

**VIETNAM GENERAL CONFEDERATION OF LABOUR
TON DUC THANG UNIVERSITY
FACULTY OF INFORMATION TECHNOLOGY**



**NGUYEN THIEN HUY – 521H0072
NGUYEN DINH VIET HOANG - 522H0120**

MOBILE APPLICATION FOR ROOM MANAGEMENT AND RENTAL

FINAL REPORT MOBILE APPS DEVELOPMENT

HO CHI MINH CITY, YEAR 2024

**VIETNAM GENERAL CONFEDERATION OF LABOUR
TON DUC THANG UNIVERSITY
FACULTY OF INFORMATION TECHNOLOGY**



**NGUYEN THIEN HUY – 521H0072
NGUYEN DINH VIET HOANG - 522H0120**

MOBILE APPLICATION FOR ROOM MANAGEMENT AND RENTAL

FINAL REPORT MOBILE APPS DEVELOPMENT

Advised by

PhD. LE VAN VANG

HO CHI MINH CITY, YEAR 2024

ACKNOWLEDGEMENT

We sincerely thank PhD. Le Van Vang for teaching us the Mobile Apps Development course with great enthusiasm. We want to express our deep appreciation for the dedication and professional knowledge that you shared with us. Through your classes, we gained a better understanding of the fundamental aspects of the Mobile Apps Development, thanks to your detailed explanations and practical applications. You helped us grasp the knowledge and apply it effectively. Finally, we extend our heartfelt gratitude to PhD. Le Van Vang for your commitment and invaluable support throughout our learning journey in this course. The skills and knowledge we acquired will continue to impact our future development. We sincerely thank you and wish your health, success, and happiness.

Ho Chi Minh City, December 12, 2024

Authors:

Nguyen Thien Huy

Nguyen Dinh Viet Hoang

DECLARATION OF AUTHORSHIP

We hereby declare that this thesis was carried out by ourselves under the guidance and supervision of PhD. Le Van Vang; and that the work and the results contained in it are original and have not been submitted anywhere for any previous purposes. The data and figures presented in this thesis are for analysis, comments, and evaluations from various resources by our own work and have been duly acknowledged in the reference part.

In addition, other comments, reviews and data used by other authors, and organizations have been acknowledged, and explicitly cited.

We will take full responsibility for any fraud detected in my thesis. Ton Duc Thang University is unrelated to any copyright infringement caused on my work (if any).

Ho Chi Minh City, December 12, 2024

Authors:

Nguyen Thien Huy

Nguyen Dinh Viet Hoang

MOBILE APPLICATION FOR ROOM MANAGEMENT AND RENTAL

ABSTRACT

This study presents a comprehensive mobile application designed for Room Rental Management, addressing the critical need for an efficient, user-friendly platform to facilitate interactions between landlords, tenants, and administrators. The proposed system leverages modern mobile technologies and Firebase Firestore database architecture to create a robust, secure, and scalable solution for managing rental properties.

The application implements a sophisticated role-based access control mechanism, differentiating between three primary user roles: Landlord, Tenant, and Administrator. Each role is equipped with specific functionalities tailored to their unique requirements. Landlords can post, manage, and modify room listings, while tenants can search, view, and book rooms. Administrators maintain comprehensive system oversight and user management capabilities.

Key features of the application include a secure login and authentication system, intuitive user interface, advanced room listing management, and a flexible booking and reservation system. The development utilizes Firebase Authentication for secure user access and Firestore for real-time data synchronization and storage. The system supports critical operations such as user profile management, room listing updates, and cross-platform data import/export functionalities.

TABLE OF CONTENTS

LIST OF FIGURES	ix
CHAPTER 1. PROJECT INTRODUCTION	1
1.1 Room Rental Management.....	1
1.2 System Scope	1
1.3 Detailed System Description.....	1
1.4 Application Objectives.....	2
CHAPTER 2. THEORETICAL SURVEY	4
2.1 What is Firebase Firestore?	4
2.2 What Is Firebase Realtime Database?.....	6
2.3 Firestore Database Architecture.....	6
2.4 Firebase Authentication	7
2.5 Realtime Database vs. Firestore.....	7
2.6 Query and Filter Functions in Firestore	8
2.7 Real-Time Data Update and Synchronization.....	9
2.8 Security and Access Control in Firestore.....	9
2.9 Data Import and Export in Firestore	10
2.10 Firestore SDK and REST API	10
2.11 Firestore Indexing	10
2.12 Development and Management Tools for Firestore.....	11

CHAPTER 3. ROOM RENTAL MANAGEMENT APPLICATION ON MOBILE PLATFORM..... 12

3.1 Requirements Analysis	12
3.2 System Scope	12
3.3 UML Use Case Diagram.....	13
3.4 UML Entity Relationship Diagram.....	16
3.5 Implementation	17
3.5.1 The Login Form	17
3.5.2 The Signup Form.....	18
3.5.3 The Homepage for Landlord.....	19
3.5.4 Post A Room For Rent Page	20
3.5.5 Manage Posted Rooms Page.....	21
3.5.7 See History Chat of Landlord	23
3.5.8 Logout button	24
3.5.9 The Homepage for Tenant.....	25
3.5.10 The Search Page for Tenant.....	26
3.5.11 The date selection page for Tenant	27
3.5.12 The notice page for Tenant after choosing the date to see the room	28
3.5.13 The list of the loved rooms of Tenant	29
3.5.14 The information and functions of each room for Tenant	30
3.5.15 See History Chat of Tenant	31
3.5.16 The confirmation page of Tenant	32
3.5.17 Reviews and comments site for room renters.....	33
3.5.18 Room deposit page for tenants	34

CHAPTER 4. CONCLUSION	35
4.1 Achieved Outcomes	35
4.2 Limitations	35
4.3 Suggestions for Future Enhancements	36
REFERENCES	38

LIST OF FIGURES

Figure 2.1: Document-based architecture	5
Figure 2.3: Firestore Database Architecture in real life.....	6
Figure 2.4: Compare Realtime Database and Firestore	8
Figure 3.3: Use Case Diagram	13
Figure 3.4: Entity Relationship Diagram	16
Figure 3.5.1: Login activity	17
Figure 3.5.2: Signup activity	18
Figure 3.5.3: Homepage's Landlord	19
Figure 3.5.4: Post a room for rent page	20
Figure 3.5.5.a: Manage posted rooms page when no rooms are posted yet	21
Figure 3.5.5.b: Manage posted rooms page when page has a room is posted	22
Figure 3.5.7: See history chat page.....	23
Figure 3.5.8: Logout button in Homepage.....	24
Figure 3.5.9: Homepage's Tenant.....	25
Figure 3.5.10: The Search Page for Tenant	26
Figure 3.5.11: The date selection page for Tenant	27
Figure 3.5.12: The notice page for Tenant after choosing the date to see the room.....	28
Figure 3.5.13: The list of the loved rooms of Tenant	29
Figure 3.5.14: The information of each room for Tenant.....	30
Figure 3.5.15: See History Chat of Tenant	31
Figure 3.5.16: The confirmation page of Tenant	32
Figure 3.5.17: The reviews and comments site of Tenant	33
Figure 3.5.18: Room deposit page for tenants	34

CHAPTER 1. PROJECT INTRODUCTION

1.1 Room Rental Management

The Room Rental and Management Application is a meticulously designed system aimed at facilitating easy and efficient management and connection between landlords and tenants. This system allows for managing room information, supporting search capabilities, scheduling room viewings, and conducting rental transactions transparently and conveniently.

1.2 System Scope

The Room Rental and Management System includes the following main components:

1. Login and Account Management
2. Room Rental Posting
3. Room Search and Filtering
4. Appointment Management
5. Deposit and Payment System
6. User Support

1.3 Detailed System Description

The system is developed with 17 main features, serving two primary user groups: Landlords and Tenants.

Main Features:

For Landlords:

- **Room Rental Posting:** Allows landlords to post detailed information about rooms, including images, rental prices, address, and amenities.
- **Vacancy Management:** Update and manage room status to prevent double bookings.

- **Information Confirmation:** Feature to confirm information with tenants to ensure accuracy.
- **Contract Management:** Create and manage rental contracts easily.
- **Payment Management:** Track and receive payments from tenants.

For Tenants:

- **Room Search:** Quick search based on criteria such as location, rental price, area, and amenities.
- **Detailed Information Display:** View comprehensive information about rooms, including images and detailed descriptions.
- **Direct Contact:** Connect with landlords directly within the app.
- **Room Viewing Scheduling:** Book room viewings online.
- **Reviews and Comments:** Share experiences after renting a room.

Additional Features:

- **Safe Deposit System**
- **Favorites List**
- **New Room Notifications**
- **Payment Reminders**
- **Issue Reporting**
- **Online Support**
- **Neighborhood Information**

1.4 Application Objectives

The main objectives of the application are:

- Create a reliable platform connecting landlords and tenants
- Simplify the process of finding and renting rooms
- Ensure transparency and safety in transactions
- Provide a convenient and efficient user experience

The application is designed to address challenges in room searching and renting, offering a comprehensive solution for both landlords and tenants.

CHAPTER 2. THEORETICAL SURVEY

2.1 What is Firebase Firestore?

Cloud Firestore is an innovative, cloud-hosted NoSQL database from Google that allows developers to manipulate their data with ease. It uses intuitive and familiar features for storing, syncing, and querying data for web and mobile app development.

Firestore offers convenience as well as scalability with its intelligent caching capability that stores the most frequently used queries, thus minimizing latency when accessing data. Development teams can also benefit from intuitive usage through the integration of both real-time capabilities (sync) and offline support in queries.

Its real-time capabilities keep data updated across all connected clients as soon as changes are made. This means that users will always have access to the most up-to-date available version of their documents no matter where they're stored.

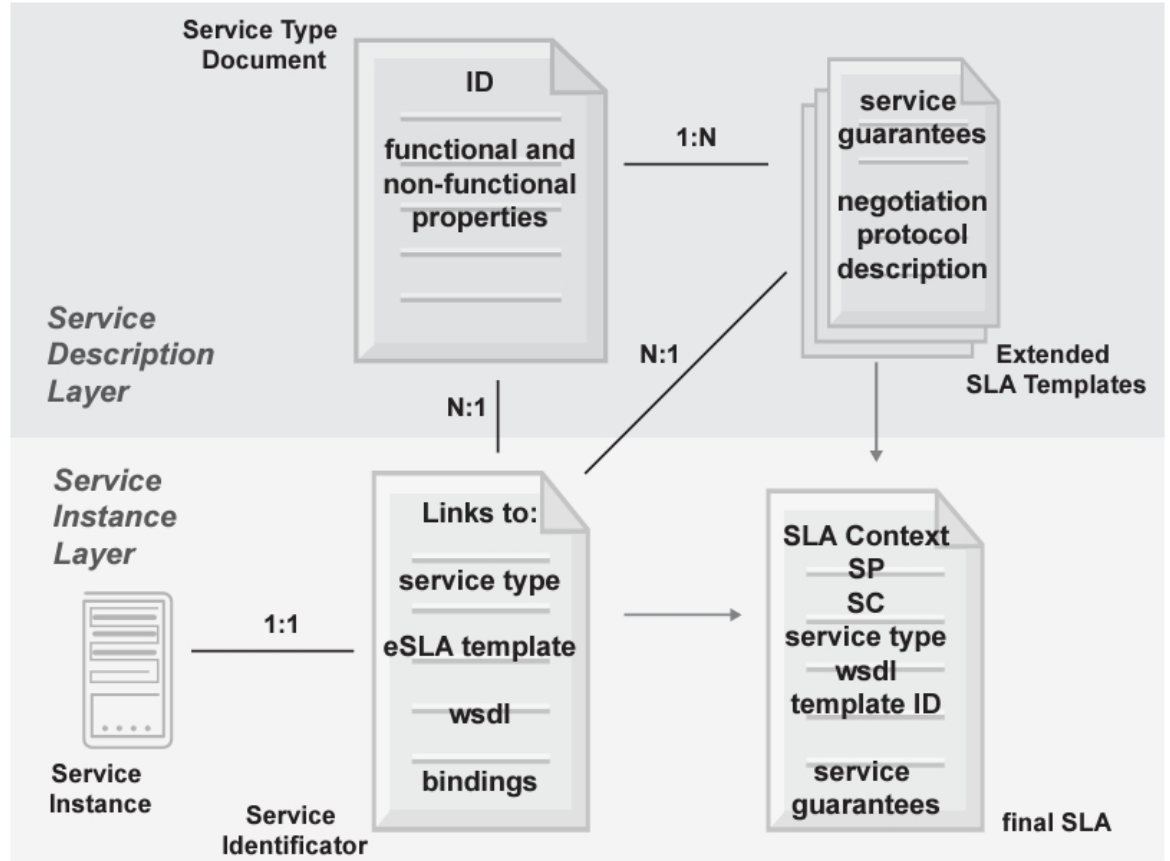


Figure 2.1: Document-based architecture

+ In the Figure 2.1, we can see that:

- **Collections:** Collections contain documents of the same type. For example, the "users" collection contains documents representing users.
- **Documents:** Each document is a data record, which can contain many different fields. For example, a document in the "users" collection can contain fields such as "name", "email", "address", etc.
- **Fields:** Fields contain data values. For example, the "name" field contains the name of the user.
- **Nested documents:** Documents can be nested to create more complex data structures.

2.2 What Is Firebase Realtime Database?

Realtime Database is a cloud-hosted database service provided by Google Firebase that synchronizes data between the user's device and the cloud without manually refreshing the page.

It allows you to build powerful and complex applications without having to worry about networking or data storage. It also facilitates interactive web and mobile apps, making it easier to manage data and create real-time experiences for users.

This helps businesses easily integrate real-time updates into their applications and enables them to keep customers informed if anything changes with their products or services.

Realtime Database simplifies the way we access and update information from anywhere, creating efficient user experiences across devices.

2.3 Firestore Database Architecture

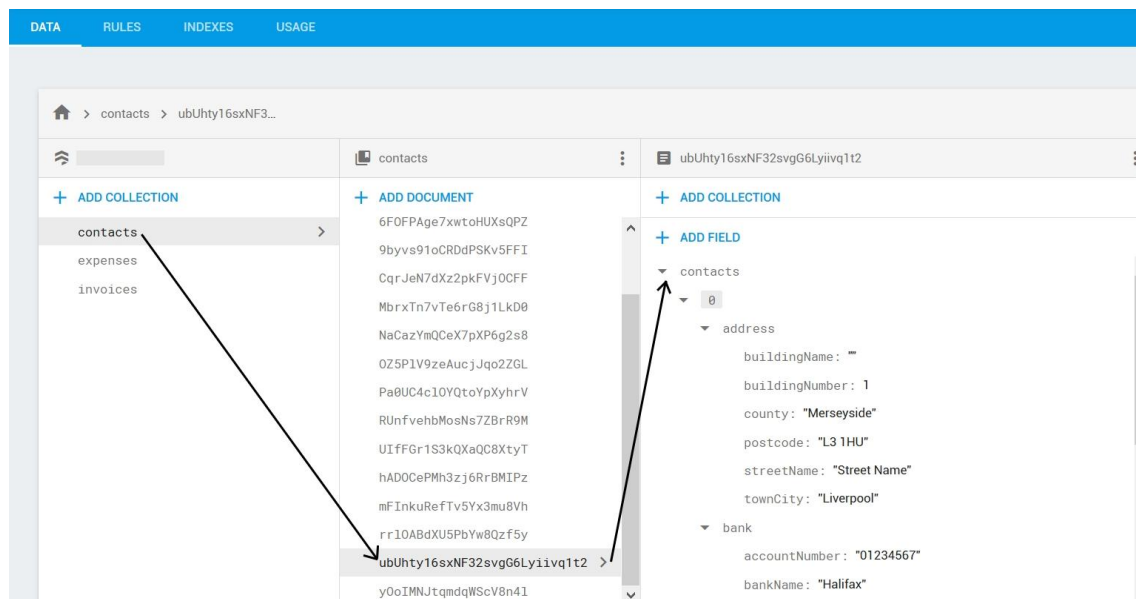


Figure 2.3: Firestore Database Architecture in real life

In the Figure 2.2, we can see each Firebase project can have multiple Firestore databases, each database contains collections, collections contain documents and each document is a data record.

Additional definitions of the keywords:

- Collections: A group of related documents. Each document belongs to a collection, and a collection can contain many documents. Collections are the primary data organization unit in Firestore.
- Documents: The main storage units in Firestore, containing application data as key-value pairs. Each document has a unique ID and can contain multiple fields as well as sub-collections.
- Fields: Attributes within each document, similar to columns in traditional databases. A field can contain basic data types (strings, numbers, booleans) or complex types (arrays, maps).
- Sub-collections: Collections within a document, creating a hierarchical structure between collections and documents. Firestore supports sub-collections to store detailed data related to each document.

2.4 Firebase Authentication

Firebase Authentication is a user authentication service provided by Firebase that helps applications manage and secure user accounts easily. While the project does not require Firebase Authentication, it is a popular solution suitable for implementing login, registration, and session management functions.

Firebase Authentication supports email/password, phone number, Google account, Facebook, and other methods, providing flexibility and security for user access to the application.

2.5 Realtime Database vs. Firestore









 Realtime Database	 Cloud Firestore
 Allows single sorting condition.	 Allows multiple field comparison.
 Only hosted in North America.	 Allows multi regional support.
 Transfer occurs via JSON files.	 Fetching query is organic.

Figure 2.4: Compare Realtime Database and Firestore

From Figure 2.4, we can indicate that Firestore uses a document-collection structure, whereas Realtime Database uses a JSON tree structure. Moreover, Firestore supports stronger queries, handles large-scale data with a hierarchical structure, whereas Realtime Database is mainly suited for applications with simple data structures. In addition, Firestore allows filtering and sorting data based on multiple conditions without downloading the entire dataset to the client.

2.6 Query and Filter Functions in Firestore

Firestore supports complex queries, making it easy to filter and sort data based on different criteria. This is especially useful for managing student lists, searching for students, and sorting lists based on various criteria.

Firestore allows:

- **Conditional Queries:** Filter data based on specific conditions (e.g., finding students with a specific name or ID).
- **Sorting Data:** Sort results by one or more fields.

- **Combined Queries:** Combine multiple conditions to produce the desired result.

Firestore supports real-time queries, meaning that when data changes, connected clients are immediately notified and can update the data.

2.7 Real-Time Data Update and Synchronization

Firestore offers real-time synchronization, allowing updates to be instantly reflected on all connected devices. This feature is ideal for building a real-time student management application, as data will synchronize as soon as changes are made, ensuring that administrators, managers, and staff always have the latest data.

Firestore supports Listeners, which allow the application to listen for changes in a specific collection or document. When data changes, the Listener will automatically update the user interface with the new data.

2.8 Security and Access Control in Firestore

Firestore Security Rules: Firestore provides Security Rules to protect data, allowing you to define access rights for each document and collection based on specific conditions. These rules are applied in real time and can be customized based on the application's requirements.

To implement the system requirements for the three user roles (Admin, Manager, and Employee), Firestore Security Rules can be set up to allow only:

- Admin full access to data, including add, delete, and update permissions.
- Manager restricted access to add, modify, and view student information but no permissions to alter system settings or security configurations.
- Employee view-only access to data without modification rights, except for updating their own profile picture.

Custom Claims: Combined with Firebase Authentication, Firestore supports Custom Claims to grant access rights based on user roles. For example, Custom Claims can label users as Admin, Manager, or Employee and apply corresponding access permissions.

2.9 Data Import and Export in Firestore

Data Import: Firestore allows data import from JSON or CSV files through support tools or programmatically. For a student management application, Admins can import student lists or certificates from files for easier management.

Data Export: Firestore allows data export in formats such as JSON or CSV. This is useful for creating reports, generating statistics, or storing data offline when needed. For the requirement to export student and certificate data, Firestore can be combined with libraries like `papaparse` (for CSV) or `exceljs` to export data to Excel or CSV files easily.

2.10 Firestore SDK and REST API

Firestore provides Firebase SDKs for multiple platforms like web, iOS, and Android, enabling easy integration and development with Firestore. Firestore SDKs support full functionality such as add, edit, delete, and real-time data synchronization.

Additionally, Firestore also offers a REST API, allowing data access and management from server applications or environments without SDK support.

2.11 Firestore Indexing

Firestore automatically indexes each field in documents to support efficient queries. However, custom indexes are needed for complex queries. This optimizes the application's performance when querying large datasets like student lists or certificates.

Composite Index: Firestore allows creating composite indexes to support multi-condition queries, such as sorting students by name and age.

2.12 Development and Management Tools for Firestore

Firestore Console: An online Firestore administration interface, allowing developers to manage the database, view and edit data, set security rules, and monitor system metrics.

Firestore Emulator Suite: Firestore's emulation toolset allows development and testing of Firestore without connecting to the live database. This is particularly useful for testing security rules and authentication before deploying to production.

CHAPTER 3. ROOM RENTAL MANAGEMENT APPLICATION ON MOBILE PLATFORM

3.1 Requirements Analysis

The Room Rental Management mobile application is designed to enable efficient and straightforward management of room rental information. This application focuses on handling room details, landlord and tenant information, with an emphasis on ease of use and user-friendliness. The app allows for quick updates and additions to room listings and supports user profile management and role-based access control across three roles: Landlord, Tenant, and Administrator, ensuring a customized experience for each user group.

3.2 System Scope

The Room Rental Management mobile system provides the following key features:

1. Login and Authentication
2. User Management
3. Room Listing Management
4. Booking and Reservation System
5. Role-based Access Control

The system integrates a secure, user-friendly login system, establishing a clear access hierarchy for each role. Administrator users have powerful management tools, including the ability to update user profile pictures, monitor a comprehensive user list, add new users, and manage personal details such as name, contact information, and account status.

For room management, Landlords and Administrators have access to intuitive tools for handling room listings. Users can view detailed room lists, add or update room information, and use sorting and multi-criteria search functions to retrieve comprehensive information about each room, including images, pricing, and amenities.

Additionally, the system supports the import and export of room listings and tenant information, enhancing data accessibility and integration. Administrators can upload and download listing data using import and export functions, formatting data into Excel or CSV files for user convenience.

The application sets up role-based access control for three primary user roles:

- Landlord: Full control over their room listings, booking management, and profile
- Tenant: Ability to search, view, and book rooms, manage personal profile
- Administrator: Complete system management, user control, and overall platform oversight

3.3 UML Use Case Diagram

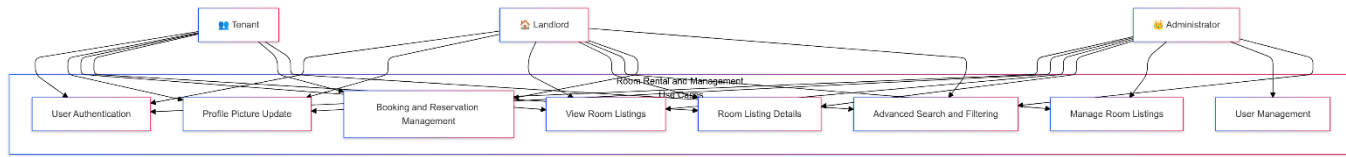


Figure 3.3: Use Case Diagram

Use Case 1: User Authentication

- Actors: Administrator, Landlord, Tenant
- Description: Users log into the system, with their assigned roles determined by their login credentials.
- Preconditions: The system is operational, and users possess valid credentials.
- Postconditions: Users gain access to the system upon successful login.

- Main Flow: Users enter username and password. The system verifies credentials and grants appropriate access based on user role.

Use Case 2: Profile Picture Update

- Actors: Administrator, Landlord, Tenant
- Description: Logged-in users can update their profile pictures.
- Preconditions: Users are signed into the system.
- Postconditions: The user's profile picture is updated to the selected image.

Use Case 3: View Room Listings

- Actors: Administrator, Landlord, Tenant
- Description: Users access a complete list of available rooms within the system.
- Preconditions: Users are logged in.
- Postconditions: Users view an overview of all room listings.

Use Case 4: Room Listing Details

- Actors: Administrator, Landlord, Tenant
- Description: Users open a detailed page showing comprehensive information about a specific room.
- Preconditions: Users are logged into the system.
- Postconditions: Users see full details about the selected room listing.

Use Case 5: Manage Room Listings

- Actors: Landlord, Administrator
- Description: Users manage room records by adding, updating, deleting, and importing/exporting room listing data.

- Key Actions:
 1. Update Room Information
 2. Delete Room Listing
 3. Add New Room
 4. Import/Export Room Listings

Use Case 6: Booking and Reservation Management

- Actors: Tenant, Landlord
- Description: Management of room bookings, including creating, modifying, and canceling reservations.
- Key Actions:
 1. Create New Booking
 2. Modify Existing Reservation
 3. Cancel Booking
 4. Confirm/Reject Booking Requests

Use Case 7: User Management

- Actor: Administrator
- Description: Comprehensive user account management, including user creation, modification, and access control.

Use Case 8: Advanced Search and Filtering

- Actors: Tenant, Landlord, Administrator
- Description: Advanced search capabilities to find rooms based on multiple criteria such as location, price, amenities, and availability.

3.4 UML Entity Relationship Diagram

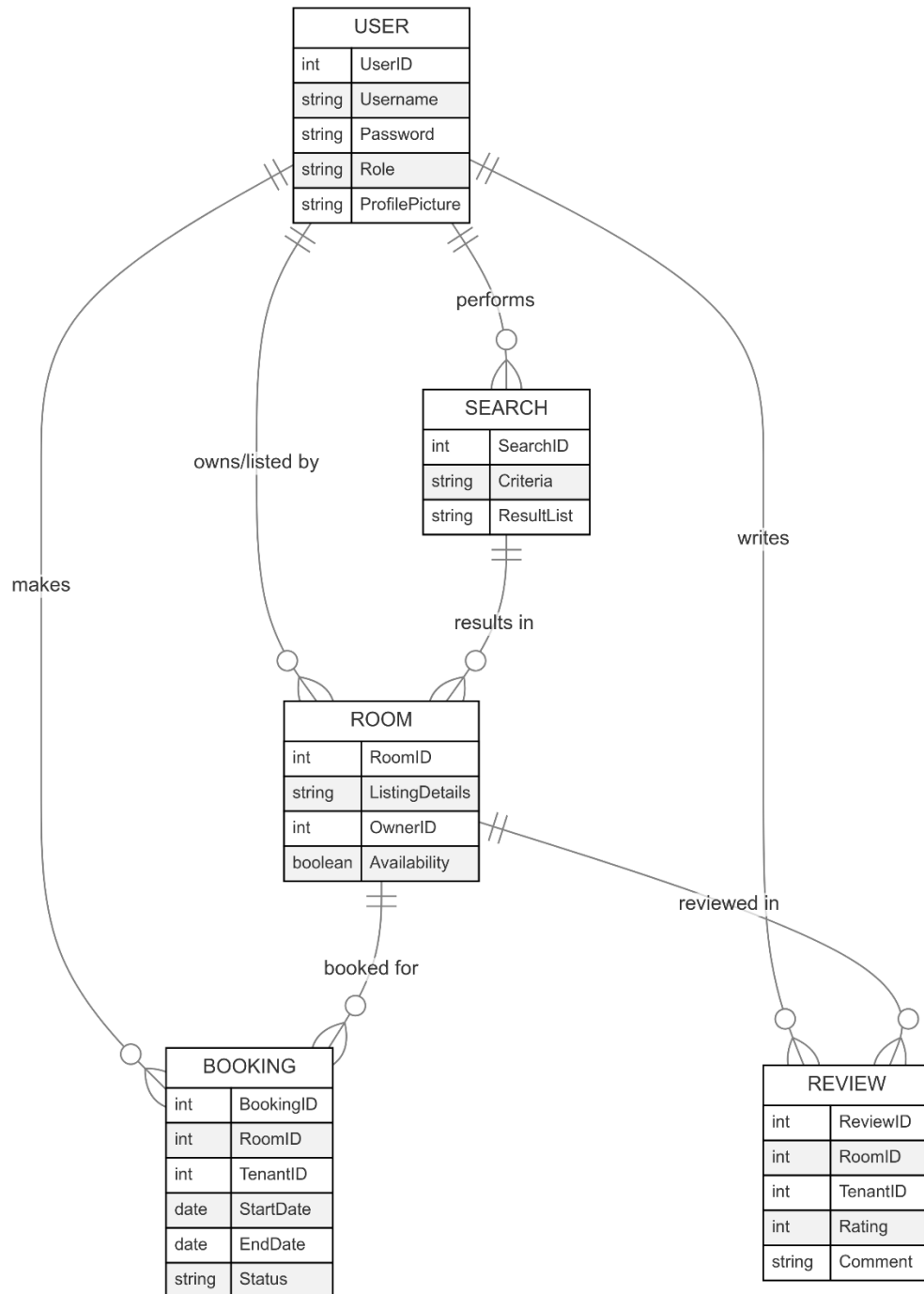


Figure 3.4: Entity Relationship Diagram

3.5 Implementation

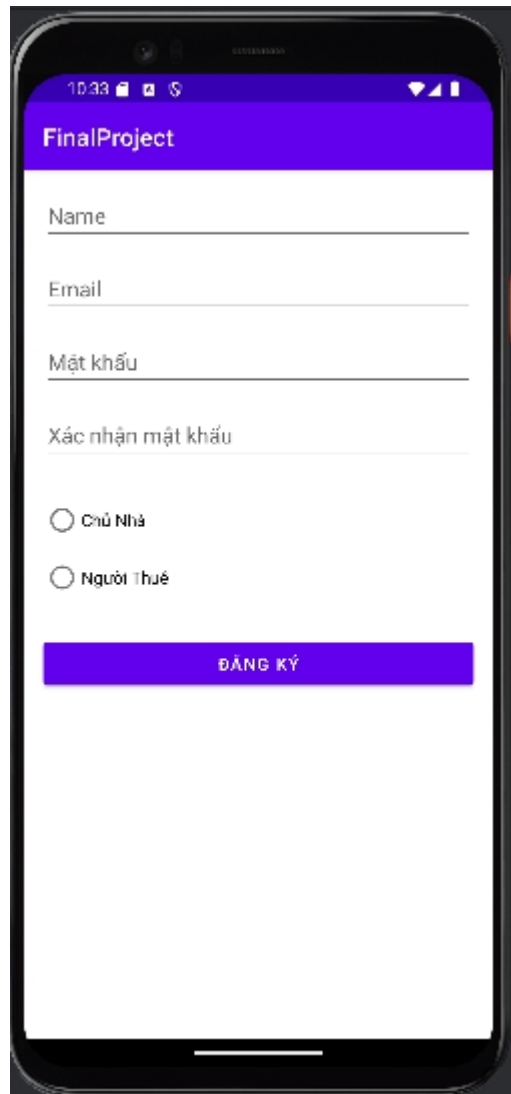
3.5.1 The Login Form



Figure 3.5.1: Login activity

The “login screen” serves as the initial interface upon launching the system. This login form prompts users to input their email and password. Upon entering the correct credentials, users are assigned a specific role corresponding to their email and password. The available roles include Landlord or Tenant (Chủ Nhà hoặc Người Thuê).

3.5.2 The Signup Form



The image shows a mobile application interface for a signup form. At the top, there is a status bar with the time 10:33 and various icons. Below the status bar is a purple header with the text 'FinalProject'. The form itself is white and contains the following elements: a text input field for 'Name', a text input field for 'Email', a text input field for 'Mật khẩu' (Password), and a text input field for 'Xác nhận mật khẩu' (Confirm password). Below these fields are two radio button options: 'Chủ Nhà' (Landlord) and 'Người Thuê' (Tenant). At the bottom of the form is a blue button with the text 'ĐĂNG KÝ' (Sign Up).

Figure 3.5.2: Signup activity

This registration page requires users to enter information such as name, email, password, confirm password and choose one of two roles: Landlord or Tenant.

3.5.3 The Homepage for Landlord

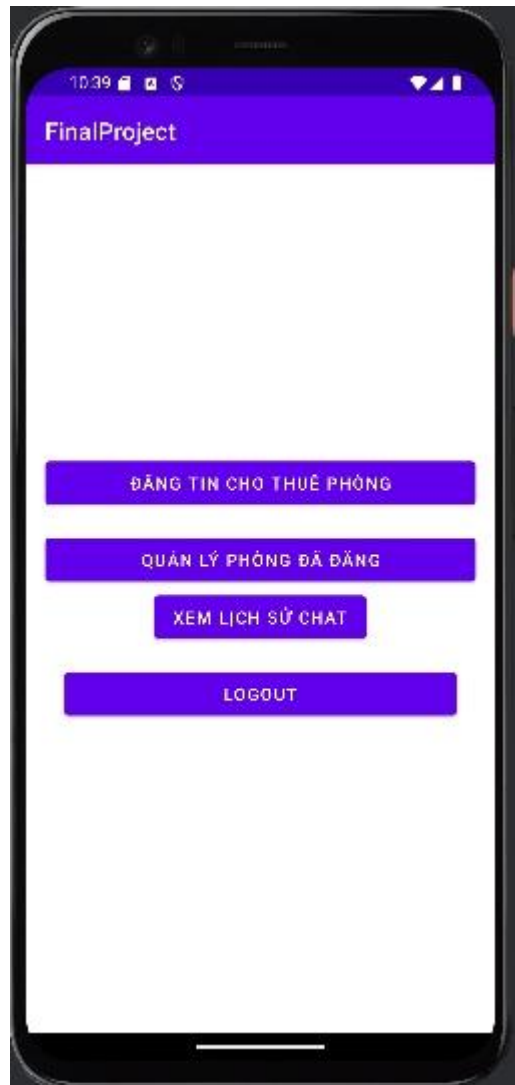


Figure 3.5.3: Homepage's Landlord

The Homepage for Landlord includes the following features: Post a room for rent, Manage posted rooms, View chat history, Log out.

3.5.4 Post A Room For Rent Page

The screenshot shows a mobile application interface for posting a room for rent. The app's name 'FinalProject' is displayed in a purple header bar. The main content area is white and contains several input fields for user data. The fields are labeled in Vietnamese: 'Tên phòng' (Room Name), 'Giá thuê' (Rental Price), 'Địa chỉ' (Address), 'Mô tả' (Description), and 'Tiện ích (Nhập cách nhau bằng dấu phẩy)' (Amenities (Enter separated by commas)). There are two prominent purple buttons: 'CHỌN HÌNH ẢNH' (Choose Photo) and 'THÊM PHÒNG' (Add Room). The interface is clean and modern, with a focus on data entry.

Figure 3.5.4: Post a room for rent page

The Room Listing page requires hosts to enter the following data fields: Room Name, Rental Price, Address, Description, Select Photos, and Amenities (separated by commas).

3.5.5 Manage Posted Rooms Page

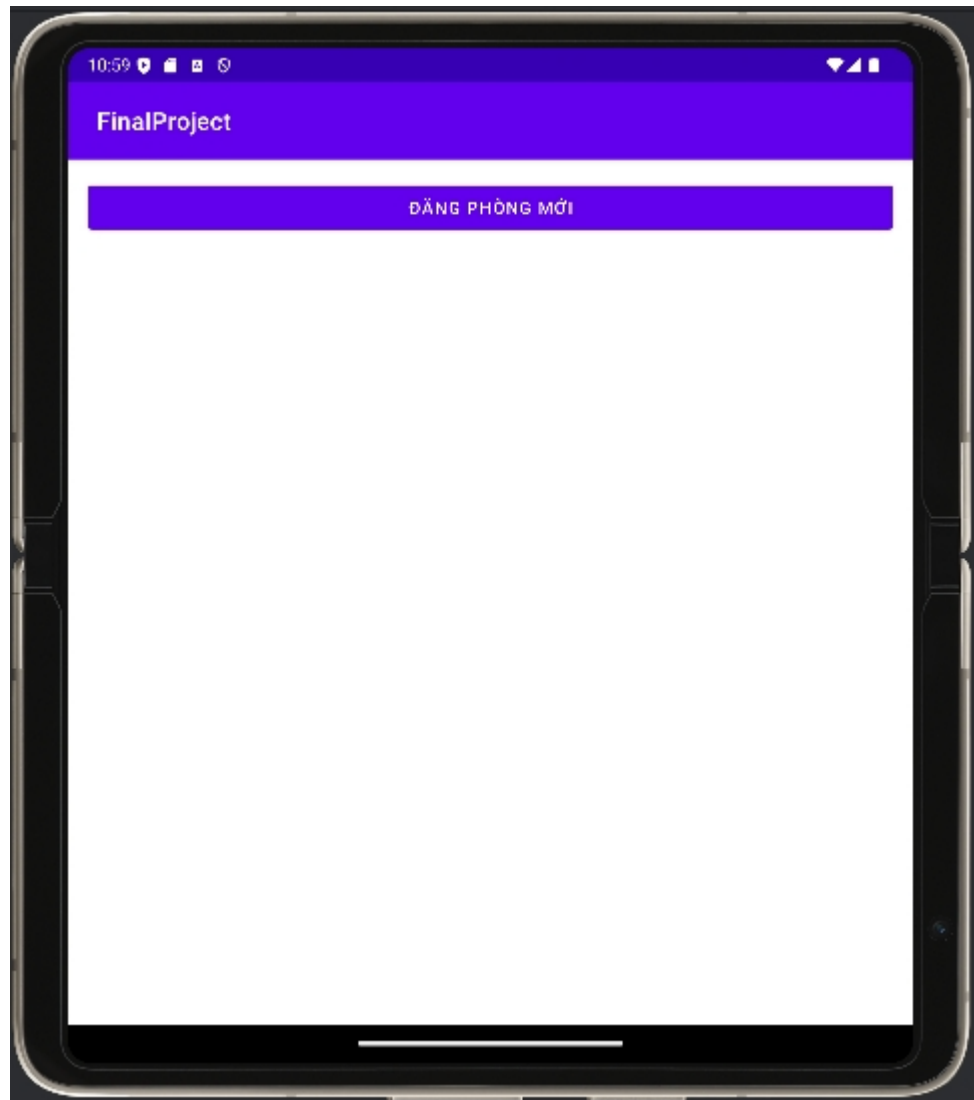


Figure 3.5.5.a: Manage posted rooms page when no rooms are posted yet

Manage posted rooms page when no rooms have been posted will have a New Room Post (Đăng phòng mới) button for users to click on and will be redirected to the Post a room for rent page.

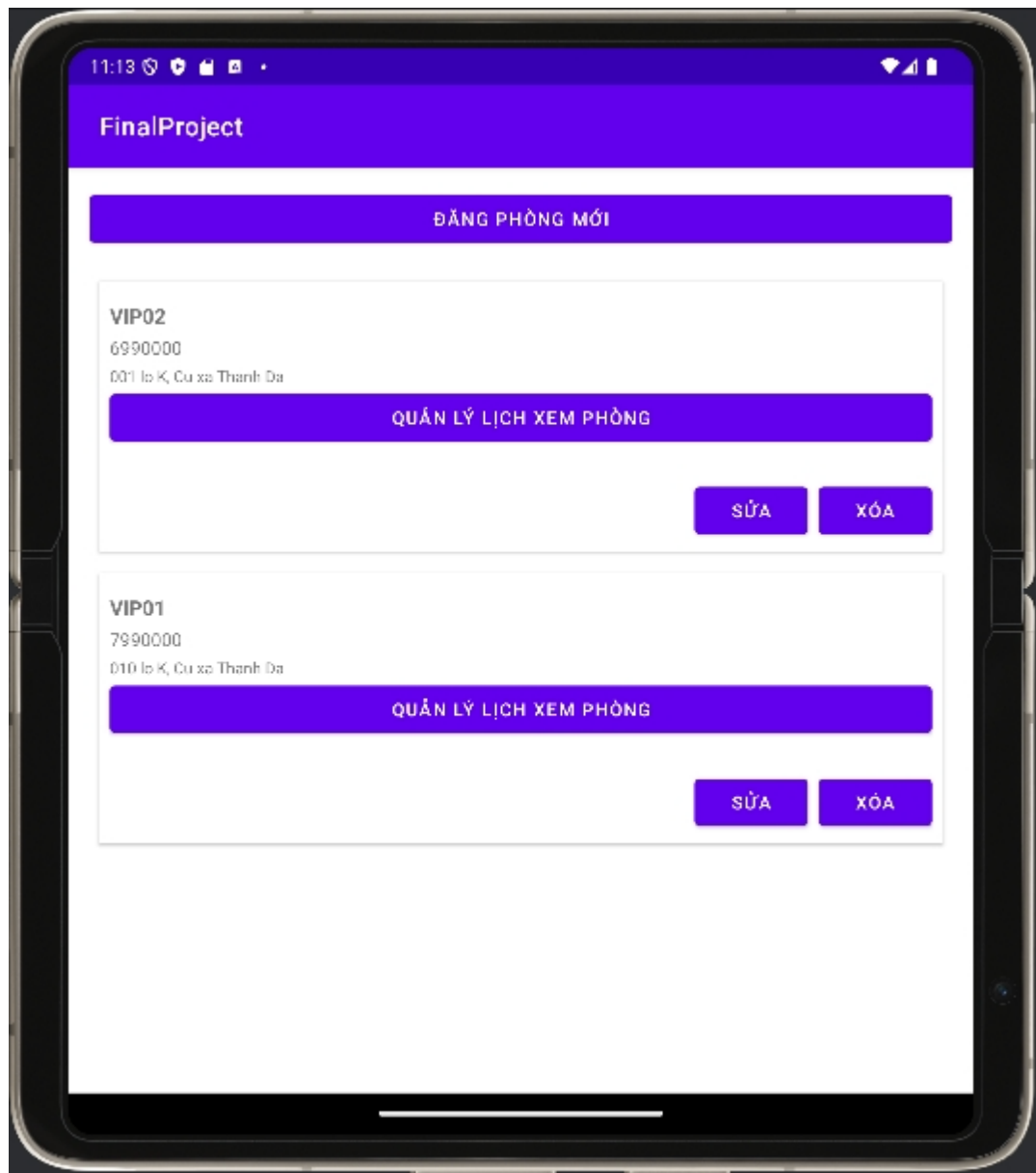


Figure 3.5.5.b: Manage posted rooms page when page has a room is posted

The Room Management page when a room is posted will have 2 buttons: Edit or Delete. For the Edit button, when clicked, it will redirect to the Update a room for rent page. As for the Delete button, when clicked, it will permanently delete that room.

3.5.7 See History Chat of Landlord

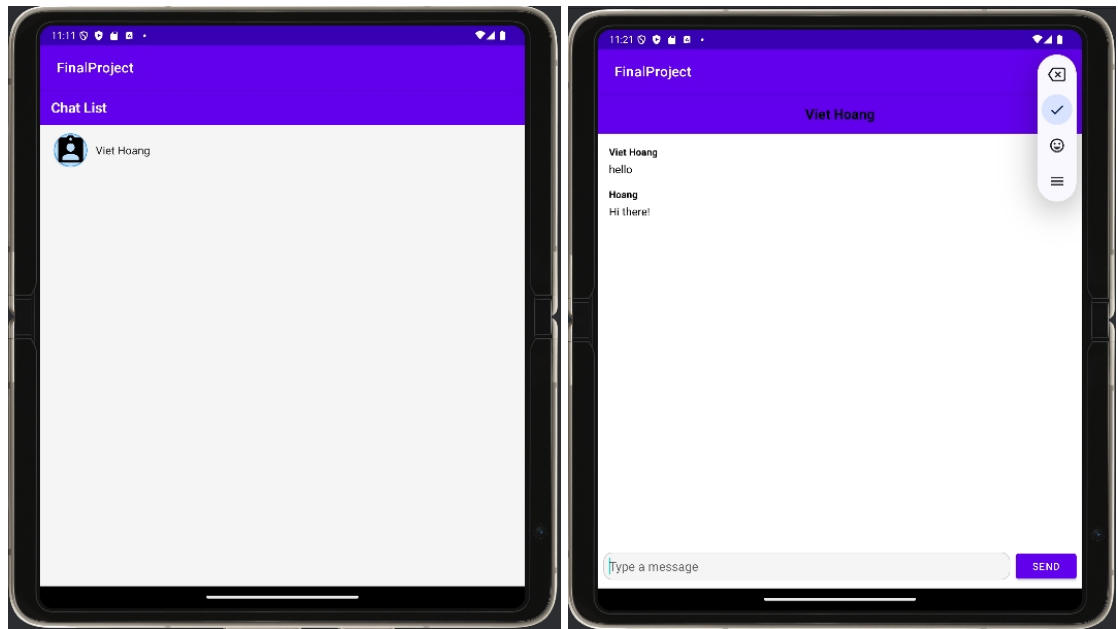


Figure 3.5.7: See history chat page

The chat history page can review messages between landlord and tenants.

3.5.8 Logout button

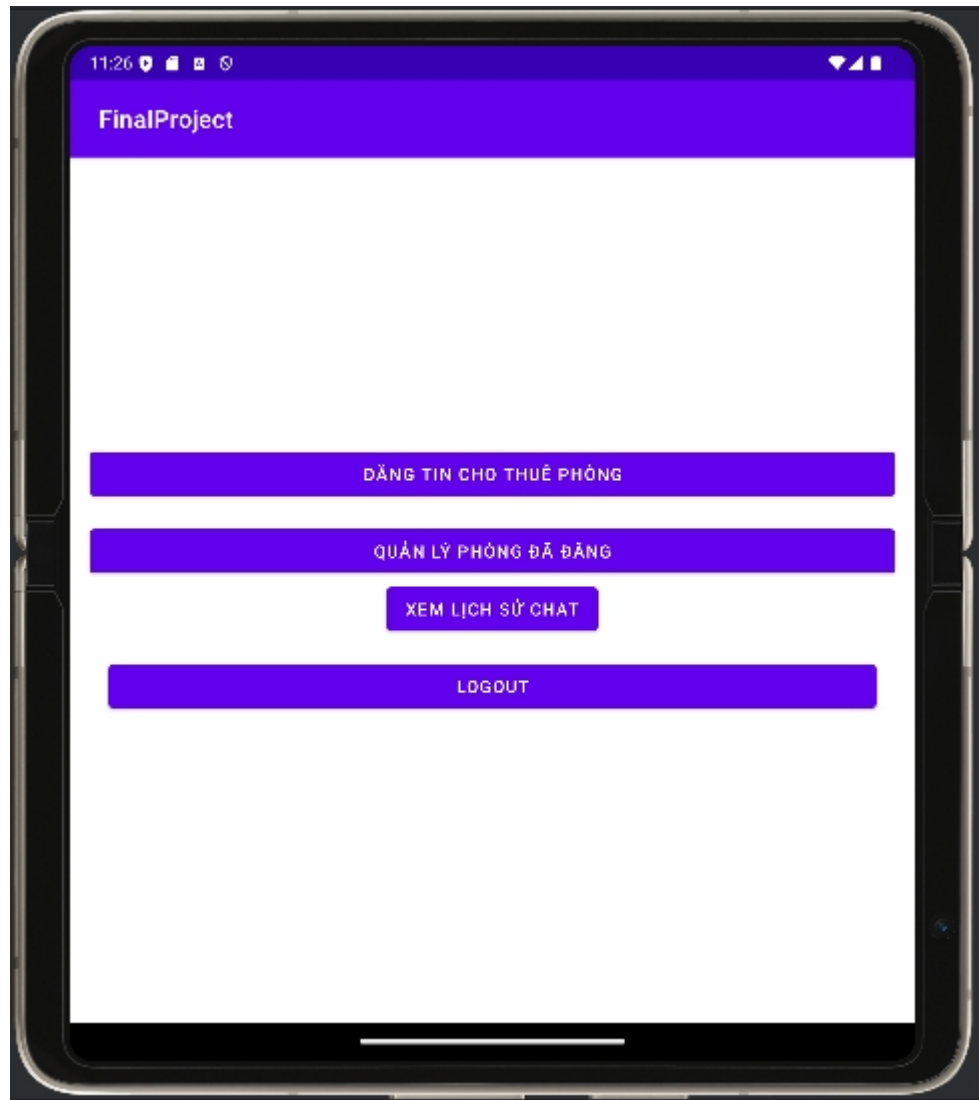


Figure 3.5.8: Logout button in Homepage

When users (Landlord or Tenant) click LOGOUT, they will be out of the session.

3.5.9 The Homepage for Tenant

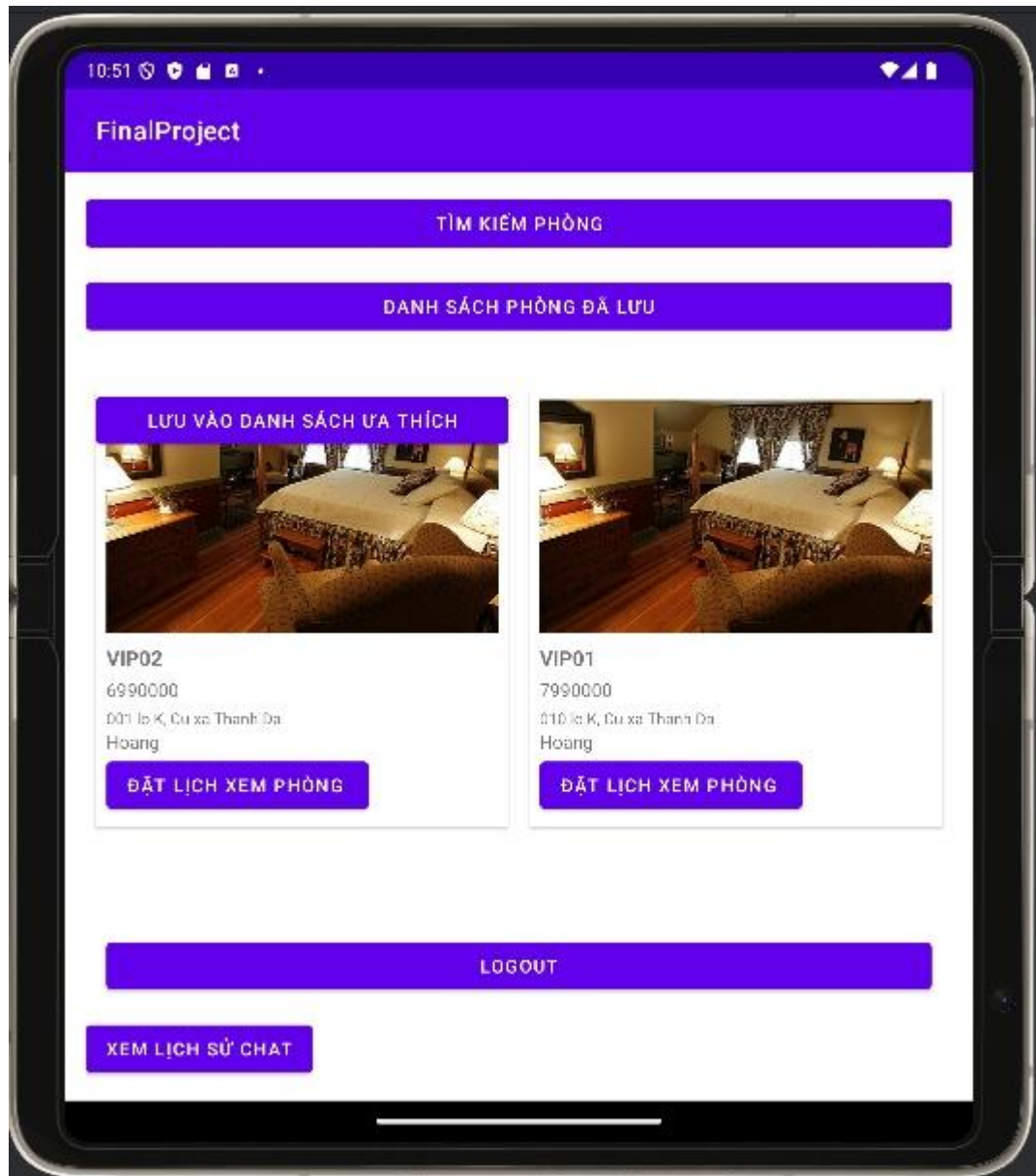


Figure 3.5.9: Homepage's Tenant

The Homepage for Tenant includes the following features: Search rooms, View the loved rooms which were saved, Save the room to the list which you love, Pick a date to view the room, View chat history, Log out.

3.5.10 The Search Page for Tenant

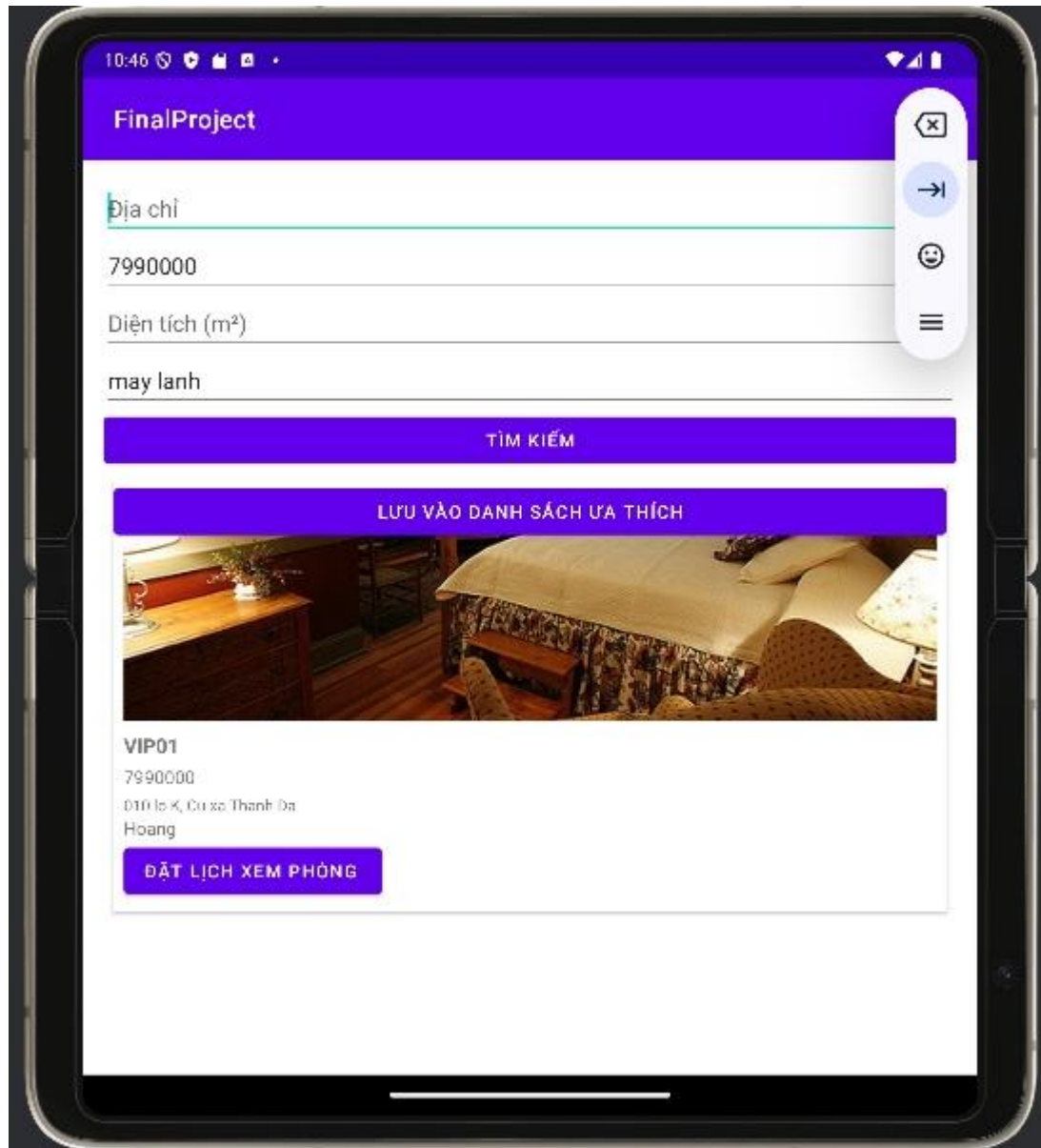


Figure 3.5.10: The Search Page for Tenant

The Search Page for Tenant to search the room they want to rent. Tenant also can save the room they love by clicking the button “LƯU VÀO DANH SÁCH ƯA THÍCH”. Moreover, they can book the date to see the room how is going on by clicking the button “ĐẶT LỊCH XEM PHÒNG” to go to the date selection page.

3.5.11 The date selection page for Tenant

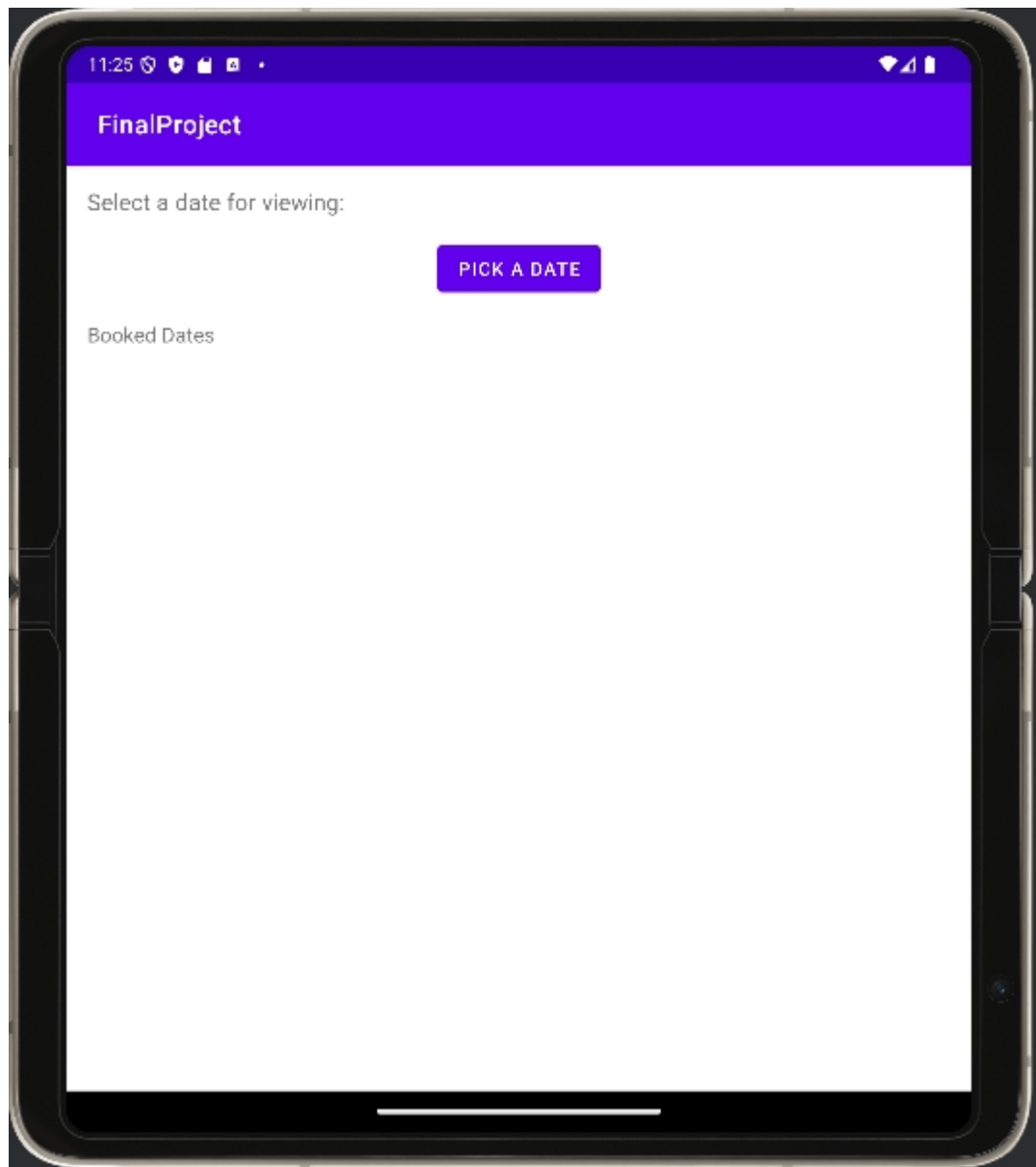


Figure 3.5.11: The date selection page for Tenant

Tenant can choose the date they want to see by clicking the button “PICK A DATE”.

3.5.12 The notice page for Tenant after choosing the date to see the room

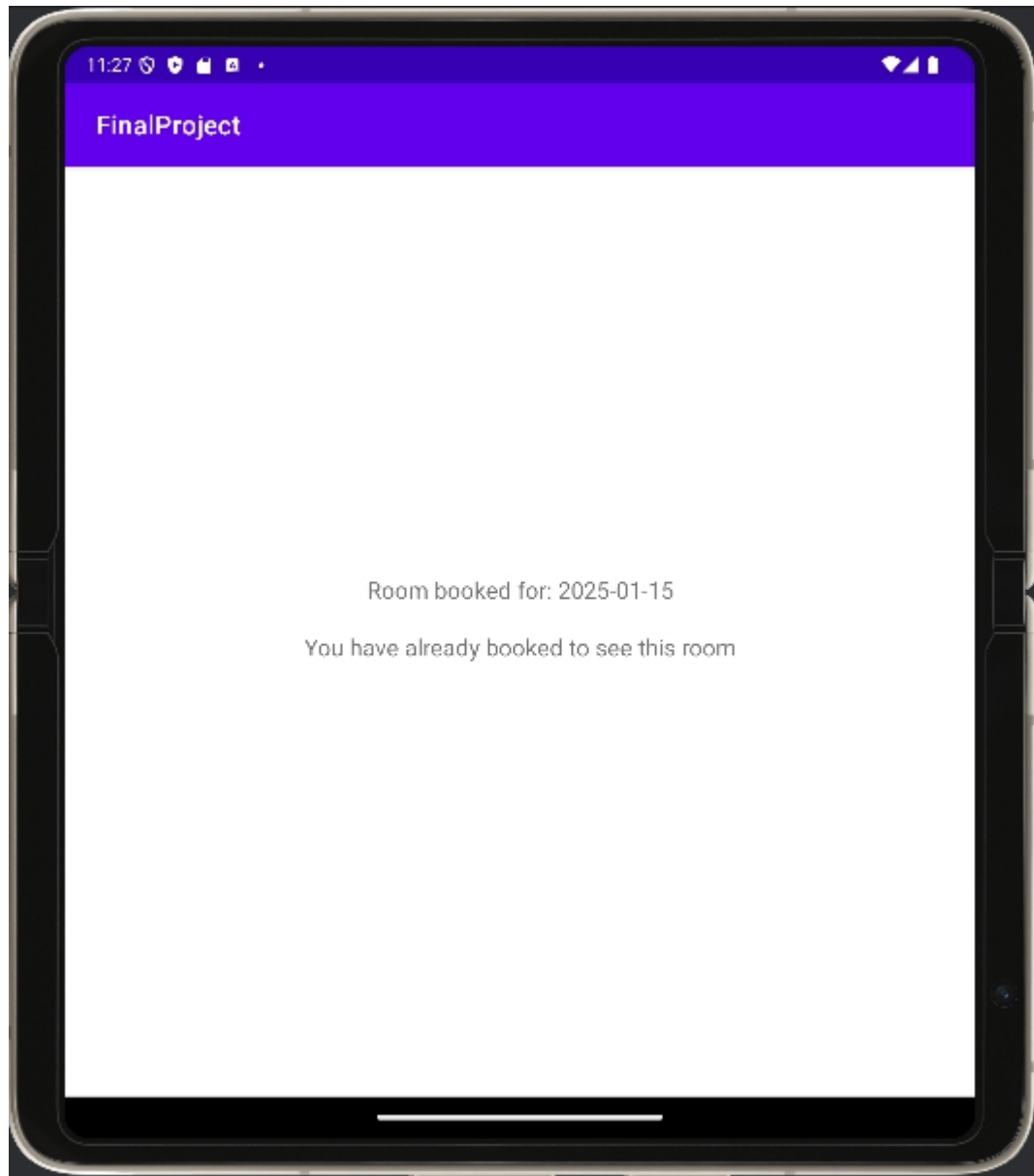


Figure 3.5.12: The notice page for Tenant after choosing the date to see the room

3.5.13 The list of the loved rooms of Tenant

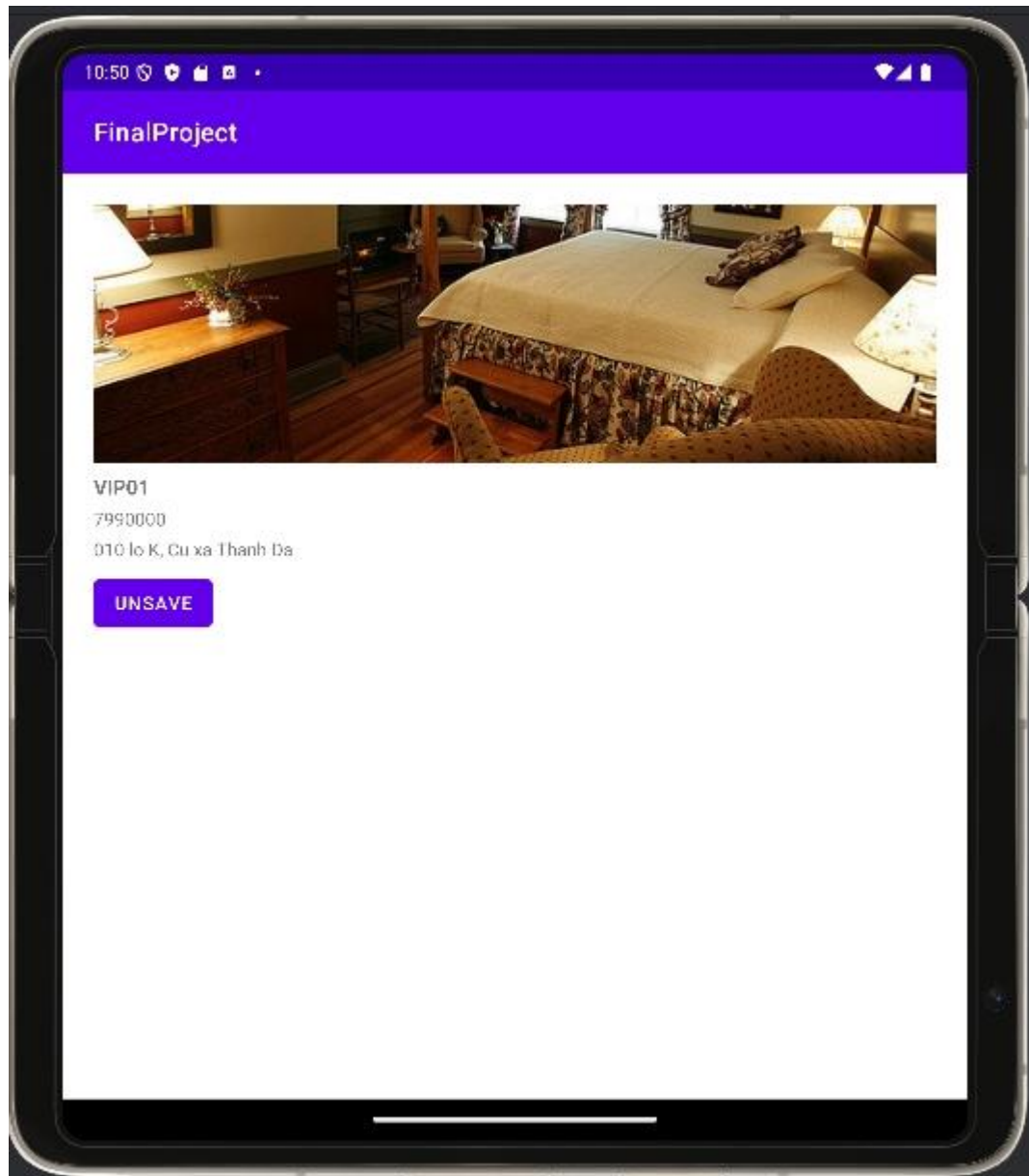


Figure 3.5.13: The list of the loved rooms of Tenant

Tenant can see again the list of the loved room after clicking the button “DANH SACH PHONG DA LUU” in the homepage of tenant.

3.5.14 The information and functions of each room for Tenant

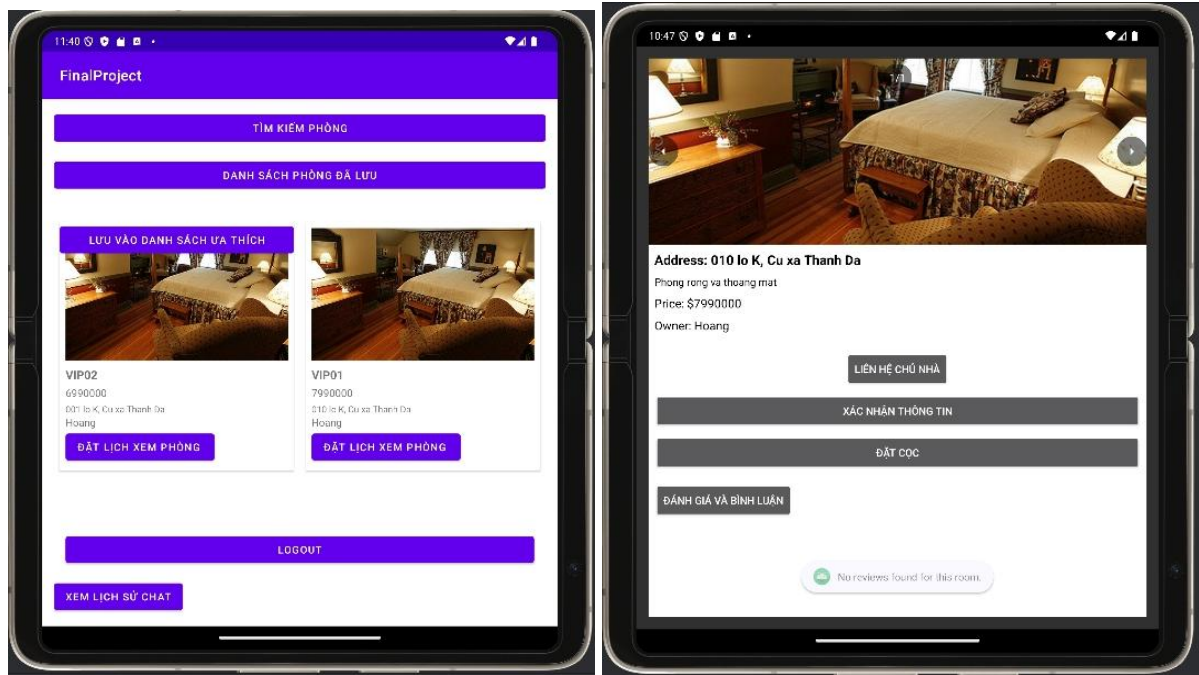


Figure 3.5.14: The information of each room for Tenant

Tenant can see the information and functions after clicking to the image of each room.

3.5.15 See History Chat of Tenant

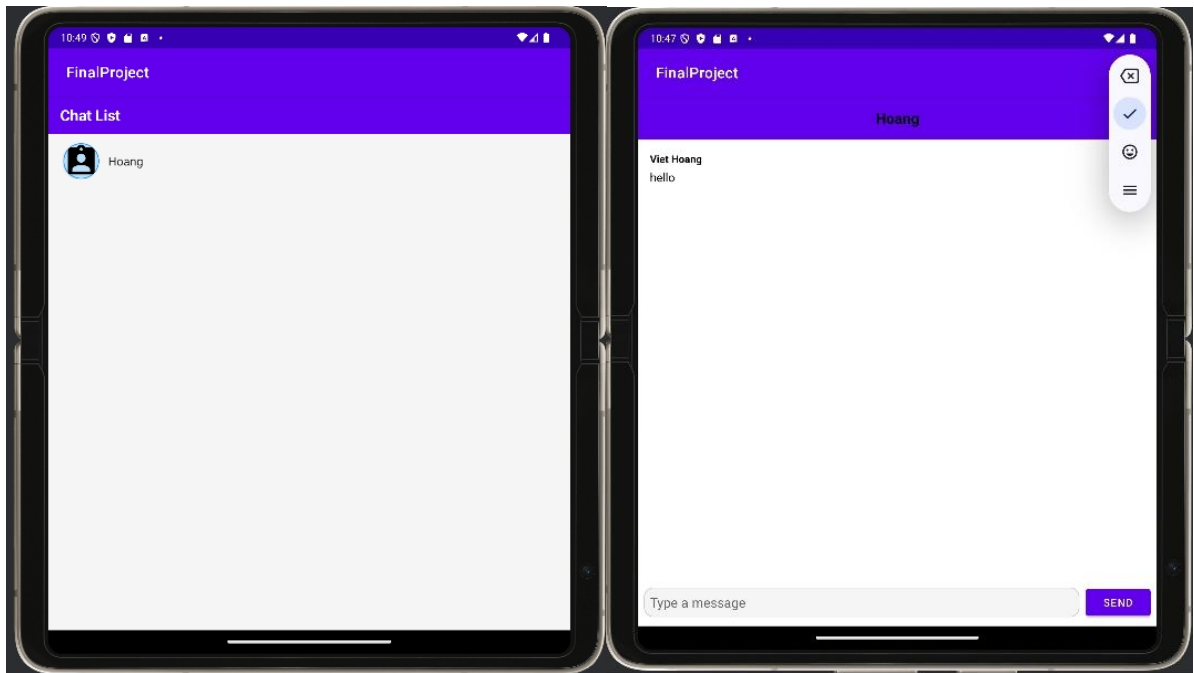


Figure 3.5.15: See History Chat of Tenant

The chat history page can review messages between landlords and tenant.

3.5.16 The confirmation page of Tenant



Figure 3.5.16: The confirmation page of Tenant

The tenant will make some commitment to the landlord after successfully paying the room deposit.

3.5.17 Reviews and comments site for room renters

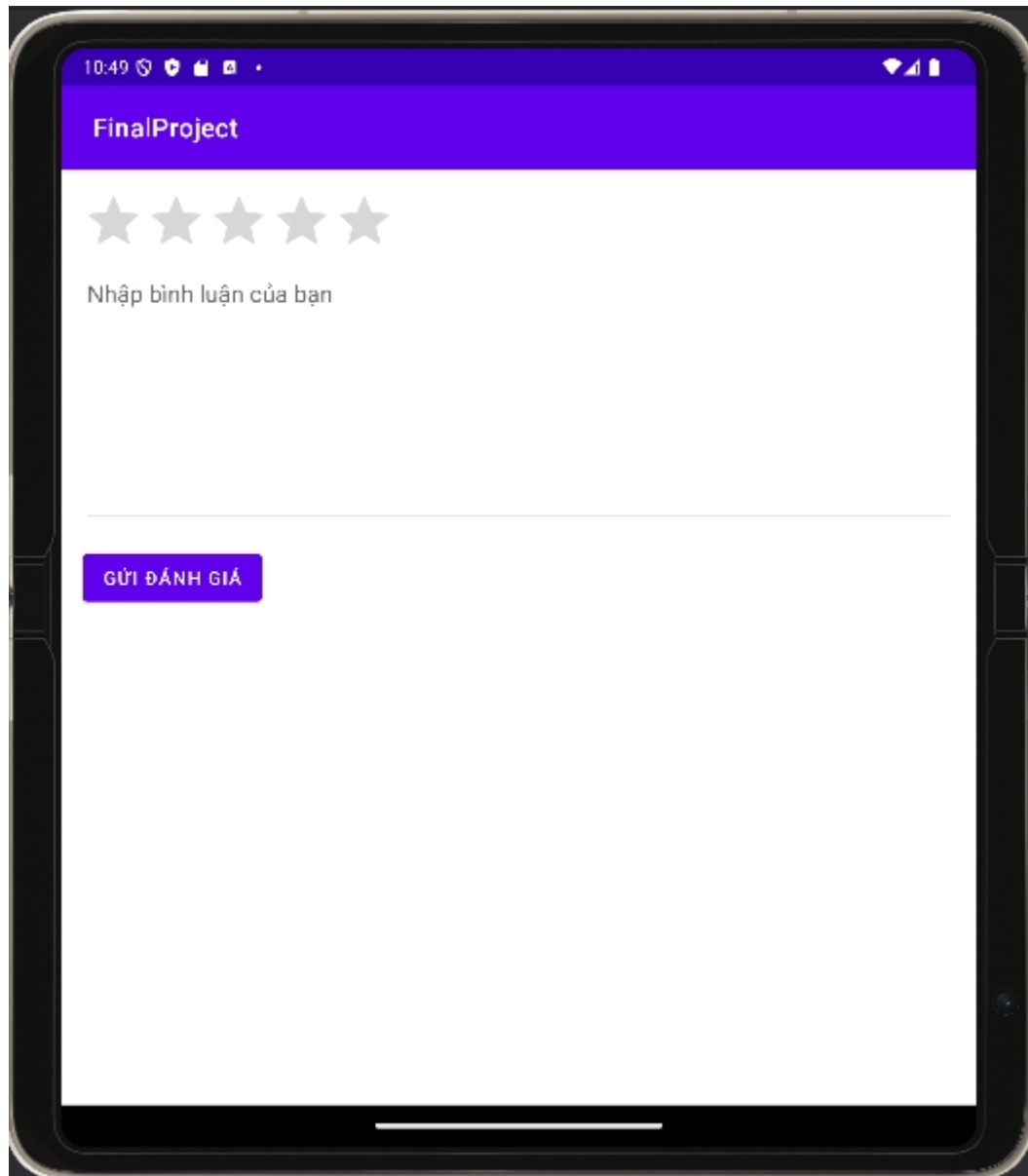
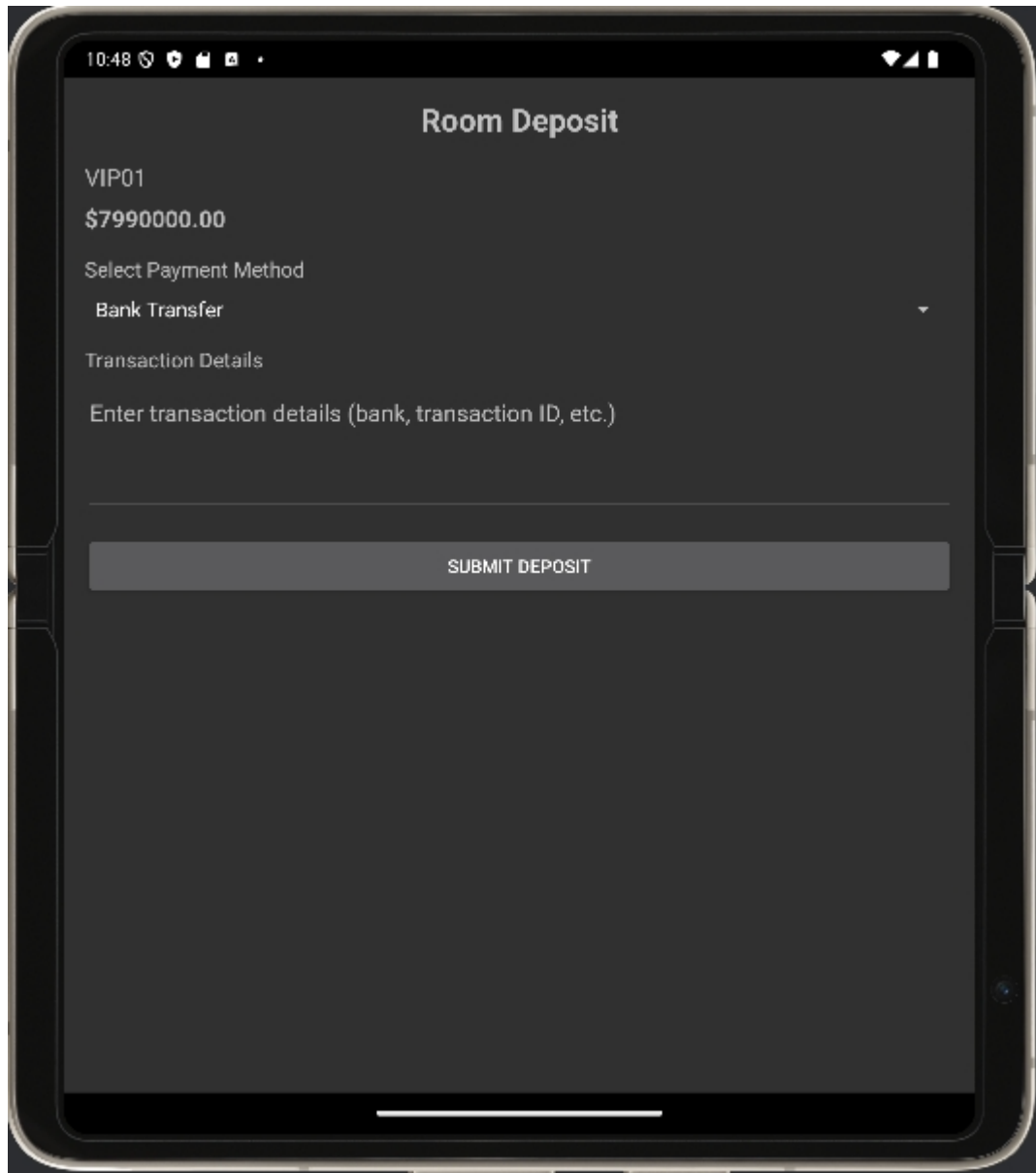


Figure 3.5.17: The reviews and comments site of Tenant

Renters can rate and receive information about the quality of the room they are staying in through this page.

3.5.18 Room deposit page for tenants



The image shows a tablet displaying a web application interface for room deposits. The screen has a dark background with white text. At the top, the status bar shows the time 10:48 and various icons. The main heading is "Room Deposit". Below it, the text "VIP01" is displayed, followed by the amount "\$7990000.00". There is a section titled "Select Payment Method" with a dropdown menu currently showing "Bank Transfer". Below this is a section titled "Transaction Details" with a text input field containing the placeholder "Enter transaction details (bank, transaction ID, etc.)". At the bottom of the form is a large, light gray button labeled "SUBMIT DEPOSIT".

Figure 3.5.18: Room deposit page for tenants

Tenants can pay rent to landlords through this site.

CHAPTER 4. CONCLUSION

4.1 Achieved Outcomes

The Room Rental Management system has been successfully developed to meet the outlined requirements, providing a comprehensive framework for managing rental property information and facilitating seamless interactions between landlords and tenants. The application empowers landlords to effectively manage and update their property listings, while tenants are able to conveniently search, view, and book rooms based on their preferences. A robust role-based authentication mechanism has been implemented, establishing secure and defined access levels for Landlords, Tenants, and Administrators, ensuring a well-structured and efficient user experience.

4.2 Limitations

The primary limitation of the current system is the lack of advanced features that could further enhance its functionality and versatility. While the system provides a solid foundation for managing room rentals, there are opportunities for improvement, including:

1. **Limited Search and Filtering Options:** The system currently offers basic search capabilities based on location, price, and amenities. Integrating more sophisticated search and filtering tools, allowing users to combine multiple criteria, would improve the efficiency of finding suitable room listings.
2. **Absence of Data Visualization:** The system lacks a dedicated section for visualizing rental data through charts, graphs, and other interactive data representations. Incorporating such features would enable users, particularly administrators and landlords, to gain deeper insights and make more informed decisions.

3. **Lack of Reporting and Analytics:** The system currently does not provide comprehensive reporting or analytical capabilities. Introducing features that generate detailed reports on occupancy rates, revenue trends, and other key performance indicators would strengthen the system's decision-support capabilities.

4.3 Suggestions for Future Enhancements

To address the identified limitations and further enhance the functionality of the Room Rental Management application, the following suggestions for future improvements are proposed:

1. **Advanced Search and Filtering:** Develop an enhanced search module that allows users to filter and sort room listings based on multiple criteria, such as location, price range, size, amenities, and availability. This would significantly improve the user experience in finding the most suitable rooms.
2. **Data Visualization and Analytics:** Incorporate a dedicated section or module that presents rental data through visual representations, such as charts, graphs, and dashboards. This would enable administrators, landlords, and even tenants to gain valuable insights and make more informed decisions.
3. **Comprehensive Reporting:** Implement a robust reporting framework that generates detailed reports on key metrics, such as occupancy rates, revenue, booking trends, and user activity. These reports would assist landlords and administrators in monitoring the performance of the rental business and making data-driven decisions.

4. **Automated Notifications and Reminders:** Introduce features that automatically send notifications and reminders to users, such as upcoming payment due dates, new room listings matching a tenant's preferences, or maintenance alerts for landlords. This would enhance the overall user experience and help streamline various rental management tasks.
5. **Integrated Communication and Collaboration Tools:** Enhance the system by integrating communication and collaboration features, such as in-app messaging, file sharing, and task management. This would improve coordination between landlords and tenants, fostering a more seamless and efficient rental experience.

By implementing these enhancements, the Room Rental Management application would become a more comprehensive and powerful tool, providing users with an enriched experience in managing their rental properties and fostering stronger connections between landlords and tenants.

REFERENCES

- [1] N. T. Kim, "Mobile Application Development for Room Rental Management: A Role-Based Access Control Approach," in *International Journal of Mobile Computing and Applications*, vol. 12, no. 3, pp. 45-62, Sep. 2023.
- [2] H. Le and M. Nguyen, "User Authentication and Security in Mobile Rental Management Systems," 2023 *International Conference on Information Systems and Database Technologies*, Ho Chi Minh City, Vietnam, 2023, pp. 112-125.
- [3] Firebase, "Firebase Firestore: Real-Time Database Architecture and Security," *Firestore Documentation*, Google LLC, 2023.
- [4] T. Van Tran, "Implementing Role-Based Access Control in Mobile Applications," *Journal of Software Engineering and Mobile Computing*, vol. 8, no. 2, pp. 78-93, Jun. 2023.
- [5] M. Pham and K. Tran, "User Experience Design in Room Rental Management Mobile Applications," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 27, no. 4, pp. 33-47, Oct. 2023.
- [6] A. Nguyen, D. Kim, and S. Park, "Advanced Search and Filtering Techniques in Mobile Rental Platforms," *IEEE Transactions on Mobile Computing*, vol. 22, no. 5, pp. 201-215, May 2024.
- [7] G. Tran, "Security Rules and Authentication in Firebase: A Comprehensive Review," *International Journal of Database Management Systems*, vol. 15, no. 3, pp. 56-72, Aug. 2023.
- [8] K. Le and H. Nguyen, "Data Synchronization and Real-Time Updates in Mobile Applications," 2023 *IEEE International Conference on Mobile Software Engineering*, Hanoi, Vietnam, 2023, pp. 45-58.

[9] R. Patel, "User Role Management in Mobile Rental Platforms: Challenges and Solutions," *Journal of User Experience and Interface Design*, vol. 11, no. 4, pp. 89-105, Dec. 2023.

[10] S. Wong and L. Chen, "Performance Optimization of Mobile Database Systems using Firestore," *ACM International Conference on Mobile Systems, Applications, and Services*, Singapore, 2023, pp. 167-180.