høie
Espen Kalleberg
Håkon Harnes

Trondheim, 08.02.2021

NTNU

Meeting Summary
Bachelor Task 31

Time: 11:00-11:25
Date: 10.02.2021
Place: Digital, Zoom

Attended:

1. Ali Alsam (Supervisor)
2. Svein Jakob Høie
3. Espen Kalleberg
4. Håkon Harnes

---

§01/2021 Introduction

Meeting leader, Svein Jakob opened the meeting. He started with explaining the different research articles the team had read through and found relevant and interesting.

§02/2021 Meeting notice feedback

The meeting notice was approved.

§03/2021 Research question

The team explained the different research articles they had read. Ali thought that the articles and their content was interesting and highly relevant.

It was agreed that the research question at hand would have to come naturally as a part of the research and topic the team would choose. As long as the research question and the articles were relevant to each other, the topic could be widely chosen from numerous topics.

§04/2021 Other matters

Ali wanted the team to start writing on the main report in order to get properly started and to have a concise perception of the work ahead.

The meeting ended at 11:25

Svein Jakob Høie Trondheim, 10.02.2021

3.6 Meeting 6

NTNU

Meeting Notice
Bachelor Task 31

Time: 11:00-12:00
Date: 10.03.2021
Place: Digital, Zoom

Invited:

1. Ali Alsam (Supervisor)
2. Svein Jakob Høie
3. Espen Kalleberg
4. Håkon Harnes

---

§01/2021 Introduction
§02/2021 Meeting notice feedback
§03/2021 Research question update
§04/2021 Other matters

Please contact us at our student email if you are unable to attend.

Sincerely,

Svein Jakob Høie
Espen Kalleberg
Håkon Harnes

Trondheim, 08.03.2021

NTNU

Meeting Summary
Bachelor Task 31

Time: 11:00-11.40
Date: 10.03.2021
Place: Digital, Zoom

Attended:

1. Ali Alsam (Supervisor)
2. Svein Jakob Høie
3. Espen Kalleberg
4. Håkon Harnes

---

§01/2021 Introduction

Meeting leader, Håkon Harnes, opened the meeting with explaining how the work with the bachelor assignment is going.

§02/2021 Meeting notice feedback

The meeting notice was approved.

§03/2021 Research question update

The team had found several subtopics in relation to the main topic of security in a web-application. Examples of these subtopics were passwords, two-factor authentication and different layers of security based on permissions.

§04/2021 Other matters

Ali specified that it was important for the team to not get tunnel vision. He wanted the team to be open-minded about the task and explore different angles that would arise during development and work.

The meeting ended at 11:40.

Svein Jakob Høie Trondheim, 10.03.2021

3.7 Meeting 7

NTNU

Meeting Notice
Bachelor Task 31

Time: 11:00-12:00
Date: 07.04.2021
Place: Room 201, Kjøpmannsgt 65

Invited:

1. Ali Alsam (Supervisor)
2. Svein Jakob Høie
3. Espen Kalleberg
4. Håkon Harnes

---

§01/2021 Introduction
§02/2021 Bachelor thesis feedback
§03/2021 Research question update
§04/2021 Other matters

Please contact us at our student email if you are unable to attend.

Sincerely,

Svein Jakob Høie
Espen Kalleberg
Håkon Harnes

Trondheim, 05.04.2021

NTNU

Meeting Summary
Bachelor Task 31

Time: 11:00-12:00
Date: 07.04.2021
Place: Digital, Zoom

Attended:

1. Ali Alsam (Supervisor)
2. Svein Jakob Høie
3. Espen Kalleberg
4. Håkon Harnes

---

§01/2021 Introduction

Meeting leader, Svein Jakob opened the meeting.

§02/2021 Meeting notice feedback

The meeting notice was approved.

§03/2021 Research question update

The team introduced the updated proposals for the research questions through a small presentation. These largely dealt with the topic of "authentication". The team and Ali agreed that the questions were good but there should also be included a question that covers a wider range of what the prosject contains. It was decided to add an research question that concerned the actual development of the application. This question can be concretized further at a later date. Directions that were discussed included 'Ease of use' and 'Development of software using only digital aids'.

§04/2021 Other matters

There were no other matters at this meeting.

The meeting finished at 12:00.

Espen Kalleberg Trondheim, 07.04.2021

3.8 Meeting 8

NTNU

Meeting Notice
Bachelor Task 31

Time: 11:00-12:00
Date: 14.04.2021
Place: Digital, Zoom

Invited:

1. Ali Alsam (Supervisor)
2. Svein Jakob Høie
3. Espen Kalleberg
4. Håkon Harnes

---

§01/2021 Introduction
§02/2021 Meeting notice feedback
§03/2021 Bachelor thesis feedback
§04/2021 Other matters

Please contact us at our student email if you are unable to attend.

Sincerely,

Svein Jakob Høie
Espen Kalleberg
Håkon Harnes

Trondheim, 14.05.2021

NTNU

Meeting Summary
Bachelor Task 31

Time: 11:00-12:00
Date: 14.04.2021
Place: Digital, Zoom

Attended:

1. Ali Alsam (Supervisor)
2. Svein Jakob Høie
3. Espen Kalleberg
4. Håkon Harnes

---

§01/2021 Introduction

Meeting leader, Svein Jakob opened the meeting.

§02/2021 Meeting notice feedback

The meeting notice was approved.

§03/2021 Bachelor thesis feedback

The research questions now cover a good range of the project. The wording can still be changed slightly but the area is defined. Ali has not had time to go through the task before the meeting so this is reviewed together in the meeting. The feedback from Ali is that the report looks good and the team will continue to work in the same way as now. The division of work in relation to writing the report and working with the application is discussed. There is talk that a deadline must be set for the first draft of the main report. Remaining time is used to review the language use and the importance of clarifying so that everything is completely clear and accurate.

§04/2021 Other matters

It is mentioned in conclusion of the meeting that the team should begin to get an overview of how the task and the main report will be assessed.

The meeting finished at 12:00.

Espen Kalleberg Trondheim, 14.04.2021

3.9 Meeting 9

NTNU

Møteinnkalling
Bacheloroppgave 31

Tid: 11:00-12:00
Dato: 20.04.2021
Sted: Rom 201, Kjøpmannsgt 65

Invitert:

1. Arne Pukstad Juliussen (Oppdragsgiver)
2. Svein Jakob Høie
3. Espen Kalleberg
4. Håkon Harnes

---

§01/2021 Introduksjon
§02/2021 Fremvisning av foreløpig arbeid gjort på applikasjon
§03/2021 Gjennomgå planen for fremtidig arbeid
§04/2021 Tilbakemelding
§05/2021 Eventuelt

Ta kontakt med undertegnede på epost dersom du ikke har anledning til å komme.

Med vennlig hilsen,

Svein Jakob Høie
Espen Kalleberg
Håkon Harnes

Trondheim, 14.05.2021

NTNU

Møtereferat
Bacheloroppgave 31

Tid: 11:00-12:00
Dato: 13.04.2021
Sted: Rom 201, Kjøpmannsgt 65

Deltagere:

1. Arne Juliussen Pukstad (Oppdragsgiver)
2. Svein Jakob Høie
3. Espen Kalleberg
4. Håkon Harnes

---

§01/2021 Introduksjon

Svein Jakob starter møtet.

§02/2021 Fremvisning av foreløpig arbeid gjort på applikasjon

Teamet viser hva som er gjort så langt. Det er hovedsaklig frontend som vises fram samt forklaring av hva som ligger klart til å implementeres fra backend.

§03/2021 Gjennomgå planen for fremtidig arbeid

Dette omhandlet mye hva som må implementeres til frontend ettersom det meste er gjort og ligger klart i backend.

§04/2021 Tilbakemelding

Arne kommer med gode tilbakemeldinger til designet og til funksjonalitet. Det kommer fram at CSV-importeringen bør gi tilbakemelding på antall rekkerregistert og at filterings alternativet bbygningerburde omskrives til "område/etasje/avdeling".

§05/2021 Eventuelt

Det ble snakket noe om mulighetene for å fortsette arbeidet etter endt bachelor enten av teamet eller nye utviklere.

3.10 Meeting 10

NTNU

Meeting Notice
Bachelor Task 31

Time: 11:00-12:00
Date: 22.04.2021
Place: Room 201, Kjøpmannsgt 65

Invited:

1. Ali Alsam (Supervisor)
2. Svein Jakob Høie
3. Espen Kalleberg
4. Håkon Harnes

---

§01/2021 Introduction
§02/2021 Bachelor thesis feedback
§03/2021 Other matters

Please contact us at our student email if you are unable to attend.

Sincerely,

Svein Jakob Høie
Espen Kalleberg
Håkon Harnes

Trondheim, 18.04.2021

NTNU

Meeting Summary
Bachelor Task 31

Time: 11:00-12:00
Date: 22.04.2021
Place: Digital, Zoom

Attended:

1. Ali Alsam (Supervisor)
2. Svein Jakob Høie
3. Espen Kalleberg
4. Håkon Harnes

---

§01/2021 Introduction

Meeting leader, Svein Jakob opened the meeting.

§02/2021 Meeting notice feedback

The meeting notice was approved.

§03/2021 Bachelor thesis feedback

Ali's feedback on the first draft was that we should focus more on writing less text that has more information in a compact way. It involves cutting sentences like "in modern day technology"and "in this subchapteretc when this is obvious. It must also be written timelessly. During the pandemic"is also written as "COVID-19 pandemic".

Otherwise, research question 1 is good. Question 2 is still a bit general but this can be fixed in due course.

§04/2021 Other matters

There where no other matters.

The meeting finished at 12:00.

Espen Kalleberg Trondheim, 22.04.2021

3.11 Meeting 11

NTNU

Meeting Notice
Bachelor Task 31

Time: 11:00-12:00
Date: 04.05.2021
Place: Room 201, Kjøpmannsgt 65

Invited:

1. Ali Alsam (Supervisor)
2. Svein Jakob Høie
3. Espen Kalleberg
4. Håkon Harnes

---

§01/2021 Introduction
§02/2021 Bachelor thesis feedback
§03/2021 Other matters

Please contact us at our student email if you are unable to attend.

Sincerely,

Svein Jakob Høie
Espen Kalleberg
Håkon Harnes

Trondheim, 04.05.2021

NTNU

Meeting Summary
Bachelor Task 31

Time: 11:00-12:00
Date: 04.05.2021
Place: Digital, Zoom

Attended:

1. Ali Alsam (Supervisor)
2. Svein Jakob Høie
3. Espen Kalleberg
4. Håkon Harnes

---

§01/2021 Introduction

Meeting leader, Svein Jakob opened the meeting.

§02/2021 Meeting notice feedback

The meeting notice was approved.

§03/2021 Bachelor thesis feedback

This meeting was largely about discussing research question 2 as there was nothing significant new about the content.

In the second question, it was concluded that the last part is not appropriate for a research question. This is reformulated into something more like "What methods are appropriate in a system like smartbooking?"

This change also gives us a different view of the method chapter of the second question. The method will probably be a literature review. Maybe dialogue if possible. It should be mentioned that the client wanted a pin code. This must be done to shed light on the fact that when you think too fast, you may overlook important aspects of safety.

§04/2021 Other matters

There where no other matters.

3.12 Meeting 12

NTNU

Meeting Notice
Bachelor Task 31

Time: 11:00-12:00
Date: 11.05.2021
Place: Room 201, Kjøpmannsgt 65

Invited:

1. Ali Alsam (Supervisor)
2. Svein Jakob Høie
3. Espen Kalleberg
4. Håkon Harnes

---

§01/2021 Introduction
§02/2021 Bachelor thesis feedback
§03/2021 Other matters

Please contact us at our student email if you are unable to attend.

Sincerely,

Svein Jakob Høie
Espen Kalleberg
Håkon Harnes

Trondheim, 11.05.2021

NTNU

Meeting Summary
Bachelor Task 31

Time: 10:00-11:00
Date: 11.05.2021
Place: Digital, Zoom

Attended:

1. Ali Alsam (Supervisor)
2. Svein Jakob Høie
3. Espen Kalleberg
4. Håkon Harnes

---

§01/2021 Introduction

Meeting leader, Svein Jakob opened the meeting.

§02/2021 Meeting notice feedback

The meeting notice was approved.

§03/2021 Bachelor thesis feedback

All the content is now in the report and it is written quite well. The meeting is used to get feedback on parts of the report that present the content. The problem and the reason for this project must be presented in a simple way without too much extra information.

§04/2021 Other matters

It is agreed to have an extra meeting on 18 May 09:00 when the report is considered complete. Then small things will be reviewed and we will weed out linguistic errors etc.

The meeting finished at 11:10.

Espen Kalleberg Trondheim, 11.05.2021

4 Time Sheets

4.1 Team

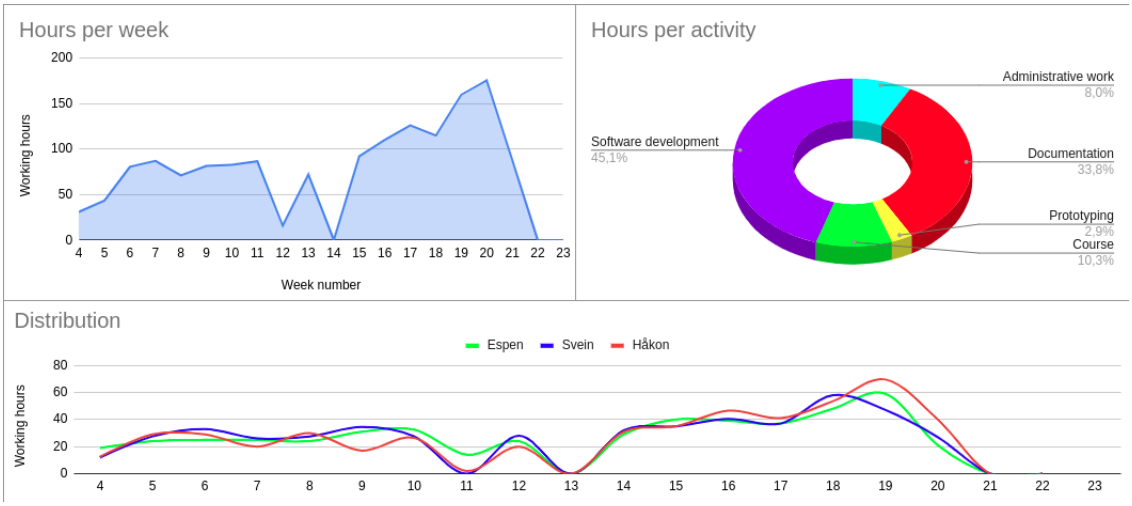


Figure 3: Working hours statistics

Category	Team working hours per activity
Administrative work	120,5
Documentation	511
Prototyping	43,5
Course	155
Software development	681

Week number	**Working hours per week**
4	31
5	43,5
6	80,5
7	87
8	71
9	81,5
10	82,5
11	86,5
12	16
13	72
14	0
15	92
16	110
17	126
18	115
19	159,5
20	175,5
21	88
Sum of working hours:	1517,5

4.2 Espen Kalleberg

4.2.1 Time sheet

Beskrivelse	Dato	Uke	Fra	Til	Timer
Forberedelse til første møte	18.01.2021	4	kl. 10.00	kl. 12.00	2
Oppstartmøte	19.01.2021	4	kl. 11.00	kl. 12.00	1
Gant Diagram	19.01.2021	4	kl. 13.00	kl. 15.00	2
Visjonsdokument	22.01.2021	4	kl. 12.00	kl. 15.00	3
Visjonsdokument	25.01.2021	5	kl. 10.00	kl. 12.00	2
Wireframes	25.01.2021	5	kl. 10.00	kl. 15.30	5,5
Wireframes	27.01.2021	5	kl. 10.00	kl. 16.00	6
Wireframes	28.01.2021	5	kl. 10.00	kl. 14.00	4
Veiledning/diskusjon	29.01.2021	5	kl. 10.00	kl. 11.30	1,5
Jobbet med problemstilling	01.02.2021	6	kl. 10.00	kl. 17.00	7
Forberedelse til møte med oppdragsgiver	02.02.2021	6	kl. 10.00	kl. 15.00	5
Veiledningsmøte + etterarbeid	03.02.2021	6	kl. 11.00	kl. 13.00	2
Forberedelse til møte med oppdragsgiver	04.02.2021	6	kl. 10.00	kl. 14.00	4
Jobbet med problemstilling	01.02.2021	6	kl. 09.00	kl. 10.00	1
Jobbet med problemstilling + Vue.js	01.02.2021	6	kl. 11.00	kl. 16.00	5
Møte + slides	09.02.2021	7	kl. 11.00	kl. 15.00	4
Vue	10.02.2021	7	kl. 09.00	kl. 10.00	1
Workshop og forebredelser	10.02.2021	7	kl. 11.00	kl. 14.30	3,5
Vue	10.02.2021	7	kl. 14.30	kl. 16.00	1,5
Vue	11.02.2021	7	kl. 08.00	kl. 10.00	2
Vue	11.02.2021	7	kl. 12.00	kl. 16.00	4
Vue	11.02.2021	7	kl. 09.30	kl. 14.30	5
Vue	12.02.2021	7	kl. 12.00	kl. 16.00	4
Vue	20.02.2021	8	kl. 12.00	kl. 16.00	4
Vue	19.02.2021	8	kl. 09.00	kl. 16.00	7
Vue	18.02.2021	8	kl. 09.00	kl. 16.00	7
Vue	17.02.2021	8	kl. 09.00	kl. 16.00	7
Databasetabeller, endepunkt, routes	22.02.2021	9	kl. 09.00	kl. 15.00	6

Beskrivelse	Dato	Uke	Fra	Til	Timer
Databasetabeller, endepunkt, routes	23.02.2021	9	kl. 09.00	kl. 15.00	6
Databasetabeller, endepunkt, routes	24.02.2021	9	kl. 09.00	kl. 16.00	7
Databasetabeller, endepunkt, routes	25.02.2021	9	kl. 10.00	kl. 15.00	5
Backend	01.03.2021	10	kl. 10.00	kl. 15.00	5
Backend	02.03.2021	10	kl. 09.00	kl. 16.00	7
Backend	03.03.2021	10	kl. 09.00	kl. 14.00	5
Backend	04.03.2021	10	kl. 09.00	kl. 16.00	7
controller backend	05.03.2021	10	kl. 09.00	kl. 16.00	7
controller backend	08.03.2021	11	kl. 09.00	kl. 16.00	7
controller backend	09.03.2021	11	kl. 09.00	kl. 15.00	6
Node kurs	10.03.2021	11	kl. 09.00	kl. 15.30	6,5
error handling	11.03.2021	11	kl. 09.00	kl. 15.30	6,5
error handling	12.03.2021	11	kl. 09.00	kl. 15.30	6,5
error handling	19.03.2021	12	kl. 09.00	kl. 16.00	7
service	19.03.2021	12	kl. 09.00	kl. 16.00	7
service	22.03.2021	13	kl. 09.00	kl. 16.00	7
service. Forarbeid rapport	23.03.2021	13	kl. 09.00	kl. 16.00	7
service. Rapport	24.03.2021	13	kl. 09.00	kl. 16.00	7
Rapport	25.03.2021	13	kl. 10.00	kl. 13.00	3
Rapport	06.04.2021	15	kl. 10.00	kl. 17.00	7
Rapport	07.04.2021	15	kl. 09.00	kl. 17.00	8
Rapport	08.04.2021	15	kl. 10.00	kl. 17.00	7
Rapport	09.04.2021	15	kl. 10.00	kl. 17.00	7
Backend	12.04.2021	16	kl. 09.00	kl. 17.00	8
Backend	13.04.2021	16	kl. 09.00	kl. 17.00	8
Backend	14.04.2021	16	kl. 09.00	kl. 17.00	8
Rapport	15.04.2021	16	kl. 09.00	kl. 17.00	8
Rapport	16.04.2021	16	kl. 09.00	kl. 17.00	8
Backend	19.04.2021	17	kl. 09.00	kl. 17.00	8
Backend	20.04.2021	17	kl. 09.00	kl. 17.00	8
Backend	21.04.2021	17	kl. 09.00	kl. 17.00	8

Beskrivelse	Dato	Uke	Fra	Til	Timer
Rapport	22.04.2021	17	kl. 09.00	kl. 17.00	8
Rapport	23.04.2021	17	kl. 10.00	kl. 17.00	7
Backend	26.04.2021	18	kl. 10.00	kl. 17.00	7
Backend	27.04.2021	18	kl. 09.00	kl. 17.00	8
Backend	28.04.2021	18	kl. 09.00	kl. 17.00	8
Rapport	29.04.2021	18	kl. 10.00	kl. 17.00	7
Rapport	30.04.2021	18	kl. 10.00	kl. 17.00	7
Backend	03.05.2021	19	kl. 09.00	kl. 17.00	8
Backend	04.05.2021	19	kl. 09.00	kl. 17.00	8
Backend	05.05.2021	19	kl. 09.00	kl. 17.00	8
rapport	06.05.2021	19	kl. 09.00	kl. 17.00	8
rapport	07.05.2021	19	kl. 09.00	kl. 17.00	8
rapport	08.05.2021	19	kl. 09.00	kl. 17.00	8
rapport	09.05.2021	20	kl. 09.00	kl. 17.00	8
rapport	10.05.2021	20	kl. 09.00	kl. 17.00	8
rapport	11.05.2021	20	kl. 09.00	kl. 17.00	8
rapport	12.05.2021	20	kl. 09.00	kl. 21.00	12
rapport	13.05.2021	20	kl. 09.00	kl. 16.00	7
rapport	14.05.2021	20	kl. 09.00	kl. 17.00	8
rapport	15.05.2021	20	kl. 09.00	kl. 17.00	8
rapport	16.05.2021	21	kl. 09.00	kl. 16.00	7
rapport	18.05.2021	21	kl. 09.00	kl. 16.00	7
rapport	19.05.2021	21	kl. 09.00	kl. 16.00	7
Sum antall timer:	**499.5**				

4.2.2 Week 4

Uke 4 var oppstartsuken for bacheloren og vi har hatt oppstartsmøte. Syntes det gikk veldig bra og det virker som en grei oppgave. Den er allerede godt gjennomtenkt av Arne (oppdragsgiver) så vi kan komme raskt i gang virker det som. Vi har satt opp et Gantt diagram og startet på visjonsdokumentet.

4.2.3 Week 5

Vi har gjort ferdig visjonsdokumentet tidlig denne uken. Resten av uken har vi brukt på Wireframes. Vi tenker å bruke disse til testing. De er laget i figma så vi kan enkelt koble disse sammen til en prototype. Vi har også hatt det første veiledningsmøtet for å komme i gang med forslag til problemstilling. Vi viste også fram visjonsdokumentet til Ali.

4.2.4 Week 6

Vi har fortsatt å tenke på mulige problemstillinger til oppgaven. Applikasjonen i seg selv virker ikke å være veldig teknisk vanskelig så vi har god mulighet her til å se på en litt større problemstilling siden vi kommer nok til å få god tid til å jobbe med denne. Noe som vi syntes var litt interessant var at Arne har foreslått at PIN kode skal brukes som passord. Vi skjønte ganske raskt at dette ikke var veldig gunstig så vi må ta dette opp på møtet i neste uke. Dette er en aktuell problemstilling også. Hva slags krav det er til sikkerhet i et system som dette. Vi har også bestemt oss for å bruke Vue.js i dette prosjektet og vi har satt av et par uker til å ta et kurs i dette.

4.2.5 Week 7

Etter møtet med Arne så er det bestemt at vi går bort fra PIN kode løsningen. Vi forstår hvorfor det ble foreslått og det virket ganske praktisk men det blir rett og slett for lav sikkerhet. Selv om systemet er relativt lav terskel så burde det absolutt implementeres noe som kan sikre brukerkontoer bedre. Dersom "hemmeligheten" om hvordan PIN kodene hadde kommet ut så hadde det blitt kaos. Når vi da velger å droppe PIN kode så er det heller ikke nødvendig med telefonnummer. Planen framover er å gå over til en email og passord kombinasjon istedet. Resten av uken har blitt brukt til kurs i Vue.js

4.2.6 Week 8

Hele denne uken har blitt brukt til kurs i Vue.js. Det er et stilig rammeverk og alltid artig å lære noe nytt. Ikke så mye mer å fortelle om denne uka.

4.2.7 Week 9

Denne uka har vi begynt utviklingen. Jeg har stort sett jobbet backend. Har satt opp databasetabellene. Har også begynt på Node.js serveren og satt opp en del routes og endepunkter for de forskjellige tabellene.

4.2.8 Week 10

Har fortsatt med serveren denne uka. Svein Jakob har begynt å sette opp frontend så vi jobber ganske tett sammen for å passe på at vi får sendt rette dataen på rett plass osv.

4.2.9 Week 11

Vi har havnet i ganske naturlig arbeidsfordeling nå der Svein Jakob jobber mest i frontend og jeg er mest i backend. Trives i grunnen godt med det. De fleste endepunktene ser ut til å være på plass nå men du kan være helt sikker på at det vil dukke opp nye etterhvert.

4.2.10 Week 12

Ettersom alle endepunktene tilsynelatende er på plass så er det på tide med litt error handling. Startet uken med å ta et lite kurs i Node og errorhandling. Resten av uken er egentlig satt av til å implementere dette.

4.2.11 Week 13

Har gjort ferdig error handlingen på backend og har nå byttet til frontend repoet. Siden jeg har satt opp endepunktene på serveren var det naturlig at jeg skulle sette opp service klassene på frontend og det har jeg startet på nå.

4.2.12 Week 14

Påskeferie.

4.2.13 Week 15

Denne uken har jeg stort sett brukt til å jobbe med rapporten. Det er allerede en del ting vi kan ta tak i så det er greit å være tidlig ute. Har fått lest endel teori om autentifikasjon og vi er vel alle enige om at er hovedtemaet vårt i denne bacheloren. Vi vil også se litt på hvordan covid-19 restriksjonene har påvirket prosessen. Da vil dette bli "utviklingsdelen" av rapporten.

4.2.14 Week 16

Etter påske så jobber vi fulle arbeidsdager hver dag med oppgaven. Vi skal prøve å til beste evne jobbe 50/50 med rapporten og utviklingen de neste ukene. Vi må se litt hvordan utviklingen går. Siden vi må ha applikasjonen ferdig i god tid før rapporten så kan det være

at arbeidsfordelingen blir litt mer 60/40 en periode med favør utvikling. Denne uken jobbet jeg mandag til onsdag på backend. Torsdag og fredag brukte jeg på hovedrapporten

4.2.15 Week 17

Denne uken gikk som planlagt og omtrent som forrige uke. Mandag til onsdag ble brukt til å jobbe i backend. Det er fremdeles noen mangler så tror 60/40 fordelingen vil fortsette neste uke også. Torsdag og fredag brukte jeg på rapporten. Der ferdigstilte jeg teorien delen

4.2.16 Week 18

Denne uken har jeg jobbet med backend mandag til onsdag. Dersom vi hadde hatt fokus på kun utviklingen denne uken kunne vi nok kommet ganske nærme mål. Vi valgte derimot å fortsette fordelingen mellom utvikling og rapporten. Har fått skrevet en god del i metodekapittelet.

4.2.17 Week 19

Nå er backend blitt såpass nære ferdig at jeg ikke lengre trenger å dedikere en hel arbeidsdag til å jobbe med utvikling. Det blir nok litt jobbing av og til når ting dukker opp men hovedfokuset mitt framover vil være på hovedrapporten. Etter denne uken så anser jeg metodedelen og diskusjonsdelen som omtrent ferdig.

4.2.18 Week 20

Denne uken har jeg stort sett brukt på rapporten. Utviklingen er så og si ferdig utenom et par bugs som vi fikser underveis. Rapporten er nesten ferdig. Er noen vedlegg som må legges til og det kan ikke gjøres før vi setter et endelig slutttidpunkt for utviklingen. Vi vil også avtale et siste møte med Ali for å se gjennom alt før innlevering.

4.2.19 Week 21

Innlevering på torsdag. Det siste dagene før innlevering har det vært endel jobb med å ferdigstille dokumentasjon og vedleggene. Det meste var på plass så det ble endel gjennomlesning og fiksing av småting.

4.3 Håkon Harnes

4.3.1 Time sheet

Beskrivelse	Dato	Uke	Fra	Til	Timer
Forberedelse til første møte	18.01.2021	4	kl. 10.00.00	kl. 12.30.00	2,5
Møte + etterarbeid	19.01.2021	4	kl. 11.00.00	kl. 15.00.00	4
Avdekking av sikkerhetsfeil i algoritme	21.01.2021	4	kl. 22.00.00	kl. 23.00.00	1
Visjonsdokument	22.01.2021	4	kl. 12.00.00	kl. 15.00.00	3
Visjonsdokument + Wireframe	25.01.2021	5	kl. 10.30.00	kl. 15.30.00	5
Visjonsdokument + Møteinnkallelse + Wireframe	27.01.2021	5	kl. 12.00.00	kl. 16.00.00	4
Logo	28.01.2021	5	kl. 12.00.00	kl. 14.00.00	2
Veiledningsmøte + etterarbeid	29.01.2021	5	kl. 10.00.00	kl. 11.30.00	1,5
Problemstilling	01.02.2021	6	kl. 09.00.00	kl. 15.00.00	6
Forberedelse til møte med oppdragsgiver	02.02.2021	6	kl. 10.00.00	kl. 15.00.00	5
Vue-kurs	02.02.2021	6	kl. 16.00.00	kl. 20.00.00	4
Veiledningsmøte + etterarbeid	03.02.2021	6	kl. 11.00.00	kl. 13.00.00	2
Vue-kurs	03.02.2021	6	kl. 16.00.00	kl. 19.00.00	3
Forberedelse til møte med oppdragsgiver	04.02.2021	6	kl. 09.00.00	kl. 14.00.00	5
Vue-kurs	04.02.2021	6	kl. 16.00.00	kl. 20.00.00	4
Forberedelse til møte med oppdragsgiver + Vue-kurs	08.02.2021	7	kl. 10.00.00	kl. 18.00.00	8
Møte med oppdragsgiver	09.02.2021	7	kl. 11.00.00	kl. 12.00.00	1
Vue-kurs	09.02.2021	7	kl. 18.00.00	kl. 20.00.00	2
Møte med veileder + workshop	10.02.2021	7	kl. 11.00.00	kl. 14.00.00	3
Vue-kurs	10.02.2021	7	kl. 15.00.00	kl. 18.00.00	3
Vue-kurs	11.02.2021	7	kl. 16.00.00	kl. 20.00.00	4
Vue-kurs	12.02.2021	7	kl. 10.00.00	kl. 18.00.00	8
Vue-kurs	17.02.2021	8	kl. 18.00.00	kl. 20.00.00	2
Vue-kurs	18.02.2021	8	kl. 14.00.00	kl. 18.00.00	4
Vue-kurs	19.02.2021	8	kl. 10.00.00	kl. 16.00.00	6
Vue-kurs	20.02.2021	8	kl. 10.00.00	kl. 18.00.00	8

Beskrivelse	Dato	Uke	Fra	Til	Timer
Vue-kurs	21.02.2021	9	kl. 12.00.00	kl. 18.00.00	6
Planlegging	22.02.2021	9	kl. 09.00.00	kl. 15.00.00	6
Frontend, navigeringselementer	23.02.2021	9	kl. 09.00.00	kl. 18.00.00	9
Frotnend, service klasse + oppsett backend	24.02.2021	9	kl. 09.00.00	kl. 14.00.00	5
Backend	25.02.2021	9	kl. 10.00.00	kl. 14.00.00	4
Backend	01.03.2021	10	kl. 11.00.00	kl. 14.00.00	3
Backend	02.03.2021	10	kl. 10.00.00	kl. 15.00.00	5
Backend	04.03.2021	10	kl. 10.00.00	kl. 15.00.00	5
Backend	05.03.2021	10	kl. 10.00.00	kl. 14.00.00	4
Backend	08.03.2021	11	kl. 10.00.00	kl. 14.00.00	4
Backend	09.03.2021	11	kl. 09.00.00	kl. 14.00.00	5
Backend	10.03.2021	11	kl. 09.30.00	kl. 15.30.00	6
Node-kurs + Backend	11.03.2021	11	kl. 11.00.00	kl. 16.30.00	5,5
Backend	12.03.2021	11	kl. 10.00.00	kl. 16.00.00	6
Backend	19.03.2021	12	kl. 10.00.00	kl. 12.00.00	2
Backend	22.03.2021	13	kl. 09.00.00	kl. 16.00.00	7
Backend	23.03.2021	13	kl. 10.00.00	kl. 17.00.00	7
Backend	24.03.2021	13	kl. 10.00.00	kl. 13.00.00	3
Rapportskriving	25.03.2021	13	kl. 10.00.00	kl. 13.00.00	3
Rapportskriving	06.04.2021	15	kl. 10.00.00	kl. 14.00.00	4
Rapportskriving	06.04.2021	15	kl. 16.00.00	kl. 18.00.00	2
Rapportskriving + Veiledningsmøte	07.04.2021	15	kl. 08.00.00	kl. 17.00.00	9
Rapportskriving	08.04.2021	15	kl. 09.00.00	kl. 15.00.00	6
Rapportskriving	09.04.2021	15	kl. 10.00.00	kl. 20.00.00	10
Rapportskriving	12.04.2021	16	kl. 10.00.00	kl. 16.00.00	6
Rapportskriving	13.04.2021	16	kl. 09.00.00	kl. 16.00.00	7
Backend + Frontend, Autentisering	14.04.2021	16	kl. 09.00.00	kl. 16.00.00	7
Backend + Frontend, Autentisering	15.04.2021	16	kl. 09.00.00	kl. 16.00.00	7
Backend + Frontend, Autentisering	16.04.2021	16	kl. 10.00.00	kl. 15.00.00	5
Backend + Frontend, Autentisering	17.04.2021	16	kl. 15.00.00	kl. 18.00.00	3

Beskrivelse	Dato	Uke	Fra	Til	Timer
Backend + Frontend, Autentisering	19.04.2021	17	kl. 10.00.00	kl. 23.30.00	13,5
Backend + Frontend, Autentisering	20.04.2021	17	kl. 08.00.00	kl. 14.00.00	6
Backend + Frontend, Autentisering	20.04.2021	17	kl. 18.00.00	kl. 20.00.00	2
Rapportskriving	21.04.2021	17	kl. 11.00.00	kl. 18.00.00	7
Frotnend, bekreftelseskode	21.04.2021	17	kl. 19.00.00	kl. 20.00.00	1
Rapportskriving	22.04.2021	17	kl. 10.30.00	kl. 15.00.00	4,5
Rapportskriving	22.04.2021	17	kl. 18.30.00	kl. 20.00.00	1,5
Rapportskriving + veiledningsmøte	23.04.2021	17	kl. 10.00.00	kl. 15.00.00	5
Backend, teori refresh tokens	24.04.2021	17	kl. 12.00.00	kl. 18.00.00	6
Backend, Autentisering redo med refresh tokens	26.04.2021	18	kl. 10.00.00	kl. 21.00.00	11
Backend, Autentisering redo med refresh tokens	27.04.2021	18	kl. 10.00.00	kl. 20.00.00	10
Backend, Autentisering redo med refresh tokens	28.04.2021	18	kl. 10.00.00	kl. 19.00.00	9
Backend, Autentisering + Rapportskriving	29.04.2021	18	kl. 10.00.00	kl. 19.00.00	9
Veiledningsmøte + Rapportskriving	30.04.2021	18	kl. 09.00.00	kl. 11.00.00	2
Rapportskriving	03.05.2021	19	kl. 09.00.00	kl. 15.00.00	6
Implementering av auth på frontend + bugfixes	04.05.2021	19	kl. 10.00.00	kl. 23.30.00	13,5
Rapportskriving	05.05.2021	19	kl. 10.00.00	kl. 16.00.00	6
Rapportskriving	06.05.2021	19	kl. 10.00.00	kl. 20.00.00	10
Rapportskriving	07.05.2021	19	kl. 10.00.00	kl. 20.00.00	10
Rapportskriving	08.05.2021	19	kl. 10.00.00	kl. 18.00.00	8
Rapportskriving	09.05.2021	20	kl. 10.00.00	kl. 20.00.00	10
Rapportskriving	10.05.2021	20	kl. 10.30.00	kl. 20.00.00	9,5
Rapportskriving	11.05.2021	20	kl. 10.00.00	kl. 16.00.00	6
Rapportskriving	12.05.2021	20	kl. 10.00.00	kl. 20.00.00	10
Rapportskriving	13.05.2021	20	kl. 10.00.00	kl. 23.00.00	13
Rapportskriving	14.05.2021	20	kl. 10.00.00	kl. 20.00.00	10
Bugfix	15.05.2021	20	kl. 10.00.00	kl. 21.00.00	11

Beskrivelse	Dato	Uke	Fra	Til	Timer
Bugfixing	16.05.2021	21	kl. 10.00.00	kl. 23.00.00	13
Rapportskriving	17.05.2021	21	kl. 14.00.00	kl. 20.00.00	6
Rapportskriving	18.05.2021	21	kl. 10.00.00	kl. 20.00.00	10
Rapportskriving	19.05.2021	21	kl. 10.00.00	kl. 21.00.00	11
Sum antall timer:	**513**				

4.3.2 Week 4

Hadde veiledningsmøte denne uken. Arbeidsgiver, Arne Pukstad Juliussen, stilte godt forberedt til møte med en overraskende detaljert kravspesifikasjon. Etter møtet starter planleggingsfasen v.h.a. Gantt diagram og visjonsdokument.

4.3.3 Week 5

Førsteutkastet av visjonsdokumentet ble ferdigstilt denne uken. Resten av uken gikk til prototyping i Figma. Brukte en del tid på å bestemme oss for design, og utforme en passende logo i Photoshop. Hadde også første veiledningsmøte med veileder.

4.3.4 Week 6

Denne uken tenkte vi på ulike problemstillinger, og vi leste oss opp på relevant stoff. Det viste seg at den foreslåtte autentiseringsmetoden med PIN-koder ikke er sikker nok. Viste arbeidsgiver hvordan metoden kunne knekkes v.h.a. lineær regresjon, og arbeidsgiver sa seg enig da vi foreslo en annen type autentiseringsmetode. Videre ble vi enige om å bruke Vue.js, og vi begynte på et kurs på Udemy.

4.3.5 Week 7

Denne uken ble brukt til kursing i Vue. Har også hatt et møte med veileder.

4.3.6 Week 8

Denne uken ble brukt til kursing i Vue.js. Vi nærmer oss slutten av kurset og ser fram til å begynne på utviklingen. Brukte også en god del tid på å sette opp dokumenter i LaTex.

4.3.7 Week 9

Denne uken begynte vi på selve utviklingen. Vi satt opp front-end og back-end med tilhørende git repositories. La inn "basekoden" for å kunne fortsette videre med utviklingen.

4.3.8 Week 10

Har brukt mye av denne uken på å lese om ulike autentiseringsmetoder samt ulike implementasjonmuligheter i Node. Vi har bestemt oss for at problemstillingen skal omhandle autentisering, så mye av tiden går til lesing.

4.3.9 Week 11

Har begynt å implementere autentiseringsmetoden. Følger et kurs om node.js på udemy, men der brukes mongoDB, så mye av koden blir annerledes. Konseptene er de samme, og jeg lærer mye.

4.3.10 Week 12

Denne uken har vi hatt eksamen, og det har ikke blitt så mye arbeid med bacheloroppgaven. Leste litt mer litteratur om autentisering, og begynt å skrive litt på rapporten.

4.3.11 Week 13

Denne uken ble den viktigste delene autentiseringsimplementasjonen ferdig. To-faktor autentisering gjenstår fortsatt, og dekkes ikke av kurset jeg følger. Har også funnet ut at vi burde implementere refresh tokens.

4.3.12 Week 14

Påskeferie.

4.3.13 Week 15

Har skrevet en del rapport, men også implementert to-faktor autentisering i systemet. Jeg er ikke helt fornøyd med koden slik den er nå, da jeg har lært mye fra andre kilder enn node kurset, og jeg føler nå autentiseringen burde implementeres på nytt.

4.3.14 Week 16

Denne uken gikk til å starte å implementere autentisering på nytt, med nye strategier og refresh tokens. Implementeres på en slik måte at XSS og CSRF ikke er mulig, slik det var i den gamle implementasjonen. Skrev også en del rapport om hvor tokens burde lagres for å unngå vanlige angrep, samt beskrivelse av noen angrep.

4.3.15 Week 17

Denne uka ble brukt til å ferdigstille autentiseringen, samt skrive en del rapport. Er en del bugs med mail o.l. som må tas tak i neste uke, og engangspassordene fungerer ikke alltid som de skal.

4.3.16 Week 18

Denne uken ble autentiseringsdelen ferdig, med funksjonalitet for både to-faktor autentisering og to-stegs verifisering, samt glemt passord og endre passord funksjonalitet. Resten av uka gikk til å skrive på hovedrapporten og vedlegg.

4.3.17 Week 19

Denne uken ble brukt til å skrive rapport, og fikse småbugs i systemet. Legger merke til at feilhåndteringen kunne vært bedre, noe som burde fikses neste uke.

4.3.18 Week 20

Denne uka ble en god del bugs fikset. I tillegg ble designet på frontend endret en god del, og gjort slik at det er kompatibelt med mobiler. Det var et gjennomgående problem hvor ting fungerte "halvveis" på mobil, men nå fungerer det slik det skal. Skrev også en del på rapporten, teori- og metodedel er så og si ferdig nå, og resultat-, diskusjon og konklusjon er der, men det må poleres.

4.3.19 Week 21

Denne uka ble brukt til å ferdigstille rapporten og annen dokumentasjon. Vi fikset også en del småbugs i systemet, og fikset litt på koden.

4.4 Svein Jakob Høie

4.4.1 Time sheet

Beskrivelse	Dato	Uke	Fra	Til	Timer
Forberedelse til første møte	18.01.2021	4	kl. 09.00	kl. 12.30	3,5
Første møte + dokumentasjon	19.01.2021	4	kl. 09.00	kl. 15.00	6
Visjonsdokument	22.01.2021	4	kl. 12.00	kl. 15.00	3
Visjonsdokument og wireframing	25.01.2021	5	kl. 10.30	kl. 15.30	5
Visjonsdokument og wireframing	27.01.2021	5	kl. 12.00	kl. 16.00	4
Wireframing	28.01.2021	5	kl. 11.00	kl. 14.00	3
Litteraturvalg og artikkelinnhenting	01.02.2021	6	kl. 09.00	kl. 15.00	6
Wireframing	02.02.2021	6	kl. 09.00	kl. 15.00	6
Møte med veileder	03.02.2021	6	kl. 09.00	kl. 13.00	4
Presentasjon og wireframing	04.02.2021	6	kl. 09.00	kl. 14.00	5
Kursing	05.02.2021	6	kl. 08.30	kl. 14.00	5,5
Kursing	06.02.2021	6	kl. 15.00	kl. 16.00	1
Kursing	07.02.2021	7	kl. 20.00	kl. 22.00	2
Wireframing og kursing	08.02.2021	7	kl. 09.00	kl. 16.00	7
Møte med arbeidsgiver	09.02.2021	7	kl. 09.00	kl. 13.00	4
Møte med veileder og workshop	10.02.2021	7	kl. 09.00	kl. 15.30	6,5
Kursing	10.02.2021	7	kl. 13.00	kl. 18.30	5,5
Kursing	10.02.2021	7	kl. 20.00	kl. 22.00	2
Kursing	12.02.2021	7	kl. 09.00	kl. 15.00	6
Kursing	17.02.2021	8	kl. 16.00	kl. 19.00	3
Kursing	18.02.2021	8	kl. 08.00	kl. 16.00	8
Kursing	19.02.2021	8	kl. 09.00	kl. 17.00	8
Kursing	20.02.2021	8	kl. 09.00	kl. 16.00	7
Kursing	21.02.2021	9	kl. 10.00	kl. 15.00	5
Planlegging	22.02.2021	9	kl. 09.00	kl. 15.00	6
Sette opp arbeidsmiljø og koding	24.02.2021	9	kl. 09.00	kl. 15.30	6,5
Styling og koding	25.02.2021	9	kl. 09.30	kl. 14.00	4,5
Jobbing med bruker-siden	26.02.2021	9	kl. 08.30	kl. 14.00	5,5
Sette opp backend	01.03.2021	10	kl. 08.30	kl. 14.30	6
Frontend	02.03.2021	10	kl. 08.30	kl. 17.00	8,5

Beskrivelse	Dato	Uke	Fra	Til	Timer
Frontend	03.03.2021	10	kl. 09.00	kl. 17.00	8
Frontend	04.03.2021	10	kl. 09.00	kl. 14.30	5,5
Frontend	05.03.2021	10	kl. 09.00	kl. 15.30	6,5
Frontend	08.03.2021	11	kl. 09.30	kl. 15.30	6
Frontend (reservasjoner)	09.03.2021	11	kl. 09.00	kl. 15.00	6
Frontend (reservasjoner) + veiledermøte	10.03.2021	11	kl. 09.30	kl. 17.00	7,5
Frontend (reservasjoner)	10.03.2021	11	kl. 09.00	kl. 17.00	8
Frontend (reservere rom)	22.03.2021	13	kl. 09.00	kl. 16.00	7
Frontend (reservere rom)	23.03.2021	13	kl. 09.00	kl. 16.00	7
Frontend (reservere rom)	24.03.2021	13	kl. 09.00	kl. 17.00	8
Ferdigstille dokumentasjon av møter	25.03.2021	13	kl. 09.00	kl. 15.00	6
Koding	05.04.2021	15	kl. 12.00	kl. 16.00	4
Hovedrapport og dokumentasjon	06.04.2021	15	kl. 09.00	kl. 14.30	5,5
Møte med veileder og koding	07.04.2021	15	kl. 10.00	kl. 17.30	7,5
Jobbing med rom-kalender	08.04.2021	15	kl. 09.00	kl. 17.00	8
Rapportskriving	09.04.2021	15	kl. 09.00	kl. 16.00	7
Koble frontend mot backend	12.04.2021	16	kl. 09.00	kl. 15.30	6,5
Koble frontend mot backend	13.04.2021	16	kl. 09.00	kl. 16.00	7
Koble frontend mot backend	14.04.2021	16	kl. 09.00	kl. 16.00	7
Koble romfunksjonalitet til backend	15.04.2021	16	kl. 09.00	kl. 16.00	7
Koble brukerfunksjonalitet til backend	16.04.2021	16	kl. 09.30	kl. 17.00	7,5
Koble romfunksjonalitet til backend	19.04.2021	17	kl. 09.00	kl. 22.00	13
Forberedelse til møte og bugfiksing	20.04.2021	17	kl. 07.30	kl. 15.00	7,5
Forberedelse til møte og bugfiksing	21.04.2021	17	kl. 09.00	kl. 16.30	7,5
Rapportskriving	22.04.2021	17	kl. 09.00	kl. 16.00	7
Møte og rapportskriving	23.04.2021	17	kl. 09.00	kl. 14.30	5,5
Kundeinnstillinger	26.04.2021	18	kl. 09.00	kl. 15.30	6,5
Kundeinnstillinger	27.04.2021	18	kl. 09.00	kl. 19.30	10,5
Brukerinnstillinger	28.04.2021	18	kl. 09.00	kl. 15.30	6,5
Brukerinnstillinger	29.04.2021	18	kl. 08.30	kl. 19.00	10,5

Beskrivelse	Dato	Uke	Fra	Til	Timer
Brukerinnstillinger	30.04.2021	18	kl. 09.00	kl. 12.00	3
Custom timepicker	03.05.2021	19	kl. 09.30	kl. 20.00	10,5
Booking kalender	04.05.2021	19	kl. 09.30	kl. 19.00	9,5
Booking kalender	05.05.2021	19	kl. 09.00	kl. 20.00	11
Booking kalender	06.05.2021	19	kl. 09.30	kl. 19.00	9,5
Administrator side	07.05.2021	19	kl. 09.00	kl. 19.00	10
Administrator side	08.05.2021	19	kl. 10.30	kl. 18.00	7,5
Organisasjoner	09.05.2021	20	kl. 10.00	kl. 15.00	5
Rapport og litt kode	10.05.2021	20	kl. 09.00	kl. 18.00	9
Rapport og litt QR	11.05.2021	20	kl. 09.00	kl. 16.00	7
QR-koder og rapport	12.05.2021	20	kl. 09.30	kl. 18.00	8,5
Rapportskriving + møte med veileder	13.05.2021	20	kl. 09.30	kl. 15.30	6
Dokumentasjon	14.05.2021	20	kl. 09.00	kl. 16.30	7,5
Systemdokumentasjon	15.05.2021	20	kl. 10.00	kl. 14.00	4
Hovedrapport og dokumentasjon	17.05.2021	21	kl. 14.00	kl. 21.00	7
Ferdigstille dokumentasjon	18.05.2021	21	kl. 09.00	kl. 18.00	9
Ferdigstille dokumentasjon	19.05.2021	21	kl. 09.00	kl. 20.00	11
Sum antall timer:	**505**				

4.4.2 Week 4

Vi hadde første møte med oppdragsgiver og det gikk meget bra. Arne, som er rektor på Sonans Trondheim stilte godt forberedt til møtet og hadde detaljerte planer om hva han så for seg. Vi startet med å planlegge; visjonsdokument og Gantt diagram.

4.4.3 Week 5

Denne uken gikk med på å ferdigstille visjonsdokumentet. Vi startet med design av prototypen i Figma. Oppdragsgiver hadde et bra utgangspunkt for designet, men vi ønsket å modernisere det. Meg og Espen har jobbet en god del sammen i Figma.

4.4.4 Week 6

Denne uka begynte vi å se på muligheter rundt problemstilling og hva vi ville lære mer om og sette oss inn i. Håkon fant ut at sikkerheten som ble presentert og foreslått av oppdragsgiver ikke var særlig sikker, da den lett kunne knekkes. Vi leste om utvikling av lignende applikasjoner og bestemte oss for å bruke rammeverket Vue.js på frontend. Vi ble enige om at alle 3 av oss skulle gjennomgå et kurs i rammeverket for å oppnå en felles kunnskap, da ingen av oss hadde brukte Vue.js tidligere.

4.4.5 Week 7

Mye av denne uken har gått med på kursing. Har også fortsatt på prototyping i Figma, og vi har hatt møte med veileder.

4.4.6 Week 8

Denne uka gikk med til å gjøre ferdig kurset vårt. Neste uka er planen å begynne å utvikle applikasjonen.

4.4.7 Week 9

I starten av denne uka satte vi oss ned og begynte å legge til rette for et godt utvikleroppsett. Vi lagde repositories på GitHub og fordelte arbeidsoppgaver. Jeg begynte å jobbe med frontend.

4.4.8 Week 10

Denne uka har gått med til å jobbe med frontend-applikasjonen. Begynner å bli mer dreven i Vue.js.

4.4.9 Week 11

Uka har gått med til å jobbe med reservasjoner. Det er frustrerende å jobbe med Date-objekter i JavaScript da de plutselig bruker lokal tidssone, og så plutselig ikke:)

4.4.10 Week 12

Denne uka har vi tatt oss fri da vi har en eksamen i et annet fag.

4.4.11 Week 13

I denne uka har jeg fortsatt å jobbe på reservering av rom og reservasjoner generelt. Det er en god del arbeid som ligger i akkurat dette.

4.4.12 Week 14

Påskeferie.

4.4.13 Week 15

Denne uka ble det en del dokumentasjonsarbeid. Vi har satt opp strukturen og planlegger hvordan alt skal se ut til slutt.

4.4.14 Week 16

Denne uka ble mye av tiden brukt sammen med Espen for å koble frontend sammen med backend. Hittil har jeg brukt mye dummy-data, men det var nå på tide å koble alt skikkelig opp mot hverandre slik at vi fikk jobbe med ekte data.

4.4.15 Week 17

I denne uka har jeg jobbet en god del med å fikse småbugs rundt forbi i frontend-applikasjonen. Har også brukt en dag på å skrive videre på hovedrapporten.

4.4.16 Week 18

Denne uka har jeg jobbet med å lage innstillinger for brukere og kunder. Det innebærer endring av passord, informasjon om konto, oppsett av to-faktor autentisering og slikt.

4.4.17 Week 19

Denne uka gikk mye av tiden til å forbedre rom-kalenderen. Den ble mer responsiv og skulle ta hensyn til eventuelle restriksjoner som var satt av organisasjonen, for eksempel maks antall reservasjoner, lengde på reservasjoner og hvor langt fram i tid man kan reservere et rom.

4.4.18 Week 20

Denne uka har jeg jobbet endel med å implementere QR-koder for rom. Jeg måtte finne kapable tredjepartspakker som kunne brukes sammen med Vue 3. Jeg måtte endre en god

del på romkomponener, for å gjøre slik at man kom inn på samme rom om man refreshet siden. Det måtte også gjøres for å kunne ha en URL som ledet til kalenderen til det aktuelle rommet. QR-koder til rom kan lastes ned separat eller i bulk.

4.4.19 Week 21

Denne uka har det blitt noen lengre dager. Vi hadde ganske god kontroll på alt, men det var en god del småting som måtte ordnes her og der. Det ble mye dokumentasjon og gjennomlesning av hovedrapporten.

E. Kalleberg, H. Harnes and S.J. Høle

Influental factos when implementing authentication

NTNU
Kunnskap for en bedre verden