Bachelor Thesis

Computer Science and Engineering, 300 Credits

An Integrated Room Booking and Access Control System for Public Spaces

Computer Science and Engineering, 15 credits

Halmstad, 2023-06-21

Jaffar Kamil, Mohamed Amer

HÖGSKOLAN
I HALMSTAD

Acknowledgments

We want to extend our most profound appreciation to our supervisor, Wagner Ourique De Morais, for his invaluable assistance in shaping our project. His commitment to organizing meetings, providing insightful feedback, addressing our queries, and consistently being available has been remarkable. Additionally, we would like to express our gratitude to our examiner, Veronica Gaspes, for her constructive feedback that has significantly enhanced the quality of our bachelor thesis.

Jaffar Kamil & Mohamed Amer

Abstract

Public spaces, especially educational institutions like universities, encounter challenges with their room booking and access control systems. These challenges commonly manifest as overlapping bookings and unauthorized entry. The latter issue, unauthorized access, specifically stems from inadequate integration between the respective systems. This bachelor thesis introduces a proof-of-concept for a cohesive room booking and access control system to address these issues. The proposed solution encompasses two mobile applications, one as the room reservation platform and the other as the access control mechanism. By integrating the management of bookings and access control, this proof-of-concept aims to overcome the prevalent shortcomings in existing systems. Halmstad University's IT department was consulted during the requirement definition phase to ensure a comprehensive understanding of the common problems, their underlying causes, and possible solutions. The proposed system utilizes common technologies such as NodeJS, Android Studio, and PostgreSQL. Additionally, Mobile BankID is integrated as a unique feature for secure user authentication, providing a trusted and widely-accepted method to verify users' identities. The final results were tested in a simulated environment and indicate that the developed system satisfies the initial requirements, addressing the problems of double bookings and unauthorized access identified during the consultation with the IT department.

Sammanfattning

Offentliga platser, särskilt utbildningsinstitutioner som universitet, står inför utmaningar med sina rum bokningssystem och åtkomstkontrollsystem. Dessa utmaningar visar sig ofta som dubbla bokningar och obehörig åtkomst. Det senare problemet, obehörig åtkomst, beror specifikt på otillräcklig integration mellan de respektive systemen. Detta kandidatexamensarbete presenterar ett proof-of-concept för ett sammanhållet rum bokningssystem och åtkomstkontrollsystem för att ta itu med dessa problem. Den föreslagna lösningen omfattar två separata mobila applikationer, där den ena fungerar som plattform för rum bokning medan den andra fungerar som åtkomstkontrollmekanism. Genom att integrera hanteringen av bokningar och åtkomstkontroll, så siktar detta proof-of-concept på att lösa de rådande bristerna i befintliga system. Halmstad Universitets IT-avdelning konsulterades under kravdefinitionsfasen för att säkerställa en omfattande förståelse för de vanliga problemen, deras bakomliggande orsaker och möjliga lösningar. Det föreslagna systemet använder vanliga teknologier som NodeJS, Android Studio och PostgreSQL. Dessutom integreras Mobilt BankID som en särskild funktion för säker användarautentisering, vilket erbjuder en betrodd och allmänt accepterad metod för att verifiera användarnas identiteter. De slutliga resultaten testades i en simulerad miljö och indikerar att det utvecklade systemet uppfyller de initiala kraven och behandlar problemen med dubbla bokningar och obehörig åtkomst som identifierats under samrådet med IT-avdelningen.

Table of contents

1. Introduction

An efficient and dependable online solution for booking and controlling access to facilities in public spaces can improve productivity, allocation of resources, safety, and security while reducing administration.

A well-designed university room booking system is vital for effectively managing classrooms, labs, and meeting rooms, catering to the university community's needs. Study spaces, in particular, play a critical role in academic success as the environment in which one studies can significantly impact productivity and efficiency [1]. Without designated study areas, maintaining focus and completing work can be challenging, leading to procrastination and decreased performance [20].

Room access control systems are also critical to ensuring security and privacy within an institution. A secure environment is created by preventing unauthorized access, and resources are used more efficiently.

However, many existing room booking and access control systems still exhibit shortcomings. For instance, Halmstad University employs a system called MazeMap to manage room bookings, and a lock system provided by Bravida, known as Integra W7, controls room access [26]. The MazeMap booking system is not integrated with the university's access control system.

Another concern is that MazeMap lacks real-time updates, which can lead to delays and confusion in the booking process. Without an integrated system, inefficiencies such as double bookings, conflicts, or missed room utilization opportunities may occur. For example, two students may attempt to book the same room simultaneously, but the system only processes one booking while erroneously displaying both as successful. This double-booking scenario can result in scheduling conflicts and lost productivity. This problem has been recurrently reported by students and acknowledged by the university's IT department.

MazeMap is exclusively used by students for room reservations, covering student rooms and classrooms. This problem is not restricted to a few rooms but is widespread across all rooms students can book through the MazeMap system.

While the university utilizes other booking alternatives such as TimeEdit and Outlook Calendar, these platforms are not accessible to students. Notably, the different systems used at the university, TimeEdit and Outlook Calendar, have not experienced double-booking issues reported with MazeMap.

This difference may be because these systems function more as schedule creators than fully-fledged room booking systems. Additionally, these alternatives are primarily used by the teaching staff, not students, which limits their exposure to potential double-booking scenarios.

This issue is not unique to Halmstad University, as many institutions face similar challenges due to purchasing booking and access control systems from different manufacturers. Such

procurement practices often result in limited customization options, hindering the systems' ability to meet unique institutional requirements. Consequently, integrating these systems may be suboptimal or fail to satisfy university standards. Additionally, disparate technologies and protocols used by room booking and access control systems can challenge seamless integration.

Furthermore, the current system's limitations contribute to security concerns. Booking a room only guarantees availability at a specific time, while access remains open to anyone with a student card. This situation increases the risk of unauthorized access, theft, and other undesired activities, compromising the privacy and security of students and staff. Addressing these shortcomings requires a comprehensive review and overhaul of existing room booking and access control systems to enhance efficiency, safety, and user satisfaction.

1.1 Purpose and Objectives

This project aims to develop a proof of concept room booking system with an integrated access control system for universities, explicitly addressing the shortcomings of current systems used by Halmstad University and others.

The goals of this project are to implement the specific functional and non-functional requirements in the requirement specification. This document will present these requirements in detail, emphasizing the key features and objectives essential to the project's success.

Functional requirements:

- The system must allow users to log in securely.
- The system must allow users to search for available rooms
- The system must allow users to create and manage room bookings.
- The system must provide users with the ability to add more users to a booking
- The system must only grant authorized users to access booked rooms
- The access control should only allow access to rooms when it has been booked
- A student can only have one active booking at a time

Non-functional requirements:

- The system must be easy to use and navigate, even for individuals who may not be technically savvy.
- The system should offer a solution for students without BankID. ‘
- Each user's booking permissions should align with their role and authorization within the university

1.2 Limitations

This project's scope is limited to developing a proof-of-concept room booking and access control system for universities. However, it will not work directly with the university access control system. Additionally, the system will be developed for Android devices and not iOS devices. The project will focus on designing and developing the system's back-end, front-end, and API components. It will not involve implementing the system in a university setting but will be tested in a simulated environment as an application. No administration system will be developed as part of this project.

2. Background

This section presents the technical aspects of room booking and access control systems, which are essential for managing university facilities. We will present the architecture and components of room booking systems. Then, we will present existing access control systems, covering frontend, backend, and integration with room booking systems. Lastly, we will analyze existing solutions, emphasizing their technical aspects and limitations.

2.1 Existing Room Booking Systems

2.1.1 Mazemap

Mazemap is an interactive indoor mapping platform that integrates with room booking systems. Mazemap is available through both a mobile and web application. It primarily utilizes web technologies like HTML, CSS, and JavaScript. Its API provides integration points for room booking systems but does not offer a comprehensive booking solution. The API is based on REST principles, enabling seamless communication between Mazemap and other systems [25].

2.1.2 Outlook

Outlook, a popular web-based email and calendar application developed by Microsoft, offers basic room booking functionality through its calendar feature. Users can invite attendees and reserve rooms for meetings, but the system lacks advanced features, such as integration with access control systems or facility management tools. Moreover, Outlook's room booking capabilities are limited by its primary function as an email and calendar application.

2.1.3 TimeEdit

TimeEdit is a web-based scheduling and resource management solution for educational institutions. It offers room booking, course scheduling, and event management features. However, TimeEdit does not include native access control functionality, requiring separate integration with an access control system to manage room access permissions. It uses web technologies like HTML5, CSS3, and JavaScript, focusing on responsive design and device compatibility. It uses a RESTful API for communication between the frontend and the backend. TimeEdit's backend uses Microsoft technologies, such as ASP.NET and SQL Server.

2.2 Existing Access Control Systems

2.2.1 Bravida Integra W7

Bravida Integra W7 is an access control system that offers a wide range of features, such as access management, alarm monitoring, and video surveillance. Although it is a powerful system, it requires manual configuration and integration with room booking systems, which can be time-consuming and prone to errors. The system's API is based on REST principles, which can be used for integration with other systems, including room booking solutions.

2.2.2 HID Global

HID Global is a well-known provider of access control solutions, including card readers, biometric devices, and mobile access solutions. Their products and solutions can be easily integrated with various room booking systems through their API. HID Global follows an open architecture approach, allowing seamless integration with different third-party systems.

2.2.3 Axis Communications

Axis Communications specializes in IP-based access control solutions, offering network door controllers, card readers, and mobile access solutions. They also provide a software development kit (SDK) and a RESTful API, which enables integration with room booking systems and other third-party applications.

2.3 Technical Overview, Types, and Features of Room Booking Systems

A room booking system can be implemented as a web-based and mobile application and comprises several architectural layers and components. Architectural layers in a room booking system include the presentation layer for user interface and interaction, the application layer for business logic, the data access layer for managing data interactions, the data storage layer for storing system data, and the integration layer for API communication with external systems.

The user interface, or front-end component, comprises the mobile application utilized by students and staff and the web-based application. These applications enable users to locate and reserve study rooms, incorporating elements like application layout, graphics, and textual content [21]. Furthermore, the front-end component is responsible for gathering user credentials and booking information, which is subsequently processed by the back-end.

The back-end component plays a crucial role in the system architecture by handling requests and managing data, making it an essintial part of the system. It incorporates a database that

stores essential information about room availability, reservation schedules, and user accounts while detecting conflicts and ensuring a smooth and efficient reservation process. In addition, the back-end component also includes an API to allow other systems to interact with the booking system.

APIs are vital in software systems' structure, enabling communication and information sharing between diverse systems. They supply endpoints that empower external systems to access and manipulate data within the target system, employing conventional HTTP requests like GET, POST, PUT, and DELETE. These requests are generally executed using the versatile RESTful API architecture [17]. In relation to a room booking system, APIs present a valuable method for external systems to obtain, supplement, adjust, and remove booking-related data. The APIs may exhibit variations depending on whether a mobile or web-based application is employed.

Room booking systems can be categorized into three main types: manual, semi-automated, and fully automated systems [27]. Manual systems rely on traditional methods such as pen and paper or spreadsheets, while semi-automated systems employ basic software solutions to streamline the booking process. On the other hand, fully automated systems use advanced software to manage every aspect of the room booking process with minimal human intervention [28].

Various features are common in room booking systems, including meeting management, scheduling, visitor management, and integration with existing systems. Meeting management allows for the organization and coordination of meetings, while scheduling features enable users to reserve rooms based on availability. Visitor management offers the ability to track and manage visitors within a facility. Integration with existing systems ensures the room booking system can communicate with other applications such as calendar and email software [28].

During the COVID-19 pandemic, the significance of health and safety features gained prominence. Monitoring the number of people and visitors to adhere to regulations became essential. Consequently, room booking systems proved invaluable for managing occupancy and ensuring compliance.

2.4 Technical Overview, Types, and Features of Access Control Systems

Access control systems consist of several architectural layers and components and are available in various types, including physical and logical access control. The frontend typically includes a graphical user interface (GUI) for system administrators and a user interface for clients, which may consist of card readers, biometric scanners, or other access control technologies such as RFID/NFC, smart cards, QR codes, and mobile apps [21].

Similar to room booking systems, access control systems use APIs to facilitate communication between the front and backend, enabling access request processing and user

data management. The backend layer manages user data and processes access requests, including a server that can be implemented using various programming languages and a database for storing user credentials and access permissions. To ensure secure access control, the server may employ authentication and authorization protocols, such as OAuth2 or SAML [22].

Each access control technology has its advantages and disadvantages. For example, RFID and NFC provide quick, contactless access, while biometrics offer a higher security level due to their unique nature. Smart cards and QR codes are cost-effective, and mobile apps provide the convenience of remote access management. Institutions must carefully evaluate their specific needs and requirements when selecting the appropriate access control technology.

2.5 Challenges and Limitations of Existing Solutions

While existing room booking and access control systems offer a range of functionalities to manage university facilities, they also present several challenges and limitations that need to be addressed to enhance their effectiveness and user experience [23].

One of the significant challenges educational institutions face is the complexity of integrating room booking systems with access control systems. The integration process often requires extensive manual configuration and customization, increasing costs and potential errors [24]. Software compatibility issues can further complicate this process, making it challenging to integrate the two systems [28]. A more streamlined and automated integration process would significantly reduce the time and effort required to set up a fully functional system.

In addition, many universities use separate room booking and access control systems, which can result in inefficiencies and inconsistencies in resource management. A unified solution combining room booking and access control functionalities would provide a more seamless experience for administrators, staff, and students. This ensures that resources are allocated effectively, and access is granted only to authorized individuals.

Educational institutions have unique requirements based on size, resources, and policies. Some existing solutions may not offer sufficient customization options to cater to these specific needs, leading to suboptimal performance and user experience. A more adaptable solution tailored to each institution's requirements would provide a more effective and user-friendly system.

Finally, ensuring the security of room booking and access control systems is crucial to prevent unauthorized access to sensitive areas and protect user data. Some existing systems may lack security protocols or regular updates, posing potential risks to the integrity and confidentiality of the information stored within the system. A robust and secure solution that employs state-of-the-art security measures and receives regular updates would help mitigate these risks.

As previously mentioned, one of the most significant issues plaguing the current room booking system at Halmstad University is the problem of double booking within the MazeMap application. This problem has been recurrently reported by students and acknowledged by the university's IT department. MazeMap is exclusively used by students for room reservations, covering student rooms and classrooms. However, instances have been reported where students arrive to find another group occupying a space for which they hold a valid booking. This problem is not restricted to a few rooms but is widespread across all rooms students can book through the MazeMap system.

While the university utilizes other booking alternatives such as TimeEdit and Outlook Calendar, these platforms are not accessible to students. Importantly, these alternatives have not experienced the double-booking issues reported with MazeMap, suggesting that the problem is specific to the MazeMap system rather than a general issue across all the university's booking systems.

A more effective and user-friendly room booking and access control system can be developed by addressing these challenges and limitations, benefiting educational institutions and their users.

3. Materials and Methods

This section comprehensively explains the steps taken to fulfill the project requirements. The discussion begins with a detailed explanation of the methodology employed, including a justification for selecting elements. Subsequently, the chapter elaborates on the methods used to analyze the results.

3.1 Methodology

The project commenced with a comprehensive research stage, exploring existing technologies that could provide the required functionality for a unified room booking and access control system.

Next, we identified the critical problems with the current room booking system at Halmstad University through consultations with the IT department. During our meeting, they identified the primary shortcomings of the existing system as double bookings and unauthorized access, both known and persistent issues.

These problems stemmed from a lack of integration between the access control and room booking systems. The IT department was not entirely certain about the root cause of the double booking issue. However, they suspected it might be due to using three different booking systems—Outlook, MazeMap, and TimeEdit—which did not share information, potentially leading to inconsistencies and conflicts in the booking process.

Guided by these findings, we designed an integrated system to address these problems. This system utilized a REST API to facilitate communication between the client-side application, developed using Android Studio, and the database, created with PostgreSQL. The API was designed to handle HTTPS requests, allowing secure data transmission between the client application and the server.

Our server, database, and API were hosted on Microsoft Azure, a reliable and robust platform for cloud computing. The selection of Azure was driven by its scalability, security, and extensive support for APIs and databases. We continuously revised and adjusted the system throughout development to ensure that the final product effectively solved the identified problems and met all outlined requirements.

Figure one below displays a conceptual representation of the integrated booking and access control system that will be created. This diagram highlights the key components and interactions between the various subsystems, providing a foundation for the project's subsequent design and development phases.

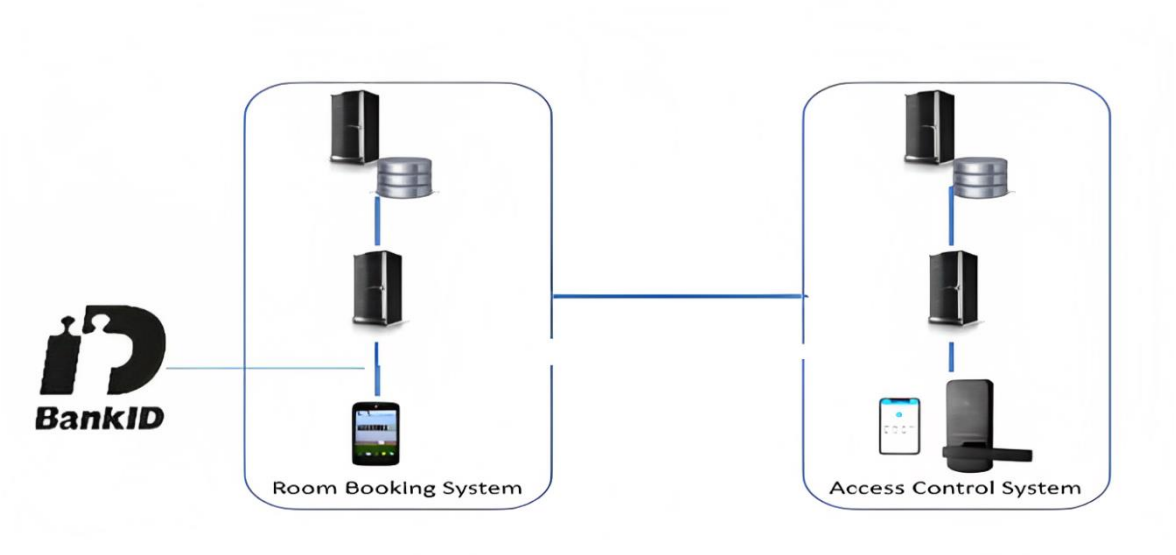


Figure 1 Conceptual representation of the integrated booking and access control system

3.1.1 System Design

The proposed university room booking system will be accessible to students through a pair of mobile applications: one serving as the booking room platform and the other functioning as the access control mechanism, both working in parallel. Upon logging in to the room booking application using their mobile BankID for secure authentication, students will be directed to a home screen displaying available rooms.

Students must select a specific date, time, and room location to reserve a room. They also have the option to invite additional users to join the reservation. Upon booking confirmation, students can unlock the reserved room using the room booking mobile application. The access control application can monitor the door's status, providing real-time updates for convenience.

Behind the scenes, the application communicates with a server and database to verify booking details and grant room access. This system ensures coordination between the user interface and back-end infrastructure. Furthermore, the system offers convenient options for canceling bookings and reviewing booking history.

3.1.2 System Integration

We propose an integration of the room booking and access control systems using a centralized approach, ensuring seamless communication and coordination between the two. This integration aims to address significant shortcomings, such as double bookings and unauthorized access, as identified in our discussions with the IT department at Halmstad University.

A centralized booking and access management system serve as the single source of truth for room availability and reservations [23]. The system stores user, room, booking, and access control information in a unified database, ensuring data consistency across both systems.

Consider, for instance, a situation where a student tries to book a room that has just been booked by someone else. With a centralized database, the room's booked status is immediately reflected across the system, preventing the second student from booking it, thereby avoiding a double booking scenario.

The room booking system's server and access control system's server communicate with the centralized database, retrieving and updating the necessary information to manage bookings and access permissions. This architecture ensures instant system updates upon every booking through real-time synchronization and data consistency [24], significantly mitigating the risk of double bookings. Our centralized approach offers real-time access control adjustments for unauthorized access prevention following each booking. This ensures that only the students who made the booking or have been invited to the booking can access the room, thereby preventing unauthorized access.

The system also leverages transactional mechanisms, such as database transactions or distributed locking systems, to maintain data integrity across both systems. Essentially, these mechanisms ensure that all parts of a transaction are completed successfully or none at all. For example, if a student attempts to book a room, the booking operation and the corresponding access control update either succeed or fail, preventing inconsistencies and partial updates.

Integrating room booking and access control systems using a centralized approach can enhance resource management in educational institutions, simplifying the reservation and access process while preventing issues such as double bookings and unauthorized access [24].

However, while the centralized approach offers numerous advantages, it's important to acknowledge potential limitations. For example, it requires robust data backup and recovery mechanisms, as a single database failure could impact the entire system. Also, maintaining optimal performance may necessitate more advanced data management techniques as the system scales up.

An alternative approach to our problem could have been to use a microservices architecture, where the room booking system and access control system would be independent services, each with its databases, communicating through well-defined APIs.

While a microservices architecture could provide high levels of scalability and allow each system to be developed and deployed independently, it comes with its challenges. For example, ensuring data consistency between two separate systems can be complex and potentially lead to issues such as double bookings. Moreover, if one system is updated without the corresponding update in the other, it could lead to compatibility issues or inconsistent functionality.

Figure two below illustrates the Entity-Relationship (ER) representation of the conceptual database design for a room booking and access control system. The database consists of several entities, each serving a distinct purpose in the system's functionality. Key entities include Users, Rooms, Bookings, and Access Control.

One essential entity in our design is the Users table, which stores system user information. The primary key, userID, is a unique identifier for each user and enables identification through Bank ID. This approach simplifies user authentication without the need for a password.

The database schema supports various functionalities, such as room search, booking, and controlled sharing of bookings. The Rooms and Bookings tables store information about available rooms and booking details. The Access Control table manages user permissions for accessing shared bookings.

In conclusion, despite the potential challenges of a centralized approach, we found it to fit our needs best. It allows us to maintain data consistency, simplify system maintenance, enhance data security, and provide a smooth user experience.

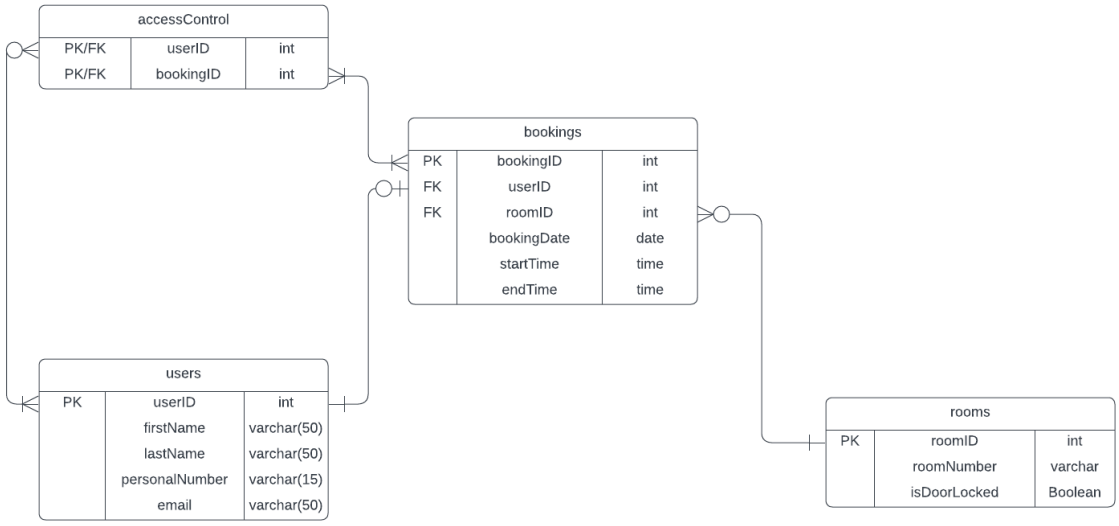


Figure 2 ER-representation of the conceptual database design

3.2 Tools and Technologies

This section will showcase the tools and technologies employed in developing the proof-of-concept room booking and access control system. The chosen technologies aim to ensure the system is reliable, providing an intuitive experience for users. The primary tools and technologies utilized in this project are BankID for secure user authentication, REST API for efficient and flexible communication between software components, and HTTPS Protocol for reliable data transfer over the Internet.

3.2.1 Bank ID

BankID is a digital identification solution financial institutions use to authenticate users and enable the digital signing of documents. That was selected as the authentication method for several reasons. Firstly, its widespread adoption, especially within Sweden, made it a familiar solution to most potential users of the university room booking and access control system. This familiarity can increase user trust and reduce potential resistance to adopting the new approach.

Secondly, BankID's robust security features, including strong encryption and two-factor authentication, offer superior protection against unauthorized access compared to more traditional username-password authentication methods [7]. While not eliminating the risk, BankID significantly reduces the chances of unauthorized access, a significant concern in any system handling sensitive data.

However, the integration of BankID did present challenges. The most significant of these was obtaining and managing the necessary certificates. BankID's high-security measures require developers to provide certificates proving their identity and the legitimacy of their application. Obtaining these certificates can be complex, and managing them requires a sound understanding of secure programming practices.

Despite these challenges, BankID was still chosen because of its apparent benefits. The certificate process, although complex, provides an additional layer of security by ensuring only authorized applications can implement BankID. Furthermore, overcoming these challenges provided invaluable learning experiences and demonstrated the ability to solve problems independently and effectively.

In the context of our university room booking system, BankID's role is crucial. By providing secure and reliable authentication, BankID ensures that only authorized users can access the system, book rooms, and manage bookings, thus protecting sensitive information and maintaining the system's integrity.

Integration of BankID into a system begins with registering as a developer with BankID. This step involves providing necessary information about the developer and the application where the BankID service is to be used. Once registered, developers receive an API key that grants access to BankID services.

Next, a certificate needs to be obtained from a bank. This certificate is used to authenticate the application when requesting the BankID servers. Certificates come with a cost and have a lifespan, after which they need to be renewed. However, BankID provides a demo service that can be used for testing purposes. This free service comes with test certificates, allowing developers to understand the system’s functionality without incurring usual costs.

Integrating BankID into the application involves using the BankID API to initiate authentication and signing requests, check the status of requests, and retrieve signed documents. The BankID API uses standard protocols and formats such as JSON, making it easy to integrate into most applications [7].

A secure connection must be established between the application and the BankID servers during the integration process. This is done using HTTPS, with the application's certificate being used to authenticate each request. The security of this connection is crucial as sensitive user information is being transferred.

Finally, thorough testing must be conducted to ensure the BankID functionality works correctly and securely. BankID provides a testing environment where developers can simulate different scenarios and verify that their application handles them correctly.

While the integration process can be complex and time-consuming, the security and user benefits offered by BankID make it an excellent choice for applications requiring secure user authentication, like our university room booking system. Overcoming these challenges provided invaluable learning experiences and demonstrated the ability to solve problems independently and effectively.

Several other options, including OAuth and SAML-based solutions, were considered, including Freja eID. However, after comparing these solutions based on security, user familiarity, ease of integration, and cost, BankID was the best fit for our project.

Freja eID is a comprehensive mobile-based solution that offers personal identification, document signing, and age verification services. It provides robust security features similar to BankID and is approved for the Swedish government's e-identification board's highest level of assurance (LOA3) [29]. However, it has a lower user base in Sweden compared to BankID. Given that our system targets university students and staff, the widespread adoption of BankID made it a more suitable choice as it increases the likelihood of users already having a BankID and being familiar with its use. However, these were discarded in favor of BankID due to the latter's robust security features and widespread acceptance among our user base.

3.2.2 REST API

A REST API, short for Representational State Transfer Application Programming Interface, is a web API that abides by the REST architectural style. This type of API facilitates communication between diverse software systems, enabling them to exchange data and functionality [17]. REST APIs support data exchange in different formats like XML, JSON, and plain text, which allows for a high degree of flexibility. It can be incorporated with various software systems, such as web applications and mobile apps.

The most notable advantage of REST APIs is their scalability. Since they are stateless, REST APIs can handle numerous concurrent requests without slowing down or crashing [17]. As a result, they are well-suited for applications with high traffic volumes or unpredictable demand.

Apart from flexibility and scalability, REST APIs are also highly secure, which is crucial for safeguarding against unauthorized access and other security risks. For our university room booking system, the REST API will enable efficient and secure communication between the

mobile application and server-side components, allowing users to interact with the system seamlessly and ensuring that the system can scale to accommodate the needs of a large user base.

3.2.3 HTTP Protocol

Hypertext Transfer Protocol (HTTP) is a widely used communication protocol for transferring data over the Internet. It is a request-response protocol, where the client sends a request to the server through HTTP or HTTPS, and the server responds with the requested data [10]. Once the response is received, the connection between the client and server is closed the reliable delivery of data packets is. In case a package is lost during transmission, TCP will automatically resend it to ensure that all data packets are received in the correct order.

HTTPS will facilitate communication between the client and server components of the university room booking system. When a student wants to book a room, the client component (e.g., the mobile application) will send an HTTPS request to the server component, asking for the available rooms at the requested date and time. The server component will then process the request and respond with the details of the available rooms. This communication between the client and server components is done using HTTPS. Figure three below will explain the HTTPS connection process from request to response.

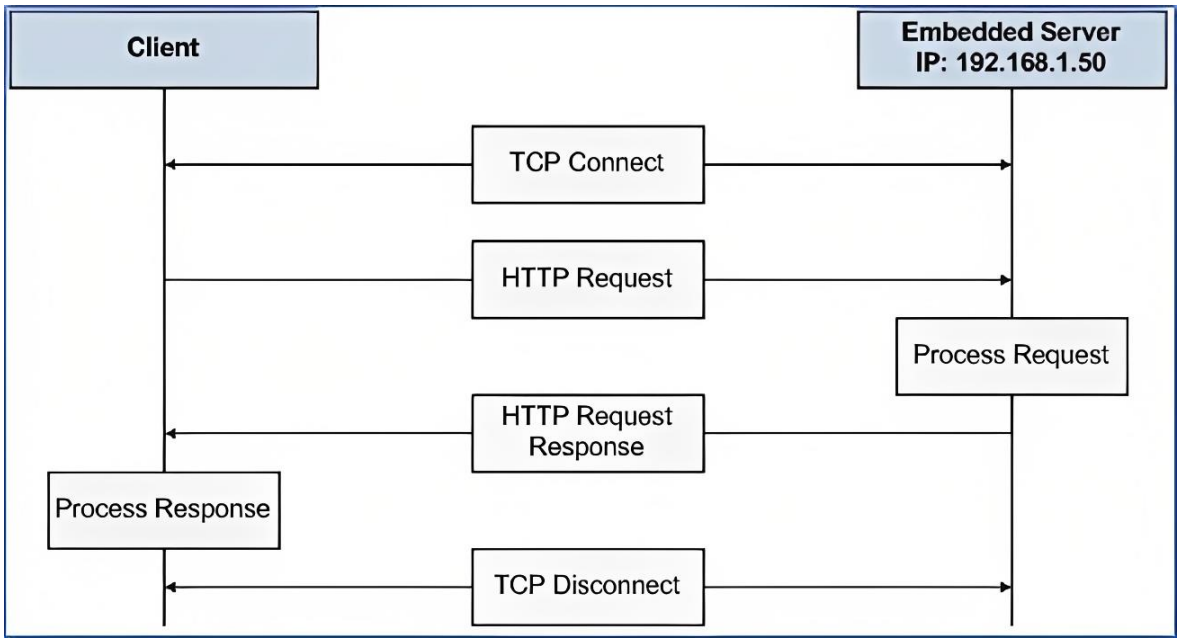


Figure 3 Overall view of HTTP connection

3.3 Experimental Setup

This section outlines the experimental setup employed to develop the room booking and access control system. The project uses a combination of well-established tools and

technologies to cater to different aspects of development. These include utilizing NodeJS development with Visual Studio Code to implement REST API, application development using Android Studio, database management via PostgreSQL, and secure user authentication through Mobile BankID. The following subsections offer an in-depth explanation of each tool and technology, illustrating their contributions to the project and the rationale behind their selection.

3.3.1 NodeJS Development with Visual Studio Code

For NodeJS development, we used Visual Studio Code (VS Code), a popular choice for Node.js development because it provides a range of features and extensions that make it easy to develop, test, and debug Node.js applications [19]. It has a built-in terminal allowing developers to run commands and scripts directly from the editor. This is especially useful for Node.js developers who must run scripts, commands, and pass arguments to start and manage their Node.js applications.

In our project, we used VS Code to develop the REST API that enables the communication between the mobile application and the server-side components of the university room booking system. The various tools and features offered by VS Code and its large and active community of developers will help us develop high-quality REST APIs quickly and efficiently.

3.3.2 Application Development with Android Studio

We used Android Studio for the application development, which offers developers a wide range of features and tools, such as a visual layout editor and a built-in emulator [15]. This will ease the process of testing and debugging the app. Java is the programming language used for the frontend of the application. The application's user interface will be created using XML files, which define the layout and design of the app's screens and UI elements.

Android Studio, combined with Java, creates a rich development ecosystem with various third-party libraries, tools, and plugins that facilitate the development of complex and feature-rich Android apps. In our project, we will use these libraries for UI development, networking, and database management, enabling us to create a seamless and efficient user experience for the university room booking system.

3.3.3 Database Management with PostgreSQL

The database management system for the university room booking and access-control system is built using PostgreSQL, a powerful free, open-source database system that suits our needs [4]. It uses the SQL language to interact with the database. PostgreSQL fits businesses and individuals needing a solid and reliable database management system with advanced features and strong community support.

In our project, we used PostgreSQL to store user, room, booking, and access control information in a unified database, ensuring data consistency across both the room booking and access control systems. While other open-source options are available, PostgreSQL's

open-source nature, reputation, and compatibility make it a popular choice for many applications.

3.3.4 Mobile BankID Integration

Mobile BankID is the login method used for the application. It provides a secure way of verifying the user's identity, essential for protecting sensitive data and preventing unauthorized access. Furthermore, it uses strong encryption and multifactor authentication based on public-key cryptography, specifically RSA, with end-to-end encryption. This means the communication is encrypted on both ends, making it much more secure than traditional username/password login methods.

In the context of our university room booking system, we will integrate Mobile BankID into the app to ensure that only authorized users can access the system, book rooms, and manage bookings. This will protect sensitive information and maintain the system's integrity.

3.4 Tests and Analyzing the results

3.4.1 Test Design

To evaluate the success of the integrated room booking and access control system, we designed and conducted a series of tests that focused on addressing the issues of double bookings and unauthorized access. During the development phase, we conducted integration tests, such as testing with HTTPS requests and the Bank ID API integration, to ensure the proper functioning of different system components. Once the development was complete, we tested whether the integrated system solved the issues we aimed to address, such as double bookings and unauthorized access. The tests were performed in a simulated environment, as the proof-of-concept system could not be tested in a university setting. The testing process consisted of several tests, including integration, functional, stress, and edge case testing.

3.4.2 Test Execution

The integrated room booking and access system was thoroughly tested to verify its functionality and performance to ensure it was ready for deployment. The testing process consisted of several types of tests, including integration testing, functional testing, stress testing, edge case testing, and user acceptance testing.

We used integration testing to test how different system components work together. Specifically, we employed HTTPS requests containing JSON code via the Postman application to test how the API interacts with the database and confirm that data can be successfully inserted into the table. Each request tested one or multiple functions of the code, and if the request successfully added valid information to the table, we concluded that the function was developed correctly.

As part of our integration testing for the Bank ID API, we sent authorization requests to verify users and obtain their personal information. Specifically, we requested the user's name

and social security number, which the API returned after successful verification. Throughout the testing process, we closely monitored the accuracy of the data provided by the API and ensured that it aligned with our expectations. Our primary goal was to confirm that the Bank ID API could successfully verify users and provide their personal information to our application.

In addition to integration testing, we performed functional testing on the system's components to ensure they worked as intended. For example, we tested the room booking feature to verify that users can select a room, date, and time and book it successfully. We also tried the room unlock feature to verify that users can unlock rooms that they have booked. Additionally, we tested the view bookings feature to ensure that users can view their current and future bookings.

We also simulated a scenario where a high volume of bookings, specifically 1000 bookings, were made simultaneously. This was done to put the system under heavy load and observe its performance and integrity. The objective was to confirm if the system could handle many such bookings simultaneously without creating double bookings and assess its response time and overall performance under high stress.

This test showed that the system maintained functionality and performance even under heavy load. Despite the influx of booking requests, there were no double bookings, indicating that the system could handle a high volume of simultaneous bookings while maintaining the integrity of the booking process.

We also tested the system's behavior in edge-case scenarios. Specifically, we attempted to book the same room at the same time. This test aimed to check if the system could correctly handle such situations without creating double bookings. The test result was positive, as only one booking was successfully processed, preventing double bookings even in extreme conditions. We developed a Python script capable of simultaneously sending 1000 booking requests to the API. We executed this script ten times, and the double booking problem only occurred once out of those attempts as figure 5 shows. Each time we performed the test, we sent 1000 requests to book the same room simultaneously. We lowered the requests to 500 and ran the script 10 more times, the double booking issue did not occur, this means that the double booking issue only occurs when the system is heavily stressed. This outcome is considered satisfactory since it is unlikely to encounter a real-life scenario where 1000 students simultaneously submit booking requests. To prevent the error from occurring, there is still work to be done in understanding the causes and potentially addressing them.

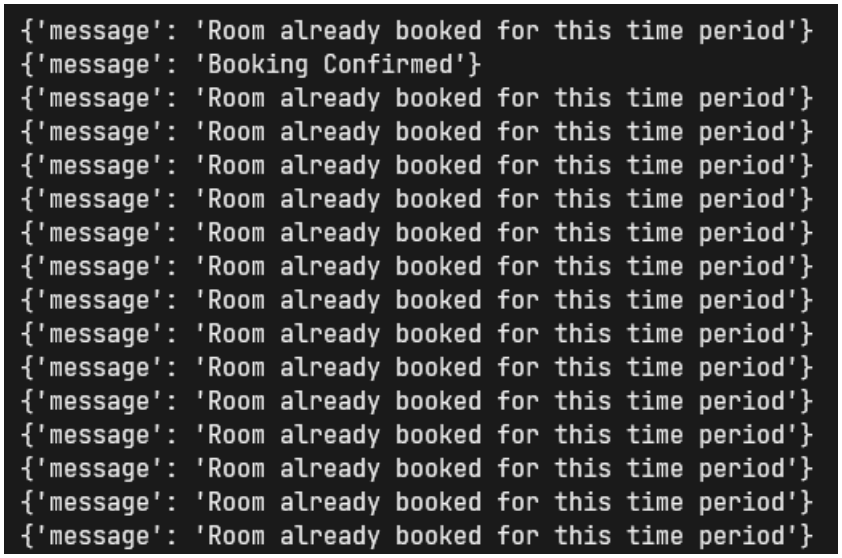


Figure 4: API responses when sending 1000 booking requests, no double booking issue

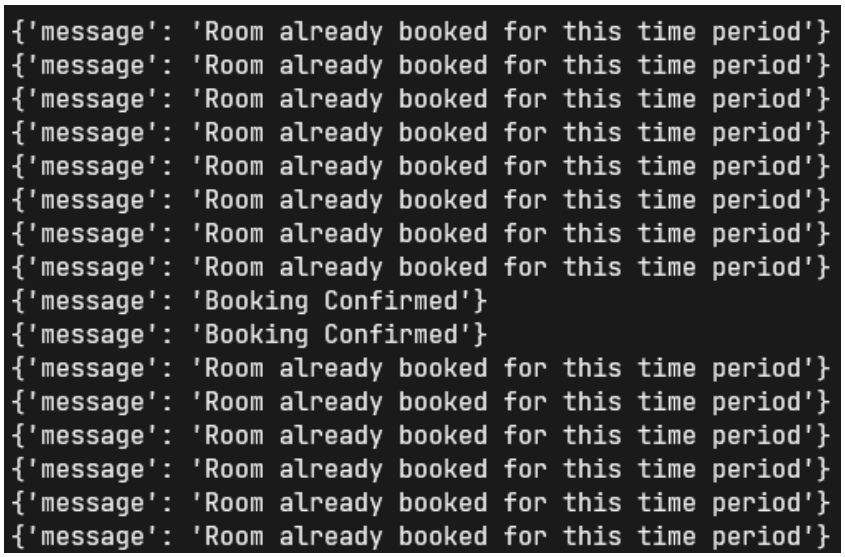


Figure 5: API responses when sending 1000 booking requests, one double booking issue

In another edge case test, we attempted to book a room for a time slot that was just beginning or ending. This was to verify if the system could adequately handle bookings made at the edge of available time slots. The results showed that the system appropriately managed such bookings, ensuring the correct allocation of rooms even when bookings were made at the very start or end of a time slot.

As part of our objective to ensure secure room access, we conducted tests to simulate attempts of unauthorized access. Specifically, we tried to open rooms not booked under the test user account. We aimed to verify the robustness of the system's access control mechanisms in denying such unauthorized requests.

The results of these tests were successful. The system reliably denied access to rooms that were not booked by the user attempting to unlock the room, thereby confirming the effectiveness of our access control system in preventing unauthorized access.

4. Results

In the following section, we will present the results of our project, which aimed to create a proof-of-concept room booking and access control system. The room booking and access control system is built as a mobile application that contains the following features:

Pages:

- Sign in page
- Booking page
- Manage bookings
- Unlock doors

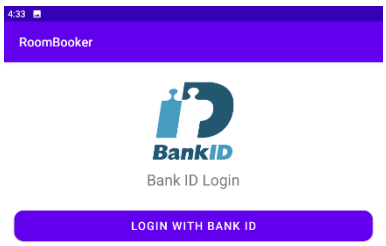
Functionalities:

- Log in using Mobile BankID
- Search for available rooms and book them
- Unlock the booked room
- Adding people to the booking

4.1 Room Booking Application

Figure 4a: Upon opening the application, the user is presented with the initial home screen. To access their account, they must log in using Mobile BankID as their authentication method, as seen in figure 4a. The system retrieves user information from BankID and verifies whether the user is already in the database. If the user exists, they are directed to the next page.

Figure 4b: If the system does not find the user in the database, the user is prompted to enter their email address. Since the user's credentials are already known through BankID, they will be registered in the database. Once registered, the user can proceed with the application.



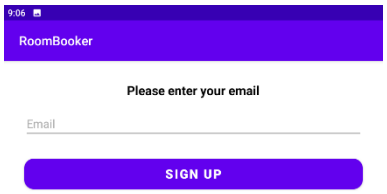


Figure 4 a) *b)*

Figures 5a and 5c: After logging in, you will be directed to a screen featuring two buttons: 'View Available Rooms' and 'View Bookings,' as shown in Figure six. By selecting 'View Available Rooms,' you will see a list of rooms that can be booked, regardless of their current availability status.

Figure 5b: Once you have chosen a room, you will be taken to another page where you can select the date, time, and duration of your booking. After confirming the booking, you can return to the previous screen. Click 'View Bookings' on the previous screen to review your reservations.

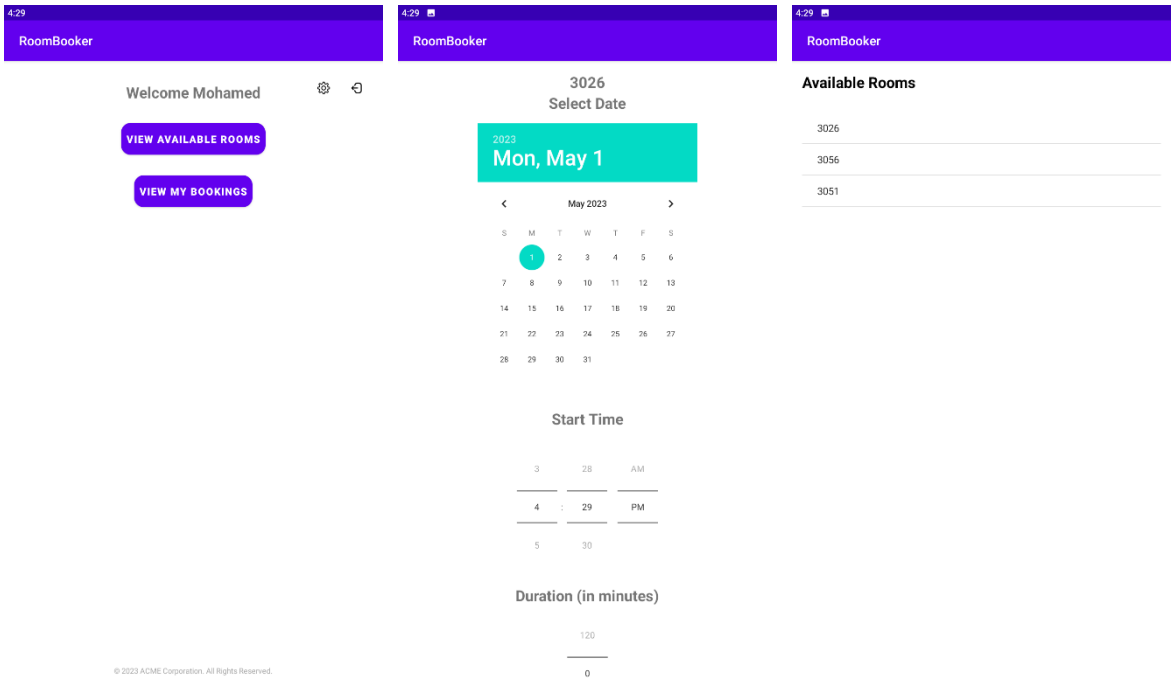


Figure 5 a) *b)* *c)*

Figures 6a and 6b: Additionally, you can share your booking with others, granting them access to unlock the room. However, until the booking has been shared, only you can unlock the door. This feature aligns with one of our key requirements established from the beginning. You can remove the person from the booking or cancel the reservation anytime.

Figure 6c: Upon sharing a booking with another individual, they can view the reservation on their 'Bookings' page. Additionally, when a booking has been shared with another individual, both parties can unlock the room using the earlier process. This ensures that all authorized users can access the room during the designated reservation period. It is important to note that a user cannot have multiple bookings concurrently, and each booking is limited to a maximum duration of two hours.

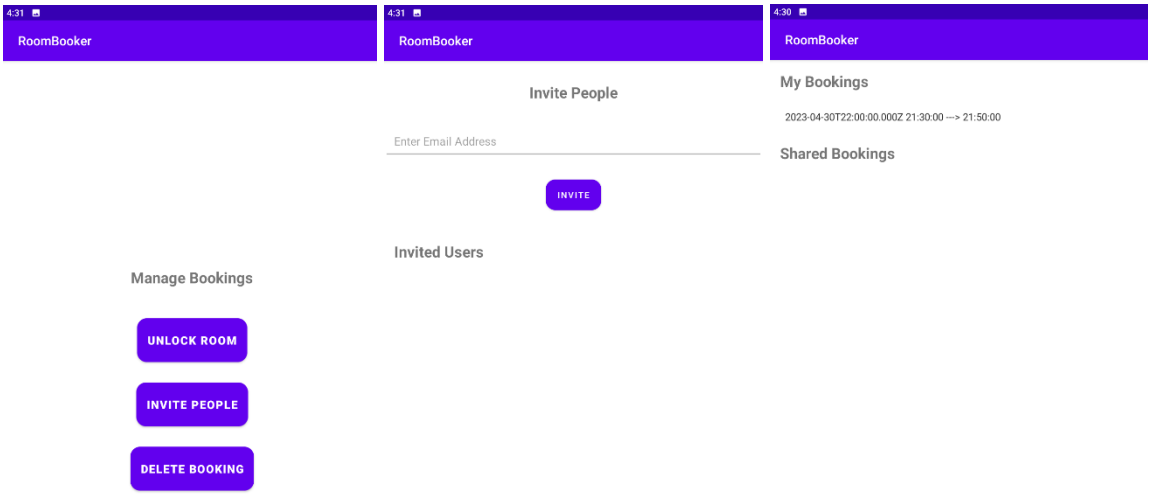


Figure 6 a) *b)* *c)*

4.2 Access Control Application

Figure 7a: In the "View Bookings" section of the proposed room booking application, users can access a button to unlock their reserved room. This functionality will be facilitated by an additional, integrated application that serves as the access control system. The access control application will give users a list of rooms, allowing them to select the one they booked.

Figures 7b and 7c: Upon choosing a room, a red screen will be displayed within the access control application. Users must press the "Unlock" button within the room booking application to unlock the room. The system will then verify whether the user has a valid booking under their name. If the booking is confirmed, the access control application screen will turn green for 10 seconds, indicating that the room is temporarily unlocked. After this period, the room will automatically lock once more.

Figure 7 a) *b)* *c)*

5. Discussion

After a thorough analysis of the results, it is evident that they align well with the project's defined objectives. We have developed a proof-of-concept room booking system with an integrated access control system that meets the essential functional requirements. Throughout the testing period, the problems we initially sought to address, such as double bookings and unauthorized access, never occurred, indicating the success of our solution. This work contributes to understanding room booking and access control systems by showcasing the benefits of integrating these functionalities, which could inspire further research and development.

In terms of functional requirements, the system allows users to securely log in using Mobile BankID, ensuring the protection of user data. Additionally, the room booking application provides an intuitive interface for users to search for available rooms, create and manage bookings, add other users to a booking, and ensures that only authorized users can unlock booked rooms, enhancing security. The access control application is designed to focus on simplicity, facilitating the unlocking of doors through the room booking application. It serves as a visual indicator of a room's status by displaying either a red or green screen, depending on whether the room is locked or unlocked.

We followed the time plan after some modifications and exceptions, but some non-primary functions were not implemented due to a lack of time. This includes a login function for students without Mobile bankID that would allow them to log in to the application using a generated temporary code. This function was time-consuming to implement, and we decided to devote more time to refining the primary functions.

A key aspect that sets our system apart from others is the integration of room booking and access control functionalities. Compared to the traditional, separate systems, this integrated approach offers several advantages, such as streamlined communication and reduced errors and inefficiencies. As previously discussed with the IT department at Halmstad University, existing systems that lack such integration may face issues arising from the need to manage and synchronize information across separate platforms.

It is essential to acknowledge the limitations of our proof-of-concept system. It has yet to be tested in a university setting, making it difficult to directly compare with more established systems regarding performance and scalability. Additionally, we lacked user feedback during the development process, which might have led to unaddressed usability issues. Moreover, while the proposed proof-of-concept room booking and access control system allows users to manage bookings, it does not include an administration system, which could be an area for future improvements. However, the system provides an API that can be a foundation for developing an administration system. These limitations should be considered when interpreting our findings and considering their implications. Nevertheless, our work is a valuable starting point for further research and development, providing a foundation for future improvements and refinements.

5.1 Societal Aspect

From a societal perspective, the successful integration of room booking and access control systems offers various benefits that can positively impact educational institutions and other organizations. By streamlining the room booking process, reducing errors and inefficiencies, and enhancing security, the system could lead to improved resource management and allocation. This could result in better utilization of available spaces and increased overall productivity.

Furthermore, adopting BankID as an authentication mechanism for the room booking and access control system aligns with the global trend towards digitalization and the phasing out of physical cards. As educational institutions and other organizations continue to modernize, there is a growing trend toward implementing technology-driven solutions, and BankID is a perfect example of this.

Integrating BankID into the system eliminates the need for physical student cards, leading to multiple benefits. For one, the reduced reliance on physical cards aligns with the shift toward sustainable practices. With less biological waste from discarded or lost cards, the digital solution is more eco-friendly, reducing the institution's environmental footprint.

Implementing this integrated system could promote collaboration and communication among students, faculty, and staff. The ability to invite others to a booking fosters a more inclusive environment, encouraging teamwork and knowledge sharing. This can contribute to a more vibrant and collaborative academic community.

6. Conclusion

In conclusion, this bachelor thesis successfully demonstrated the development of a proof-of-concept room booking system with an integrated access control system that meets essential functional requirements. The project's unique contribution lies in integrating room booking and access control functionalities, offering a streamlined approach compared to traditional, separate systems. The benefits of this integrated approach include improved communication and reduced errors, specifically targeting double bookings and unauthorized access to study rooms.

Despite the limitations, such as not being tested in a university setting and needing more user feedback during development, the project is a valuable starting point for further research and development in room booking and access control systems. It provides a foundation for future improvements and refinements to enhance the system's functionality and usability.

In light of the project's outcomes, future work should address the system's limitations by conducting tests in real-world university settings and gathering user feedback to identify and resolve potential usability issues. Additionally, implementing the non-primary functions not included in the current version, such as the temporary code login feature for students without Mobile BankID, would be beneficial for making the system more accessible to a broader range of users.

7. References

[1] The University of Southern Queensland. (2016). Academic success: A guide for university students. Study Space.

[2] Agarwal, A., George, M., Jeyaraj, A., & Schwarzkopf, M. (2021). Retrofitting GDPR compliance onto legacy databases. Proceedings of the VLDB Endowment, 15(4), 958-970.

[3] Connolly, T. M., & Begg, C. E. (2005). Database systems: A practical approach to design, implementation, and management. Pearson Education.

[4] Drake, J. D., & Worsley, J. C. (2002). Practical PostgreSQL. O'Reilly Media, Inc.

[5] Harrington, J. L. (2016). Relational database design and implementation. Morgan Kaufmann.

[6] DuBois, P. (2008). MySQL. Pearson Education.

[7] BankID. (2021). BankID relying party guidelines (Version 3.8) [PDF document]. Retrieved from https://www.bankid.com/assets/bankid/rp/bankid-relying-party-guidelines-v3.8.pdf

[8] Jacobson, D., Brail, G., & Woods, D. (2012). APIs: A strategy guide. O'Reilly Media, Inc.

[9] IBM. (n.d.). API. Retrieved March 2, 2023, from https://www.ibm.com/topics/api

[10] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., & Berners-Lee, T. (1999). Hypertext transfer protocol--HTTP/1.1 (No. rfc2616).

[11] Jamison, D. C. (2003). Structured Query Language (SQL) fundamentals. Current Protocols in Bioinformatics, Chapter 9, Unit 9.2. doi: 10.1002/0471250953.bi0902s00. PMID: 18428713.

[12] IBM. (n.d.). SQL vs. NoSQL Databases: What's the Difference? Retrieved March 3, 2023, from [https://www.ibm.com/cloud/blog/sql-vs-nosql

[13] Educative.io. (n.d.). What is Object Relational Mapping? Retrieved March 3, 2023, from [URL not provided]

[14] Arnold, K., Gosling, J., & Holmes, D. (2005). The Java programming language. Addison-Wesley Professional.

[15] DiMarzio, J. (2016). Beginning Android Programming with Android Studio. John Wiley & Sons.

[16] Tilkov, S., & Vinoski, S. (2010). Node.js: Using JavaScript to build high-performance network programs. IEEE Internet Computing, 14(6), 80-83.

[17] Masse, M. (2011). REST API design rulebook: Designing consistent RESTful web service interfaces. O'Reilly Media, Inc.

[18] Express.js. (n.d.). Express - Node.js web application framework. Retrieved March 4, 2023, from https://expressjs.com

[19] Microsoft. (n.d.). Learn - Visual Studio Code. Visual Studio Code. Retrieved March 6, 2023, from https://code.visualstudio.com/learn

[20] Brown, P. C., Roediger III, H. L., & McDaniel, M. A. (2014). Make it stick: The science of successful learning. Harvard University Press.

[21] Smith, P. (2012). Professional website performance: Optimizing the front end and back end (1st ed.). Wrox. ISBN: 978-1118487525.

[22] Baker, M. (2022). OAuth2. In Secure Web Application Development: A Hands-On Guide with Python and Django (pp. 351-397). Berkeley, CA: Apress.

[23] Kjeldskov, J., & Paay, J. (2010). Indexicality: Understanding mobile human-computer interaction in context. ACM Transactions on Computer-Human Interaction (TOCHI), 17(4), 1-28.

[24] King, T. M., Ghansah, I., & Ahuja, S. (2020). Smart Campus: Integrating Information and Communication Technologies in Higher Education Institutions. CRC Press.

[25] Mazemap. (n.d.). Home. Retrieved March 10, 2023, from https://www.mazemap.com/

[26] Halmstad University. (n.d.). Group rooms and study spaces. Retrieved April 3, 2023, from https://www.hh.se/student/innehall-a-o/hitta-pa-campus/grupprum-och-studieplatser.html#h-SabokardurumiMazemap

[27] Computerworld. (n.d.). How to choose desk booking software for the hybrid workplace. Retrieved from https://www.computerworld.com/article/3628210/how-to-choose-desk-booking-software-for-the-hybrid-workplace.html

[28] AFPA. (n.d.). Room booking systems: a comparative study. Retrieved from https://www.afpa.admin.cam.ac.uk/files/roombookingreport.pdf

[29] Freja eID. (n.d.). Get Freja eID. Retrieved May 30, 2023, from https://frejaeid.com/en/get-freja-eid/