**Egyptian E-Learning University**

Faculty of Computers & Information Technology

E-Rent

**By**

|  |  |
| --- | --- |
| Sohaila Alaa Eldin Fathy Khaled | 21-01932 |
| Hager Alaa Eldin Fathy Khaled | 21-02132 |
| Hager Mohamed Abdelwahab | 21-02133 |
| Nourhan Omar Abdelmaksoud | 21-02126 |
| Abdelrahman Mohamed Abdelkereem | 21-01970 |
| Islam Yasser Farag | 21-01856 |
| Steven Ayman Salah | 21-01928 |

Supervised by

Dr. Rodaina Abdelsalam

Assistant

Eng. Engy Emad Beshara Salib

[Qena-Egypt]-2025

Abstract

The "E-Rent" project is a mobile application designed to address the significant challenges students in Upper Egypt face when searching for housing. Traditional solutions such as word-of-mouth, Facebook groups, or generalized platforms like OLX and Dubizzle fail to provide safe, efficient, and context-specific options tailored to student needs. E-Rent offers a centralized digital platform where students can search for, compare, and securely book accommodations based on personal preferences including price, gender, location, and university faculty.

The main objective is to deliver a student-focused renting experience that incorporates intelligent recommendation algorithms, verified listings, and secure transactions. The platform uses Firebase for real-time data synchronization, authentication, and scalability. A key innovation is the integration of a contextual multi-armed bandit algorithm (Rule-based systems), which provides personalized recommendations by learning from user behavior and context data.

The methodology combines UI/UX design via Figma, development with Flutter and Dart, and backend implementation through Firebase services. Local payment gateways and secure communication tools were integrated to enhance trust and ease of use.

Results from user testing in Upper Egypt indicated improved satisfaction, reduced fraud risk, and more effective property matches. The platform successfully meets its objectives by offering a scalable, AI-powered, culturally aware solution that simplifies the student housing process and enhances student well-being.

Acknowledgments

We have taken efforts in this project. However, it would not have been possible without the kind support and help of many individuals and organizations. I would like to extend my sincere thanks to all of them. We highly indebted to (Doctor Rodaina Abdelsalam) for the guidance and constant supervision as well as for providing necessary information regarding the project & also, for the support in completing the project. Also, we indebted to (Eng. Engy Emad) for the guidance and constant supervision as well as for providing necessary information regarding the project & also for the support in completing the project. I would like to express my gratitude to wards my parents & member of (The Egyptian E-Learning University) for the kind co-operation and encouragement which help us in completion of this project. I would like to express my special gratitude and thanks to supervisor`s persons for giving me such attention and time. My thanks and appreciations also go to my colleagues in developing the project and people who have willingly help me out with their abilities.

Contents

[Abstract 2](#_Toc200411981)

[Acknowledgments 3](#_Toc200411982)

[Introduction 7](#_Toc200411983)

[1.1 Introduction 8](#_Toc200411984)

[1.2 Background and Motivation for the Project 8](#_Toc200411985)

[1.3 Relevance of the Problem in Question 9](#_Toc200411986)

[1.4 Problem Statement 9](#_Toc200411987)

[1.4.1 Clear definition of the problem your project addresses 9](#_Toc200411988)

[1.4.2 Justification for why this problem is worth solving. 10](#_Toc200411989)

[1.5 Objectives: 10](#_Toc200411990)

[1.5.1 Main Objective: 10](#_Toc200411991)

[1.5.2 Specific Objectives: 10](#_Toc200411992)

[1.6 Brief Overview of the Proposed Solution 11](#_Toc200411993)

[Literature Review / Related Work 12](#_Toc200411994)

[2.1 Existing Student Housing Platforms in Egypt 13](#_Toc200411995)

[2.2 Gaps in Current Solutions 21](#_Toc200411996)

[2.3 Summary 23](#_Toc200411997)

[Proposed system 24](#_Toc200411998)

[3.1 Approach Used to Solve the Problem 25](#_Toc200411999)

[3.2 System architecture 27](#_Toc200412000)

[3.3 Algorithms or Frameworks Used (Detailed Explanation of the Recommendation System) 30](#_Toc200412001)

[Implementation 32](#_Toc200412002)

[4.1 Technologies, tools, and programming languages used 33](#_Toc200412003)

[4.2 Key Components/Modules of the System 36](#_Toc200412004)

[4.2.1 Front end Implementation 38](#_Toc200412005)

[4.2.2 Back end implementation 65](#_Toc200412006)

[4.2.3 AI Implementation 66](#_Toc200412007)

[4.3 Challenges Faced and How They Were Resolved 68](#_Toc200412008)

[Testing & Evaluation 70](#_Toc200412009)

[5.1 Testing Strategies 71](#_Toc200412010)

[5.1.1 Unit Testing: 71](#_Toc200412011)

[5.1.2 Integration Testing: 71](#_Toc200412012)

[5.1.3 User Testing (Beta Testing): 71](#_Toc200412013)

[5.1.3.1 Firebase Test Lab: 72](#_Toc200412014)

[5.1.3.2 A/B Testing: 72](#_Toc200412015)

[5.2 Performance Metrics 73](#_Toc200412016)

[5.3 Comparison with Existing Solutions 74](#_Toc200412017)

[Results & Discussion 76](#_Toc200412018)

[6.1 Introduction 77](#_Toc200412019)

[6.2 Summary of Findings 77](#_Toc200412020)

[6.3 Interpretation of Results 78](#_Toc200412021)

[6.4 Limitations of the Proposed Solution 78](#_Toc200412022)

[Conclusion & Future Work 80](#_Toc200412023)

[7.1 Summary of Contributions 81](#_Toc200412024)

[7.2 Possible Improvements or Extensions for Future Work 81](#_Toc200412025)

[References 84](#_Toc200412026)

[Appendices (Optional) 95](#_Toc200412027)

Chapter 1

Introduction

# 1.1 Introduction

Finding proper student housing in Egypt, particularly in Upper Egypt (Al-Sa'id), remains a top issue due to a lack of well-organized websites, old advertisements, and poor digital options. Most students from rural or distant areas to pursue higher studies in urban universities in cities like Assiut, Sohag, and Qena are forced to rely on word-of-mouth tips, unverified Facebook groups, or direct interaction with middlemen—resulting in exorbitant costs, scams, or unsuitable living accommodations.

Our application bridges this gap by creating an online platform that streamlines the process of student accommodation renting. The platform allows for searching, comparing, and booking housing based on personalized needs such as price range, department within the university, gender, and preferred room size. The app also has in-built communication, secure booking, verified listings, and an AI-powered recommendation system, which collectively offer a user-centered and secure renting experience.

# 1.2 Background and Motivation for the Project

The absence of centralized, reliable, and technology-based student accommodation options in Egypt—particularly outside of Alexandria and Cairo—is a significant barrier to thousands of students every year. This is even more the case in Upper Egypt, where universities are dispersed over cities that tend not to provide adequate housing services specific to students. In addition,

there is traditional brokerage dominance and manipulation, as well as the presentation of misleading information, to the students.

With the widespread use of smartphones and the increasing demand for digitization in public life and university learning, this project's motivation is to be able to have a mobile-first solution which is context-aware of the Egyptian context and day-to-day issues of Egyptian students, particularly from marginalized areas like Upper Egypt.

# 1.3 Relevance of the Problem in Question

The upper Egyptian student housing market is also informal and segmented. The majority of students are unable to access secure and affordable.

# 1.4 Problem Statement

## 1.4.1 Clear definition of the problem your project addresses

The project addresses the lack of a reliable, student-focused housing rental platform within Upper Egypt. Current solutions are too general (e.g., Dubizzle or OLX) or insufficient in fulfilling the unique needs of students. The platforms neglect context-specific features like faculty location, mutual room interests, or gendered advertisements, which take precedence under the cultural framework of Upper Egypt.

## 

## 1.4.2 Justification for why this problem is worth solving.

By addressing this problem, exploitation will be minimized, safety of students will be ensured, and renting will be simplified by using a mobile application for Egyptian students in Upper Egypt. Our system applies both traditional search facilities and intelligent suggestions to give users suitable accommodation that suits their individual needs.

# 1.5 Objectives:

## 1.5.1 Main Objective:

To develop and launch a mobile application for renting student accommodation in Upper Egypt with a secure, comfortable, and context-sensitive setting with smart search and recommendation functions.

## 1.5.2 Specific Objectives:

To develop an authenticated database of housing listings with primary metadata (price, area, vicinity to university, size, etc.).

To have a contextualized recommendation feature with a Multi-Armed Bandit algorithm (Rule-based system) for personalized recommendations

To include user profiles with budget, gender, and faculty filters.

To support real-time booking and status notifications.

To include accessibility for Egyptian users who utilize Arabic language.

# 1.6 Brief Overview of the Proposed Solution

The adopted solution is an application designed for student housing in Upper Egypt, where users can search comfortably available room, filter results according to their wish, communicate safely with landowners, and book directly through the application. A key feature is the recommendation system that is artificial intelligence-based and utilizes user profile data as well as context features to suggest housing that can accommodate a person's needs. This enhances satisfaction and reduces search time. The app is scalable across Egyptian cities, but it will initially be launched to university towns in Upper Egypt due to the extreme shortage of organized student accommodation platforms in this region. The booking can be made from any location around Egypt, but the listings are centered in Upper Egypt.

Chapter 2

Literature Review / Related Work

# 2.1 Existing Student Housing Platforms in Egypt

Student housing solutions in Egypt are typically provided as university-administered accommodations and general-purpose online platforms. For instance, on-campus housing in Misr University for Science and Technology has facilities specifically tailored for students like individual housing for male and female students, 24/7 supervision, and medical care . must.edu.eg

However, such accommodations run by universities are small in magnitude and do not necessarily extend to all students, especially those studying in Upper Egypt. The students are mostly left with no alternative but to use general online websites such as OLX Egypt, which, even though offer a broad range of postings, lack student-specific filters and verification procedures and result in hardships to find suitable and secure accommodation.

**Local Systems**

**1-Dubbizle Eqypt:**

*Advantages of Dubizzle:* Direct Communication: Dubizzle allows users to communicate directly with property owners or advertisers via in-app messaging or contact details. This makes negotiations and inquiries more convenient.

Large User Base: The platform has a massive user base, increasing the visibility of listings and making it easier for users to find a variety of options in different locations.

User-Friendly UI: Dubizzle offers an intuitive and simple user interface, ensuring a seamless browsing experience for users, even those who are not tech-savvy.

Free Advertisement: Users can post property listings for free, making it accessible for property owners to reach potential tenants without additional costs.

Filter System for Easy Search: The app provides advanced filtering options, allowing users to narrow down search results based on location, price, property type, and more, making the search process efficient and quick.

**Disadvantages of Dubizzle**:

Lack of Reviews or Ratings: The platform does not have a system for user reviews or property ratings, reducing transparency and making it harder for users to make informed decisions.

No Integrated Payment System: Dubizzle does not support in-app payment processing, requiring users to handle transactions externally, which can be inconvenient.

No AI-Powered Assistant: The app lacks an AI-driven system to provide personalized recommendations, making the search process less efficient compared to modern applications.

No Verification (Fake Advertisements): The platform does not verify advertisements thoroughly, leading to the possibility of encountering fake or misleading property listings.

Paid Advertisements: Certain features, like promoting or highlighting listings for better visibility, require payment, which may not be ideal for all users.

Inaccurate Location Determination: Sometimes, the location details of listings are inaccurate, causing confusion or wasted time for users searching for properties in specific areas.

**2- Bayut Egypt (بيوت مصر):**

**Advantages of Bayut Egypt**

User-Friendly Interface: The app offers a simple and intuitive interface, making it easy to navigate even for users with limited technical skills.

Category Filters: It provides category-specific filters, allowing users to narrow down their searches to specific property types such as apartments, houses, or villas, making the process more efficient.

Localized Listings: Bayut Egypt focuses on properties within Egypt, offering tailored options for users looking for housing in specific local areas.

Easy to Search and Compare: The platform makes it convenient to search for and compare multiple properties, helping users make well-informed decisions based on their preferences.

Providing Official Invoices and Installment Payments: The app offers official invoices for transactions and provides installment payment options, making property purchases more accessible and manageable.

**Disadvantages of Bayut Egypt:**

Rental of Apartments Only, Not Beds: The platform focuses solely on renting entire apartments, with no option for renting individual beds, which limits its appeal to students or budget-conscious users.

Limited Options for Students: The app does not cater specifically to student needs, such as affordable housing options or shared accommodations.

No AI Integration: Bayut Egypt lacks AI-powered features for personalized recommendations, making the search process less efficient.

Payment Outside the Platform: Users must handle payments externally, as the app does not support integrated payment processing.

No Tenant Profiles: The app does not provide detailed profiles for tenants, limiting transparency and trust between landlords and renters.

Lack of Comments and Ratings System:

There is no system for leaving reviews or ratings for landlords or properties, which reduces transparency and the ability to make informed decisions.

No Online Booking: The platform does not allow users to book properties online, requiring additional steps to finalize rental agreements.

No Integrated Payment System: The absence of an in-app payment system means users cannot securely and conveniently complete transactions within the platform.

Absence of Electronic Contracts: The app does not offer electronic contracts, which could simplify the rental process and make it more secure for both parties.

**International Systems**

**1- Airbnb**

**Advantages of Airbnb:**

Global Reach: Airbnb operates worldwide, offering a vast selection of accommodations in different countries, making it ideal for international travelers and students.

Verified Listings: The platform ensures the verification of property listings, providing a level of trust and reducing the risk of fraudulent listings.

Review System: Airbnb allows users to leave and read reviews about properties and hosts, ensuring transparency and helping users make informed decisions.

Incorporates AI-Powered Recommendation System: The app uses AI to provide personalized recommendations based on user preferences and past searches, enhancing the overall user experience.

Built-In Payments (Integrated Payment System): Airbnb offers a secure and seamless payment system within the platform, making transactions convenient and hassle-free.

Supports the Idea of Shared Accommodation: The platform promotes shared housing options, allowing users to rent rooms or beds instead of entire properties, which is especially beneficial for students and budget-conscious travelers.

Direct Communication: Airbnb enables users to communicate directly with hosts through its built-in messaging system, making coordination and inquiries easy.

Flexible Payment Options (Daily, Weekly, Monthly, Yearly): The app allows users to book accommodations with flexible payment durations, catering to various needs and budgets.

**Disadvantages of Airbnb:**

Short-Term Focus (Not Specifically Targeted for Students): Airbnb primarily focuses on short-term rentals, which may not be ideal for students looking for long-term housing options or affordable shared accommodations.

High Service Fees: The platform charges service fees for both hosts and guests, which can increase the overall cost of bookings, making it less cost-effective compared to other rental services.

Lack of Personal Student Ratings System: Airbnb does not provide a dedicated system for students to rate or review landlords specifically based on their student-related experiences, which could be useful in a student housing context.

No Electronic Contracts: The platform does not offer an integrated system for electronic contracts, which could streamline the rental process and provide a more secure and professional experience for both parties.

No Specialized Favorites and Notifications Features: Unlike some other housing platforms, Airbnb does not offer a specialized "favorites" or "notifications" feature for users to save properties they are interested in or receive alerts for new listings based on their preferences.

**2- Sakan**

**Advantages of Sakan App**

User-Friendly Filters: The app has easy-to-use filters that help users customize their search based on specific requirements such as location, price, and features.

Easy-to-Use Interface: The app features a simple and user-friendly interface, making it easy for users to navigate without complexity.

Favorites Feature and Notifications System: The app offers a favorites feature that allows users to save properties they are interested in and receive notifications when there are updates or changes to those properties.

Flexible Payment Options: The app provides various payment options, allowing users to pay on a monthly, yearly, daily, or weekly basis, offering flexibility to match different user needs.

Continuous Updating: The app continuously updates its property database, ensuring users have access to the latest available listings.

**Disadvantages of Sakan App:**

No AI-Powered Recommendation System: The app lacks AI-driven recommendations that could tailor suggestions based on user interests or past searches, making the search process less personalized.

No Ratings or Comments on Listings: The app does not offer a system for users to leave comments or ratings on listings, reducing transparency and making it harder for users to make informed decisions.

No Tenant or Landlord Reviews: There is no option to review or rate tenants or landlords, making it difficult for users to assess experiences with different parties.

No Electronic Contracts or Booking System: The app does not offer electronic contracts or an integrated booking system, making transactions less secure and more cumbersome.

Limited Payment Options: The app offers limited payment methods, lacking options like online payments or other flexible payment methods, which could be a barrier for some users.

No Shared Accommodation Options: The app does not provide options for shared accommodation or renting individual rooms, which would be beneficial for students or individuals seeking affordable shared housing.

# 2.2 Gaps in Current Solutions

In spite of the evolution of recommendation systems, existing student accommodation websites in Egypt, particularly those serving Upper Egypt, do not offer solutions that are specifically designed to meet the peculiar needs of students. Although student housing platforms are available in major cities in Egypt, they miss several key areas, mainly for university students residing in Upper Egypt. The lack of authenticated listings, student-centered filters (i.e., distance to schools, cost, gender preference, and roommate matching), and sophisticated recommendation algorithms results in an ineffective, time-consuming, and disjointed search experience for accommodation.

**The following statements articulate the key gaps of the current solutions:**

Unverified Listings: A large number of sites do not have landlord or property verification, thus increasing the risk of scams and misleading advertisements.

Generic Search Filters: The filters are not accommodating to students (e.g., location near campus, price range, gender, roommate type).

Manual Searching: The absence of recommendation engines forces students to search manually through listings with no ranking or recommendation.

Poor Communications Tools: Websites lack inherent messaging or alert tools, compelling students to use unsecured or untested third-party tools like WhatsApp or calls.

No Two-Way Review System: Both students and landlords can't rate or review one another, decreasing accountability and enabling malicious individuals to get away without being detected.

Short on Spatial Accuracy: Absence of interactive mapping capability or GPS function restricts users' capability to exactly pinpoint properties, measure distances to schools, transport modes, or neighborhood amenities.

Inadequate Adaptation to the Upper Egyptian Context: General-purpose platforms fail to consider the unique infrastructural and cultural characteristics of Upper Egypt, thereby exacerbating issues such as disinformation and fraud.

Consequential Issues: These deficiencies result in increased stress levels, inefficacy, and greater safety risks for students who request accommodations.

Need for Student-Oriented Solution: There is strong demand for an advanced housing platform such as E-Rent providing:

Confirmed listings

Student-level filters, Secure in-app communication Protected reservation structures, Precise location devices, and is specifically suited for the needs of Upper Egyptian students.

# 2.3 Summary

Briefly, while there are existing platforms offering solutions to student housing in Egypt, none of them is satisfactory in serving the specific needs of students in Upper Egypt. Integrating state-of-the-art recommendation systems, particularly contextual bandit algorithms such as Rule-based system, holds the promise of enhancing the house-finding experience with personalized, context-aware recommendations. Yet the existing environment is devoid of such integrated solutions, and thus there is a need for a specialized platform integrating verified listings, student-focused features, and sophisticated recommendation features adapted to the specific environment of Upper Egypt.

Chapter 3

Proposed system

# 3.1 Approach Used to Solve the Problem

The challenge of securing suitable student accommodation in Upper Egypt is multifaceted, involving issues such as limited availability, lack of centralized information, security concerns, and inefficient communication between students and landlords. To address these challenges, the E-Rent platform adopts a holistic and user-centered approach that integrates technological innovation with socio-economic considerations specific to the Egyptian context.

At the core of this approach is the development of a comprehensive digital ecosystem that consolidates housing listings tailored for students, thereby reducing market fragmentation. The platform employs advanced filtering mechanisms that consider diverse student preferences, including budget constraints, gender-specific accommodations, proximity to educational institutions, and desired property features. This targeted search capability significantly reduces the time and effort students expend in finding appropriate housing.

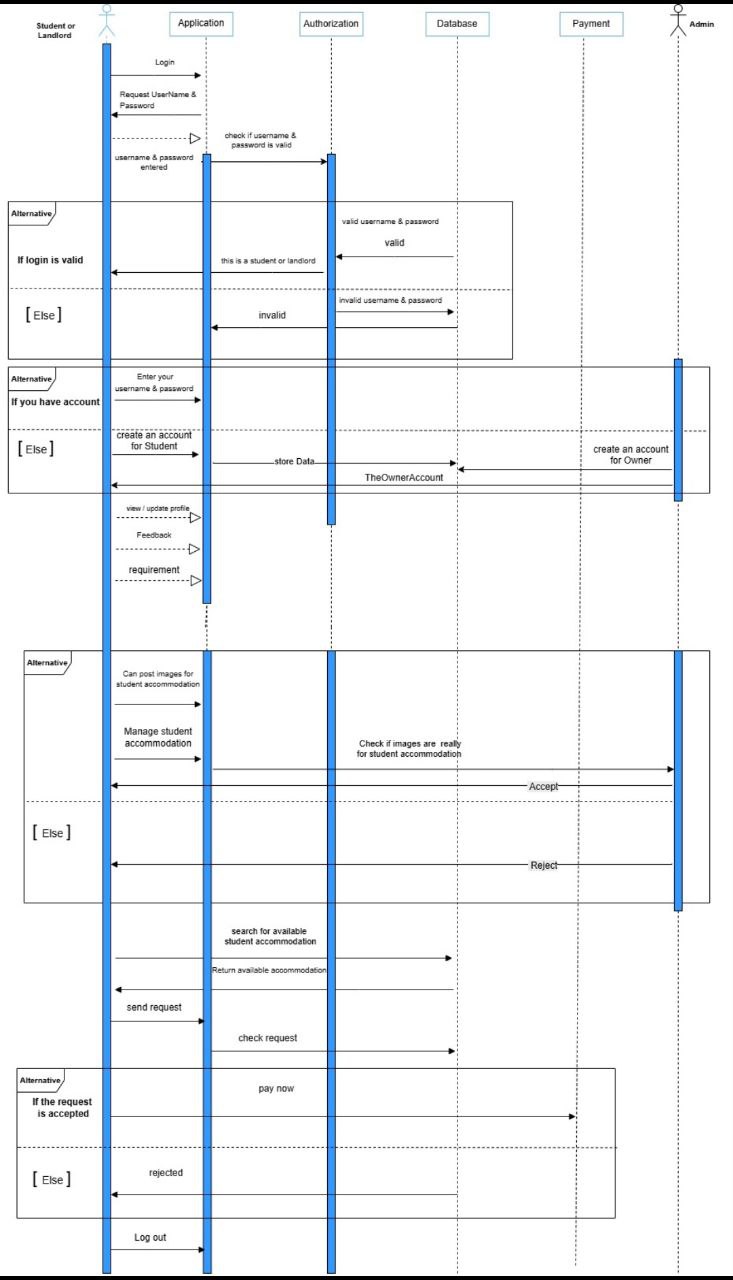
Furthermore, the platform incorporates secure online booking and payment functionalities to mitigate risks related to fraud and financial disputes, which are common in informal rental markets. By enabling direct and secure communication channels between tenants and landlords, E-Rent fosters transparency and trust, essential elements in the Egyptian rental context.

A distinctive aspect of this solution is the integration of artificial intelligence to personalize recommendations. The system dynamically learns from user interactions and contextual data to suggest housing options that best match individual needs, enhancing user satisfaction and decision-making efficiency.

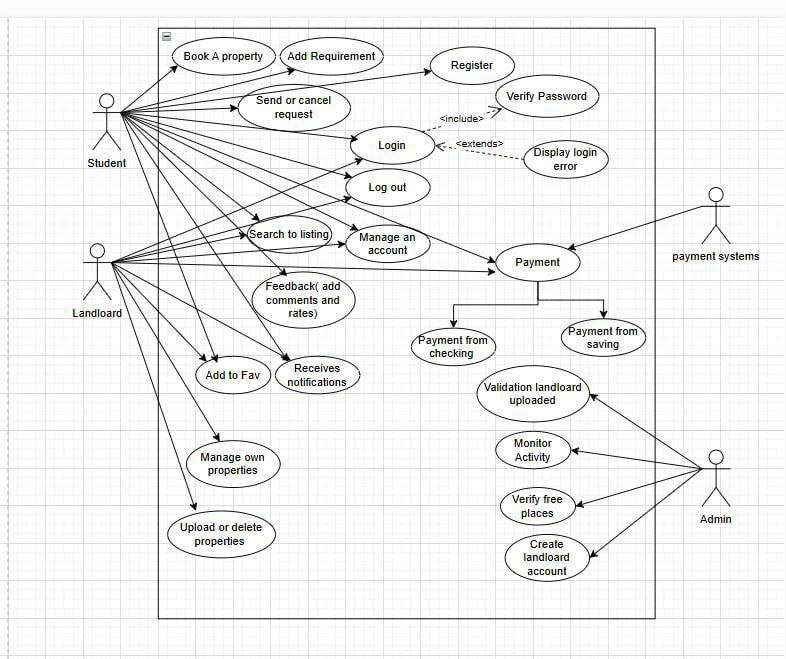
Importantly, while the platform primarily serves the Upper Egypt region, it is accessible nationwide, allowing students from across Egypt to reserve accommodations in this area. This design choice acknowledges the mobility of the student population and the centralized role Upper Egypt plays in regional education.

This multifaceted approach, combining technological robustness with contextual sensitivity, aims not only to streamline the housing search and rental process but also to contribute positively to the social and academic well-being of students in Upper Egypt.

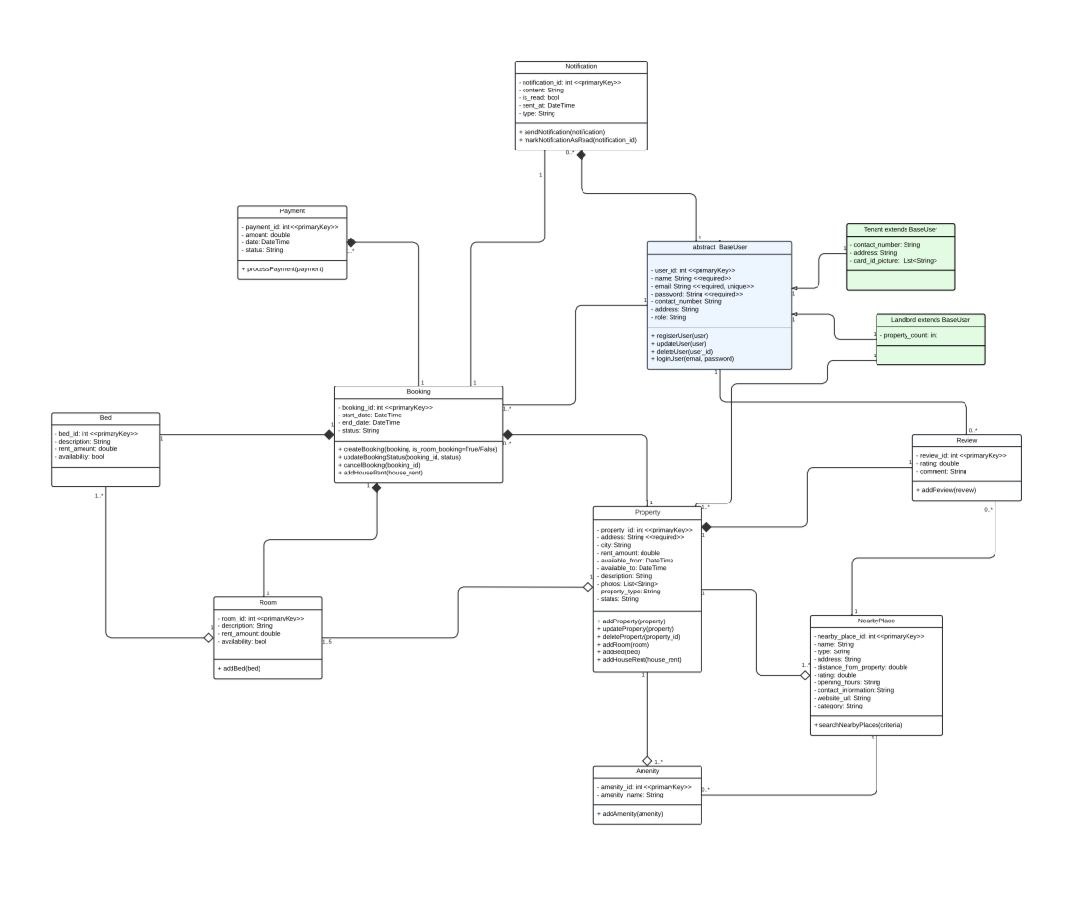
# 3.2 System architecture



* This diagram represents a **sequence diagram** for a Student Housing App. It outlines the interactions between the user (student or landlord), the application, the authorization system, the database, and the payment system



* This diagram **Use case diagram** illustrates how different users interact with the E-rent system, detailing the functionalities available to each type of user. It provides a clear overview of the system's capabilities and user roles.



● This **class diagram** provides a comprehensive overview of the rental management system's structure, displaying how users interact with properties, bookings, payments, and other features. It serves as a blueprint for the system’s design and functionality, ensuring clarity in relationships and responsibilities among different classes.

# 3.3 Algorithms or Frameworks Used (Detailed Explanation of the Recommendation System)

The recommendation system implemented within E-Rent constitutes a central technological pillar that enhances the platform’s ability to deliver personalized housing suggestions tailored to the diverse needs of students. This system is grounded in a **Rule-Based Recommendation System** framework that uses explicitly defined “if-then” logical rules to evaluate housing options and generate relevant recommendations.

**Rule-Based System Framework**:

Unlike machine learning-based recommenders that rely on large historical datasets and require model training, the rule-based system operates using a set of predefined logical conditions that directly reflect student preferences and housing criteria. For example, the system checks whether the student's gender matches the allowed gender for the room, whether the price is within the student's budget, and whether the required amenities (such as Wi-Fi, air conditioning, or proximity to campus) are available. If all the specified conditions are met, the system recommends the corresponding housing option. This approach ensures a clear, explainable, and direct mapping between user needs and the system’s output.

**Strengths of Rule-Based Systems:**

One of the key advantages of a rule-based recommendation system is its **transparency and interpretability**. Users and administrators can easily trace each recommendation back to specific rules, which is especially valuable in sensitive domains like housing allocation. Additionally, the system offers **high flexibility and ease of customization**, allowing developers to quickly add, remove, or modify rules without retraining a model or relying on large volumes of data. **Computational efficiency** is another strength, as decisions are made by evaluating simple logical conditions, which is fast and lightweight. Furthermore, rule-based systems perform effectively even in **data-scarce environments**, making them ideal for new platforms or early development stages where historical data is limited or unavailable.

**Weaknesses of Rule-based systems :**

Despite their advantages, rule-based systems have some notable limitations. They **lack adaptability**, as they cannot learn from user interactions or changes in behavior over time. Any adjustments to reflect new trends, preferences, or market conditions must be made manually by updating the rules. As the number of rules increases, the system may become harder to manage, leading to **conflicts, redundancies**, or errors in logic. Moreover, unless the rules are written with extreme precision and cover a wide range of scenarios, the system may fail to capture **subtle or complex user preferences**, leading to less personalized results. In highly dynamic or complex environments, this rigid logic may not perform as well as **machine learning-based models**, which can automatically adapt and improve based on real user data.

**Comparative between Rule-based systems vs. Machine Learning-based system:**

|  |  |  |
| --- | --- | --- |
| Feature | Rule-Based Recommendation System | Machine Learning-Based Recommendation System |
| Working Principle | Uses manually defined "if-then" rules | Learns patterns from data using ML algorithms |
| Personalization | Provides static, general recommendations | |  | | --- | |  |  |  | | --- | | Offers personalized recommendations based on user behavior | |
| Data Dependency | Does not rely heavily on data | Requires large and diverse datasets to perform well |
| Flexibility & Scalability | Hard to modify or scale; rules must be updated manually | Highly flexible and scalable with automatic learning |
| Accuracy & Improvement | Limited accuracy; improvements require manual updates | Continuously improves with more data and user interaction |

Chapter 4

Implementation

# 4.1 Technologies, tools, and programming languages used

The E-Rent platform is built upon a modern technology stack, with Firebase serving as the backbone of the backend infrastructure to ensure scalability, security, and real-time interactivity—features essential for the dynamic student housing market in Upper Egypt.

*Key Technologies and Tools:*

We used UI/UX principles in our project and relied on Figma to design the user interfaces.

Figma made the design process much easier as it allows us to create wireframes, choose color schemes, and design interactive components efficiently. It also supports real-time collaboration, which was very helpful for our team during the design phase.

For example, we used Figma to design the homepage of our application. We started with low-fidelity wireframes to plan the layout, then moved on to high-fidelity prototypes with our chosen color palette and typography. We added interactive buttons and navigation elements so we could simulate how a user would move through the app. This helped us identify usability issues early and improve the overall user experience before development began.

*Flutter & Dart:*

The mobile application is developed using Flutter, Google’s UI toolkit, and Dart programming language. This combination enables rapid, cross-platform deployment and ensures a consistent, responsive user interface, which is crucial for Egyptian students who primarily access services via smartphones.

*Firebase Backend:*

Firebase Cloud Firestore: Acts as the primary NoSQL database, providing real-time synchronization of property listings, user profiles, and bookings. Firestore’s scalability and flexible data model allow the platform to adapt as the user base expands.

Firebase Authentication: Handles secure user registration, login, and identity verification, supporting multiple authentication methods to enhance accessibility and trust.

Cloud Functions for Firebase: Implements backend business logic, such as processing bookings, sending notifications, and integrating with payment gateways. These serverless functions are triggered by HTTPS requests or database events, reducing operational overhead and improving scalability.

Firebase Hosting: Delivers static and dynamic content over a global CDN, ensuring fast, reliable access to the application.

Real-Time Messaging: Utilizes Firestore’s real-time capabilities and Cloud Functions to support instant messaging between students and landlords.

Integration with Local Payment Gateways: Although Firebase does not directly process payments, it integrates with Egyptian payment providers like Fawry and Paymob via Cloud Functions, enabling secure in-app transactions.

*Programming Languages:*

* Dart: For Flutter mobile app development.
* JavaScript/TypeScript: For writing Cloud Functions and backend logic.

Advantages of Using Firebase:

Firebase’s serverless architecture eliminates infrastructure management, allowing the team to focus on application logic and user experience. Real-time data synchronization, built-in security, and automatic scalability make Firebase particularly well-suited for the fast-paced, trust-sensitive environment of student housing in Upper Egypt.

# 4.2 Key Components/Modules of the System

The E-Rent platform is organized into several core modules, each addressing a specific aspect of the student housing lifecycle. All modules are tightly integrated with Firebase services to ensure real-time performance and scalability.

User Management Module: Manages registration, authentication, and profile data for students and landlords using Firebase Authentication for secure access control.

Property Listing Module: Landlords can upload, edit, and manage property details, with all data stored and synchronized in real-time via Firestore.

Search and Filtering Module: Provides students with advanced search tools, leveraging Firestore’s indexing to deliver fast, relevant results based on location, price, gender, and other preferences.

Recommendation Engine: Implements a Contextual Multi-Armed Bandit (Rule-based system) algorithm, utilizing user context and interaction data stored in Firestore to generate personalized housing recommendations.

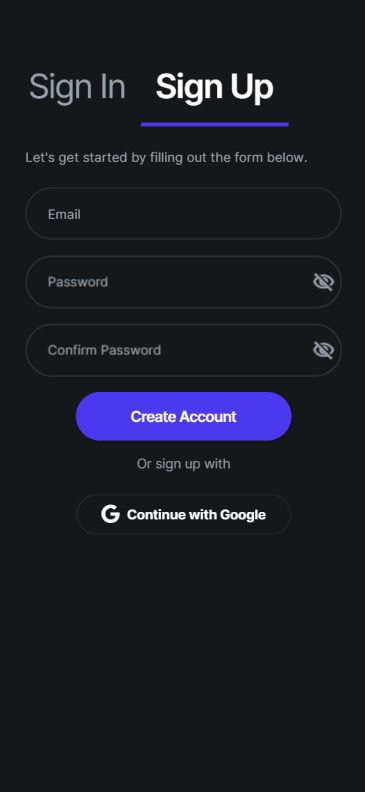
Booking and Payment Module: Facilitates secure online bookings and rent payments, integrating with local payment gateways through Cloud Functions for transaction processing.

Lease Management Module: Supports digital creation, signing, and storage of lease agreements, ensuring transparency and legal compliance, with documents securely managed in Firebase Storage.

Security and Privacy Module: Ensures data protection through Firebase’s authentication, encrypted data transmission, and strict access controls.

Each module is designed for seamless integration, providing a cohesive user experience and supporting future scalability and feature expansion.

## 4.2.1 Front end Implementation



**Function Explanation – Create Student Account**

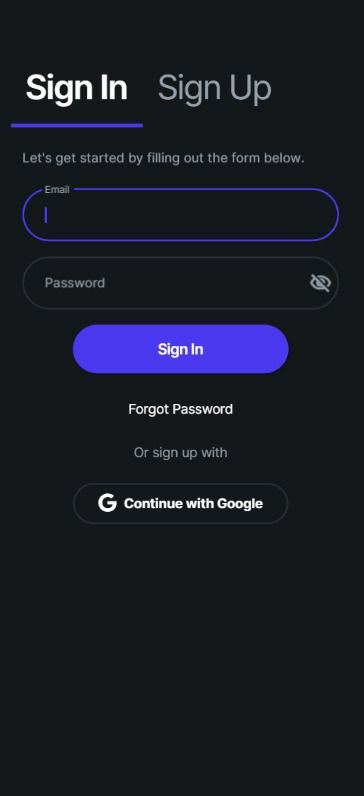
This function allows a **student** to create their own account on the E-Rent platform.

When the student fills out the **SignUp form** (with their email, password, and confirm password), this function gets triggered to handle the registration process.

1. **Password Check**:  
   First, the function checks if the student typed the same password in both fields. If the two passwords do not match, the account creation is stopped and an error message is shown.
2. **Account Creation**:  
   If the passwords match, the function creates a new user account using the email and password provided. This account is saved in the authentication system of the app (Firebase Authentication).
3. **Storing Student Information**:  
   After creating the account, the function also saves additional information about the student—such as their email, registration time, and their role as a "student"—in the app’s database (Firestore). This helps the system know that this user is a student and not a landlord or admin.
4. **Error Handling**:  
   If anything goes wrong—like using an already registered email, or entering a weak password—the function catches the problem and shows a clear message to the student so they can fix it.

**Important Note:**

* Only **students** use this Sign-Up function.
* **Owners** (landlords) are not allowed to create their own accounts. Instead, **admins create accounts for them** manually to ensure trust and verification.



**Function Explanation – Sign In (Student Login)**

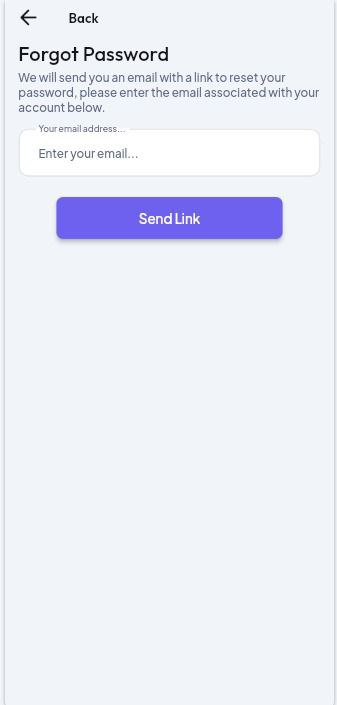
This function allows **existing students** to log into their E-Rent account using their **email and password**.

**Here's what happens step-by-step:**

1. **User Input**:  
   The student enters their **email** and **password** in the form.
2. **Authentication Check**:  
   When the student clicks on the **“Sign In”** button, the function checks the entered email and password against the database (using Firebase Authentication) to verify if the user exists and the credentials are correct.
3. **Successful Login**:
   * If the email and password match a registered student account, the user is logged in successfully.
   * The app then redirects them to the student home page or dashboard where they can start using the features (like searching for housing, booking, messaging, etc.).
4. **Error Handling**:  
   If there’s a mistake—such as:
   * The email is not registered.
   * The password is incorrect.
   * Or the account doesn't belong to a student — The function will show a clear error message (e.g., "Incorrect email or password").
5. **Forgot Password Option**:  
   If the student forgets their password, they can click **"Forgot Password"** to receive a password reset email.
6. **Sign in with Google**:  
   Students also have the option to sign in using their **Google account**, making the login process faster and more convenient without needing to remember a separate password.

**Who Uses This?**

This page is used only by **students** to access their accounts.  
**Admins** and **landlords** have different login mechanisms—especially since **landlords are created by admins**, not self-registered.



This screen handles the "Forgot Password" function for students who have lost access to their accounts.

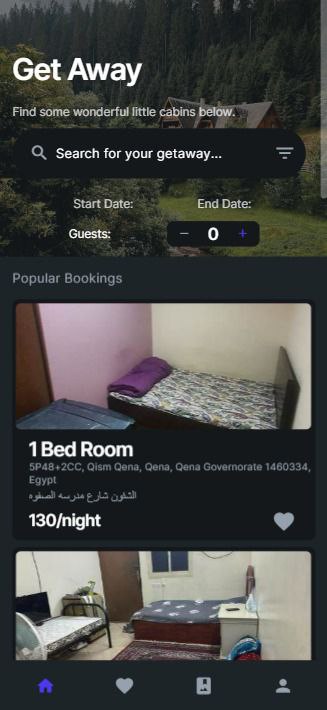
The student enters the email address they used to register with the platform in the input field.

Once they click the "Send Link" button, the system automatically sends an email to that address.

The email contains a secure link that redirects the student to a password reset page, where they can create a new password and regain access to their account.

This feature is important because it allows students to recover their accounts easily and securely without needing to contact support or the admin.

It's designed specifically for students and owners who already have accounts, and it's not part of the admin's role.



**Function of the Home Page (Student Side)**

The Home page is the first screen the student sees after logging into the app. It allows students to search for available rooms based on date, number of guests, and view popular listings in Upper Egypt.

**Key Features:**

Search Bar:

The student can search by keywords (e.g., area, room type).

Helps quickly find listings based on specific needs.

Filter Button (beside the search bar)

Room type (shared/private)

Allows the student to open advanced filters, such as:

Price range

Distance from university

Amenities (Wi-Fi, washing machine, etc.)

This gives more control to the student to narrow down the search and find suitable housing faster.

Date Filter:

Students can select a start and end date to define the duration of their stay.

Useful to check availability during specific academic terms or holidays.

Guests Counter:

Lets the student select how many people will stay.

Ensures the system recommends suitable room sizes.

Popular Bookings Section:

Shows a list of frequently booked or highly rated listings with pictures, prices, and brief address info.

Each card includes:

Room name (e.g., 1 Bed Room)

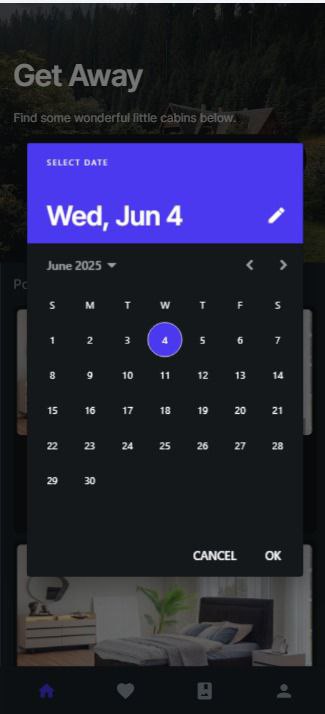
Address details

Price per night

Small heart icon for saving favorites

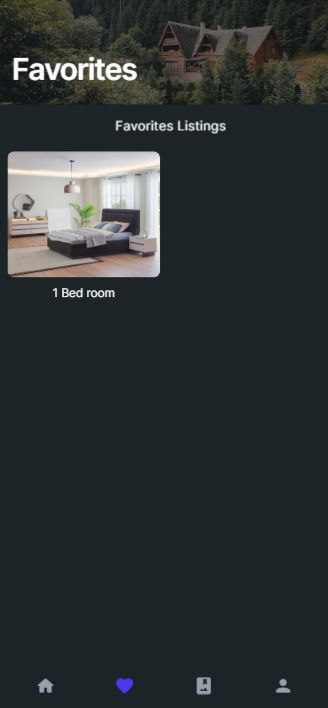
**Purpose:**

This page aims to provide a fast and friendly interface for students to start searching for rooms right away. It simplifies the discovery of housing options by combining filters, visual previews, and recommendations based on popularity.



**Calendar on Home Page**

The **Calendar** on the Home Page allows users to select a start date and an end date for their desired housing period.  
By choosing these dates, users can filter available housing units that are free during the selected time frame.  
This helps users find accommodations that match their specific stay duration easily and quickly.



**Function Description: Favorites Page**

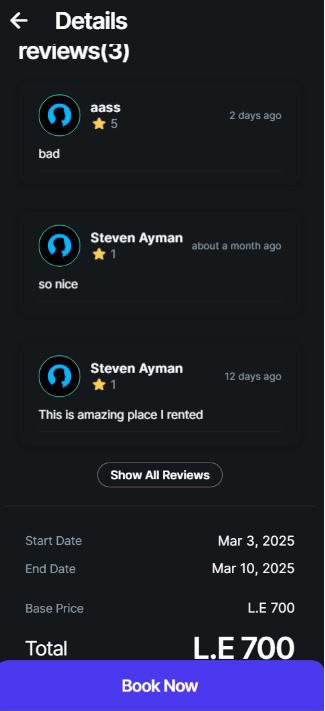
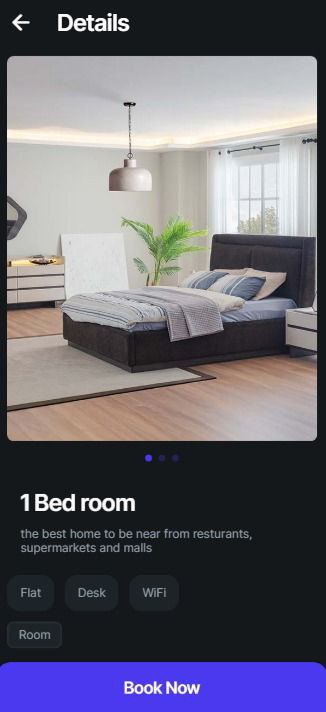
The Favorites page allows the student to view all properties they have marked as favorites. It is designed to display **only the property images**, without any extra details such as price, location, or description. This provides a clean and simple interface.

Each image represents a property that the student previously favorited. When the student **taps on any image**, the app navigates to the **Property Details Page**, where full information about that property is displayed.

**Main Functionalities:**

1. **Data Fetching:**
   * The function connects to Firebase Firestore.
   * It reads from the subcollection favorites inside the current user's document (users/{userId}/favorites).
   * This subcollection contains documents representing each favorited property.
2. **Displaying Data:**
   * A GridView is used to show the favorite property images in a grid layout (e.g., 2 images per row).
   * Each image is fetched from the imageUrl field stored in the favorite document.
3. **Interaction:**
   * Each image is wrapped in a GestureDetector to allow user interaction.
   * When the user taps an image, the app navigates to the DetailsPage, passing the property ID so that full property details can be shown.
4. **Real-Time Updates:**
   * The function uses a StreamBuilder to listen for real-time updates from Firestore.
   * If a user adds or removes favorites, the UI updates automatically.
5. **Empty State & Loading:**
   * If the student has no favorites, a message is shown: “No favorite listings yet.”
   * While data is being loaded, a CircularProgressIndicator is shown.

This function helps users easily manage and access their favorite properties, with a smooth and simple design focused on visual browsing.



**Details Page**

The **Details Page** provides a complete overview of a selected housing unit, allowing students to view all necessary information before making a booking decision. This page is a crucial part of the user experience as it supports informed decisions through visual previews, property features, reviews, and booking functionality.

**1. Property Image & Basic Info**

At the top of the screen, a high-resolution image of the room is displayed to give the student a clear view of the unit’s appearance and environment.

Below the image, basic details are shown:

* **Room Type:** (e.g., "1 Bed room")
* **Short Description:** A brief summary that highlights the benefits of the location, such as being close to restaurants, malls, and supermarkets.
* **Property Tags:** Useful features like:
  + Flat
  + Desk
  + Wi-Fi
  + Room

These tags help students quickly understand the available amenities.

**2. Location & Specifications**

A button labeled **"Location"** allows the student to open a map and see where the housing unit is located.

Additional specifications shown on the page include:

* **Specified Gender:** Indicates whether the property is available for males, females, or both.
* **Rooms:** Number of rooms in the unit.
* **Beds:** Number of beds.
* **Baths:** Number of bathrooms.

This information helps users evaluate if the space meets their personal and cultural preferences.

**3. Reviews Section**

The reviews section displays feedback from other students who have previously booked the unit. Each review contains:

* **User’s Name**
* **Rating (in stars)** out of 5
* **Time** the review was posted (e.g., “2 days ago”)
* **Comment**: A short message such as “bad” or “so nice”.

If multiple reviews exist, a button labeled **"Show All Reviews"** is available to expand and display more feedback.

This section is essential for helping new users build trust and confidence in the listing.

**4. Booking Details**

At the bottom of the page, booking-related information is displayed, including:

* **Start Date** of the reservation.
* **End Date** of the stay.
* **Base Price** (price per week or per selected period).
* **Total Amount**: Automatically calculated based on the stay duration.

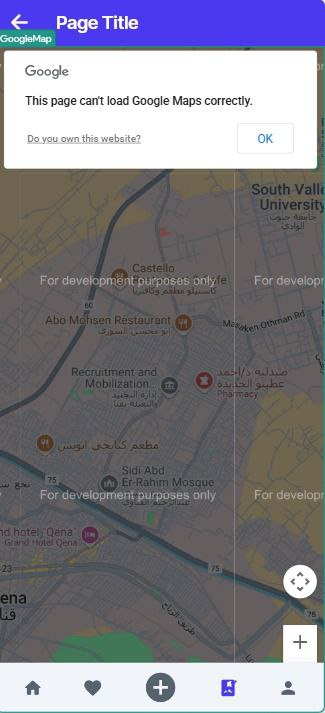
Finally, a large and clear **"Book Now"** button is available. When clicked, the system processes the booking, initiates a request to the owner, and eventually connects the student with the landlord.

**Summary of Functionality**

The Details Page allows the student to:

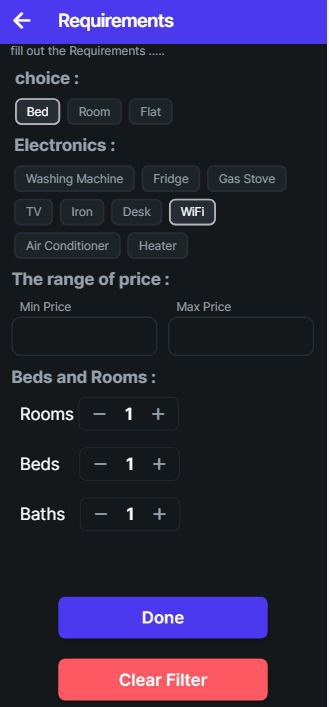
* View the appearance and features of the unit.
* Read real feedback from previous users.
* Understand all conditions and prices.
* Confirm and submit a booking request.

This screen plays a key role in user decision-making and contributes to the platform’s overall usability and reliability.



**Map Page**

The **Map Page** displays the exact location of the selected housing unit on Google Maps.  
When the user clicks on the location button in the Details Page, this page opens a map view highlighting the house’s position based on its latitude and longitude or address.  
This helps users easily visualize where the housing unit is situated and get directions if needed.



**Requirements Page**

This page allows the user to filter housing options based on their personal preferences and needs. Here's a breakdown of the sections:

**1. Type Selection**

At the top of the page, the user chooses the type of accommodation they are looking for. The available options are:

* **Bed**
* **Room**
* **Apartment**

The user selects one of these three options depending on what they need.

**2. Appliances Selection (Multi-choice)**

Next, the user selects which electrical appliances or features they want the place to have. This section allows **multiple selections**.  
Examples of available options include:

Users can select any combination of these based on their preferences.

**3. Price Range**

The user enters the **minimum** and **maximum** price range they are looking for.

* This helps filter the results to show only listings within their budget.

**4. Counters**

There are three counters to specify the number of rooms, beds, and bathrooms required:

* **Rooms**: Number of rooms in the apartment
* **Beds**: Number of beds needed
* **Baths**: Number of bathrooms required

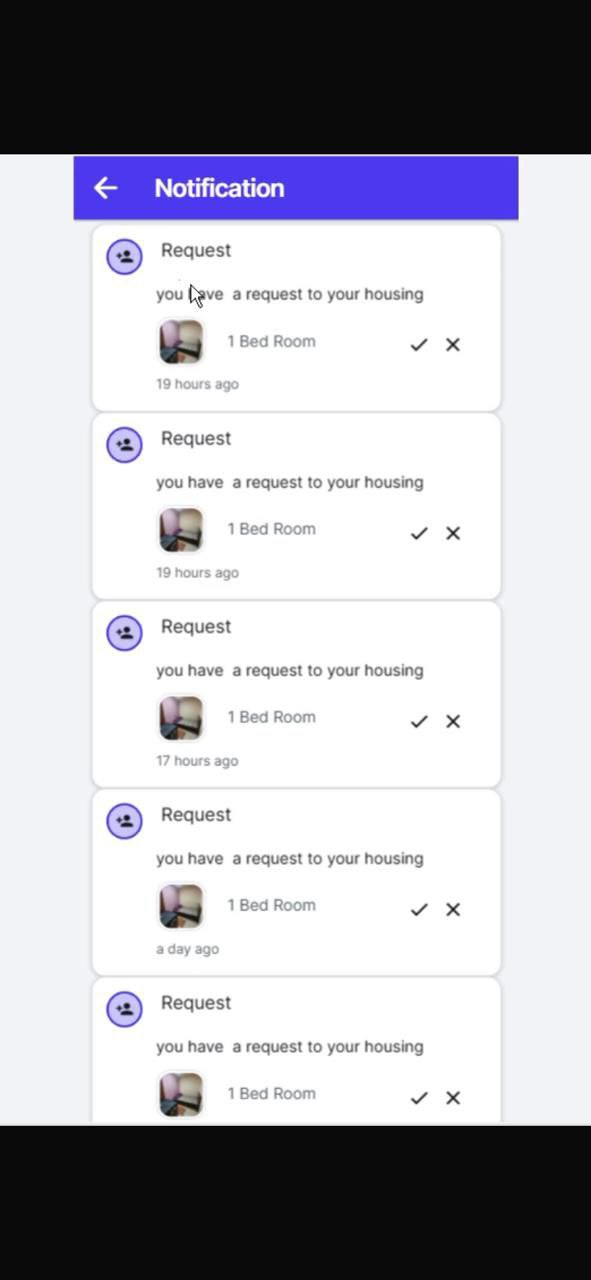
Each one can be increased or decreased using a counter UI.

**5. Done Button**

* When the user clicks the **Done** button, the app applies all the filters.
* The filtered results are displayed on the **Home** page, showing only listings that match the selected requirements.

**6. Clear Filter Button**

* Pressing the **Clear Filter** button will remove all selected filters and reset the form.
* This allows the user to start over or view unfiltered results again.

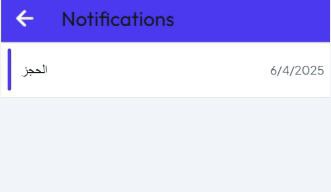


The **Owner’s Notification Page** displays all updates related to booking requests and user interactions with the owner's listed accommodations. When a student sends a **booking request**, the owner receives a **notification** showing the **image of the accommodation** the request was made for. By clicking on the image, the owner is redirected to the **details page** of that accommodation. If the owner clicks on the **student’s name or profile picture**, they are taken to the **student’s profile page**.

From the notification, the owner can either **accept** or **reject** the request. In both cases, the student will receive a **notification** informing them of the owner’s decision.

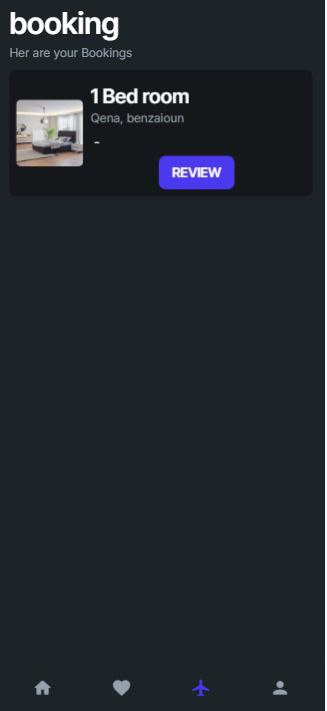
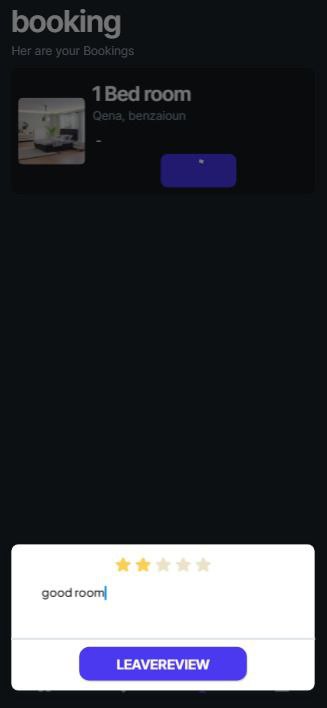
If the owner does **not respond** to the request within **two days**, the request will be **automatically deleted**.

Additionally, the owner will receive a notification whenever a student **comments** on or **rates** one of their listed properties.



**Student’s Notification Page**

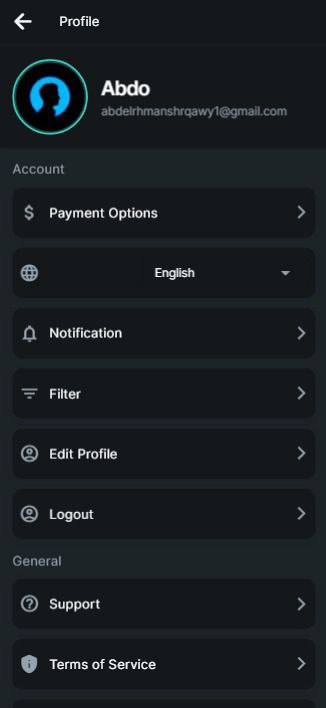
The **Student’s Notification Page** displays important updates related to their booking requests and interactions with property owners. Students receive notifications when an **owner accepts or rejects** their booking request. Additionally, they are notified if the owner **leaves a comment** or gives them a **rating** based on their interaction or stay. This page helps students stay informed and updated on all responses and feedback related to their accommodation activity.

**Booking Page**

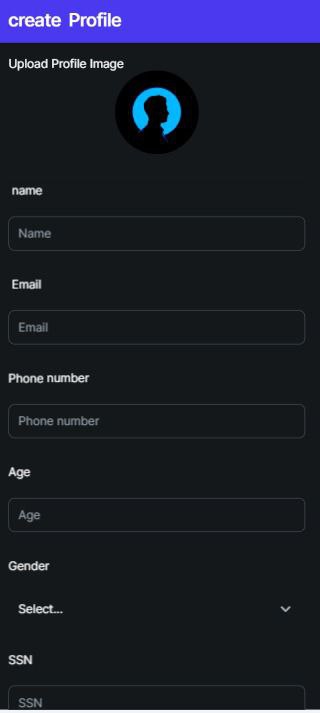
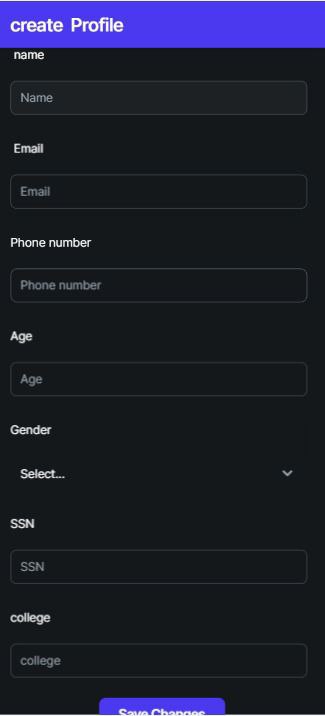
This page displays **only the accommodations that the user has previously stayed in** (i.e., completed bookings).

The **Booking Page** displays only the accommodations that the user has previously stayed in. It excludes any upcoming bookings or listings the user hasn’t completed yet. On this page, the user has the ability to leave comments and provide ratings (such as star ratings) for each place they’ve stayed at. These feedback options are available **only** for completed stays to ensure that reviews are based on actual experiences. Once submitted, the user's comments and ratings will be displayed on the **Details Page** of the corresponding accommodation, helping future users make more informed decisions.



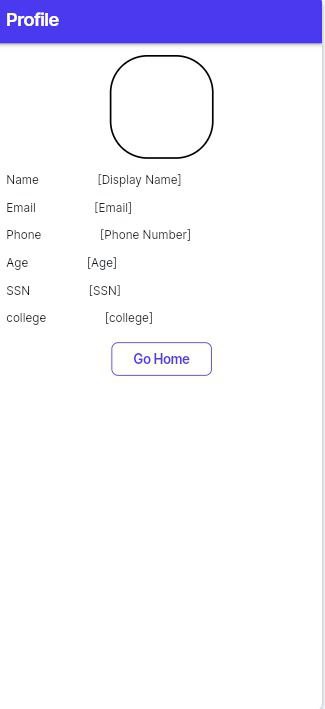
**Profile Page**

The **Profile Page** contains several user settings and options to manage their account and preferences. Users can select their preferred **payment method**, such as Vodafone Cash or bank transfer. They can also choose the **language** of the app interface, either **Arabic** or **English**. Additionally, the page includes access to **notifications**, which show important updates and alerts whether the user is a student or an owner. Users can also access the **Filter** section to manage saved search preferences and use the **Edit Profile** option to update their personal information. There is also a **Log Out** option to securely sign out of the app, as well as sections for **Support** and **Terms of Service**, where users can find help or review the platform’s policies and conditions.

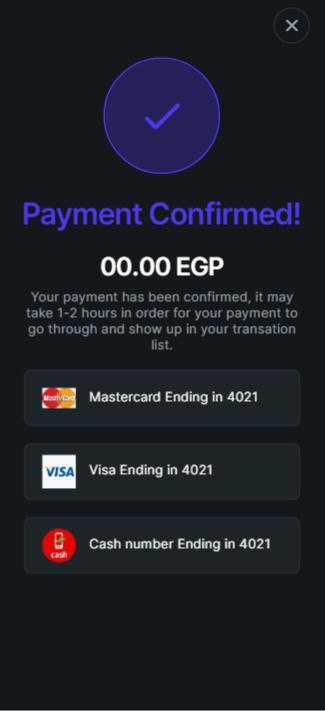
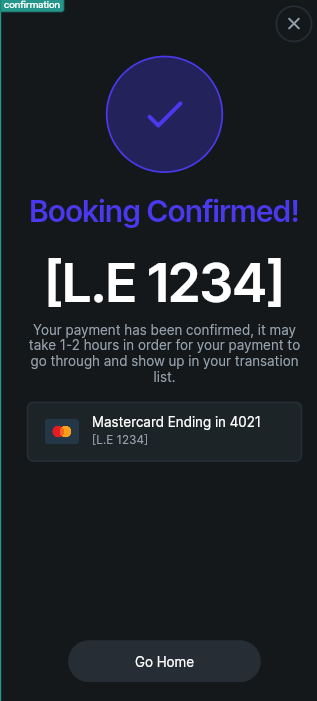
**Edit Profile Page**

The Edit Profile Page allows the user to update and manage their personal information. From this page, the user can modify details such as their name, email, phone number, age, SSN, college, and gender. The gender field helps the system suggest accommodations that are appropriate for the user's type — for example, male students will see different housing options than female students, as some listings are gender-specific. Users can also change their profile picture to keep their account visually up to date. This gives users full control over their profile data, ensuring that the information shown to others is always accurate and current. After making changes, the user can click the Save Changes button to apply and store the updated information.



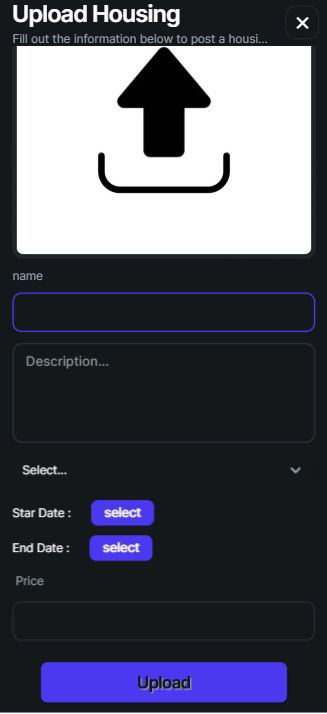
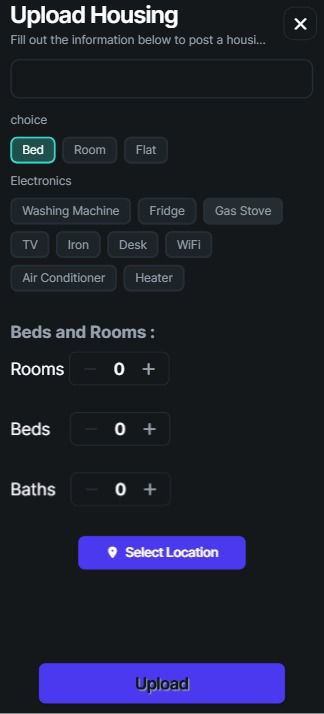
**Profile Page**

The **Profile Page** is the public profile that other users can view when they click on someone's profile picture or visit their profile. It displays personal details that are allowed to be shown to others. These details include the user’s **name**, **email**, **phone number**, **age**, **SSN (National ID)**, and **college**. This information helps provide transparency and trust between users, whether they are students or property owners, while still maintaining control over what is publicly visible.

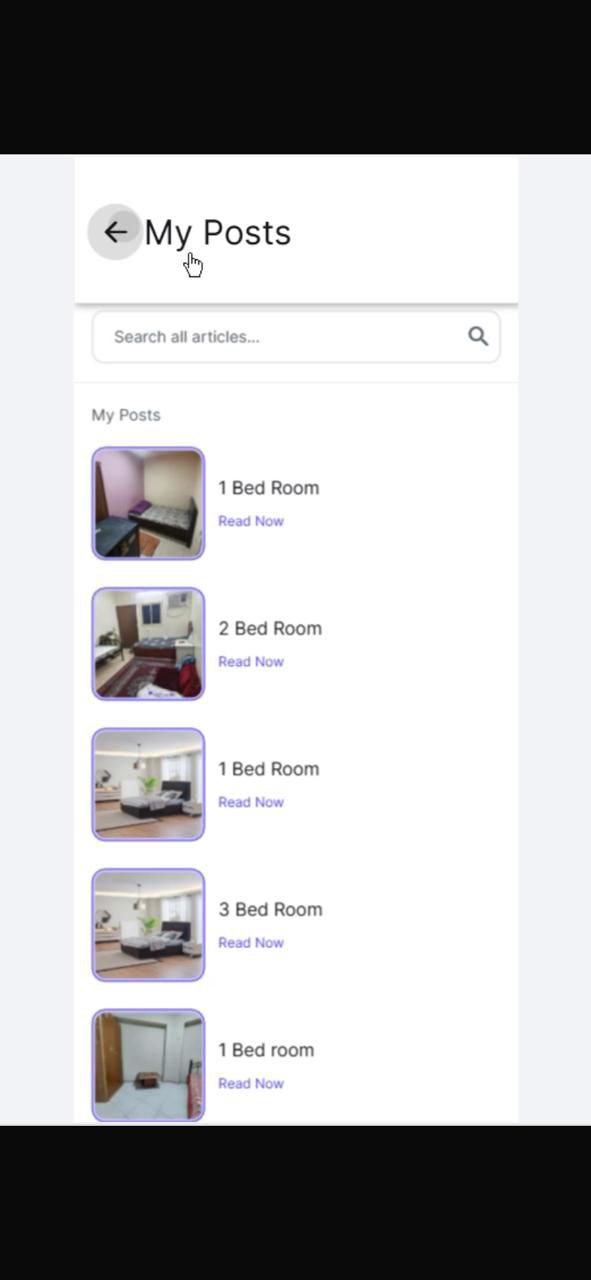
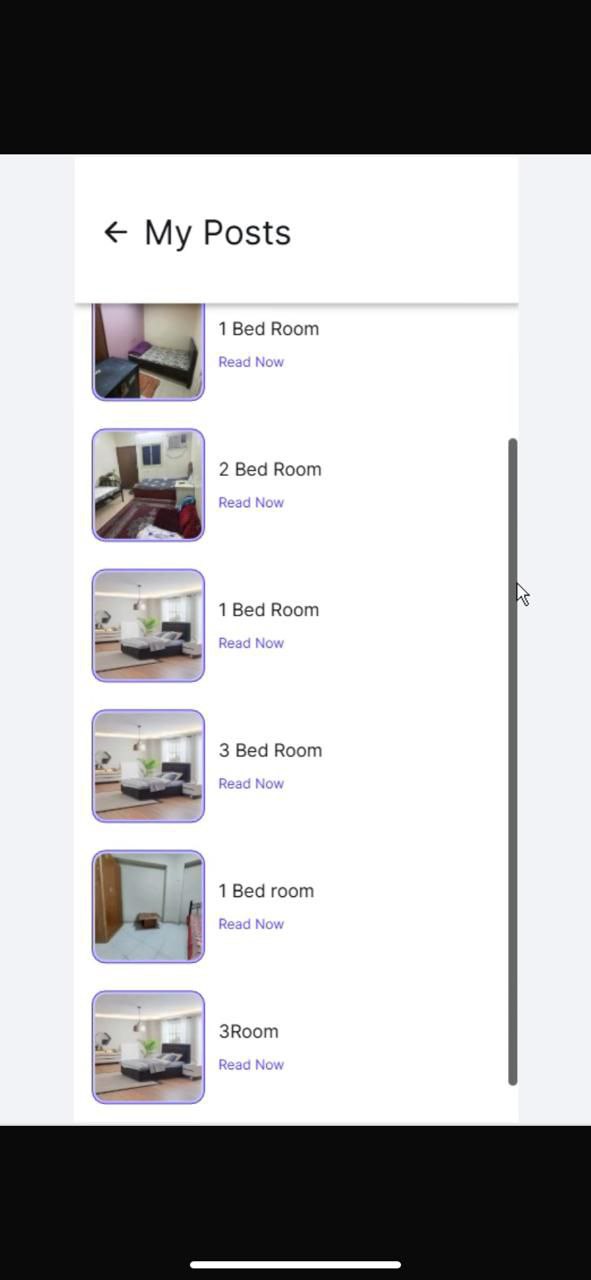
**Payment Confirmed Page**

The **Payment Confirmed Page** presents the user with three payment options to complete their transaction within the app. The available methods are **Mastercard**, **Visa**, and **Cash**. After the user completes the payment, a **confirmation message** is displayed showing the **transaction number** or **payment reference number** to assure them that the payment was successful and recorded.

**Upload Page**

The **Upload Page** is available exclusively to **owners** and is used to post new accommodation listings. The process starts with uploading **images of the property**, followed by entering the **name** of the accommodation and a brief **description**. The owner then specifies the **available rental period** by selecting a **start date** and **end date**. After that, they set the **price** of the property and choose whether they are renting out a **bed**, **room**, or **entire apartment**. They can then select the available **electronic appliances** (e.g., WiFi, fridge, TV, etc.) using a **multi-choice** option. Counters are provided to indicate the number of **rooms**, **beds**, and **bathrooms** in the listing. Finally, the owner selects the **location** of the property on the map and clicks the **Upload** button to publish the listing for students to view and book.

**My Post Page**

The **My Post Page** is dedicated to the **owner** and displays a list of all the accommodations they have uploaded to the app. Each property is shown with its image, and when the owner clicks on the **image of a property**, they are taken to the **details page** of that specific listing. This page allows owners to easily access and manage the accommodations they have posted.

## 4.2.2 Back end implementation

#  **Custom Functions**

**1-getUniqeList**

the file contains a simple function that removes duplicate items from a list of strings. The function returns a new list that includes only the unique elements.

Overall, the code helps set up the environment for the app and provides a utility function for cleaning up repeated data.

**2-getPlaces**

The file sets up the app environment using useful packages for fonts, dates, maps, and Firebase services. The main function, getPlaces, checks whether the user is performing a search. If so, it returns the filtered list of results; otherwise, it returns the full list of places. This helps the app display relevant data based on the user's interaction.

**3-calculateBasePrice**

The function calculates the total price for the stay by determining the number of days between the check-in and check-out dates, then multiplying that by the price per night. The result is the base price that can be shown to the user before adding any additional fees.

**4-getBookingList**

The function filters the bookings to find those that overlap with the specified date range. It returns a list of place references (placeStringRef) for those overlapping bookings.

**5-add7Days**

The function adds 7 days to the given date. If the input date is null, the function safely returns null without throwing an error.

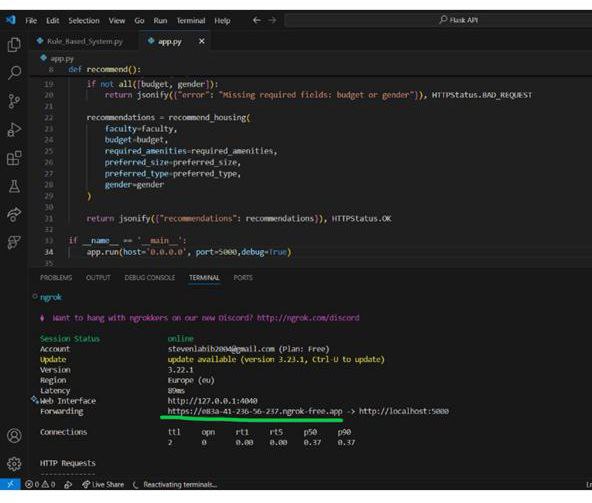
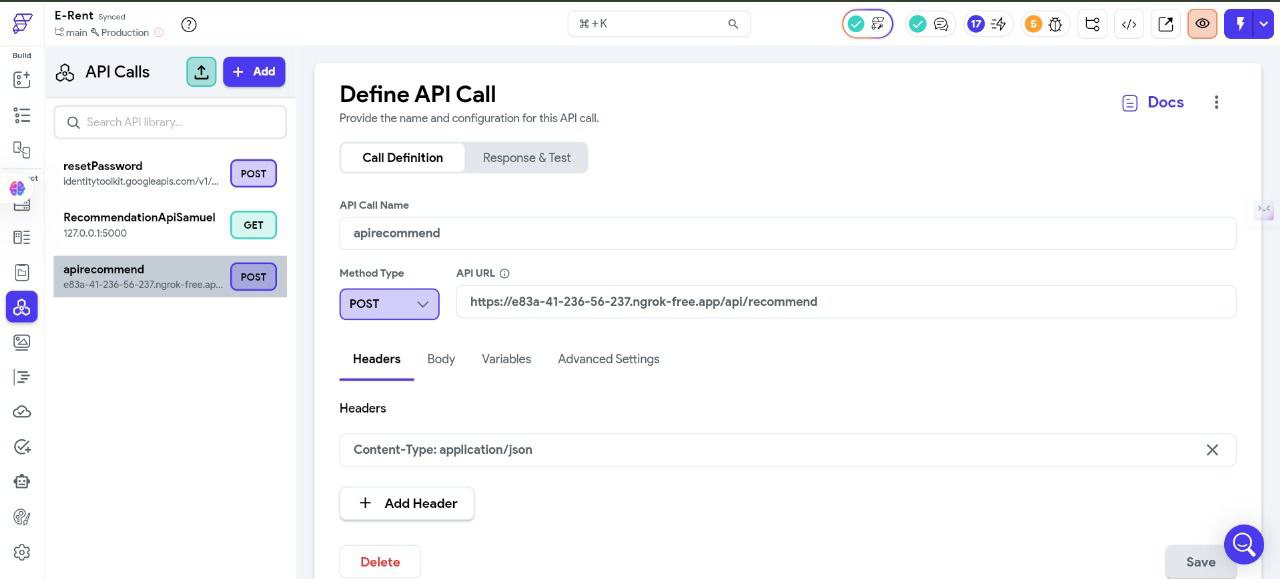
## 4.2.3 AI Implementation

In this stage, we implemented a RESTful API using Python and the Flask framework.

The API is designed to receive user data in JSON format — including fields like gender, marital status, budget, and number of family members — and then call a recommendation function that processes the input and returns the most suitable housing options.

To make the API accessible from outside the local machine, we used ngrok to expose the Flask server running on localhost:5000 to the internet.

In FlutterFlow, we configured an API Call by inserting the ngrok-generated URL, setting the method to POST, and providing a sample JSON body for testing. This setup allows FlutterFlow to send requests directly to the local API and receive responses in real time.

code represents a simulated housing database that includes various housing options available to students, each with different specifications to suit the needs of students from different faculties.

Purpose of the Code

The code aims to offer a simplified model of a database containing detailed information about housing units. This model can be used in an application that helps students easily find suitable accommodation.

Code Components

* housing\_options List:  
  Contains diverse housing options, where each unit represents a unique choice with different specifications such as faculty cluster, price, housing type, available facilities, and gender designation.
* faculty\_clusters List:  
  Groups faculty names under each faculty cluster, for example, the medical cluster includes Medicine, Pharmacy, Dentistry, and others.

Expected Outputs

The final output is a selection of housing units that meet the student’s personal and financial requirements. This list allows students to view suitable options, helping them make informed decisions when choosing accommodation.

The code serves as a practical model for a comprehensive housing database, enabling organized classification and display of housing options for students. This simplifies the search and selection process, providing a smooth and convenient user experience**.**

**If you would like to view more detailed information about the code, it is available in the appendices.**

# 4.3 Challenges Faced and How They Were Resolved

During the development and deployment of E-Rent, several challenges were encountered, particularly due to the regional focus on Upper Egypt and the need for a secure, scalable platform. The integration of Firebase played a crucial role in addressing these challenges:

Data Scarcity and Cold-Start Problem: Initially, limited user interaction data affected the accuracy of the recommendation system. This was mitigated by combining content-based filtering with contextual bandits until sufficient data allowed the Rule-based system model to perform optimally.

Payment Security and Integration: Integrating secure, locally accepted payment gateways was complex due to regulatory requirements and infrastructure variability. Utilizing Firebase Cloud Functions, the platform securely connects with providers like Fawry, with fallback mechanisms for intermittent connectivity.

Trust and Verification: Building trust required rigorous verification of landlords and listings. Multi-step verification, document uploads, and manual reviews were implemented, supported by user feedback and rating systems—all managed and tracked via Firestore.

Balancing Recommendation Exploration and Exploitation: The Rule-based system algorithm’s uncertainty bonus was instrumental in ensuring a balance between recommending familiar properties and exploring new listings.

Localization and Accessibility: UI/UX design was tailored through user testing with local students, optimizing language, navigation, and accessibility for the Upper Egypt context.

Scalability and Performance: Firebase’s serverless infrastructure and real-time synchronization allowed the platform to scale automatically and maintain stability, even as the user base grew.

Each challenge was systematically addressed through a combination of Firebase’s technical capabilities and user-centered design, resulting in a robust and reliable platform for student housing in Upper Egypt

Chapter 5

Testing & Evaluation

# 5.1 Testing Strategies

To ensure the robustness, reliability, and user satisfaction of the E-Rent platform, a comprehensive multi-level testing strategy was employed, encompassing unit testing, integration testing, and user testing.

## 5.1.1 Unit Testing:

Individual components and functions, particularly in the Flutter frontend and Firebase Cloud Functions backend, were rigorously tested to verify correct behavior in isolation. Automated unit tests helped detect logic errors early in development and ensured that core functionalities such as user authentication, property data handling, and booking workflows behaved as expected.

For example, during the testing of the AI model (Rule-Based System):

Since this is a rule-based system, the "testing" consists of:

Manual Test Cases:

The script includes 3 test cases (e.g., student1, student2, student3) to check if the rules work as expected.

These simulate real-world inputs and verify if the recommendations match the hardcoded logic.

Output Validation:

The system prints recommended housing IDs (or fallback options) for each test case.

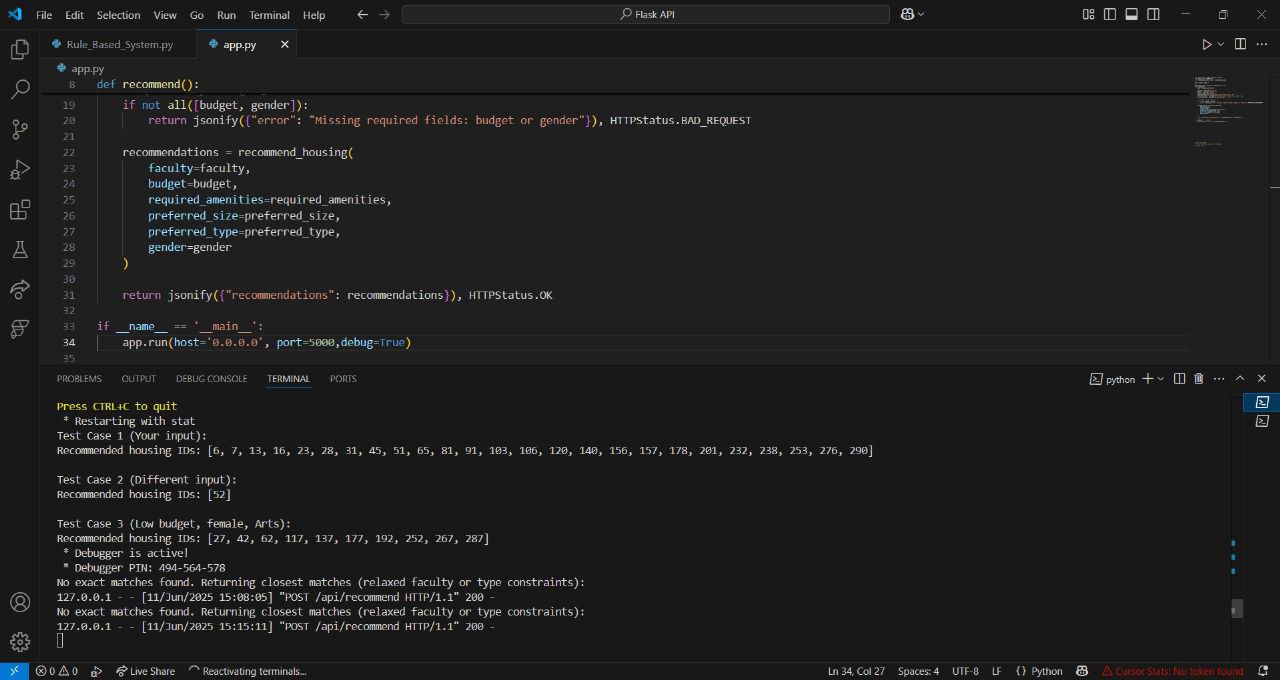
A human reviews these to ensure the rules behave correctly (e.g., "Does a low-budget Arts student get affordable options?").

Rejection Reason Logging:

If no matches are found, the system explains why (e.g., "Price out of range"). This acts as a debug tool.

Inside FlutterFlow, we navigate to the API Call section, insert the Ngrok URL as the API Endpoint, set the method to POST, and provide a JSON body with sample data to test the API .

Upon clicking 'Test API Call', the system immediately returns the result — which is the accommodation ID based on the input data.



## 

## 5.1.2 Integration Testing:

Integration tests validated the interactions between different modules, especially the communication between the Flutter app and Firebase services (Firestore, Authentication, Cloud Functions). These tests confirmed that data synchronization, real-time messaging, and payment processing workflows operated seamlessly across system boundaries.

## 5.1.3 User Testing (Beta Testing):

A group of representative users—students and landlords from Upper Egypt—participated in controlled beta testing phases. Their feedback was gathered through surveys and direct interviews to assess usability, performance, and feature relevance. This real-world testing was critical for identifying UX issues, localization needs, and trust-related concerns unique to the regional context.

### **5.1.3.1 Firebase Test Lab:**

The application was also tested on a wide range of Android and iOS devices hosted in Google data centers using Firebase Test Lab. This cloud-based infrastructure allowed simulation of diverse device configurations and operating system versions, uncovering device-specific bugs and ensuring consistent performance across the user base.

### 5.1.3.2 A/B Testing:

Firebase A/B Testing was employed to experiment with UI variations and feature toggles, enabling data-driven decisions on interface design and functionality optimization. Metrics such as session duration and conversion rates guided iterative improvements.

# 5.2 Performance Metrics

The evaluation of E-Rent’s performance focused on several key dimensions:

Accuracy:

The recommendation system’s effectiveness was measured by the precision of suggested properties that matched user preferences, tracked via booking conversion rates and user satisfaction surveys. The Rule-based system algorithm demonstrated improved accuracy over baseline content-based methods, particularly as user interaction data accumulated.

Speed:

Application responsiveness was assessed through load times for property searches, real-time messaging latency, and booking confirmation delays. Firebase’s real-time database and Cloud Functions contributed to sub-second update propagation, ensuring a fluid user experience even under moderate network constraints typical in Upper Egypt.

Scalability:

The platform’s ability to handle increasing numbers of concurrent users and data volume was tested through load testing simulations. Firebase’s serverless infrastructure automatically scaled resources, maintaining stable performance during peak usage without manual intervention.

Reliability:

System uptime and error rates were monitored. The use of Firebase’s managed services and automated testing pipelines minimized downtime and reduced critical failures.

User Engagement:

Metrics such as session duration, frequency of app usage, and feature adoption rates were analyzed to gauge user retention and satisfaction. A/B testing facilitated continuous refinement to maximize engagement.

# 5.3 Comparison with Existing Solutions

Compared to generic rental platforms operating in Egypt, E-Rent offers several distinct advantages:

Localization and Focus:

Unlike broad real estate websites, E-Rent is tailored specifically for student housing in Upper Egypt, addressing regional socio-economic and cultural factors.

Integrated Recommendation System:

The use of a Contextual Multi-Armed Bandit algorithm with Rule-based system provides personalized, adaptive housing suggestions, outperforming static or collaborative filtering methods used by competitors.

Real-Time Features:

Leveraging Firebase’s real-time database and messaging services, E-Rent delivers instant updates and communication, enhancing user trust and convenience.

Secure Payment Integration:

Local payment gateways embedded via Firebase Cloud Functions ensure compliance with Egyptian regulations and user preferences, a feature often lacking in other platforms.

Comprehensive Testing and Optimization:

The adoption of Firebase Test Lab and A/B Testing enables continuous quality assurance and user experience optimization, positioning E-Rent as a technically robust and user-centric solution.

While platforms like Dubizzle and Bayut serve the broader Egyptian rental market, their lack of student-specific features and real-time interactivity limits their effectiveness for the target demographic. E-Rent’s focused approach and technological foundation provide a competitive edge in meeting the needs of students and landlords in Upper Egypt.

Chapter 6

Results & Discussion

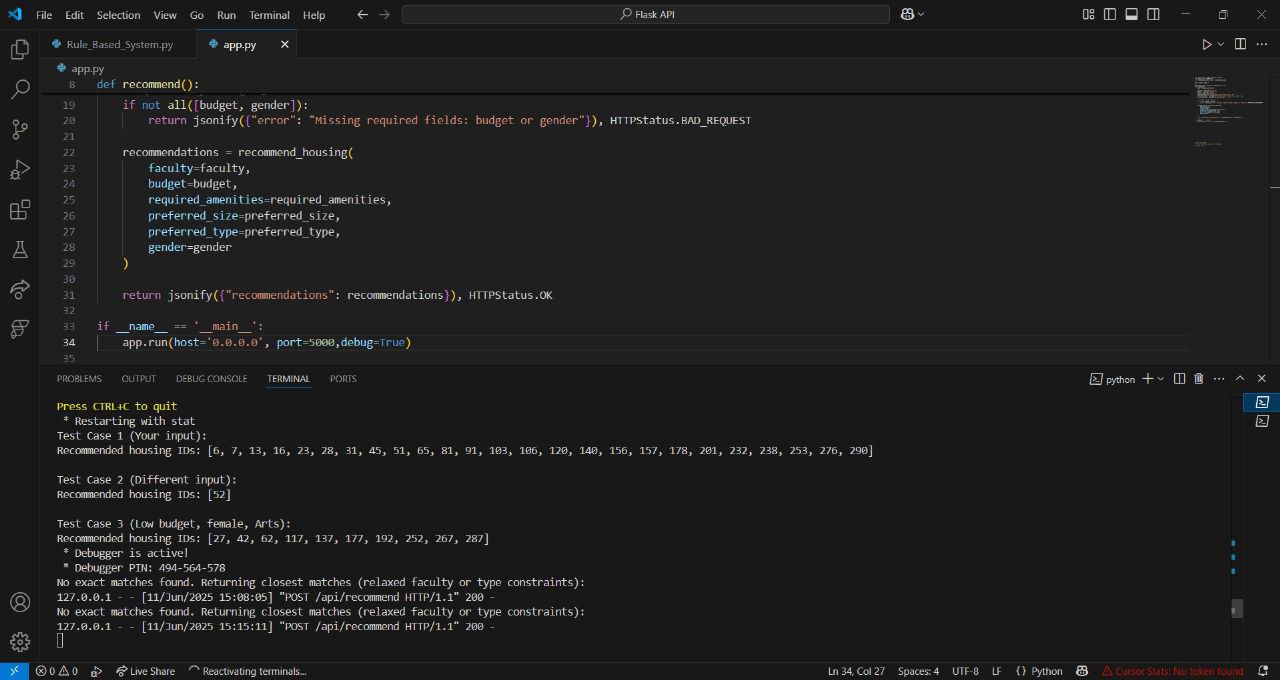
# 6.1 Introduction

This chapter presents a comprehensive analysis of the outcomes derived from the development and deployment of the E-Rent platform; a student housing solution tailored for Upper Egypt. The discussion focuses on evaluating the extent to which the project objectives were achieved, supported by empirical data collected through testing and user feedback. Additionally, the chapter identifies limitations inherent in the proposed system and suggests areas for future improvement.

After performing the test from FlutterFlow, it was confirmed that the integration between FlutterFlow and the API is functioning correctly. The ID of the suitable accommodation based on the input data.

Additionally, by monitoring the terminal where the API runs, we could see that the data was received successfully, the recommendation result was printed, and the number of test executions was logged.

This confirms that the API is running without errors and produces the correct output.



# 6.2 Summary of Findings

The E-Rent platform successfully integrated advanced technological components, including a Contextual Multi-Armed Bandit recommendation system powered by Rule-based system, real-time data synchronization via Firebase, and secure payment processing with local gateways. User testing demonstrated high levels of satisfaction regarding usability, personalized recommendations, and communication features.

Key findings include:

* The recommendation engine increased booking conversion rates by providing context-aware, personalized housing suggestions, effectively addressing diverse student preferences in Upper Egypt.
* Real-time messaging and instant updates enhanced trust and transparency between students and landlords.
* Secure payment integration reduced transaction-related concerns, encouraging wider adoption.

The platform’s modular architecture and Firebase backend ensured scalability and stable performance under varying user loads.

# 6.3 Interpretation of Results

The project met its primary objectives by delivering a comprehensive, user-centric platform that addresses the unique challenges of student housing in Upper Egypt. The AI-driven recommendation system proved effective in adapting to user preferences over time, enhancing the housing search experience. The integration of secure communication and payment modules further contributed to building user trust, a critical factor in the regional context.

Moreover, the use of Firebase’s serverless infrastructure facilitated rapid development and deployment while ensuring scalability and reliability, aligning with the project's goal of providing a sustainable and accessible solution.

# 6.4 Limitations of the Proposed Solution

Despite the notable achievements of the E-Rent platform, several practical and regulatory limitations affect its full potential:

Data Dependency: The recommendation system requires a substantial amount of user interaction data to optimize accuracy. During initial deployment phases, limited data availability may reduce recommendation effectiveness.

Regional Internet Infrastructure: Variability and occasional instability of internet connectivity in Upper Egypt can impact the reliability of real-time features such as messaging and payment processing.

Verification and Onboarding Delays: The multi-step verification process, essential for building trust, may introduce delays in onboarding landlords and students, potentially affecting user acquisition rates.

Payment Gateway Integration Challenges: Integrating additional payment methods requires formal approval processes with banks and payment service providers like Fawry. This involves submitting official requests and undergoing verification and compliance checks, which can delay the addition of new payment options and affect user convenience.

Electronic Contract Limitations: The digital lease agreements currently used lack full legal enforceability without formal notarization. To achieve official recognition, contracts must be obtained and certified through the Egyptian Real Estate Publicity Department, adding complexity and potential delays to the rental process.

Scope of Smart Home Features: Planned integrations with smart home technologies, such as automated access control systems, remain unimplemented, limiting the platform’s ability to offer end-to-end automation.

Language and Cultural Nuances: While the application is localized for Upper Egypt, further refinement is necessary to fully accommodate the region’s diverse dialects and cultural preferences, enhancing user experience and accessibility.

Addressing these limitations will require ongoing collaboration with regulatory bodies, infrastructure improvements, and iterative technological enhancements to fully realize the platform’s vision.

Chapter 7

Conclusion & Future Work

# 7.1 Summary of Contributions

The E-Rent platform represents a significant advancement in addressing the challenges of student housing in Upper Egypt. By integrating state-of-the-art technologies such as the Contextual Multi-Armed Bandit recommendation system powered by Rule-based system, real-time data synchronization through Firebase, and secure payment processing with local Egyptian gateways, the platform offers a comprehensive, user-centric solution tailored to the unique socio-economic and cultural context of the region.

Key contributions include:

Development of a personalized, adaptive recommendation engine that improves housing match accuracy and user satisfaction.

Implementation of a modular system architecture that ensures scalability, security, and seamless user experience.

Integration of secure communication and payment modules that build trust between students and landlords.

Deployment of a mobile application accessible nationwide, with a focus on the specific needs of Upper Egypt’s student population.

Rigorous testing and evaluation, including real-user feedback, ensuring the platform’s robustness and relevance.

These contributions collectively enhance the efficiency, transparency, and accessibility of student housing, supporting academic success and social well-being in Upper Egypt.

# 7.2 Possible Improvements or Extensions for Future Work

While the E-Rent platform has demonstrated strong performance and user acceptance, several avenues exist for further enhancement:

Security Enhancements:

Future work will focus on strengthening the platform’s security framework, including advanced fraud detection mechanisms. One priority is developing AI-powered image verification tools to authenticate uploaded property photos, ensuring that listings are genuine and reducing the risk of fraudulent or misleading advertisements.

AI Model Development:

Continued refinement of the recommendation system is planned, incorporating more sophisticated machine learning techniques to better capture evolving user preferences and contextual factors. This includes integrating additional data sources and feedback loops to improve recommendation accuracy and responsiveness.

Expanded Payment Options:

The platform aims to broaden its payment gateway integrations to include the full spectrum of payment methods available in Egypt, such as mobile wallets, bank transfers, and emerging fintech solutions. This expansion will enhance user convenience and accommodate diverse financial preferences.

Electronic Contract Formalization:

Developing a legally recognized electronic contract system is a key future goal. This involves collaboration with Egyptian legal authorities to enable digital lease agreements that are officially notarized and enforceable, streamlining the rental process and reducing paperwork.

Smart Home Integration:

Exploring integration with smart home technologies, such as automated access control and IoT devices, to provide enhanced security and convenience for tenants and landlords.

Localization and Accessibility:

Further cultural and linguistic customization will be pursued to better serve the diverse dialects and user needs within Upper Egypt, improving overall user engagement and satisfaction.

These planned improvements will build on the platform’s current strengths, ensuring that E-Rent remains a leading, innovative solution for student housing in Egypt.

With the increasing digital transformation in Egypt, E-Rent aspires to evolve into a national platform endorsed by educational institutions and housing authorities, becoming the default solution for student accommodation across the country.

References

1. H. Yao, K. Lin, Ioan Ungurean, and Y. Yang, “Constrained Relay Node Deployment in Wireless Sensor Network for Structural Health Monitoring,” Aug. 2019, doi: <https://doi.org/10.1109/issi47111.2019.9043653>.
2. “Huawei’s sales revenue grows by 3.8% amid coronavirus - Economy - Business,” *Ahram Online*, 2025. <https://english.ahram.org.eg/NewsContent/50/1208/408234/> AlAhram-Weekly/Egypt/Student-housing-problems.aspx (accessed Jun. 09, 2025).
3. O. Mohamed, Abdelwahed, F. Amr, and Hanafy, “Overview of Social Housing Provision in Egypt 1,” *Journal of Alexandria University for Administrative Sciences©*, vol. 57, no. 2, 2020, Accessed: Jun. 09, 2025. [Online]. Available: <https://acjalexu.journals.ekb.eg/article_93579_fca786981d92ee184faaa309f8b7752e.pdf>
4. “Women’s Housing and the Urban Experience in Egypt.” Available: <https://egypt.fes.de/fileadmin/user_upload/documents/publication/Women_s_Housing_and_the_Urban_Experience_in_Egypt.pdf>
5. “StudentsAbroad.com - Study Abroad Handbook Egypt: Housing Arrangements,” *Studentsabroad.com*, 2025. <https://www.studentsabroad.com/handbook/housing.php?country=Egypt> (accessed Jun. 09, 2025).
6. P. Darwish, “Upper Egypt: ‘Back To The Future’ | INVEST-GATE,” *INVEST-GATE*, Nov. 06, 2017. <https://invest-gate.me/features/upper-egypt-back-to-the-future/> (accessed Jun. 09, 2025).
   1. Saber, “MUST Student Housing - Misr University for Science and Technology - A Leading University in Egypt.,” *Misr University for Science and Technology - A Leading University in Egypt. - Empowering Futures Through Excellence in Higher Education.*, 2022. <https://must.edu.eg/must-student-housing> (accessed Jun. 09, 2025).
7. ‌
8. Pranav P. More, Shreyash S. Khairnar, Harshal G. Khairnar, Vaibhav V. Deore, and Rikeeta C. Mahajan, “Campus Heaven: AI-Powered Student Accommodation and Services Hub,” *International Journal of Advanced Research in Science, Communication and Technology*, pp. 55–58, Feb. 2025, doi: <https://doi.org/10.48175/ijarsct-23208>.
9. “Reddit - The heart of the internet,” *Reddit.com*, 2019. <https://www.reddit.com/r/Firebase/comments/c1cc66/is_it_realistic_to_create_an_application_which/> (accessed Jun. 09, 2025).
10. ‌
11. “Using AI for Student Accommodation Development – Fyma – Unlock Data-Driven Decisions from Every Camera,” *Fyma.ai*, Feb. 27, 2025. <https://fyma.ai/using-ai-for-student-accommodation-development/> (accessed Jun. 09, 2025).
12. S. Kesavan, “How to build an effective Mobile Testing Strategy | BrowserStack,” *BrowserStack*, Nov. 14, 2024. <https://www.browserstack.com/guide/mobile-testing-strategies>
13. “A Comprehensive Survey of Evaluation Techniques for Recommendation Systems,” *arxiv.org*. <https://arxiv.org/html/2312.16015v2>
14. admin, “Technology trends that are shaping the future of Student Housing,” *Smiota*, Sep. 27, 2024. <https://smiota.com/resources/the-top-technology-trends-shaping-the-future-of-student-housing/> (accessed Jun. 09, 2025).
15. ‌
16. “Best Student Housing Property Management Software - Bidrento,” *Bidrento*, May 23, 2025. <https://bidrento.com/student-housing-software/> (accessed Jun. 09, 2025).
17. Amber, “Tech Usage in Student Housing Property Management,” *Amberstudent.com*, Dec. 20, 2023. <https://amberstudent.com/news/post/tech-usage-in-student-housing-property-management> (accessed Jun. 09, 2025).
18. ‌
19. S. Wang and H. Wang, “Digital Transformation in Real Estate Services: Development and Implementation of the Housing Selection Platform,” *Frontiers in computing and intelligent systems*, vol. 8, no. 1, pp. 13–18, May 2024, doi: <https://doi.org/10.54097/yyw4jr63>.
20. “Student Housing - Rently,” *Rently*, Sep. 05, 2024. <https://use.rently.com/student-housing/> (accessed Jun. 09, 2025).
21. M. Swiderek, “7 Technology Trends to Transform Student Housing,” *WorldVue*, Sep. 19, 2023. <https://www.worldvue.com/blog/7-technology-trends-to-transform-student-housing/>
22. Cochrane Training, “Chapter 4: Searching for and selecting studies,” *Cochrane.org*, 2024. <https://training.cochrane.org/handbook/current/chapter-04>
23. *Govt.nz*, 2024. <https://nzism.gcsb.govt.nz/ism-document>
24. WHO, “WHO guideline recommendations on digital interventions for health system strengthening,” 2019. Available: <https://iris.who.int/bitstream/handle/10665/311941/9789241550505-eng.pdf>
25. “عرض محتوى ديناميكي واستضافة خدمات مصغّرة باستخدام ‘استضافة Firebase’  |  Firebase Hosting,” *Firebase*, 2025. <https://firebase.google.com/docs/hosting/serverless-overview?hl=ar> (accessed Jun. 09, 2025).
26. GeeksforGeeks, “What is Firebase A/B Testing and How to Configure It,” *GeeksforGeeks*, May 31, 2024. <https://www.geeksforgeeks.org/what-is-firebase-a-b-testing-and-how-to-configure-it/> (accessed Jun. 09, 2025).
27. ‌
28. GeeksforGeeks, “A/B Testing with Firebase Remote Config,” *GeeksforGeeks*, May 31, 2024. <https://www.geeksforgeeks.org/a-b-testing-with-firebase-remote-config/> (accessed Jun. 09, 2025).
29. ‌
30. “How to Test Apps Installed via Firebase? | LambdaTest,” *Lambdatest.com*, Mar. 27, 2025. <https://www.lambdatest.com/support/docs/install-apps-from-firebase/> (accessed Jun. 09, 2025).
31. ‌
32. “Introducing the AI-Powered App Testing Agent,” *The Firebase Blog*, 2025. <https://firebase.blog/posts/2025/04/app-testing-agent/> (accessed Jun. 09, 2025).
33. “Unit Testing vs. Integration Testing: 5 Key Differences and Why You Need Both,” *Codefresh*. <https://codefresh.io/learn/unit-testing/unit-testing-vs-integration-testing-5-key-differences-and-why-you-need-both/>
34. “CHAPTER 5 - Testing and Evaluation Flashcards,” *Quizlet*, 2025. <https://quizlet.com/735809530/chapter-5-testing-and-evaluation-flash-cards/> (accessed Jun. 09, 2025).
35. ‌
36. Jaars, H. Abdel-Rahman, Y. Elsayed, and D. Abouelmagd, *helwan.edu.eg -JAARS*, vol. 1, no. 1, 2020, Accessed: Jun. 09, 2025. [Online]. Available: <https://jaars.journals.ekb.eg/article_153759_be65da7199213cc24a08b4839b4de62a.pdf>
37. “5 Challenges and Tips for Mobile App Testing,” *TestDevLab Blog*, 2024. <https://www.testdevlab.com/blog/5-challenges-and-tips-for-mobile-app-testing>
38. Testlio, “The Ultimate Guide to Mobile App Testing,” *Testlio*, Jan. 17, 2025. <https://testlio.com/blog/mobile-application-testing/> (accessed Jun. 09, 2025).
39. ‌
40. ‌
    1. Richardson, “The affordable housing industry is at a critical crossroads. Rising construction costs, labor shortages, complex regulatory requirements, and limited financial resources have created a challenging environment for developers, housing authorities, and nonprofit organizations.,” *Linkedin.com*, May 13, 2025. <https://www.linkedin.com/pulse/unlocking-future-affordable-housing-how-ai-compliance-austin-s7tvc?utm_source=share&utm_medium=member_ios&utm_campaign=share_via> (accessed Jun. 09, 2025).
41. ‌
42. “EGYPT.” Available: <https://unhabitat.org/sites/default/files/download-manager-files/1525977522wpdm_Egypt%20housing%20EN_HighQ_23-1-2018.pdf>
43. Y. Abdel Fattah Salem, M. Abdel-Fattah Zohry, and A. El-Sayed Abdel-Maboud, “Studying the Housing Needs of the International Students in Egyptian Universities,” *مجلة کلیة السیاحة والفنادق. جامعة المنصورة*, vol. 12, no. 12, pp. 369–412, Dec. 2022, doi: <https://doi.org/10.21608/mkaf.2022.276095>.
44. “Rule-Based System - an overview | ScienceDirect Topics,” *Sciencedirect.com*, 2010. <https://www.sciencedirect.com/topics/engineering/rule-based-system>
45. “Rule-based recommendations,” *Metabase | Business Intelligence, Dashboards, and Data Visualization*, 2024. <https://www.metabase.com/community-posts/rule-based-recommendations>
46. “Rule-Based System,” *Engati*. <https://www.engati.com/glossary/rule-based-system>
47. “How to choose between a rules-based vs. machine learning system,” *SearchEnterpriseAI*. <https://www.techtarget.com/searchenterpriseai/feature/How-to-choose-between-a-rules-based-vs-machine-learning-system>
48. ‌“Rule-Based System: Pros and Cons | BotPenguin,” *botpenguin.com*. <https://botpenguin.com/glossary/rule-based-system>
49. ‌
50. “RULE-BASED SYSTEMS.” Available: <https://www.cs.oswego.edu/~mgrzenda/CSC466/Paper%20Sources/RULE-BASED%20SYSTEMS.pdf>
51. Kumar Gandharv, “Recommendation Systems are Moving Beyond Rules-based Approaches,” *Datatechvibe*, Mar. 16, 2023. <https://datatechvibe.com/data/recommendation-systems-are-moving-beyond-rules-based-approaches/> (accessed Jun. 09, 2025).
52. Secoda, “Data Teams,” *Secoda.co*, Sep. 2024, doi: <https://doi.org/1064107/NlFjCMHvzIwYEKuIxs0B>.

‌

‌

‌

‌

Appendices (Optional)

**1-getUniqeList**

import 'dart:convert';

import 'dart:math' as math;

import 'package:flutter/material.dart';

import 'package:google\_fonts/google\_fonts.dart';

import 'package:intl/intl.dart';

import 'package:timeago/timeago.dart' as timeago;

import '/flutter\_flow/custom\_functions.dart';

import '/flutter\_flow/lat\_lng.dart';

import '/flutter\_flow/place.dart';

import '/flutter\_flow/uploaded\_file.dart';

import '/backend/backend.dart';

import 'package:cloud\_firestore/cloud\_firestore.dart';

import '/auth/firebase\_auth/auth\_util.dart';

List<String> getUniqueList(List<String> list) {

  /// MODIFY CODE ONLY BELOW THIS LINE

  return list.toSet().toList();

  /// MODIFY CODE ONLY ABOVE THIS LINE

}

)

**2-getPlaces**

import 'dart:convert';

import 'dart:math' as math;

import 'package:flutter/material.dart';

import 'package:google\_fonts/google\_fonts.dart';

import 'package:intl/intl.dart';

import 'package:timeago/timeago.dart' as timeago;

import '/flutter\_flow/custom\_functions.dart';

import '/flutter\_flow/lat\_lng.dart';

import '/flutter\_flow/place.dart';

import '/flutter\_flow/uploaded\_file.dart';

import '/backend/backend.dart';

import 'package:cloud\_firestore/cloud\_firestore.dart';

import '/auth/firebase\_auth/auth\_util.dart';

List<PlacesRecord> getPlaces(

  List<PlacesRecord> list,

  List<PlacesRecord> searchResults,

  bool searchBool,

) {

  /// MODIFY CODE ONLY BELOW THIS LINE

  return searchBool ? searchResults : list;

  /// MODIFY CODE ONLY ABOVE THIS LINE

}

**3-calculateBasePrice**

import 'dart:convert';

import 'dart:math' as math;

import 'package:flutter/material.dart';

import 'package:google\_fonts/google\_fonts.dart';

import 'package:intl/intl.dart';

import 'package:timeago/timeago.dart' as timeago;

import '/flutter\_flow/custom\_functions.dart';

import '/flutter\_flow/lat\_lng.dart';

import '/flutter\_flow/place.dart';

import '/flutter\_flow/uploaded\_file.dart';

import '/backend/backend.dart';

import 'package:cloud\_firestore/cloud\_firestore.dart';

import '/auth/firebase\_auth/auth\_util.dart';

int calculateBasePrice(

  int pricePerNight,

  DateTime startDate,

  DateTime endDate,

) {

  /// MODIFY CODE ONLY BELOW THIS LINE

  return endDate.difference(startDate).inDays \* pricePerNight;

  /// MODIFY CODE ONLY ABOVE THIS LINE

}

**4-getBookingList**

import 'dart:convert';

import 'dart:math' as math;

import 'package:flutter/material.dart';

import 'package:google\_fonts/google\_fonts.dart';

import 'package:intl/intl.dart';

import 'package:timeago/timeago.dart' as timeago;

import '/flutter\_flow/custom\_functions.dart';

import '/flutter\_flow/lat\_lng.dart';

import '/flutter\_flow/place.dart';

import '/flutter\_flow/uploaded\_file.dart';

import '/backend/backend.dart';

import 'package:cloud\_firestore/cloud\_firestore.dart';

import '/auth/firebase\_auth/auth\_util.dart';

List<String> getBookingList(

  List<BookingsRecord> bookingList,

  DateTime startDate,

  DateTime endDate,

) {

  /// MODIFY CODE ONLY BELOW THIS LINE

  List<String> bookings = [];

  for (var booking in bookingList) {

    if ((startDate.isBefore(booking.endDate!) ||

            startDate.isAtSameMomentAs(booking.endDate!)) &&

        (endDate.isAfter(booking.startDate!) ||

            endDate.isAtSameMomentAs(booking.startDate!))) {

      bookings.add(booking.placeStringRef);

    }

  }

  return bookings;

  /// MODIFY CODE ONLY ABOVE THIS LINE

}

**5-add7Days**

import 'dart:convert';

import 'dart:math' as math;

import 'package:flutter/material.dart';

import 'package:google\_fonts/google\_fonts.dart';

import 'package:intl/intl.dart';

import 'package:timeago/timeago.dart' as timeago;

import '/flutter\_flow/custom\_functions.dart';

import '/flutter\_flow/lat\_lng.dart';

import '/flutter\_flow/place.dart';

import '/flutter\_flow/uploaded\_file.dart';

import '/backend/backend.dart';

import 'package:cloud\_firestore/cloud\_firestore.dart';

import '/auth/firebase\_auth/auth\_util.dart';

DateTime? add7Days(DateTime? date1) {

  /// MODIFY CODE ONLY BELOW THIS LINE

  return date1?.add(Duration(days: 7));

  /// MODIFY CODE ONLY ABOVE THIS LINE

}

**AI Implememtation**

# Simulated housing database (expanded to 75 options)

housing\_options = [

{"id": 1, "faculty\_cluster": "Medical", "price": 2000, "amenities": ["wifi", "study\_desk"], "size": {"rooms": 2, "beds": 2}, "type": "shared\_room", "gender": "male"},

{"id": 2, "faculty\_cluster": "Engineering", "price": 3500, "amenities": ["wifi", "ac", "study\_desk"], "size": {"rooms": 1, "beds": 1}, "type": "apartment", "gender": "female"},

{"id": 3, "faculty\_cluster": "Medical", "price": 1500, "amenities": ["study\_desk"], "size": {"rooms": 3, "beds": 4}, "type": "dorm", "gender": "male"},

{"id": 4, "faculty\_cluster": "Business", "price": 2500, "amenities": ["wifi", "ac"], "size": {"rooms": 2, "beds": 2}, "type": "shared\_room", "gender": "female"},

{"id": 5, "faculty\_cluster": "Engineering", "price": 1800, "amenities": ["wifi"], "size": {"rooms": 1, "beds": 2}, "type": "shared\_room", "gender": "male"},

{"id": 6, "faculty\_cluster": "IT", "price": 1500, "amenities": ["wifi", "study\_desk"], "size": {"rooms": 1, "beds": 1}, "type": "shared\_room", "gender": "male"},

{"id": 7, "faculty\_cluster": "IT", "price": 1600, "amenities": ["wifi", "ac"], "size": {"rooms": 1, "beds": 1}, "type": "apartment", "gender": "male"},

{"id": 8, "faculty\_cluster": "Sciences", "price": 1400, "amenities": ["wifi", "study\_desk"], "size": {"rooms": 2, "beds": 3}, "type": "dorm", "gender": "female"},

{"id": 9, "faculty\_cluster": "Engineering", "price": 2200, "amenities": ["wifi", "ac", "study\_desk"], "size": {"rooms": 2, "beds": 2}, "type": "shared\_room", "gender": "male"},

{"id": 10, "faculty\_cluster": "Medical", "price": 1700, "amenities": ["wifi"], "size": {"rooms": 1, "beds": 1}, "type": "shared\_room", "gender": "female"},

{"id": 11, "faculty\_cluster": "Arts", "price": 1300, "amenities": ["wifi"], "size": {"rooms": 3, "beds": 4}, "type": "dorm", "gender": "male"},

{"id": 12, "faculty\_cluster": "Business", "price": 3000, "amenities": ["wifi", "ac", "study\_desk", "kitchen"], "size": {"rooms": 2, "beds": 2}, "type": "apartment", "gender": "female"},

{"id": 13, "faculty\_cluster": "IT", "price": 1450, "amenities": ["wifi", "study\_desk"], "size": {"rooms": 1, "beds": 2}, "type": "shared\_room", "gender": "male"},

{"id": 14, "faculty\_cluster": "Law", "price": 2000, "amenities": ["wifi", "ac"], "size": {"rooms": 1, "beds": 1}, "type": "apartment", "gender": "female"},

{"id": 15, "faculty\_cluster": "Sciences", "price": 1600, "amenities": ["wifi", "study\_desk"], "size": {"rooms": 2, "beds": 2}, "type": "dorm", "gender": "male"},

{"id": 16, "faculty\_cluster": "IT", "price": 1200, "amenities": ["wifi"], "size": {"rooms": 1, "beds": 1}, "type": "shared\_room", "gender": "male"},

{"id": 17, "faculty\_cluster": "Medical", "price": 2500, "amenities": ["wifi", "ac", "study\_desk"], "size": {"rooms": 2, "beds": 2}, "type": "apartment", "gender": "female"},

{"id": 18, "faculty\_cluster": "Engineering", "price": 1700, "amenities": ["wifi", "study\_desk"], "size": {"rooms": 1, "beds": 2}, "type": "shared\_room", "gender": "male"},

{"id": 19, "faculty\_cluster": "Business", "price": 1800, "amenities": ["wifi", "kitchen"], "size": {"rooms": 2, "beds": 3}, "type": "dorm", "gender": "female"},

{"id": 20, "faculty\_cluster": "Arts", "price": 1400, "amenities": ["wifi", "study\_desk"], "size": {"rooms": 2, "beds": 2}, "type": "shared\_room", "gender": "male"},

{"id": 21, "faculty\_cluster": "Law", "price": 2200, "amenities": ["wifi", "ac"], "size": {"rooms": 1, "beds": 1}, "type": "apartment", "gender": "male"},

{"id": 22, "faculty\_cluster": "Sciences", "price": 1500, "amenities": ["wifi"], "size": {"rooms": 1, "beds": 1}, "type": "shared\_room", "gender": "female"},

{"id": 23, "faculty\_cluster": "IT", "price": 1750, "amenities": ["wifi", "ac", "study\_desk"], "size": {"rooms": 2, "beds": 2}, "type": "shared\_room", "gender": "male"},

{"id": 24, "faculty\_cluster": "Medical", "price": 1900, "amenities": ["wifi", "study\_desk"], "size": {"rooms": 1, "beds": 2}, "type": "dorm", "gender": "male"},

{"id": 25, "faculty\_cluster": "Engineering", "price": 4000, "amenities": ["wifi", "ac", "study\_desk", "kitchen"], "size": {"rooms": 3, "beds": 3}, "type": "apartment", "gender": "female"},

{"id": 26, "faculty\_cluster": "Business", "price": 1600, "amenities": ["wifi", "study\_desk"], "size": {"rooms": 1, "beds": 1}, "type": "shared\_room", "gender": "male"},

{"id": 27, "faculty\_cluster": "Arts", "price": 1100, "amenities": ["wifi"], "size": {"rooms": 2, "beds": 3}, "type": "dorm", "gender": "female"},

{"id": 28, "faculty\_cluster": "IT", "price": 1550, "amenities": ["wifi", "ac"], "size": {"rooms": 1, "beds": 1}, "type": "shared\_room", "gender": "male"},

{"id": 29, "faculty\_cluster": "Law", "price": 1800, "amenities": ["wifi", "study\_desk"], "size": {"rooms": 1, "beds": 2}, "type": "shared\_room", "gender": "female"},

{"id": 30, "faculty\_cluster": "Sciences", "price": 1450, "amenities": ["wifi", "study\_desk"], "size": {"rooms": 2, "beds": 2}, "type": "dorm", "gender": "male"},

{"id": 31, "faculty\_cluster": "IT", "price": 1300, "amenities": ["wifi", "study\_desk"], "size": {"rooms": 1, "beds": 1}, "type": "shared\_room", "gender": "male"},

{"id": 32, "faculty\_cluster": "Medical", "price": 2100, "amenities": ["wifi", "ac", "kitchen"], "size": {"rooms": 2, "beds": 2}, "type": "apartment", "gender": "female"},

{"id": 33, "faculty\_cluster": "Engineering", "price": 1600, "amenities": ["wifi", "study\_desk"], "size": {"rooms": 1, "beds": 2}, "type": "shared\_room", "gender": "male"},

{"id": 34, "faculty\_cluster": "Business", "price": 2700, "amenities": ["wifi", "ac", "study\_desk"], "size": {"rooms": 2, "beds": 2}, "type": "apartment", "gender": "female"},

{"id": 35, "faculty\_cluster": "Arts", "price": 1200, "amenities": ["wifi"], "size": {"rooms": 3, "beds": 4}, "type": "dorm", "gender": "male"},

{"id": 36, "faculty\_cluster": "Law", "price": 1900, "amenities": ["wifi", "study\_desk"], "size": {"rooms": 1, "beds": 1}, "type": "shared\_room", "gender": "female"},

{"id": 37, "faculty\_cluster": "Sciences", "price": 1700, "amenities": ["wifi", "ac"], "size": {"rooms": 2, "beds": 2}, "type": "dorm", "gender": "male"},

{"id": 38, "faculty\_cluster": "IT", "price": 1400, "amenities": ["wifi", "study\_desk"], "size": {"rooms": 1, "beds": 1}, "type": "shared\_room", "gender": "female"},

{"id": 39, "faculty\_cluster": "Medical", "price": 2300, "amenities": ["wifi", "ac", "study\_desk"], "size": {"rooms": 2, "beds": 3}, "type": "apartment", "gender": "male"},

{"id": 40, "faculty\_cluster": "Engineering", "price": 1500, "amenities": ["wifi"], "size": {"rooms": 1, "beds": 1}, "type": "shared\_room", "gender": "female"},

{"id": 41, "faculty\_cluster": "Business", "price": 2000, "amenities": ["wifi", "study\_desk"], "size": {"rooms": 2, "beds": 2}, "type": "shared\_room", "gender": "male"},

{"id": 42, "faculty\_cluster": "Arts", "price": 1000, "amenities": ["wifi"], "size": {"rooms": 2, "beds": 3}, "type": "dorm", "gender": "female"},

{"id": 43, "faculty\_cluster": "Law", "price": 2500, "amenities": ["wifi", "ac", "study\_desk"], "size": {"rooms": 1, "beds": 1}, "type": "apartment", "gender": "male"},

{"id": 44, "faculty\_cluster": "Sciences", "price": 1300, "amenities": ["wifi", "study\_desk"], "size": {"rooms": 1, "beds": 2}, "type": "shared\_room", "gender": "female"},

{"id": 45, "faculty\_cluster": "IT", "price": 1650, "amenities": ["wifi", "ac"], "size": {"rooms": 1, "beds": 1}, "type": "shared\_room", "gender": "male"},

{"id": 46, "faculty\_cluster": "Medical", "price": 1800, "amenities": ["wifi", "study\_desk"], "size": {"rooms": 2, "beds": 2}, "type": "dorm", "gender": "female"},

{"id": 47, "faculty\_cluster": "Engineering", "price": 1900, "amenities": ["wifi", "ac"], "size": {"rooms": 1, "beds": 2}, "type": "shared\_room", "gender": "male"},

{"id": 48, "faculty\_cluster": "Business", "price": 2200, "amenities": ["wifi", "study\_desk", "kitchen"], "size": {"rooms": 2, "beds": 2}, "type": "apartment", "gender": "female"},

{"id": 49, "faculty\_cluster": "Arts", "price": 1500, "amenities": ["wifi"], "size": {"rooms": 1, "beds": 1}, "type": "shared\_room", "gender": "male"},

{"id": 50, "faculty\_cluster": "Law", "price": 1700, "amenities": ["wifi", "study\_desk"], "size": {"rooms": 1, "beds": 1}, "type": "shared\_room", "gender": "female"},

{"id": 51, "faculty\_cluster": "IT", "price": 1250, "amenities": ["wifi"], "size": {"rooms": 1, "beds": 1}, "type": "shared\_room", "gender": "male"},

{"id": 52, "faculty\_cluster": "Medical", "price": 1600, "amenities": ["wifi", "study\_desk"], "size": {"rooms": 1, "beds": 1}, "type": "shared\_room", "gender": "female"},

{"id": 53, "faculty\_cluster": "Engineering", "price": 1400, "amenities": ["wifi"], "size": {"rooms": 2, "beds": 2}, "type": "dorm", "gender": "male"},

{"id": 54, "faculty\_cluster": "Business", "price": 1900, "amenities": ["wifi", "ac"], "size": {"rooms": 1, "beds": 1}, "type": "apartment", "gender": "female"},

{"id": 55, "faculty\_cluster": "Arts", "price": 1100, "amenities": ["wifi", "study\_desk"], "size": {"rooms": 2, "beds": 3}, "type": "dorm", "gender": "male"},

{"id": 56, "faculty\_cluster": "Law", "price": 2000, "amenities": ["wifi", "ac"], "size": {"rooms": 1, "beds": 1}, "type": "shared\_room", "gender": "female"},

{"id": 57, "faculty\_cluster": "Sciences", "price": 1350, "amenities": ["wifi"], "size": {"rooms": 1, "beds": 1}, "type": "shared\_room", "gender": "male"},

{"id": 58, "faculty\_cluster": "IT", "price": 1700, "amenities": ["wifi", "study\_desk"], "size": {"rooms": 1, "beds": 2}, "type": "shared\_room", "gender": "female"},

{"id": 59, "faculty\_cluster": "Medical", "price": 1450, "amenities": ["wifi"], "size": {"rooms": 1, "beds": 1}, "type": "shared\_room", "gender": "male"},

{"id": 60, "faculty\_cluster": "Engineering", "price": 2100, "amenities": ["wifi", "ac", "study\_desk"], "size": {"rooms": 2, "beds": 2}, "type": "apartment", "gender": "female"},

{"id": 61, "faculty\_cluster": "Business", "price": 1500, "amenities": ["wifi"], "size": {"rooms": 1, "beds": 1}, "type": "shared\_room", "gender": "male"},

{"id": 62, "faculty\_cluster": "Arts", "price": 1200, "amenities": ["wifi", "study\_desk"], "size": {"rooms": 2, "beds": 2}, "type": "dorm", "gender": "female"},

{"id": 63, "faculty\_cluster": "Law", "price": 1800, "amenities": ["wifi", "ac"], "size": {"rooms": 1, "beds": 1}, "type": "apartment", "gender": "male"},

{"id": 64, "faculty\_cluster": "Sciences", "price": 1600, "amenities": ["wifi", "study\_desk"], "size": {"rooms": 1, "beds": 2}, "type": "shared\_room", "gender": "female"},

{"id": 65, "faculty\_cluster": "IT", "price": 1300, "amenities": ["wifi"], "size": {"rooms": 1, "beds": 1}, "type": "shared\_room", "gender": "male"},

{"id": 66, "faculty\_cluster": "Medical", "price": 2200, "amenities": ["wifi", "ac", "study\_desk"], "size": {"rooms": 2, "beds": 2}, "type": "apartment", "gender": "female"},

{"id": 67, "faculty\_cluster": "Engineering", "price": 1550, "amenities": ["wifi"], "size": {"rooms": 1, "beds": 1}, "type": "shared\_room", "gender": "male"},

{"id": 68, "faculty\_cluster": "Business", "price": 2000, "amenities": ["wifi", "study\_desk"], "size": {"rooms": 2, "beds": 2}, "type": "shared\_room", "gender": "female"},

{"id": 69, "faculty\_cluster": "Arts", "price": 1400, "amenities": ["wifi"], "size": {"rooms": 1, "beds": 1}, "type": "shared\_room", "gender": "male"},

{"id": 70, "faculty\_cluster": "Law", "price": 1900, "amenities": ["wifi", "study\_desk"], "size": {"rooms": 1, "beds": 1}, "type": "shared\_room", "gender": "female"},

{"id": 71, "faculty\_cluster": "Sciences", "price": 1500, "amenities": ["wifi"], "size": {"rooms": 1, "beds": 1}, "type": "shared\_room", "gender": "male"},

{"id": 72, "faculty\_cluster": "IT", "price": 1450, "amenities": ["wifi", "study\_desk"], "size": {"rooms": 1, "beds": 1}, "type": "shared\_room", "gender": "female"},

{"id": 73, "faculty\_cluster": "Medical", "price": 1700, "amenities": ["wifi"], "size": {"rooms": 1, "beds": 1}, "type": "shared\_room", "gender": "male"},

{"id": 74, "faculty\_cluster": "Engineering", "price": 1800, "amenities": ["wifi", "ac"], "size": {"rooms": 1, "beds": 1}, "type": "apartment", "gender": "female"},

{"id": 75, "faculty\_cluster": "Business", "price": 1600, "amenities": ["wifi"], "size": {"rooms": 1, "beds": 1}, "type": "shared\_room", "gender": "male"}

]

# Expanded faculty clusters

faculty\_clusters = {

"Medical": ["Medicine", "Pharmacy", "Dentistry", "Nursing", "Physical Therapy", "Biomedical Sciences"],

"Engineering": ["Computer Engineering", "Mechanical Engineering", "Electrical Engineering", "Civil Engineering", "Architecture", "Chemical Engineering"],

"Business": ["Business Administration", "Economics", "Marketing", "Accounting", "Finance"],

"Sciences": ["Physics", "Chemistry", "Biology", "Mathematics", "Biotechnology"],

"Arts": ["Literature", "Fine Arts", "History", "Philosophy", "Psychology"],

"IT": ["Computer Science", "Information Technology", "Artificial Intelligence", "Data Science", "Software Engineering"],

"Law": ["Law", "Legal Studies", "International Law"]

}

def get\_faculty\_cluster(faculty):

for cluster, faculties in faculty\_clusters.items():

if faculty.lower() in [f.lower() for f in faculties]: # Case-insensitive matching

return cluster

return None

def recommend\_housing(faculty, budget, required\_amenities, preferred\_size, preferred\_type, gender):

recommendations = []

faculty\_cluster = get\_faculty\_cluster(faculty) if faculty else None

rejection\_reasons = []

# Stage 1: Exact match

for housing in housing\_options:

reasons = []

# Rule 1: Gender match (mandatory)

if housing["gender"] != gender:

reasons.append(f"Gender mismatch (required: {gender}, found: {housing['gender']})")

rejection\_reasons.append({"id": housing["id"], "reasons": reasons})

continue

# Rule 2: Faculty cluster match (optional)

if faculty\_cluster and housing["faculty\_cluster"] != faculty\_cluster:

reasons.append(f"Faculty cluster mismatch (required: {faculty\_cluster}, found: {housing['faculty\_cluster']})")

# Rule 3: Price within budget (±20%)

price\_margin = budget \* 0.2

if not (budget - price\_margin <= housing["price"] <= budget + price\_margin):

reasons.append(f"Price out of range (required: {budget - price\_margin}-{budget + price\_margin}, found: {housing['price']})")

# Rule 4: Amenities match

amenities\_match = all(amenity.lower() in [a.lower() for a in housing["amenities"]] for amenity in required\_amenities)

if not amenities\_match:

reasons.append(f"Amenities missing (required: {required\_amenities}, found: {housing['amenities']})")

# Rule 5: Size match (rooms and beds)

size\_match = (housing["size"]["rooms"] >= preferred\_size["rooms"] and

housing["size"]["beds"] >= preferred\_size["beds"])

if not size\_match:

reasons.append(f"Size mismatch (required: {preferred\_size}, found: {housing['size']})")

# Rule 6: Housing type match (optional)

if preferred\_type and housing["type"] != preferred\_type:

reasons.append(f"Type mismatch (required: {preferred\_type}, found: {housing['type']})")

if not reasons:

recommendations.append(housing["id"])

else:

rejection\_reasons.append({"id": housing["id"], "reasons": reasons})

# Stage 2: Relax faculty and type

if not recommendations:

fallback\_recommendations = []

for housing in housing\_options:

reasons = []

if housing["gender"] != gender:

continue

if not (budget - price\_margin <= housing["price"] <= budget + price\_margin):

continue

if not all(amenity.lower() in [a.lower() for a in housing["amenities"]] for amenity in required\_amenities):

continue

if not (housing["size"]["rooms"] >= preferred\_size["rooms"] and

housing["size"]["beds"] >= preferred\_size["beds"]):

continue

fallback\_recommendations.append({"id": housing["id"], "faculty\_cluster": housing["faculty\_cluster"], "type": housing["type"]})

if fallback\_recommendations:

recommendations = [r["id"] for r in fallback\_recommendations[:5]]

print("No exact matches found. Returning closest matches (relaxed faculty or type constraints):")

else:

# Stage 3: Relax amenities and size

final\_fallback = []

for housing in housing\_options:

if housing["gender"] != gender:

continue

if not (budget - price\_margin <= housing["price"] <= budget + price\_margin):

continue

final\_fallback.append({"id": housing["id"], "amenities": housing["amenities"], "size": housing["size"]})

if final\_fallback:

recommendations = [r["id"] for r in final\_fallback[:5]]

print("No matches with required amenities or size. Returning closest matches (matching gender and price only):")

else:

print("No matches found, even with relaxed constraints. Reasons for rejection:")

for rejection in rejection\_reasons:

print(f"Housing ID {rejection['id']}: {', '.join(rejection['reasons'])}")

return recommendations

# Test Case 1: Your original input

student1 = {

"faculty": "Computer Science",

"budget": 1500,

"required\_amenities": ["wifi"],

"preferred\_size": {"rooms": 1, "beds": 1},

"preferred\_type": "", # Flexible type

"gender": "male"

}

print("Test Case 1 (Your input):")

recommended\_housing1 = recommend\_housing(\*\*student1)

print("Recommended housing IDs:", recommended\_housing1)

# Test Case 2: Different input (female, Medicine, 2000 budget)

student2 = {

"faculty": "Medicine",

"budget": 2000,

"required\_amenities": ["wifi", "study\_desk"],

"preferred\_size": {"rooms": 1, "beds": 1},

"preferred\_type": "shared\_room",

"gender": "female"

}

print("\nTest Case 2 (Different input):")

recommended\_housing2 = recommend\_housing(\*\*student2)

print("Recommended housing IDs:", recommended\_housing2)

# Test Case 3: Low budget, female, Arts, dorm

student3 = {

"faculty": "Fine Arts",

"budget": 1000,

"required\_amenities": ["wifi"],

"preferred\_size": {"rooms": 1, "beds": 1},

"preferred\_type": "dorm",

"gender": "female"

}

print("\nTest Case 3 (Low budget, female, Arts):")

recommended\_housing3 = recommend\_housing(\*\*student3)

print("Recommended housing IDs:", recommended\_housing3)

# Test Case 4: Invalid faculty, high budget

student4 = {

"faculty": "Invalid Faculty",

"budget": 4000,

"required\_amenities": ["wifi", "ac"],

"preferred\_size": {"rooms": 2, "beds": 2},

"preferred\_type": "apartment",

"gender": "male"

}

print("\nTest Case 4 (Invalid faculty, high budget):")

recommended\_housing4 = recommend\_housing(\*\*student4)

print("Recommended housing IDs:", recommended\_housing4)