LAB University of Applied Sciences

Optimizing Amenity Usage in Residential Complexes

A Web-Based System for Booking and Managing Shared Facilities

LAB University of Applied Sciences

Bachelor of Technology, Industrial Information Technology

2025

Yoseph Tibebu

Abstract

Author	Publication type	Completion year
Yoseph Tibebu	Thesis, UAS	2025
	Number of pages	
	33	

Title of the thesis

Optimizing Amenity Usage in Residential Complexes

A Web-Based System for Booking and Managing Shared Facilities

Degree, Field of Study

Bachelors (UAS), Industrial information Technology

Abstract

This thesis presents a web-based reservation system for managing shared apartment amenities such as laundry and sauna facilities. It replaces manual booking methods that often-caused conflicts and limited access. The new system improves convenience and coordination by allowing real-time booking through a user-friendly calendar interface. Developed using React for the frontend and Firebase for authentication and real-time data management, the system disables past and reserved slots to prevent scheduling issues. Findings show that this web application improves the use and management of shared facilities, offering a practical solution for both residents and administrators.

Keywords

apartment amenities, reservation system, web application, Firebase, real-time booking, React, shared facilities

Contents

Abbreviation	Full Form
2FA	Two-Factor Authentication
API	Application Programming Interface
CSS	Cascading Style Sheets
CCPA	California Consumer Privacy Act
GDPR	General Data Protection Regulation
JS	JavaScript
MFA	Multi-Factor Authentication
PWA	Progressive Web Application
RBAC	Role-Based Access Control
SSL	Secure Sockets Layer
TLS	Transport Layer Security
UI	User Interface
WCAG	Web Content Accessibility Guidelines

1 Introduction

As apartment living becomes more common in urban settings, the demand for efficient management of shared facilities such as laundry rooms and saunas has grown significantly. In many buildings, outdated manual reservation methods create confusion and frustration among residents, often leading to double bookings and disputes over usage times.

This thesis presents a modern web-based reservation system designed to improve the coordination of shared apartment amenities. Using React.js for the user interface and Firebase for backend services such as authentication and real-time data storage, the system allows residents to view availability, make bookings, and avoid scheduling overlaps. The core aim of this project is to develop and evaluate a digital platform that supports fair access, enhances usability, and addresses the practical challenges of shared facility management in residential settings.

2 Background

The advancement of technology has transformed many aspects of residential property management, including how shared amenities such as laundry rooms, saunas, gyms, and pools are accessed. Traditionally, tenants booked these facilities using physical logbooks or sign-up sheets methods that are prone to errors, double bookings, and limited accessibility.

As apartment populations increase, manual reservation systems become inefficient, frustrating both tenants and property managers. These systems lack real-time updates, enforceable booking rules, and scalability features that are critical in modern residential settings. To address these challenges, digital solutions have emerged, particularly web-based reservation systems. These platforms allow tenants to view available slots and make bookings online, with added benefits like booking restrictions, real-time updates, and user authentication enhancing both fairness and transparency.

This study presents the development of a web-based reservation system tailored for apartment buildings. Built with React and Firebase, the system provides an intuitive interface, real-time availability, usage limits and secure access. It replaces outdated manual methods with a modern, efficient approach to facility booking. By improving accessibility, fairness, and administrative ease, the system aims to streamline how shared apartment amenities are managed, offering a practical model for integrating digital tools into everyday residential life.

2.1 Problem Statement

In many apartment buildings, shared amenities like laundry rooms, gyms, and saunas are still managed using manual reservation systems typically physical sign-up sheets. While simple, these methods become inefficient and error prone as resident numbers and demand grow. Manual systems lack real-time availability, are prone to disorganization or loss, and do not support features like booking limits. This often leads to confusion, double bookings, and unfair access, frustrating both tenants and managers. Moreover, enforcing usage policies manually is difficult, and the absence of real-time updates prevents residents from reliably knowing availability.

As buildings scale, managing bookings manually becomes increasingly unsustainable. Facility managers struggle to maintain order and transparency, resulting in reduced satisfaction and operational inefficiencies. To address these challenges, a web-based reservation system offers a modern, scalable solution. With features like real-time updates, usage restrictions, and

digital records, such systems improve fairness, transparency, and ease of access. By leveraging technologies like React for the frontend and Firebase for backend services, the proposed system streamlines booking, enhances user experience, and simplifies facility management. This study aims to develop and evaluate such a web-based reservation system, addressing the problems inherent in traditional manual booking methods and providing a more effective solution for managing shared apartment amenities.

2.2 Objectives of the Study

The primary objective of this study is to design, develop, and evaluate a web-based system for managing the reservation of shared amenities in apartment buildings. Specifically, the study seeks to:

- develop a user-friendly platform for booking shared facilities such as laundry rooms and saunas through a web-based interface accessible to residents at any time
- implement a real-time reservation system that allows tenants to view the availability of amenities and make bookings directly from the interface, eliminating the need for physical booking sheets
- incorporate booking restrictions within the system to ensure equitable access by limiting usage (e.g., the system restricts the number of hours per week a tenant can book certain amenities such as laundry)
- provide an intuitive calendar view that displays available time slots for each facility, helping users navigate and select their desired booking times with ease
- implement a secure user authentication system that allows tenants to create and man age accounts, ensuring that only registered users can access reservation features, and that personal information is protected
- evaluate the system's performance in terms of reliability, efficiency, and scalability to ensure it handles multiple users and bookings without compromising speed or accuracy
- explore potential future improvements such as integrating payment systems or developing a mobile application to enhance the system's functionality and reach

Through these objectives, the study aims to address current challenges in managing facility reservations and provide a digital solution that streamlines the process for both tenants and apartment managers.

2.3 Scope of the Study

This study focuses on the design, development, and evaluation of a web-based system for managing shared amenities in apartment buildings, aiming to enhance convenience, efficiency, and fairness in the reservation process. The system is intended for apartment residents who require access to facilities such as laundry rooms and saunas, with an emphasis on improving their interaction with the platform. Core functionalities include real-time booking, a calendar interface for intuitive navigation, secure user authentication, and built-in usage limits for instance maximum of four hours per week for laundry and one hour per week for sauna reservations.

The platform is developed using React for the frontend and Firebase for backend services, including authentication and data storage. Implementation is currently limited to selected residential buildings, with no immediate plans for broader deployment. The study does not cover mobile application development or payment integration, nor does it extend to general property management functions. System evaluation focuses on usability and performance, with scalability identified as a potential area for future enhancement. Overall, the study presents a targeted digital solution to streamline amenity reservation in residential settings.

2.4 Significance of the Study

This study presents a digital solution to the inefficiencies of manual facility reservations in apartment buildings. By introducing a web-based booking system, it delivers operational improvements for both tenants and property managers.

The key contributions include:

- Operational Efficiency**:** Automates reservations, reducing administrative work and eliminating common errors like double bookings.
- Improved Resident Experience**:** Enables convenient, real-time bookings, promotes fairness through usage restrictions, and minimizes resident conflicts.
- Transparency and Fairness**:** Displays real-time availability and enforces booking limits, ensuring equitable access to shared amenities.
- Cost-Effectiveness**:** Reduces the need for manual oversight and helps avoid facility overuse, saving time and administrative costs.
- Scalability**:** Designed to support more residents and additional amenities as needed, making it adaptable for future expansion.

- Technology Innovation**:** Demonstrates the effective use of React and Firebase in modern property management.
- Foundation for Future Enhancements**:** Provides a base for future integration of features like payments, mobile access, and analytics.

In summary, the system offers a scalable, user-focused solution that modernizes facility management and sets a model for digital transformation in residential housing.

2.5 Structure of the Study

This thesis is organized into eight chapters, each addressing a specific aspect of the research on the development of a web-based reservation system for shared apartment amenities. The structure follows a logical progression from identifying the problem to designing, implementing, and evaluating a solution

Chapter 1**:** Introduction

This chapter introduces the study by outlining the challenges of managing shared apartment amenities using traditional, manual methods. It presents the problem statement, research objectives, scope, and significance, thereby establishing the need for a web-based reservation system.

Chapter 2: Literature Review

This chapter reviews existing literature on facility reservation systems, with a focus on apartment management. It covers the evolution of manual and digital systems, highlights key features and challenges.

Chapter 3: System Design and Development

This chapter details the system design and development process. It covers the functional and non-functional requirements, system architecture, and technologies employed in the creation of the reservation platform.

Chapter 4: System Design and Architecture

This chapter describes the implementation phase, including both frontend and backend development. It explains the use of React for building the user interface and Firebase for authentication and data management. It also discusses development challenges and how they were addressed.

Chapter 5: System Implementation

This chapter outlines the implementation of the system, including its architecture, technologies used, and key design decisions. It also addresses major development challenges, and the solutions applied to ensure a secure and functional reservation platform.

Chapter 6: Testing and Evaluation

This chapter presents the testing and evaluation of the system, confirming that key functionalities such as user authentication, booking management, and real-time data updates perform reliably and meet the intended objectives.

Chapter 7: Discussion

This chapter highlights the benefits, limitations, and planned enhancements of a web-based apartment facility reservation system, focusing on improvements in user experience, efficiency, security, and scalability, along with future upgrades like offline access, advanced features, and a mobile app.

Chapter 8: Summary of Findings

This final chapter highlights the key results of the study, showing how the digital system improved booking efficiency, user experience, and facility management. It also outlines current limitations and suggests future enhancements to expand the system’s functionality.

3 Literature Review

3.1 Facility Reservation System

Facility reservation systems are digital platforms that simplify the booking of shared resources, such as meeting rooms or apartment amenities like laundry rooms and saunas. These systems automate the reservation process, offering a more efficient, accessible, and organized way for users to schedule facility usage. Over time, these systems have advanced from basic manual methods to sophisticated web-based solutions that support multiple users, real-time updates, and customizable restrictions.

3.2 Web-Based Systems in Apartment Management

Web-based systems have transformed apartment management by digitizing and centralizing key operations such as facility reservations, rent collection, maintenance tracking, and resident communication. These platforms enhance efficiency, reduce administrative burden, and offer greater convenience for both property managers and tenants. By enabling real-time access and automation, web-based systems provide a scalable and user-friendly solution for modern apartment living.

3.3 Key Features of Modern Facility Reservation Systems

Modern web-based facility reservation systems include essential features that significantly enhance booking efficiency and user experience. They offer real-time availability checks, allowing users to instantly see open slots and avoid overbooking without needing to contact managers. Automated booking confirmations are sent via email or in-app notifications to keep users informed of their reservations. Administrators can enforce user restrictions and policies, such as limiting hours per week or differentiating access between residents and guests. A visual calendar view simplifies navigation for users and helps administrators monitor bookings easily. Additionally, these systems support multiple users simultaneously, ensuring fair access to facilities and minimizing scheduling conflicts.

3.4 Challenges and limitations

Despite their advantages, facility reservation systems face several challenges:

- System Complexity**:** Developing and maintaining these platforms requires technical expertise and can be challenging to integrate with existing systems.
- User Adoption**:** Not all users are tech-savvy, and some may resist switching from manual methods without adequate support or training.
- Internet Dependence**:** Web-based systems rely on stable internet connectivity, which can be a barrier in areas with poor access; offline support requires additional development.
- Security and Privacy: These systems must rigorously protect user data and booking information from unauthorized access and breaches.

In summary, while facility reservation systems enhance efficiency and user experience, addressing these challenges is crucial for successful adoption and long-term sustainability.

4 System Design and Architecture

4.1 System Requirements

The development of a web-based reservation system for apartment amenities requires a detailed understanding of both functional and non-functional requirements. These requirements ensure the system meets the needs of both residents and the property managers while providing a smooth, efficient, and secure platform for facility booking. Below are the system requirements for the web-based reservation system developed for managing apartment amenities, such as laundry and sauna.

4.2 Functional Requirements

Functional requirements define the essential features the system must support to ensure it performs effectively and meets user needs. The system must provide secure user authentication management, enabling both residents and administrators to register, log in, and manage their profiles. Residents should be able to create accounts, view their booking history, and update profile settings, while administrators can manage user accounts, approve or reject bookings, and configure system settings.

A secure password reset function must also be included for users who forget their credentials. Additionally, the platform should offer a dynamic calendar interface that displays real-time facility availability, allowing residents to view and reserve time slots according to predefined restrictions. A complete facility overview should be available, listing all reservable amenities along with individual time slot options and booking limits.

Booking Restrictions and Validation

To ensure fair use, the system must enforce rules such as time limits, booking frequency, and availability checks. Each facility should have a maximum usage duration per user per week to prevent overuse. Additionally, limits on how often a facility can be reserved must be applied to promote equal access for all users. To further ensure fairness, real-time slot validation during reservations should be implemented, preventing double-booking by confirming the availability of the facility before finalizing any booking.

Booking Confirmations and Notifications

The system must notify users upon successful booking and provide timely reminders. Email or in-app notifications should automatically be sent for booking confirmations, cancellations, and reminders. Users are alerted at least one day before their scheduled reservation.

Booking modifications and cancelations

Users must have the flexibility to change or cancel bookings within a defined time window. They can reschedule reservations if the new time slot is available, or cancel them within the allowed timeframe, thereby freeing up the slot for others.

Reporting and Admin Dashboard

A centralized dashboard allows administrators to efficiently monitor and manage the reservation system. They can view, approve, or reject bookings, track facility usage and occupancy trends, and generate reports on peak hours, user activity, and overall system performance.

4.2.1 Non-functional Requirements

Non-functional requirements define the quality attributes and operational constraints of the system. These ensure the platform performs reliably, securely, and efficiently under various conditions.

Usability

The interface must be intuitive and easy to navigate for both residents and administrators, regardless of their technical background. It should follow a mobile-first design, ensuring responsiveness and full functionality on smartphones and tablets. Additionally, the platform must be accessible to users with disabilities, adhering to basic Web Content Accessibility Guidelines (WCAG).

Performance

The system must load quickly and handle bookings in real-time to ensure a smooth user experience. Key operations such as viewing availability and submitting bookings should occur within 1 to 2 seconds. Additionally, the system should be scalable, capable of supporting an increasing number of users and facilities without any decline in performance.

Availability and Reliability

The platform must ensure high availability with minimal downtime, targeting a system uptime of at least 99.5%. It should also be robust in handling unexpected inputs and system failures, providing clear and appropriate error messages to maintain usability.

Security

User data must be safeguarded against unauthorized access and security breaches. All sensitive information should be encrypted both in transit and at rest. Secure login mechanisms must include password hashing, with optional two-factor authentication (2FA) for added protection. Additionally, role-based access control must be implemented to manage permissions across different user types.

Maintainability

The system should be easy to update, debug, and enhance as requirements evolve. Its architecture must support a modular codebase to simplify maintenance and future development. Comprehensive documentation should also be maintained to assist both developers and administrators.

4.3 System Architecture and Design

The system is built on a cloud-based architecture to ensure scalability, flexibility, and accessibility (Google Firebase a). It leverages modern technologies like React.js and Firebase to deliver a responsive and real-time user experience.

Visual Studio Code and IntelliJ IDEA are used as the main development environments due to their robust support for JavaScript and React. Their features, including integrated terminals, intelligent code completion, and plugin ecosystems, significantly streamlined the development process. (Microsoft, JetBrains.)

The architecture will consist of the following components:

4.3.1 Frontend Architecture

- Technology: React.js is utilized to build the frontend, offering a component-based approach for dynamic user interfaces (React a).

- Responsibilities: The frontend serves as the user interface for residents and administrators, providing a dynamic and responsive experience.

Structure:

- UI Components: Built using React and Styled Components to offer an intuitive layout, including elements like calendars, booking slots, and user profiles.
- State Management: React's built-in useState and useContext hooks manage the application state and data flow across components (React b).
- Routing: React Router facilitates seamless navigation between pages such as Home, Booking, and Profile (React Router).
- User Authentication: Firebase Authentication handles user registration, login, and session management.

4.3.2 Backend Architecture

The backend architecture will leverage:

- Technology: Firebase services, including Firestore and Firebase Functions.
- Responsibilities: Managing user authentication, real-time database operations, and cloud storage.

Structure

The backend will include the following key components:

- User Authentication: Firebase Authentication ensures secure and scalable login mech anisms (Google Firebase b).
- Real-Time Data: Firebase Firestore stores and manages user data, booking reservations, and logs, enabling real-time synchronization across all devices (Google Firebase c).
- Business Logic: Firebase Functions implement business rules, such as booking constraints and time slot management (Google Firebase d).

4.3.3 API Layer

The API layer will facilitate communication between the frontend and backend, ensuring smooth data transfer and interaction between the user interface and the database.

- Components

The API layer will include the following components:

- RESTful API/SDK: Used to send requests from the frontend to the backend and retrieve booking and user data.
- Authentication & Authorization: Secure API endpoints manage access, ensuring only authorized users can make bookings or access data.

4.3.4 Database Architecture

The database architecture is designed to ensure efficient data storage and real-time access.

- Technology: Firebase Firestore serves as the cloud database solution.
- Responsibilities: Storing user data, reservation details, and logs while offering real-time synchronization to keep all users updated instantly.

Structure

The database will be organized as follows:

- Data Model: Designed to store user profiles, booking slots, and logs to support reserva tion tracking and system performance monitoring.
- Real-Time Updates: Firestore's real-time data capabilities ensure users and administrators always have the most current booking and availability information (Google Firebase e).

In conclusion, the system will combine these components to create a cloud-based, scalable, and efficient solution for apartment building amenities. By addressing both functional and non-functional requirements, the system will ensure a seamless, secure, and efficient experience for residents, property managers, and administrators.

4.3.5 Security Architecture

To ensure that sensitive data is protected, several security measures are incorporated into the system architecture:

- Authentication: Firebase Authentication ensures that only authorized users can log in. This includes secure password management, multi-factor authentication (MFA), and token-based authentication.

- Data Encryption: All data transmitted between the client and server is encrypted using SSL/TLS protocols. Sensitive data, such as passwords and payment information, is encrypted both in transit and at rest. (OWASP Foundation.)
- Role-Based Access Control (RBAC): The system defines user roles (e.g. Residents, admin) to restrict access to certain features and data. Administrators can manage the entire system, while residents only have access to their booking data.
- Data Privacy Compliance: The system ensures compliance with relevant data protection regulations (e.g., GDPR, CCPA). User consent is obtained for data collection, and users can delete their accounts or modify their data as needed. (W3C.)

The figure below is an illustration of how the overall system structure is built.

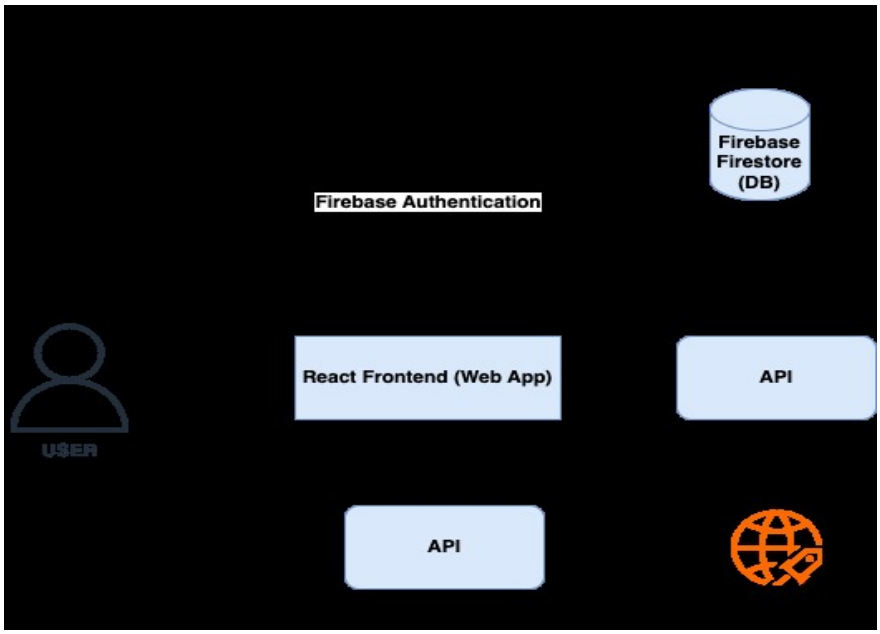

Figure 1. Overall System Architecture of the Facility Booking Platform

4.4 System Components

The system is composed of several key components that work together to enable the efficient booking and management of amenities. The frontend developed using React.js, which handles the user interface, including booking management, user authentication, and navigation. (React c.)

The backend, powered by Firebase, is responsible for managing authentication, the real-time database operations, and cloud storage. An API layer facilitates smooth communication between the frontend and backend, ensuring reliable data exchange the database implemented using Firebase Firestore, stores user profiles, booking information, and logs, all while providing real-time synchronization. Together, these components form an integrated and seamless system that delivers a responsive and user-friendly experience.

4.5 Scalability and Future Enhancements

The system is designed to scale as the number of residents and facilities increases. Firebase's cloud infrastructure allows the system to scale without worrying about managing servers or physical infrastructure

The modular nature of the architecture allows easy integration of new features, such as:

- Expanded facility types can be easily added to the system as the property grows, including options such as event rooms and fitness centres.
- Advanced analytics may be integrated into future versions of the system to track user behaviour, booking trends, and facility utilization.
- Although the web-based system is mobile-responsive, a dedicated mobile application could be developed in the future to enhance user engagement and enable features like push notifications.

4.6 Frontend Design

The frontend of the web-based reservation system is designed with a strong emphasis on user experience, responsiveness, and ease of use. Developed using React, the interface delivers dynamic interactions tailored for both residents and administrators. To ensure the user interface adhered to usability standards, Nielsen Norman Group’s ten usability heuristics served as a practical guideline. These heuristics helped in evaluating navigation clarity, user control, and consistency throughout the application. (Nielsen Norman Group.)

The layout is fully responsive, adapting seamlessly to various screen sizes, including desktops, tablets, and mobile devices, to ensure consistent usability (MDN Web Docs a). Navigation is clean and intuitive, featuring a simple navigation bar that provides quick access to essential sections such as booking, profile management, and the admin dashboard. A visual calendar allows users to view available time slots for each facility and make bookings effortlessly. Once a booking is completed, a confirmation message appears on the screen, providing immediate feedback and clarity to the user.

User Flow

The current version of the reservation system is designed with simplicity and clarity in mind, focusing solely on essential booking functionality. Users can log in, view available facilities, and select a time slot to make a reservation. Once a booking is successfully completed, a confirmation message is displayed on their dashboard. Users also have access to a section where they can view their upcoming reservations. If they exceed the allowed booking limit, a message will appear informing them of the restriction. Features such as profile management or email reminders are not included in the current scope and are considered for future enhancements. The diagram below illustrates the user flow for facility booking, showing the step-by-step process from login to reservation confirmation.

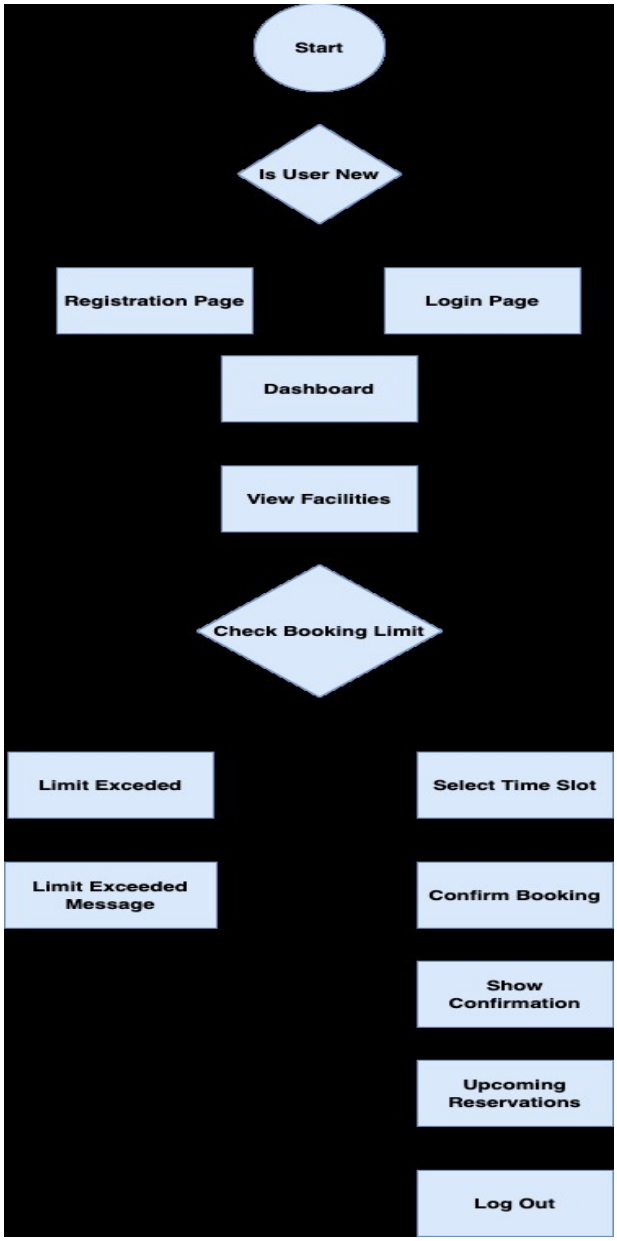

Figure 2. User Flow Diagram

UI Components

The user interface is kept minimal and user-friendly, providing only the components necessary for core functionality. A basic booking form allows users to select a facility, choose a date and time, and submit their reservation. A dashboard displays booking confirmations and a list of upcoming reservations. Additionally, messages appear on the dashboard to inform users if they have exceeded their weekly booking limit. This focused design ensures that users can interact with the system easily and without unnecessary distractions.

Styling and Aesthetics

The styling and aesthetics of the system are designed to offer a clean and modern user experience. It follows contemporary design principles, featuring a minimalistic colour scheme and intuitive icons that make navigation simple and effective. Styling the interface relied heavily on best practices and examples provided by MDN Web Docs. The CSS documentation helped implement responsive designs and maintain visual consistency across devices. (MDN Web Docs b.)

Layouts are user-friendly, with forms and buttons arranged logically to facilitate smooth interactions and streamline the booking process. In conclusion, the frontend design is crafted to provide a seamless, responsive, and efficient experience for residents, while also considering potential enhancements for property administrators in the future. To ensure faster load times and a better user experience, performance optimization techniques recommended by Google Web.dev were carefully considered during the development process (Google Web.dev).

4.7 Backend Design

The backend of the web-based reservation system is responsible for handling business logic, user authentication and data storage. Built using Firebase, it ensures scalability, real-time data synchronization, and secure operations. Below is a summary of the backend design:

Core Components

Firebase Authentication serves as a core component of the system, managing both user registration and login processes. It supports secure email and password authentication, ensuring that only authorized users can access the platform. User sessions are maintained using secure tokens, allowing users to remain authenticated throughout their interaction with the system without repeated logins.

Firebase Firestore Database

Firebase Firestore acts as the primary real-time database for the application, storing and managing critical data such as user profiles, bookings, facilities, and configuration settings. Its real-time capabilities ensure that any updates made to the data are instantly reflected across all connected clients.

The database is structured into organized collections such as Users, Bookings, and Facilities making it easy to query, filter, and retrieve relevant information efficiently. When a user submits a booking request, the backend performs necessary validations, including checking facility availability and enforcing time-based restrictions, before storing the booking data.

Firebase Functions

Firebase Functions handle the business logic processing of the application, executing server-side operations such as validating booking requests, checking facility availability, and managing cancellations. When a resident initiates a booking, the backend verifies that the selected time slot is available and that the request complies with booking rules, such as avoiding double-booking and staying within permitted usage limits. Additionally, users can view their own dashboard and booking history, enabling them to track their activity within the system.

Security and Data Privacy

The system prioritizes security and data privacy by implementing robust protection mechanisms. All data stored in Firebase Firestore is encrypted both in transit and at rest, ensuring that sensitive user information is securely handled. Access to data is controlled through role-based access rules, where residents can only view and interact with their own information. These restrictions are enforced using Firebase's built-in security rules to prevent unauthorized access and maintain data confidentiality (OWASP Foundation).

4.8 Scalability and Performance

The backend design is built for scalability and responsiveness, leveraging Firebase's cloud infrastructure to automatically handle growing numbers of users and bookings without compromising performance. Firestore's real-time synchronization ensures that users receive immediate updates on booking statuses and facility availability, eliminating the need for manual page refreshes. In conclusion, the backend architecture provides a secure, efficient, and scalable foundation that supports seamless communication with the frontend. The use of Firebase technologies ensures that the system remains both high-performing and easy to maintain.

5 System Implementation

5.1 System Architecture

The application uses a client-server architecture, where the front-end (client) communicates with the back end (Firebase services) via API calls. The front-end is built with React Native and deployed as a web- app, while Firebase handles authentication, database storage (Firestore), and hosting. For implementing client-side routing, the React Router Crash Course by Traversy Media on YouTube was consulted. It provided practical demonstrations and clarified complex routing concepts within React applications. (Traversy Media 2020.)

- React Native Frontend**:** User interface for residents to book facilities.
- Firebase Authentication**:** Manages user login and sign-up.
- Firebase Firestore**:** Stores user data and booking records.
- Firebase Hosting (if used): Hosts the app dashboard or admin view (if implemented later).
- Firebase Functions (future extension): For handling logic like sending reminders.

5.1.1 Implementation Process

The system is implemented in a series of stages:

- Frontend Development**:** The React.js interface is built, focusing on dynamic components, user navigation, and booking management.
- Backend Setup**:** Firebase is configured for user authentication, real-time database, and cloud storage.
- API Integration**:** A RESTful API is developed to manage communication between the frontend and backend.
- Testing and Deployment**:** The system undergoes through testing for functionality, usabil ity and performance before being deployed.

Technologies Used

- Frontend uses React.js, Styled Components, React Router
- Backend uses Firebase Authentication, Firebase Firestore, Firebase Functions
- API uses Custom RESTful API for data interaction between frontend and backend

Security and Data Handling

- Authentication**:** Secure login and session management using Firebase Authentication.
- Data Storage**:** User and booking data stored securely in Firebase Firestore with real-time synchronization.
- Encryption**:** All sensitive data is encrypted using industry-standard protocols.
- Access Control**:** Role-based access ensures that only authorized users can view sens itive information.

5.2 Design Considerations

The following design principles guided the development:

- Simplicity and Usability: A clean and intuitive interface.
- Offline Functionality: The app works without continuous internet access and syncs data when online.
- Booking Restrictions: Each facility has weekly usage limits (e.g., 4 hours for laundry).
- Scalability: Designed to support additional amenities or buildings in the future.

5.3 Challenges and Solutions in Implementation

The development and implementation of the web-based reservation system for apartment amenities presented several challenges that required careful planning and problem-solving. Below are the key challenges encountered during the project, along with the corresponding solutions that were implemented to address them.

5.3.1 Real time Data Synchronization

Challenge:

One of the major challenges is ensuring that the system can provide real-time updates to users, especially when booking availability changed. This was crucial to prevent double-booking of facilities and ensure accurate reservation data.

Solution:

To address this, the system leveraged Firebase Firestore's real-time database features. By utilizing Firestore's real-time synchronization, any changes made in the database, such as new bookings are immediately reflected on all connected devices. This ensured that users could always see up-to-date availability and avoid booking conflicts.

5.3.2 Booking Restrictions and Conflicts

Challenge:

Handling booking restrictions, such as time limits for specific facilities (e.g., 4 hours per week for laundry), posed a significant challenge. It was crucial to ensure that residents could not exceed these limits, and that double-booking or overbooking did not occur.

Solution:

To solve this, the system uses business logic on the backend using Firebase Functions. When a resident attempted to make a booking, the system checks the user's past bookings to verify if the requested facility time had been exceeded. If the booking violated the time restrictions or if the facility was already reserved for the selected time, the system would prevent the booking from being made and notify the user. This was done by querying the Bookings collection in Firestore**,** ensuring that the rules were respected in real-time.

5.3.3 User Authentication and Security

Challenge:

Ensuring that user data was securely managed, and that the authentication process was seamless while providing access for residents is a key concern. Additionally, implementing secure login methods and protecting sensitive data posed significant challenges.

Solution:

The system uses Firebase Authentication to handle user registration, login. Firebase provided secure methods such as email/password authentication.

5.3.4 Ensuring Data Privacy and Compliance

Challenge:

Ensuring that user data was kept private, and that the system complied with data protection regulations (e.g., GDPR) is a critical challenge, especially when handling sensitive personal information.

Solution:

Robust data protection measures are implemented using Firebase's security rules, ensuring that only authorized users can access sensitive data. The implementation of the web-based reservation system involves a series of challenges, each obstacle being overcome through careful planning, the use of reliable technologies like Firebase, and continuous testing.

6 Testing and Evaluation

6.1 Testing Strategy

The testing strategy for the web-based reservation system focused on ensuring that all functional requirements were met and the system operated as intended. Rather than conducting a formal multi-layered testing approach, the emphasis was placed on verifying the core functionalities and ensuring seamless user interactions. The primary goal was to ensure that each component, from user interface elements to backend processes, performed as expected, delivering a reliable and user-friendly experience.

6.2 Testing Results

The testing process confirmed that the system's primary functionalities were fully operational and met the outlined requirements. During the development phase, key features such as user authentication, booking management, and data synchronization were manually validated to ensure smooth operation across both the frontend and backend. Minor adjustments were made to improve the user experience based on informal testing.

6.3 System Testing

System testing focused on confirming that the core workflows, such as making bookings and managing facility reservations, functioned correctly. Through basic manual checks, it was verified that users could interact with the system intuitively, and critical functions such as booking time slots and receiving booking confirmations operated without issues. The system was also tested in various scenarios to ensure there were no disruptions in user tasks.

6.4 Evaluation of System Effectiveness

The system effectively meets its intended goals by delivering a smooth user experience and reducing administrative burden. It’s simple, intuitive interface has been positively received, supporting ease of use for both residents and future administrators. Basic functionality tests, conducted during development, confirmed that the system performs reliably and efficiently under typical usage conditions. While no major issues were observed, a few minor refinements were identified to further improve overall usability.

7 Discussion

System Benefits

The web-based reservation system for managing apartment facilities offers several key benefits, primarily enhancing the user experience and improving the booking process. These benefits include:

Improved User Experience

The developed system significantly enhances the user experience by offering a range of features that improve accessibility and convenience. Firstly, the platform is easy to use, allowing residents to quickly view facility availability, make reservations, and receive instant confirmation notifications. This streamlined process reduces the complexity often associated with traditional booking methods. Additionally, the system provides convenience by enabling users to make reservations from any internet-connected device, removing the need for physical sign-ups or relying on intermediaries. Real-time updates further improve usability by ensuring that residents always have access to the latest availability status, effectively preventing double bookings or scheduling conflicts.

Increased Efficiency

The system enhances operational efficiency by automating several key tasks and streamlining the reservation process. Booking confirmations, cancellations, and notifications are fully automated, significantly reducing the need for manual intervention and minimizing administrative workload. Users have instant access to their bookings, allowing them to view, manage, or modify reservations without relying on physical records or assistance. Furthermore, the system proactively prevents scheduling conflicts by blocking double bookings, ensuring a smoother and more reliable facility reservation experience for all residents.

Enhanced Data Security

The system incorporates robust security measures to protect user information and maintain data integrity. Personal and booking details are securely encrypted, ensuring that sensitive information remains private and protected from unauthorized access. Additionally, access control mechanisms are in place, allowing users to view and manage only their own data. This en-

sures a high level of confidentiality and builds user trust in the system's ability to safeguard their information.

Time and Resource Saving

The system contributes to significant time and resource savings by minimizing manual tasks and optimizing operations. Automation reduces the effort and time traditionally spent on booking management, resulting in improved efficiency. By eliminating the need for physical records, the system supports a paperless workflow that is both environmentally friendly and cost-effective. Additionally, users have quick access to their booking history and related reports, removing the need for manual tracking and enabling easier management of past reservations.

Scalability and Flexibility

The system is designed with scalability and flexibility in mind, making it adaptable to future demands. Built using Firebase, it can seamlessly accommodate an increasing number of users and booking volumes without requiring significant structural changes. Its architecture supports long-term growth and ensures that the system remains relevant by adapting to evolving user needs and technological advancements.

Cost Savings

The implementation of the system leads to notable cost savings through improved efficiency and reduced errors. Automated processes minimize the likelihood of mistakes such as double bookings, thereby reducing user dissatisfaction and potential conflicts. The system enhances operational efficiency by streamlining bookings and notifications, which lowers labour costs. Real-time availability updates further contribute to cost-effectiveness by preventing overbooking and ensuring clear and accurate scheduling.

Conclusion

This web-based reservation system greatly enhances the booking experience for users by automating processes, providing real-time updates, and ensuring data security. Its scalability and flexibility make it an adaptable solution that can evolve with future needs. The system improves communication, reduces errors, and results in cost savings, offering long-term value for residents and ensuring efficient management of apartment facilities.

7.1 Limitations of the System

The current version of the system offers basic functionality aimed at improving the facility reservation experience. However, several advanced features are not yet implemented. Below are the key limitations of the system, with a focus on future enhancements:

- User Authentication: Basic login and registration are in place, but authentication relies solely on standard credentials without options for enhanced security or recovery.
- Facility Booking: Users can reserve available time slots, but the system lacks advanced filtering, booking modifications, or preference settings.
- Booking Management: Upcoming reservations are viewable, but users cannot edit or cancel bookings.
- Offline Access: The system requires an internet connection; no offline capabilities are currently supported.
- Administrator Role: There is no admin interface for managing bookings, user accounts, or system settings.
- Notifications: Users do not receive booking confirmations, reminders, or alerts.

Future development will address these limitations to enhance functionality, improve user experience, and introduce administrative controls.

7.2 Future Enhancements

While the current web-based reservation system has improved user experience and operational efficiency, several enhancements are planned to expand functionality and support future scalability:

Administrator Dashboard: A role-based admin panel will allow for user management, facility oversight, and reporting on usage patterns and maintenance needs.

Offline Functionality: Users will be able to view and make bookings without internet access. Data will sync automatically once online.

Notification System: Booking confirmations, reminders, and cancellations will be delivered via push notifications, email, and SMS.

Advanced Filtering & Calendar Views: Users will filter time slots by facility type and time range, with a flexible calendar offering daily, weekly, and monthly views.

User Account Enhancements: Profile customization, secure password recovery, and improved privacy settings will create a more personalized user experience.

Analytics & Insights: Administrators will gain access to data on usage trends and user behaviour to optimize resources and personalize recommendations.

Mobile App Development: A dedicated mobile app for iOS and Android will provide full platform functionality with offline support and mobile-optimized features.

These upgrades will transform the system into a robust, user-cantered platform with improved management capabilities and long-term adaptability.

8 Summary of Findings

The web-based reservation system significantly enhances the booking process, user experience, and management of shared apartment facilities. Transitioning from manual to digital reservations has improved transparency, automation, and scalability. Key benefits include a user-friendly interface with real-time availability, a color-coded calendar, and responsive design, all of which simplify navigation and reduce booking conflicts. Operational efficiency has also improved through automated confirmations and cancellations, minimizing administrative workload.

The system's modular architecture supports future enhancements such as administrator dashboards, offline access, and integrated payments. Data security is ensured through encryption, secure authentication, and role-based access control, aligning with privacy standards. However, the current version lacks some features, including an admin interface, offline functionality, and a notification system—identified as priorities for future development. Overall, the system marks a significant advancement in facility management, offering a strong foundation for continued innovation and expanded functionality.

8.1 Contributions to Knowledge

8.1.1 Recommendations

To further enhance the web-based reservation system, key improvements are recommended. Incorporating offline functionality will allow users to access bookings without internet connectivity, increasing reliability in low-network areas. A notification system should be introduced to send automated alerts for booking confirmations, reminders, and cancellations, improving user engagement and reducing missed appointments. Administrator access is essential for managing users, resolving conflicts, and generating usage reports to optimize resources.

Advanced filtering options and enhanced calendar views will streamline the booking process, while analytics tools will offer valuable insights into user behaviour and facility utilization. Strengthening security measures and ensuring compliance with privacy standards will protect user data and maintain trust. Scalability should be prioritized to accommodate future growth and platform integration. Finally, establishing a user feedback loop will support continuous improvement and ensure the system evolves in line with resident needs.

References

React – Official Documentation
ReactJS. n.d. React documentation. Retrieved on 06 May 2024. Available at https://reactjs.org/docs/getting-started.html

React Router – Declarative Routing for React
React Router. n.d. React Router documentation. Retrieved on 13 May 2024. Available at https://reactrouter.com/en/main

React Hooks – useEffect, useState, etc.
ReactJS. n.d. Introducing Hooks. Retrieved on 20 May 2024. Available at https://reactjs.org/docs/hooks-intro.html

JavaScript – Mozilla Developer Network (MDN)
MDN Web Docs. n.d. JavaScript documentation. Retrieved on 27 May 2024. Available at https://developer.mozilla.org/en-US/docs/Web/JavaScript

CSS – Mozilla Developer Network (MDN)
MDN Web Docs. n.d. CSS documentation. Retrieved on 03 June 2024. Available at https://developer.mozilla.org/en-US/docs/Web/CSS

Firebase – Google Developers
Google Firebase. n.d. Firebase documentation. Retrieved on 10 June 2024. Available at https://firebase.google.com/docs

Firebase Authentication
Google Firebase. n.d. Firebase Authentication. Retrieved on 17 June 2024. Available at https://firebase.google.com/products/auth

Visual Studio Code – Microsoft Docs
Microsoft. n.d. Visual Studio Code documentation. Retrieved on 24 June 2024. Available at https://code.visualstudio.com/docs

IntelliJ IDEA – JetBrains
JetBrains. n.d. IntelliJ IDEA documentation. Retrieved on 01 July 2024. Available at https://www.jetbrains.com/idea/

W3C – Web Accessibility Initiative (WAI)

W3C. n.d. Web Accessibility Initiative (WAI). Retrieved on 08 July 2024. Available at https://www.w3.org/WAI/

Nielsen Norman Group – Usability Heuristics

Nielsen Norman Group. n.d. Ten Usability Heuristics for User Interface Design. Retrieved on 15 July 2024. Available at https://www.nngroup.com/articles/ten-usability-heuristics/

MDN – Progressive Web Apps (PWA)

MDN Web Docs. n.d. Progressive Web Apps (PWA). Retrieved on 22 July 2024. Available at https://developer.mozilla.org/en-US/docs/Web/Progressive_web_apps

OWASP Top Ten – Web Application Security Risks

OWASP Foundation. n.d. OWASP Top Ten 2021. Retrieved on 13 January 2025. Available at https://owasp.org/www-project-top-ten/

Google Web.dev – Performance Optimization

Google Web.dev. n.d. Web Performance Optimization. Retrieved on 20 January 2025. Available at https://web.dev/fast/

YouTube – React Router Tutorial

Traversy Media. 2020. React Router Crash Course. Retrieved on 17 February 2025. Available at https://www.youtube.com/watch?v=Law7wfdg_ls

Appendix 1. Project Documentation

This appendix provides a brief overview of the project structure, along with key details not covered in the GitHub repository. The full source code and additional documentation are available in the GitHub repository linked below.

GitHub Repository:

https://github.com/yostib/reservation_system.git

Appendix 2. Firebase Firestore Database Structure

This section describes the structure of the Firestore database used in the project.

- Users Collection
    - userId (String): Unique identifier for the user.
    - email (String): User's email address.
    - name (String): User's full name.
    - role (String): User's role (e.g., 'admin', 'resident').
- Bookings Collection
    - bookingId (String): Unique identifier for the booking.
    - userId (String): Reference to the user who made the booking.
    - facilityId (String): Reference to the booked facility.
    - startTime (Timestamp): Start time of the booking.
    - duration (Number): Duration of the booking in minutes.
- Facilities Collection
    - facilityId (String): Unique identifier for the facility.
    - name (String): Name of the facility.
    - type (String): Type of facility (e.g., 'laundry', 'sauna').
    - bookingLimits (Object): Constraints on bookings (e.g., max hours per week).

For the full Firestore structure, see:

https://console.firebase.google.com/u/0/project/reservation-sys-718bf/firestore/databases/-default-/data/~2Fbookings~2F0jio3Or5c7Eb8cMVJzfl

Appendix 3. Glossary of Key Terms

Term	Definition
User Interface (UI)	The visual part of an app users interacts with.
Application Programming Interface (API)	A set of rules that lets apps talk to each other.
Firebase	A Google platform for building web/mobile apps.
Calendar Interface	A visual tool to pick dates or time slots.
React	A JavaScript library for building UI components.
Real-time Database	A database that updates instantly with changes.
Progressive Web App (PWA)	A web app with features like a native mobile app.
Role-Based Access Control (RBAC)	Access control based on user roles.