Design and Implementation of Student Accommodation System

(Preference Matching)

BY

Maryam Abba Yusuf

BU/22B/IT/6965

IN PARTIAL FULFILMENT OF THE REQUIREMENT FOR THE AWARD OF BACHELOR OF SCIENCE IN SOFTWARE ENGINEERING, FACULTY OF COMPUTING AND APPLIED SCIENCE, BAZE UNIVERSITY, ABUJA.

September,2024

DECLARATION

I hereby declare that this research project has been written by me under the supervision of Dr. Usman Bello Abubakar. The work has been presented in any previous research for the award of B.Sc degree to the best of my knowledge. The work is entirely mine and I accept the sole responsibility for any errors that might be found in the work, while the reference to publish material have been duly acknowledge.

______________________________ ________________________

Maryam Abba Yusuf Date
BU/22B/IT/6965

APPROVED BY

____

Head of Department,

Department of Computer Science

CERTIFICATION

This project entitled "Design and Implementation of Student Accommodation System (preference matching)" meets the requirements governing the award of Bachelor of Science in Software Engineering in Baze University, Abuja.

APPROVAL

This is to certify that the research work title Design and Implementation of Student Accommodation System (Preference Matching) by Maryam Abba Yusuf with BU/22B/IT/6965 has been approved by the Department of Computer Science, Faculty of Computing and Applied Science, Baze University, Abuja, Nigeria.

By

Dr Usman Bello Abubakar

Supervisor Date

Dr Usman Idris Abubakar

Head of Department Date

Prof Helen Negbenebor

Dean, Faculty of Computing and Applied Science Date

Prof Choji Davou Nyap

External Examiner Date

DEDICATION

I dedicate this work to Allah (SWA), my source of inspiration, knowledge and understanding, who gave me the grace and strength throughout this program. I also dedicate this work to my parents, for their love and support throughout my life.

ACKNOWLEDGMENT

All praise, first and foremost, is to God Almighty for His guidance and abundant blessings. My deepest appreciation goes to my parents and siblings for their unwavering love and support. I would also like to acknowledge my project supervisor, Mr. Usman Abubakar, for his invaluable assistance, enduring the stress of reviewing my work and providing necessary feedback. I am deeply grateful to Mr. Alee Dauda for his constant support and guidance throughout this project. His insights and encouragement played a key role in its successful completion.

ABSTRACT

Many students encounter difficulties finding suitable accommodation after being admitted to university. To address this issue, a Student Accommodation System has been developed to assist students in finding hostels that meet their specific requirements through preference matching. This system offers a comprehensive list of hostels, which are regularly updated by administrators. Through the platform, students receive personalized recommendations for hostels that align with their preferences. By leveraging preference matching, the system ensures that recommendations are tailored to each student's unique needs, providing a pre-visit insight that helps them secure appropriate accommodation with greater ease. The Student Accommodation System benefits both students and hostel owners. For students, it simplifies the search process, reduces the stress associated with finding accommodation, and ensures that recommendations reflect their individual preferences. For hostel owners, it enhances visibility and extends their reach within the city, potentially resulting in increased bookings. The system serves as a bridge between students and hostel providers, making the accommodation search more efficient and effective for all parties involved.

Table of Contents

LIST OF TABLES

LIST OF FIGURES

CHAPTER ONE

INTRODUCTION

1.1 Overview

This project aims to develop a modern student accommodation system for students in Abuja, leveraging a recommender system. The goal is to deliver personalized housing recommendations tailored to individual preferences. By automating the search process, this system will significantly reduce the time and effort students spend finding suitable accommodation, while also optimizing house allocation.

1.2 Background and Motivation

Student accommodation is a critical aspect of university life, ensuring that students have safe, affordable, and convenient places to live while pursuing their studies. In Abuja, the capital city of Nigeria, the demand for student housing has surged with the growing number of higher education institutions and students. Options range from university to private student housing complexes, shared apartments. However, finding the right accommodation that meets the needs and preferences of students remains a challenge in Abuja's dynamic real estate market.

Initially, the process of finding student accommodation was manual, relying on physical visits, word of mouth, and simple listings. This approach was time-consuming, and often resulted in mismatches between students' needs and available housing options (Johnson, 2019). As student populations grew and the variety of housing options expanded, these manual processes became increasingly inadequate.

The system developed for this project leverages modern algorithms to match students based on their stated preferences. This approach is both personalized and data-driven, ensuring that users are matched with roommates and rooms that align with their living style.

The aim of this system is to improve the student living experience by minimizing potential conflicts between roommates and promoting positive interactions. Students can specify their detailed preferences, which the system uses to suggest compatible rooms and roommates. This process is efficient, benefiting both students and accommodation managers by saving time and effort.

Additionally, the system gives students a sense of empowerment, knowing their living preferences are considered, which can result in higher satisfaction and increased retention for housing providers.

1.3 Statement of the Problem

Students often struggle to find suitable accommodation due to limited knowledge of available options and difficulty matching their preferences with available housing and roommates. This project addresses these challenges by proposing a **Student Accommodation System Using Preference Matching**, which helps students find accommodation that aligns with their specific needs, simplifying the search process and improving satisfaction.

1.4 Aim and Objectives

The main aim of this research is to design and implement a student accommodation using recommender system to match users with a roommate base on their preferences.

1. To develop a comprehensive and user-friendly software that assists student in finding suitable Accommodation
2. To create a booking and reservation system that allows users to secure their chosen Accommodation, manage bookings, and handle payment transactions securely.
3. To develop a user-friendly platform where students can create profiles specifying their accommodation requirements (location, lifestyle preferences).

1.5 Significance of the Project

The Student Accommodation system using preference matching is significant as it offers many benefits:

1. It ensures that students find suitable accommodation quickly and accurately.
2. Users are provided with a system that enables them to select accommodations according to their preferences
3. It reduces the time and effort required for manual accommodation.

4. Helps in efficient allocation and management of housing resources.

1.6 Project Risks Assessment

Table 1.1 Project Risks Assessment

RISK	IMPROVEMENT
Failure to carry out inquire about due to misfortune of hardware/software assets.	Be mindful of and observe school IT security strategies How to keep your Android phone secure when not in use
Preventing work loss in case of equipment failure or loss	Day by day Reinforcement of information to numerous sources of capacity such as flash drive, hard drives, google drive, etc. for assortment.
We will explore alternative APIs if the required APIs are not available	We will identify the software requirements early to address any potential software conflicts.

1.7 Scope/Project Organization

This project was arranged into five chapters: Chapter one as an introduction to the general aim and objective of the project, and the ideas at focus presented. Chapter two deals with relevant literatures of components used in realizing this project while Chapter 3, is design methodology, Chapter 4, is implementation of the methodology and testing. Chapter 5 covered conclusions, limitations, and suggested improvements for the system.

1.8 Definition of Terms

Accommodation: refers to any space, whether a single room, a group of rooms, or an entire building, where a person can reside or stay.

Preference Matching: refers to a process where individuals or entities provide their preferences, and a system or algorithm tries to match those preferences with available options to find the best possible outcome.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

The literature review chapter aims to provide a comprehensive overview of the existing knowledge and research related to the development of a student accommodation system using preference matching. This chapter will look into the historical evolution of accommodation systems, examine previous research and implementations of using preference matching systems in the housing sector, and identify gaps and limitations in current solutions. By **fundamentally** analyzing the literature, this study seeks to build upon existing knowledge and contribute to the advancement of an efficient and personalized student accommodation system. The exploration of literature will inform the design and implementation of a robust recommendation system tailored to meet the diverse needs of students in Abuja, ensuring optimal matching of preferences with available housing options.

2.2 Historical Overview

Before the advent of digital technology, the process of finding student accommodation was largely manual and time-consuming. Students relied on notice boards, real estate agents, or university housing offices. There was little focus on matching accommodation options to student preferences, and the process lacked personalization. It was mostly driven by availability and affordability, with students often having limited access to detailed information about their potential living spaces. By the late 1990s and early 2000s, the rise of the internet revolutionized the housing search process, including student accommodation. Early platform and various student housing websites, offered digital listings, allowing students to browse available properties more easily. However, these systems typically featured basic filtering options like price, location, and number of rooms.

The 2000s and early 2010s saw the emergence of specialized student housing platforms designed specifically for university students. Websites like Uniplaces, Student.com, and HousingAnywhere catered directly to student accommodation needs, offering more user-friendly interfaces and better filtering options.

During this time, these platforms began to recognize the importance of preference matching. Basic algorithms were developed to rank listings based on simple student preferences, such as proximity to university campuses or rent budgets. However, the ability to match students with accommodations based on more detailed criteria (such as personal preferences) remained limited.

2.3 Preference system

How the Preference System works

Collecting User Preferences: After student’s log in, they are required to specify their roommate preferences, which include key factors like cleanliness, sleep habits, quietness, and social behavior. These attributes play a vital role in helping the system identify compatible roommates.

Filtering Available Rooms: When a student begins searching for accommodation, the system first narrows down available rooms by considering the user’s preferences. The get filtered room recommendations function filters out rooms that do not meet the student’s gender preference, only showing rooms with available spots that align with the user's requirements.

Calculating Match Percentages: A significant part of the preference-matching process is carried out by the calculate match percentage function. This function compares the student's preferences with those of the current occupants in a room, calculating a match percentage. The algorithm evaluates how closely the student's preferences for cleanliness, quietness, and socialness align with those of the existing occupants. It does this by calculating the absolute difference between the user's and occupants' preferences. The closer the values are, the better the compatibility and, as a result, the higher the match percentage.

Ranking Rooms: Once match percentages are computed for all available rooms, the system ranks them from the highest to the lowest match percentage. This ensures that the student is presented with rooms that best match their preferences, increasing the likelihood of finding a room that meets their expectations.

Example:

If the user prefers a high level of cleanliness (e.g., a score of 10) and the current occupant prefers a medium level of cleanliness (e.g., a score of 5), the system records a difference of 5. The process is repeated for other preferences, such as quietness and social behavior, and the differences are summed up. If a room has no current occupants, it is treated as a perfect match, with a score of 100%.

2.4 Related Work

According to Su and Khoshgoftaar, (2019) and Lakshmi and Lakshmi, (2014); when a recommendation system is first faced with an unfamiliar object or its new potential user, at this stage, historical data such as ratings, preferences, and search histories that may be questions are not in possession. This is rated as the cold start problem. In its other form it is commonly called the ‘new item’ or the ‘new user’ problem. One such way of tackling this problem is to utilize the information about the user’s demographic characteristics that usually exist in their profiles. This method is not very good and not entirely precise as profiles that are ethnically the same may result in different preferences regarding the same product.

Lu et al., (2015). The recent years have seen the research community facing the challenge of making improvements to better the applicability and performance of Recommendation systems. The problem of recommending in case of new users or new items has been accepted. One possible approach is demographic information from user profiles, but the disadvantages are that it is insufficient and inaccurate since two people with the same demographic will gravitate towards different items. These algorithms typically cover the demographic filters, the content-based ones, the collaborative ones, and the hybrid ones.

Dejo et al., 2015; Zhang et al., 2016; Bernardes et al., 2015 With the advent of social networking sites such as Facebook and Twitter, Recommendation systems have become an integral part of social networking sites. Social networking sites represent a real gold mine with regard to user information, which in turn creates a unique opportunity to develop new and innovative recommendation techniques to increase the accuracy of recommendations. Social sites supply a certain amount of contextual information, such as timestamp and the location of the users as well as groups, and the emotional status of individuals and groups. The best way to exploit all of the above is to develop a new type of Recommendation system, called contextual Recommendation system. The new Recommendation system also opens up a possibility to make a Recommendation system dynamic. Another significant application field of context-aware recommendations is seasonal marketing and conference recommendations. The recommendation system in recent times is a widely known and well-accepted concept. The popular saying that ‘a man is known by the company you keep’ suggests that if two or more users have had similar interests in the past, then it’s likely that their interests will remain the same in the future. For example, if user1 and user2 have followed a completely similar purchase history, then it is almost certain that, when user1 makes a purchase, user2 will also buy the same or a similar product. In collaborative Filtering methods, a user’s earlier reviews and ratings of items are analysed to recommend similar items in the future. Even if the user has not interacted (tried, purchased, rated, liked) with a specific item, it could be recommended to the user in view of either the user’s interactions with the same or similar items and in the same or similar situation/context or because other users whom the user trusts have used the same item. It goes without saying that a reasonable amount of recommendation accuracy is based on the number of user groups considered. Trust plays a huge role in a successful recommendation.

This refers to the point made by Tewari, A.S. and Priyanka, K. who wrote in 2015 that ‘the scalability of the system plays a key role in recommendation processes as these systems have to deal with live requests over the Internet and provide users with the best possible recommendations based on their previous purchase and rating history, As Recommendation systems get more and more complex while dealing with large datasets, more specifically large amount of users and millions of unique items, it is crucial that those systems are able to handle the online requests with high efficiency and have good abilities to scale based on user’s purchase and rating history, because the underlying systems are highly unlikely to scale well during online interactions.

As Meymandpour and Davis (2015) have pointed out on the Amazon review site, and as Amazon.in (2017a, 2017) say in response to complaints, most recommendation systems have problems with items that have different names (synonyms) such as the inability to match ‘children movie’ and ‘children film’ in memory-based CF systems to calculate similarity.

Lakshmi and Lakshmi (2014), Sarwat et al.(2015), Mayeku et al. (2015) and Orellana-Rodriguez et al. (2015) state that the system might miss the abbreviations which are a common part of the users’ online interactions, thereby recommending the wrong recommendations. One way to deal with this is by collecting the list of abbreviated words with their full forms and then including both versions into the system. In addition, when the system gets the user contextual information, say who is the companion you are going out with, and the location, the time of the day and the mood, we can define a context. Demographic information is different to the contextual information. Demographic information does not change over a long period of time, unlike contextual information, which changes with the situation of the user. This way we can capture the emotional context. This requires collection of a lot of data, which may lead to a number of issues and a system that could be added into a database system. Besides, context-aware online learning platform has been introduced along with an efficient method for extracting meaningful contexts from user comments on YouTube.

Wang et al. (2015) present a Bayesian network classifier, a probabilistic model, is utilized to address classification problems in large networks such as social networks. To tackle the user's cold start problem and enhance recommendation accuracy, a trust-based probabilistic recommendation model is proposed for social networks.

Zhang and Zhou (2014) have employed support vector machine, a supervised learning method, along with an associated learning algorithm to analyze data using classification (linear and nonlinear) and regression analysis to identify profile injection attacks in conjunction with the Hilbert-Huang transform.

Table 2.1 Comparative Analysis of the Related work

Related Work	**Method/Approach**	**Strengths**	**Weaknesses**
Chen et al. (2015)	Fuzzy-based decision-making	Handles uncertainty in preferences	Limited scalability and Requires precise preference weights
Abdullah et al. (2019)	Student Accommodation Preference Analysis Using Analytic Hierarchy Process	Effective in analyzing student accommodation preferences	Limited geographical scope (single university)

Aggarwal, C. C. (2016)	Recommender systems based on knowledge.	Comprehensive suite of tools, user-friendly interface, supports various constraints and preferences	Requires significant technical expertise and resources for implementation and customization
Basavesh et al, (2023).	Location-Based Recommendation System	Hostel Finder: for Hostels and PGS with Transit	Adapting to users and user interaction.
Dejo, el tal, (2015)	Recommendation systems	Fundamental concepts, techniques, and assessment	Specific limitations not mentioned in the document
Ekstrand, et al, (2015)	Recommendation systems utilizing collaborative filtering.	Foundations and Trends in Human-Computer Interaction	Theoretical Aspects and Real Applications
Elahi, et al, (2016)	An extensive assessment of recommender systems using collaborative filtering.	Active learning in collaborative is the subject of a survey	The document does not discuss specific strengths and weaknesses.
Elkahky, et al, (2015)	Utilizing modeling in recommendation systems.	An approach to cross-domain user using a multi-view deep learning technique.	The document does not discuss specific strengths and weaknesses.

FoxTrit (2017)	Personalized Recommender System for Digital Libraries	Reduced scheduling process time, improved allocation fairness	Limited information provided in the document
Lakshmi, et al. (2014)	Recommendation systems	Challenges and problems with recommendations.	The document does not discuss specific strengths and weaknesses.
Lu, et al, (2015)	Evaluation of advancements in recommender system applications.	The use of decision support systems in recommendation system applications.	The document does not discuss specific strengths and weaknesses.
Meymandpour, R. and Davis, J. (2015)	Utilizing recommender systems using linked.	Perform semantic analysis of items based on open data.	The document does not discuss specific strengths and weaknesses.
Moghaddam, et al, (2014)	Recommendation algorithms utilizing item-based collaborative filtering.	Maximizes student accommodation enrollment placements, minimizes	The document does not discuss specific strengths and weaknesses.
Peis, et al, (2018)	Semantic recommender systems	Provides insights into the analysis of the state of the software solutions	The document does not discuss specific strengths and weaknesses.

Pelánek, R. (2018)	Development of a web-based system for recommending housing options.	Creating, implementing, and assessing data mining and knowledge in engineering.	The document does not discuss specific strengths and weaknesses.
Pyo, S., Kim, E. and Kim, M. (2015)	Unified topic modeling based on LDA to group similar TV users and provide TV program recommendations.	study impact on grouping TV users.	The document does not discuss specific strengths and weaknesses.
Sarwat, et al, (2015)	Middleware designed for context-aware recommendations.	Investigating the impact on database systems.	The document does not discuss specific strengths and weaknesses.
Singh, et al (2019f)	Enhancing the accuracy of collaborative filtering-based recommendation systems	by considering temporal variance of top-N.	The document does not discuss specific strengths and weaknesses.
Su, X. and Khoshgoftaar, T.M. (2019)	An overview of collaborative filtering techniques.	Progress in the field of Artificial Intelligence.	The document does not discuss specific strengths and weaknesses.
Tewari, A.S. and Priyanka, K. (2015)	Book recommendation system based on collaborative filtering	Association rule in mining for college students	The document does not discuss specific strengths and weaknesses.

Wang, et al (2015a)	A trust-based probabilistic model for recommendations in social networks.	Model for social networks	The document does not discuss specific strengths and weaknesses.
Wilson, J., Chaudhury, S. and Lall, B. (2014)	Improving collaborative filtering-based recommenders	using topic modelling	The document does not discuss specific strengths and weaknesses.

2.5 Summary

The chapter explores the development of systems that provide personalized suggestions, tracing their progress from basic models to more advanced solutions by the mid-2010s. While initial systems like those used by Spotify and Netflix focused on media recommendations, the underlying principles of matching user preferences can be applied to other areas, such as student accommodation, In the case of student accommodation systems, rather than relying on complex algorithms, a simpler preference matching approach can be equally effective. By allowing students to input their preference, the system can match them with suitable accommodation options that meet their unique needs. This tailored approach helps students find housing that aligns closely with their preferences, simplifying what can otherwise be a stressful process.

Although much of the research on recommendation systems centers on complex methodologies, preference matching provides a straightforward and efficient alternative, particularly for smaller-scale applications like a student accommodation system. The chapter reviews various methods, showing that while advanced algorithms have their place, simpler techniques like preference matching can offer significant value in environments where individual preferences play a key

CHAPTER 3

REQUIREMENTS, ANALYSIS, AND DESIGN

3.1 Overview

This chapter focuses on determining the requirements, performing analysis, and developing the system design for Student Accommodation (Using Preference matching). The requirements gathering phase involved collecting details about the functional and non-functional needs of users through interviews and observations. Various diagrams have been used to depict the system analysis and design including use cases, activity diagrams, data flow diagrams and entity.

3.2 Adopted Methodology

Agile methodology is chosen for its flexibility, continuous feedback, collaboration, and incremental delivery, ensuring the student accommodation system adapts to changing needs, incorporates user input, and delivers a high-quality product efficiently.

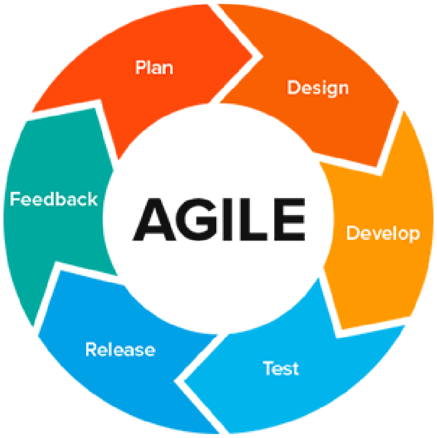


figure 3.1 Agile Methodology (Aguayo, M. 2021).

3.2.1 Interview

Interviews were conducted with students to understand their accommodation preferences. These interviews helped identify common needs and preferences, which informed the functional requirements of the system.

3.3 Tools and Techniques.

Next.js is been used on both front-end and back-end for structure, styling, and preference. PostgreSQL is used on the back-end to generate dynamic content and store/access data from the database.

3.4 Ethical Consideration

1. Student data privacy and security
2. Transparency on how student data is used
3. Accessibility and inclusion requirements
4. Fairness, accountability in recommender systems

3.5 Requirement Analysis

3.5.1 Software Requirements

1. Operating System: Windows
2. Database: PostgreSQL
3. Application program: VS Code
4. Next.js

3.5.2 Hardware Requirements

1. HP EliteBook
2. 8GB RAM
3. Browser

3.6 Requirements Specifications

Table 3.1 Functional Requirement Specifications

Req. No.	**Description**	**Type**
R-101	The operating system must be Windows 7 or a later version.	Configuration
R-102	Users must have the ability to register using their email and password.	Functional
R-103	Users should have the capability to see the list of available rooms..	Functional
R-104	Users need to be able to complete the payment process.	Functional
R-105	Users must be allowed to create their profiles.	Functional
R-106	Users should have the option to view potential matches..	Functional
R-107	The administrator should have access to pending payment records.	Functional
R-108	The administrator needs to have the ability to add rooms.	Functional
R-109	The administrator must be able to update room information.	Functional
R-110	The administrator should be able to delete a room from the system.	Functional
R-111	The system must permit users to input 4 preferences for filtering desirable rooms.	Functional
R-112	The system must use the student's preferences to match and display suitable rooms.	Functional

Table 3.2 Non-Functional Requirement Specifications

Req. No.	**Description**	**Type**
NR-101	When launched, the application will continue to operate unless there is a deliberate shutdown of the application or the platform..	Performance
NR-102	Availability the system is available to everyone	Performance
NR-103	The system should be easy to use and user-friendly	Usability
NR-104	Efficient maintenance of the application is required.	Efficiency

3.7 System Design

3.7.1 Application Architecture

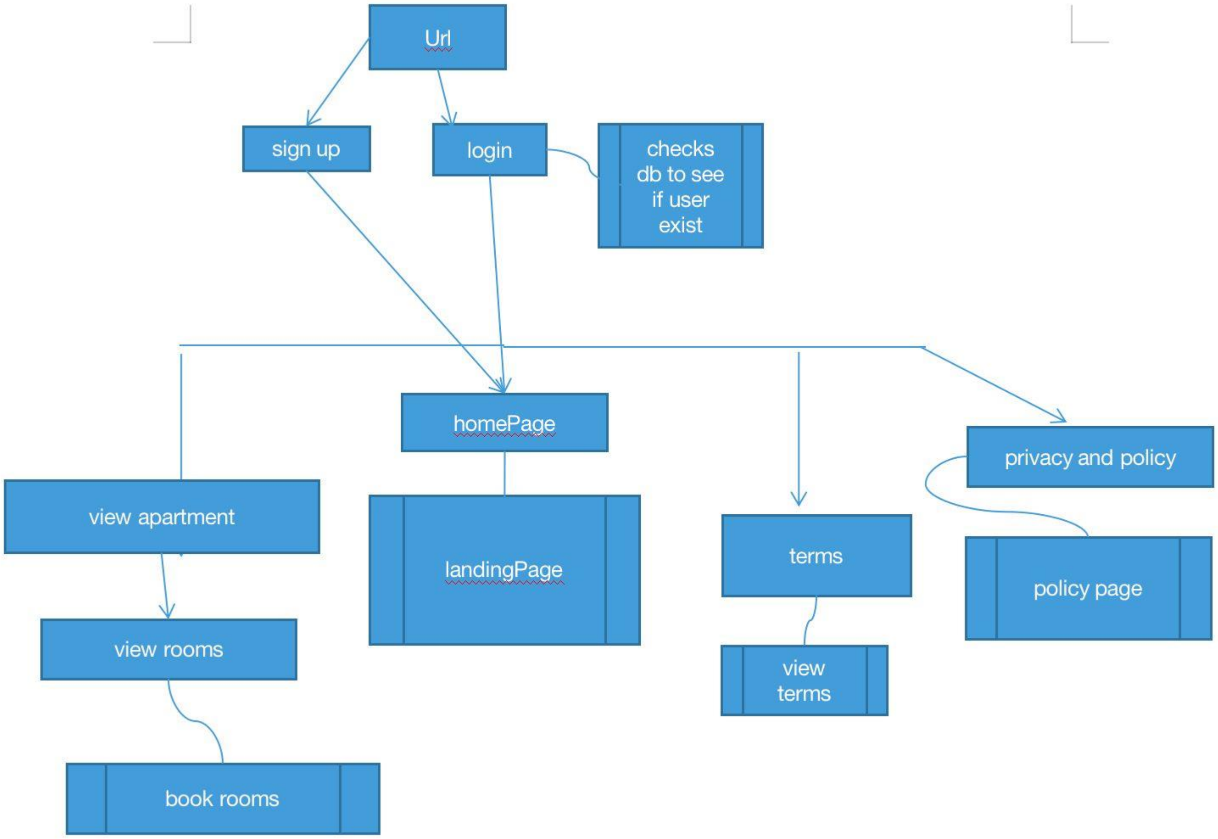


Figure 3.2 System Architecture for Student

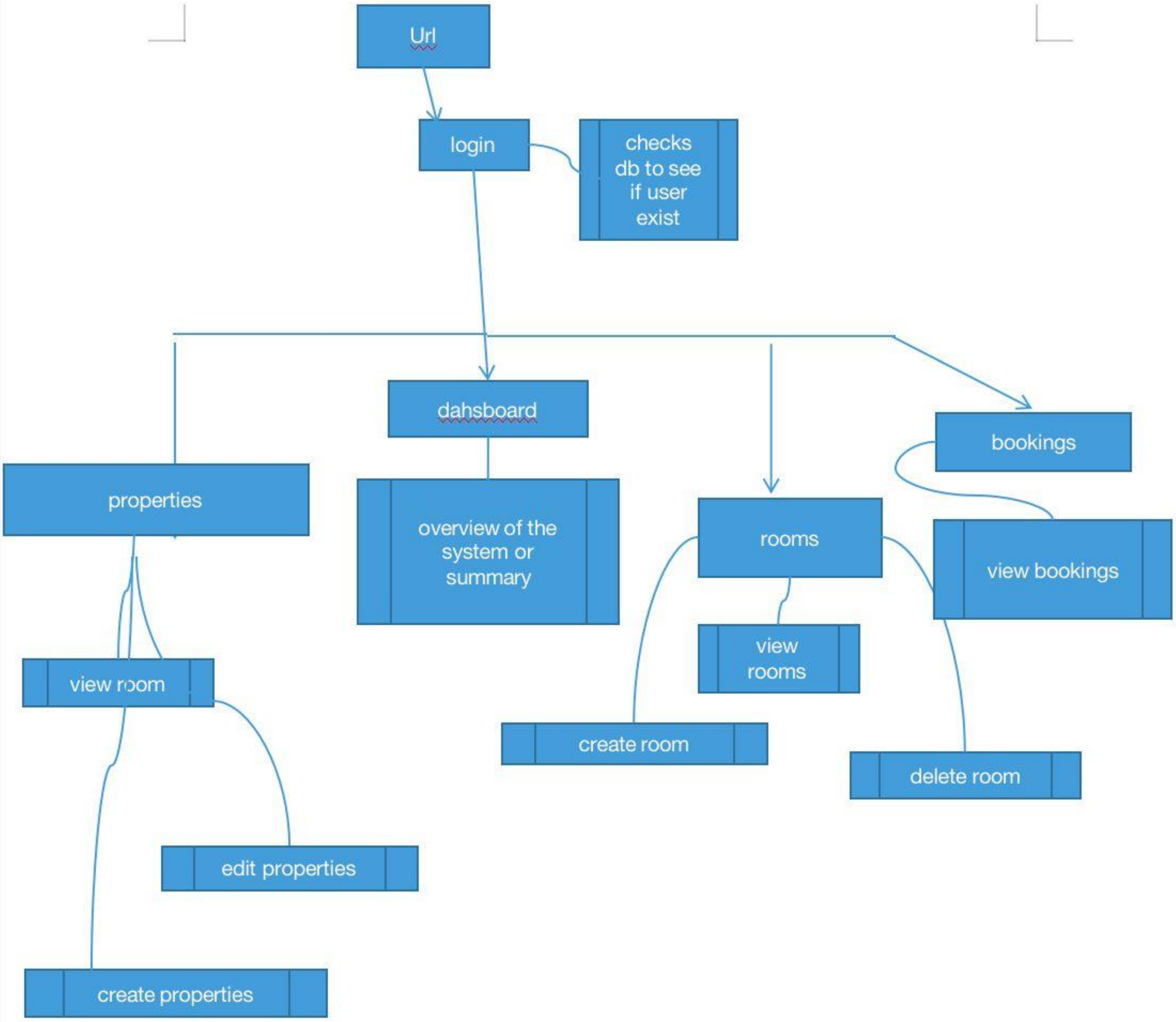


Figure 3.3 System Architecture for Admin

3.7.2 Use Case

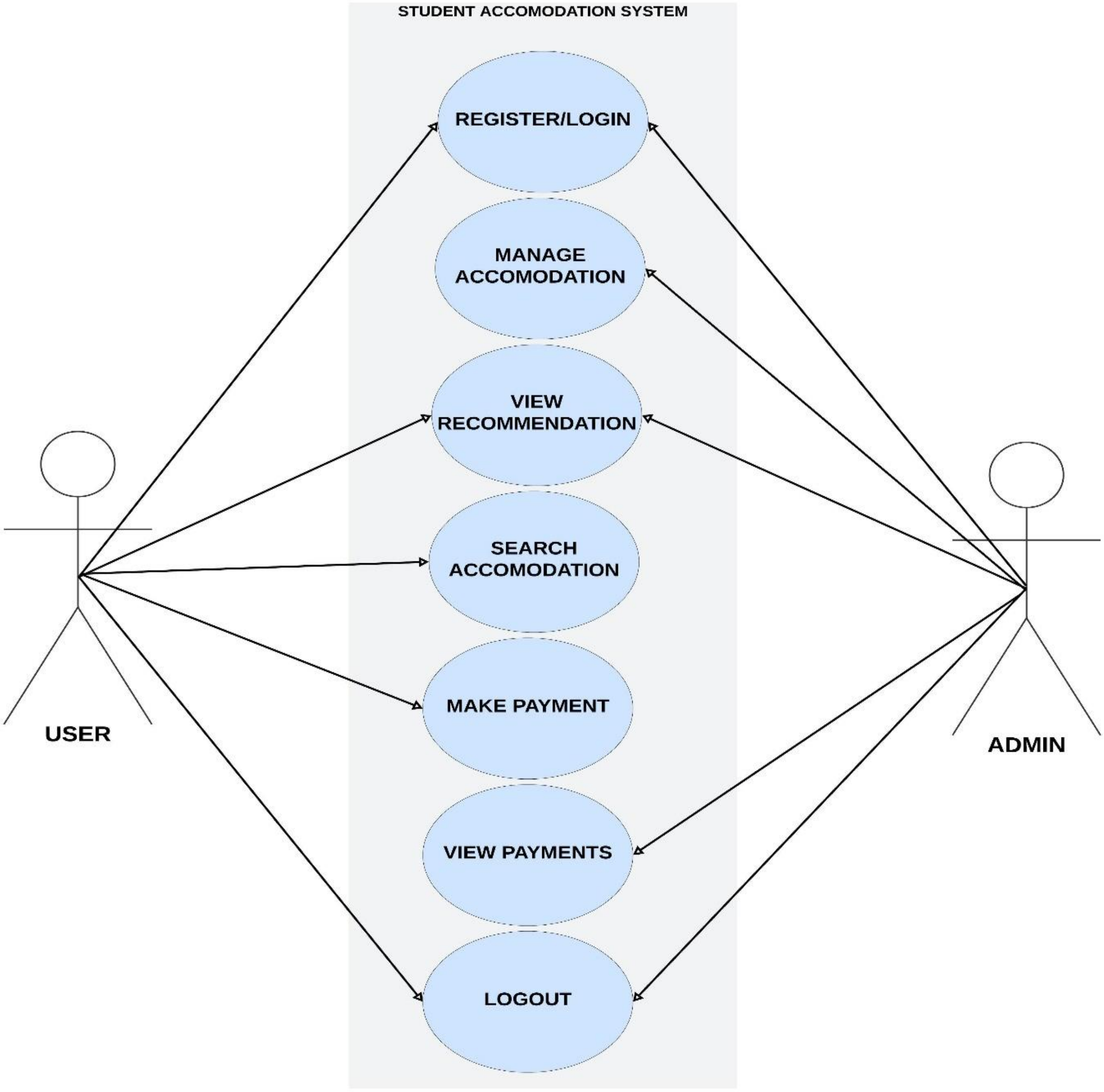


Figure 3.4: Use case Diagram

3.7.3 Use-Case Description

Table 3.3 Use-Case Description for Login/Register

USE CASE	LOGIN/REGISTER	
Description:	This use case outlines the procedure for accessing an existing account or creating a new user account in the accommodation management system.	
Actors:	User	
Pre-condition:	none	
Post condition	Access to the system is granted upon successful login. A new user account is created upon successful registration.	
Main flow	User 1. The user has the option to select either the login or registration process. 2. Upon choosing to log in, the user must input their login credentials. 3. If the user selects register, the user provides their registration details.	System 1. System validates credentials. 2. The system validates the provided login or registration details. 3. If the validation is successful, the user will be logged into or registered in the application. 4. Use case ends
Exception Conditions:	Incorrect login credentials will lead to an error message. The user can choose to retry or cancel, effectively ending the use case. Incorrect registration will result in an error message. The user can choose to retry or cancel, effectively ending the use case..	

Table 3.4 Use-Case Description for Search Accommodation

Use Case:	Search Accommodation	
Description:	a user is looking for accommodations in a specific location.	
Actor:	User	
Preconditions:	User is logged in Accommodation data is available in the system	
Post conditions:	The user is presented with a list of accommodations that match the search criteria.	
Main Flow:	User 1. User Selects “Search Accommodation”. 2. User selects Available location. 3. User selects Available Rooms.	System 4. The system searches the database to find available accommodations. 5. The system then shows a list of search results to the user.
Exception condition:	No accommodations found matching the searched location	

Table 3.5 Use-Case Description for View Payment(Admin)

Use Case:	View Payment
Description:	This use case outlines the process of an administrator checking the payment history for accommodations.
Actor:	Admin
Preconditions:	The admin is currently logged in.
Post conditions:	Payment information is displayed.

Main Flow:	Admin 1. Admin selects "View Payments". 2. Admin can filter or search for specific payments.	System 3. System searches the database for available Accommodation. 4. System displays list of search result
Exception condition:	No payment data available. System error during data retrieval.	

3.7.4 Activity Diagrams

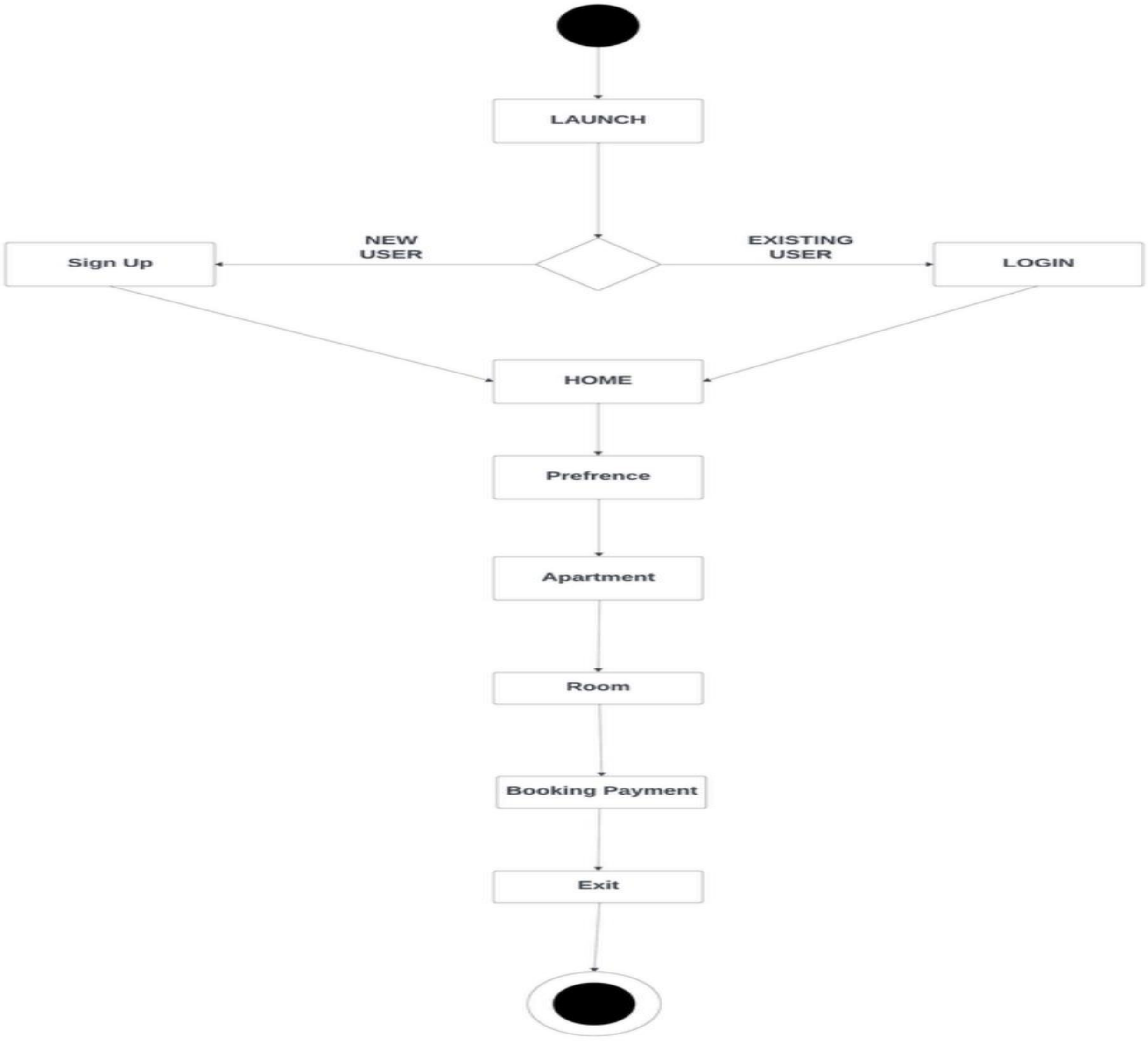


Figure 3.5 Activity Diagram (User)

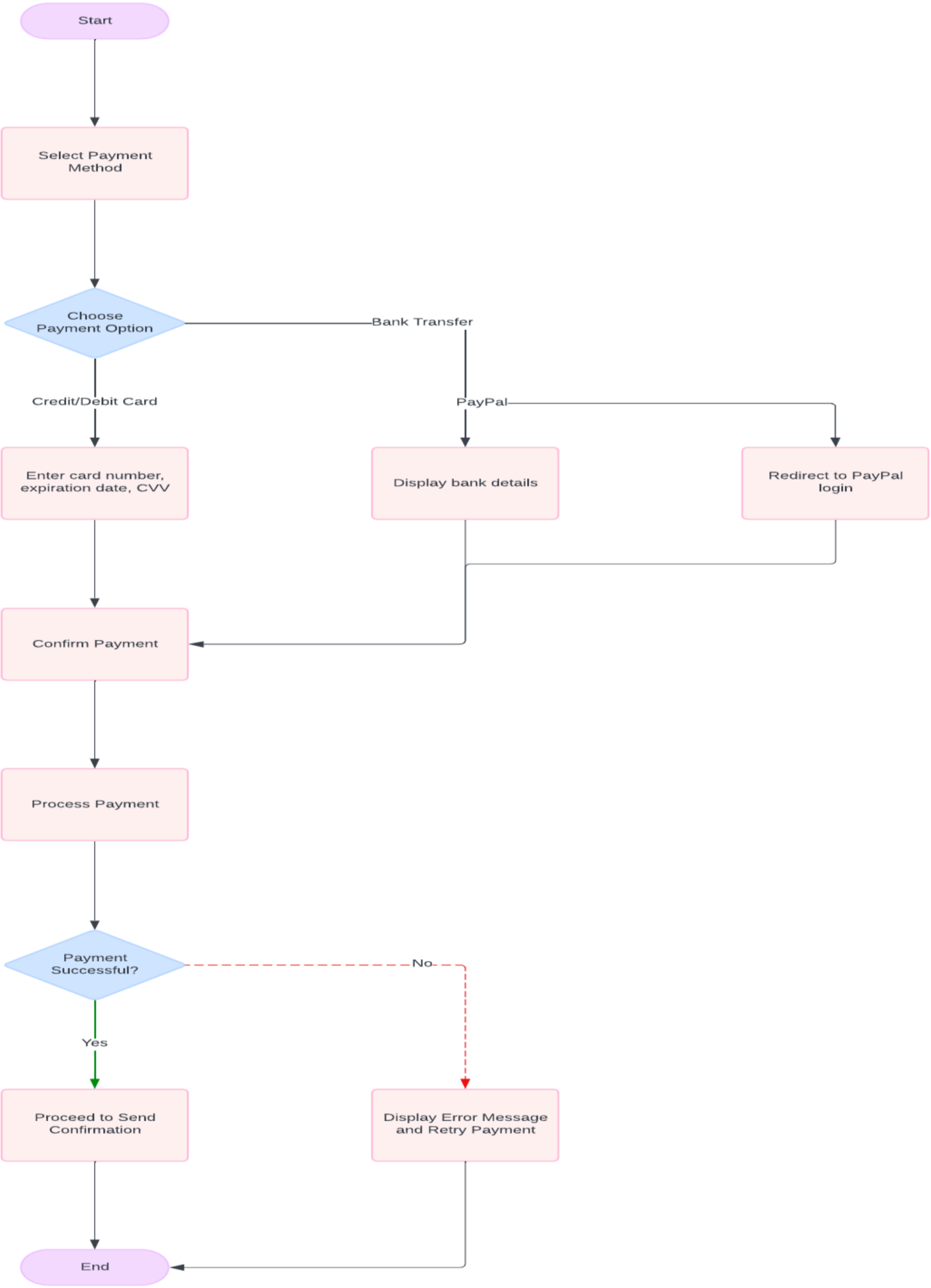


Figure 3.6 Activity Diagram (Payment)

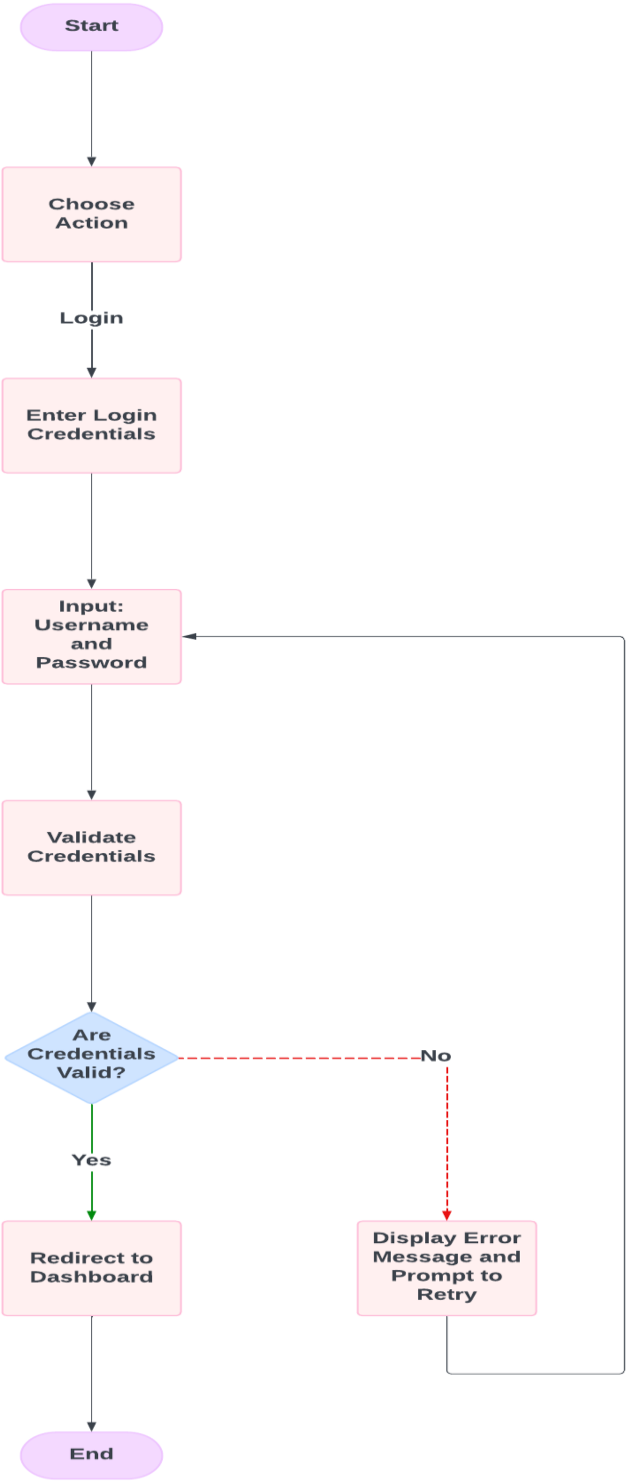
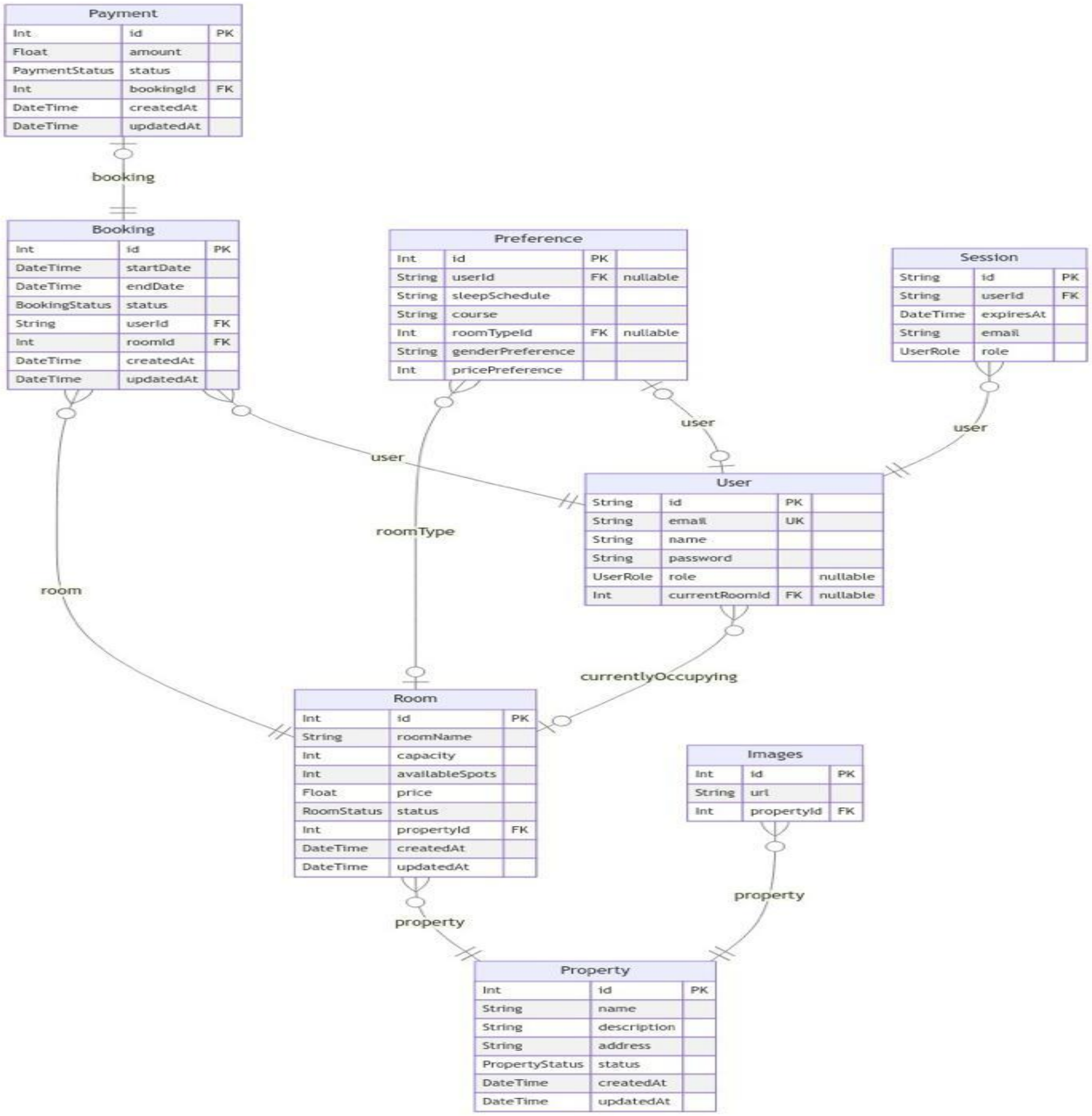


figure 3.7 Activity Diagram(Registration)

3.7.5 Entity Relationship Diagram



Payment

Int	id	PK
Float	amount	
PaymentStatus	status	
Int	bookingId	FK
DateTime	createdAt	
DateTime	updatedAt	

Booking

Int	id	PK
DateTime	startDate	
DateTime	endDate	
BookingStatus	status	
String	userId	FK
Int	roomId	FK
DateTime	createdAt	
DateTime	updatedAt	

Preference

Int	id	PK	
String	userId	FK	nullable
String	sleepSchedule		
String	course		
Int	roomTypeId	FK	nullable
String	genderPreference		
Int	pricePreference		

Session

String	id	PK
String	userId	FK
DateTime	expiresAt	
String	email	
UserRole	role	

User

String	id	PK	
String	email	UK	
String	name		
String	password		
UserRole	role		nullable
Int	currentRoomId	FK	nullable

Room

Int	id	PK
String	roomName	
Int	capacity	
Int	availableSpots	
Float	price	
RoomStatus	status	
Int	propertyId	FK
DateTime	createdAt	
DateTime	updatedAt	

Images

Int	id	PK
String	url	
Int	propertyId	FK

Property

Int	id	PK
String	name	
String	description	
String	address	
PropertyStatus	status	
DateTime	createdAt	
DateTime	updatedAt	

Figure 3.8 Entity Relationship Diagram

CHAPTER 4

IMPLEMENTATION AND TESTING

4.1 Overview

In this chapter, we will cover the practical implementation of the system and other relevant aspects that contributed to its development, including the front-end, back-end, database, as well as the challenges faced and their corresponding solutions.

4.2 Main Features

1. The Student Accommodation System Using Preference is designed to make the process of finding accommodation smoother and more personalized for students while streamlining management for administrators.
2. User Registration and Login: The system includes secure login functionality, where both students and administrators can sign in using their credentials. Role-specific access ensures that users can see and interact only with the features relevant to their role.
3. Personalized Roommate Matching: Students input their living preferences, such as cleanliness, social habits, sleep schedules, and noise tolerance. The system uses these preferences to match them with available rooms and suitable roommates, helping students find accommodations that best fit their lifestyle.
4. Accommodation Availability Filtering: Based on the student’s preferences and budget, the system filters through available rooms to recommend the most compatible options. Only rooms that meet the student’s criteria are shown, simplifying the decision-making process.
5. Roommate Compatibility Algorithm: The system calculates a match score between the student and potential roommates by comparing their preferences. Rooms are then ranked based on compatibility, ensuring students are paired with roommates who are most likely to get along.
6. Payment System: Once students select their preferred accommodation, they can proceed to make payments directly through the platform. The system supports secure payment processing, allowing students to pay for their accommodation fees or deposits online.

7. User Management: Administrators are responsible for managing user accounts, which involves tasks such as adding or removing users, assigning roles, and handling permissions. Additionally, they are able to monitor student activities and address issues regarding room assignments or payments.

4.3 Implementation Problems

Challenges faced during development included:

1. Incomplete information from students.
2. Verifying the authenticity of the provided data is difficult.
3. Insufficient data to make accurate recommendation.
4. Data inconsistencies and formatting issues.
5. Technical issues and errors.

4.4 Overcoming Implementation problem

1. Ensuring user's personal information id protected and secured.
2. Standardized data formats to ensure consistency.
3. Ensuring secure payment processing.

4.5 Testing

The application underwent testing to ensure that it fulfilled its requirements. Testing was done from a user's perspective to validate the efficiency and reliability of the application. Various forms of testing, including unit testing, were employed to thoroughly examine and test the application's functions from the front-end to the back-end.

Table 4.1 Testing for User Sign up

Test Case	User sign up
Related Requirement	FR01
prerequisites	The user can access the sign-in page
Test procedure	Navigate to the sign up page

	Enter the required information Click on “sign up” button
Test Data	User Information
Expected Result	Account Created
Actual Result	Account Created
Status	pass
Remark	None
Created by	Maryam Abba Yusuf
Date of Creation	10th August,2024
Executed By	Maryam Abba Yusuf
Date of Execution	10th August, 2024
Test Environment	HP Laptop

Table 4.2 Testing for User sign in

Test Case	User sign in
Related Requirement	FR02
prerequisites	User has a valid account User has access to sign in page.
Test procedure	Navigate to the log in page Enter valid username and password Click on the “Sign in” button
Test Data	Valid Username and password
Expected Result	The user should be able to successfully sign in and then be redirected to the home page or a page to set preferences if they haven't been set yet.
Status	pass

Remark	None
Created By	Maryam Abba Yusuf
Date of Creation	10th August,2024
Executed By	Maryam Abba Yusuf
Date of Execution	10th August, 2024
Test Environment	HP Laptop

Table 4.3 Testing for Admin

Test Case	Admin
Related Requirement	FR03
prerequisites	Admin has a valid account and is signed in. Admin has access to the admin management dashboard.
Test procedure	Login as Admin. Review current room availability and occupancy status. Update room availability (e.g., mark rooms as available or occupied). save changes and verify that updates are reflected in the system. View pending payment
Test Data	Room and Apartment status updates
Expected Result	Room availability and preferences should be updated successfully. The changes should be reflected in the system for users to see.
Status	pass
Remark	None
Created By	Maryam Abba Yusuf
Date of Creation	10th August,2024

Executed By	Maryam Abba Yusuf
Date of Execution	10th August, 2024
Test Environment	HP Laptop

Table 4.4 Testing for preference

Test Case	User Preference
Related Requirement	FR04
prerequisites	User has a valid account User has access to the preference page
Test procedure	Select gender preference (Male, Female, Other). Adjust sliders for cleanliness, socialness, quietness, and sleep schedule. Click on the “Submit” button
Test Data	Selected gender preference and values for cleanliness, socialness, quietness, and sleep schedule.
Expected Result	Once preferences are set, they should be saved successfully, and the user should then be redirected to a page with room recommendations.
Status	pass
Remark	None
Created By	Maryam Abba Yusuf
Date of Creation	10th August,2024
Executed By	Maryam Abba Yusuf
Date of Execution	10th August, 2024
Test Environment	HP Laptop

Table 4.5 Testing for Room Selection

Test Case	User Room Selection
Related Requirement	FR05
prerequisites	User has a valid account and is signed in. User has chosen an apartment and set their preferences. User has navigated to the room selection page.
Test procedure	Navigate to the room selection page. Review available rooms based on the chosen apartment and user preferences. Select a room from the available options. Confirm the room selection and proceed to the next step (e.g., payment).
Test Data	Selected apartment and user preferences
Expected Result	The system is expected to show available rooms that align with the selected apartment and preferences
Actual Result	The available rooms that correspond to the chosen apartment and preferences need to be shown by the system. A room should be selectable by the user.
Status	pass
Remark	None
Created By	Maryam Abba Yusuf
Date of Creation	10th August,2024
Executed By	Maryam Abba Yusuf
Date of Execution	10th August, 2024
Test Environment	HP Laptop

Table 4.6 Testing for payment

Test Case	User Payment
Related Requirement	FR06
prerequisites	User has a valid account and is signed in. User has selected a room and ready to make a payment. Payment gateway is integrated with the system.
Test procedure	Navigate to the payment page after selecting a room. Enter valid payment details(credit card, Bank Transfer etc.). Confirm the amount and click the "Pay Now" button. Receive confirmation of successful payment.
Test Data	Valid payment details (e.g., credit card number, Bank Transfer)
Expected Result	The payment should be successfully processed, and a confirmation message or receipt should be displayed. The user should be redirected to a "Payment Successful"
Status	pass
Remark	None
Created By	Maryam Abba Yusuf
Date of Creation	10th August,2024
Executed By	Maryam Abba Yusuf
Date of Execution	10th August, 2024
Test Environment	HP Laptop

4.5 Use Guide

1. **User Registration and Login:**

Access the system through a web browser.

New users should select the "Register" button and input the necessary information to establish an account.

For existing users, choose the "Login" button and input your credentials.

2. **Administrators**

After logging in, administrators will have access to administrative functions.

View and manage student and accommodation accounts.

Perform administrative tasks, such as managing room listings, updating accommodation details, assigning students to rooms, and overseeing payment processes.

3. **Student (User)**

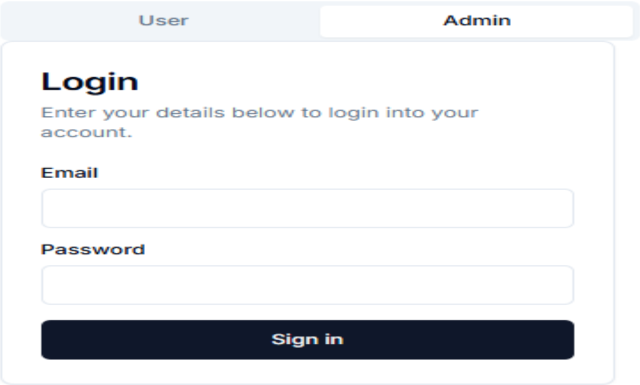
After logging in, students can input their preferences for accommodation and roommates (cleanliness, quietness, social habits, etc.).

View recommended rooms and potential roommates based on their preferences.

Make secure payments for their selected accommodation directly through the system.

4.7 User Interface Design

Admin's Page



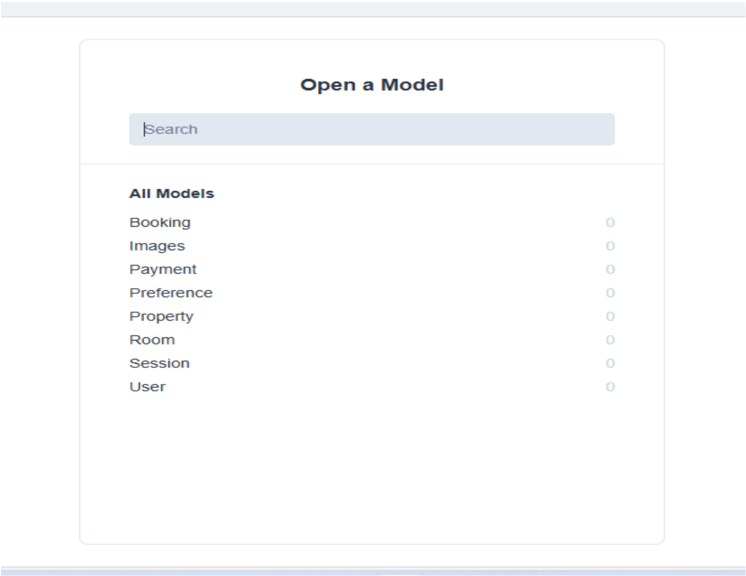
Login

Enter your details below to login into your account.

Email

Password

Figure 4.1: Login Page



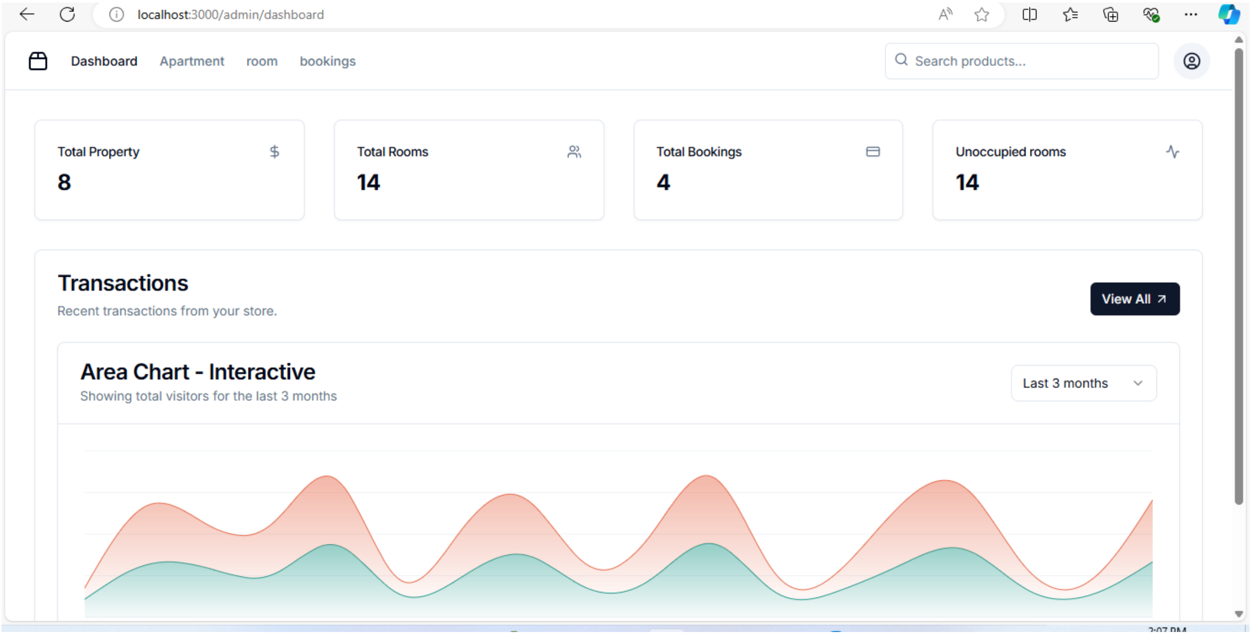
Open a Model

Search

All Models

Booking	0
Images	0
Payment	0
Preference	0
Property	0
Room	0
Session	0
User	0

Figure 4.2: Model page



localhost:3000/admin/dashboard

Dashboard Apartment room bookings

Search products...

Total Property	Total Rooms	Total Bookings	Unoccupied rooms
8	14	4	14

Transactions

Recent transactions from your store.

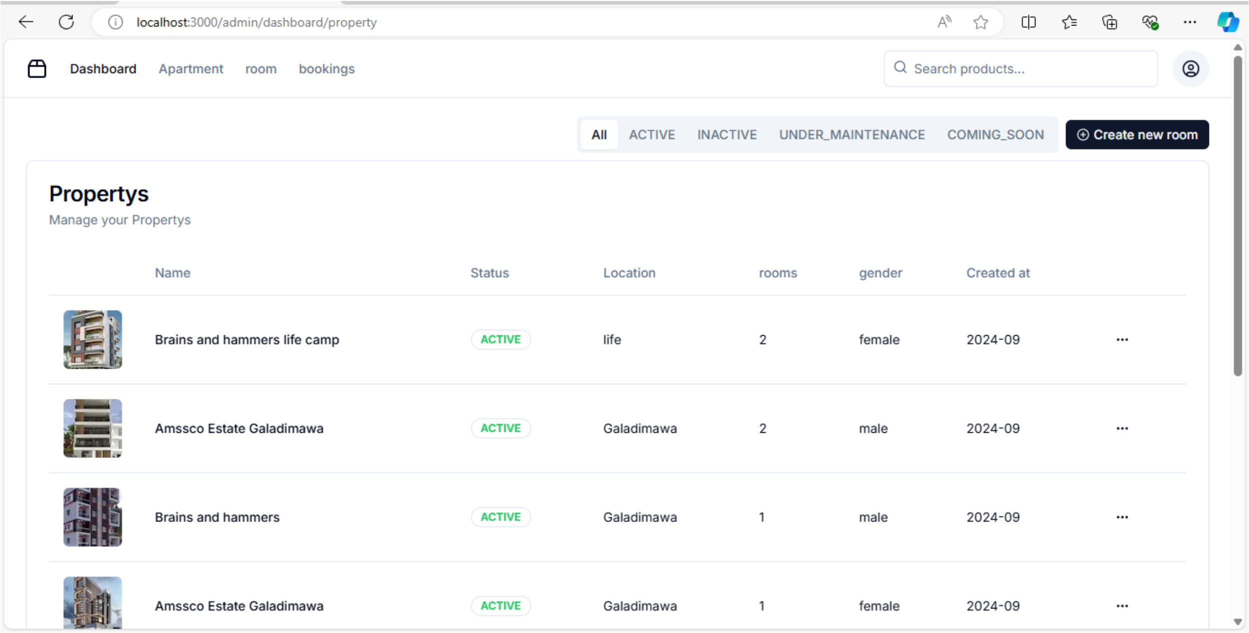
View All ↗

Area Chart - Interactive

Showing total visitors for the last 3 months

Last 3 months

Figure 4.3: Dashboard

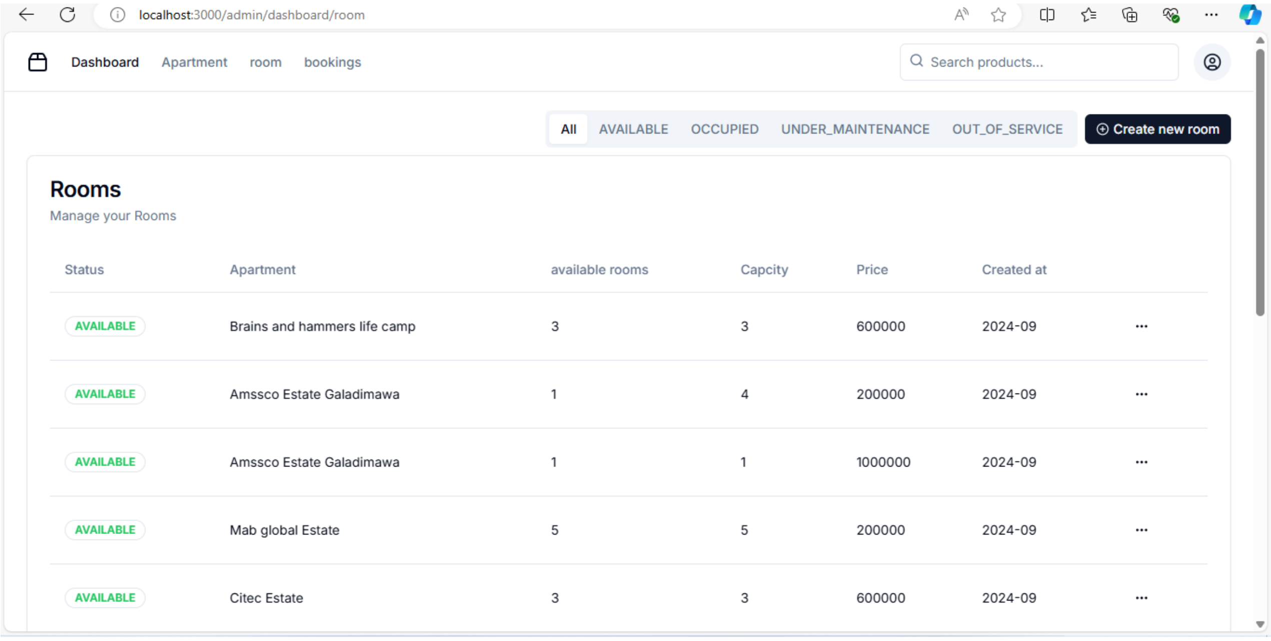


Propertys

Manage your Propertys

	Name	Status	Location	rooms	gender	Created at	
	Brains and hammers life camp	ACTIVE	life	2	female	2024-09	···
	Amssco Estate Galadimawa	ACTIVE	Galadimawa	2	male	2024-09	···
	Brains and hammers	ACTIVE	Galadimawa	1	male	2024-09	···
	Amssco Estate Galadimawa	ACTIVE	Galadimawa	1	female	2024-09	···

Figure 4.4: Apartment page

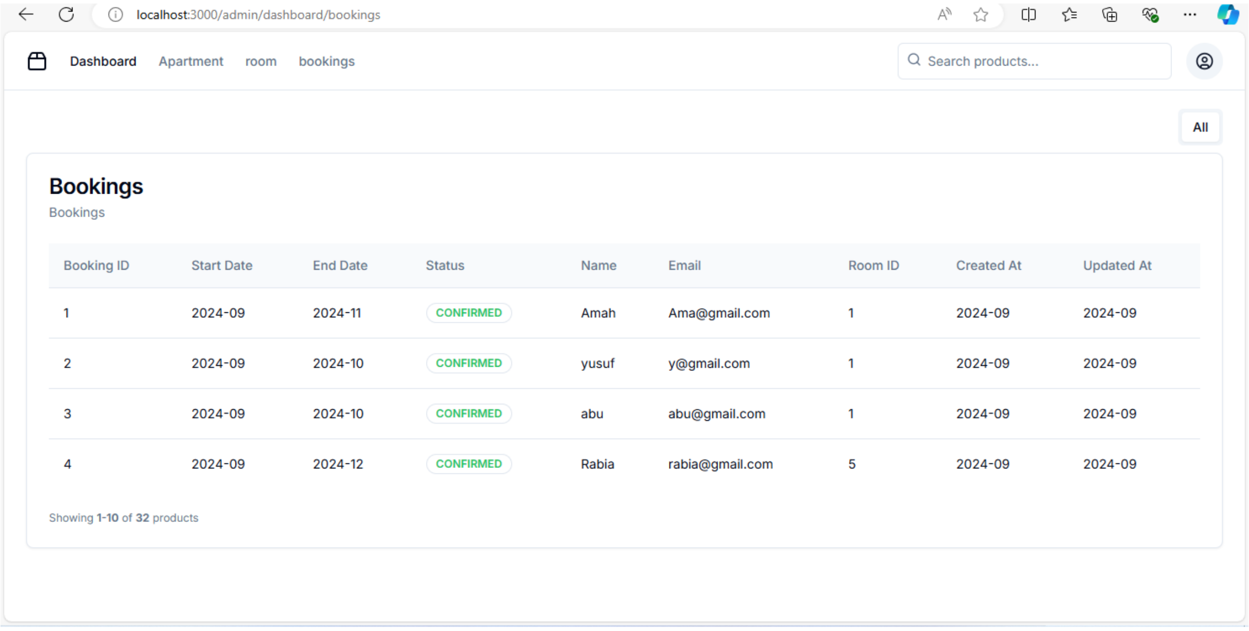


Rooms

Manage your Rooms

Status	Apartment	available rooms	Capcity	Price	Created at	
AVAILABLE	Brains and hammers life camp	3	3	600000	2024-09	···
AVAILABLE	Amssco Estate Galadimawa	1	4	200000	2024-09	···
AVAILABLE	Amssco Estate Galadimawa	1	1	1000000	2024-09	···
AVAILABLE	Mab global Estate	5	5	200000	2024-09	···
AVAILABLE	Citec Estate	3	3	600000	2024-09	···

Figure 4.5: Room page



Bookings

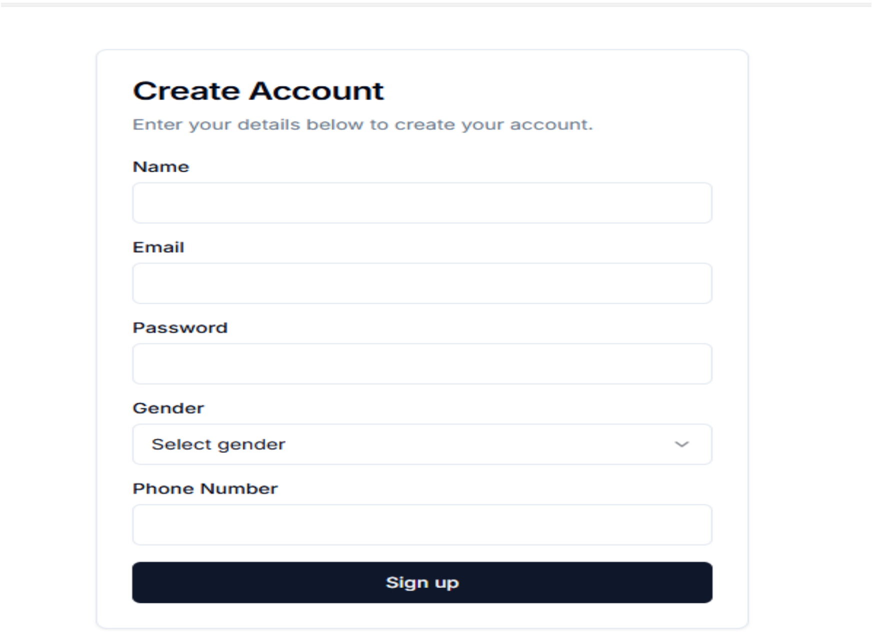
Bookings

Booking ID	Start Date	End Date	Status	Name	Email	Room ID	Created At	Updated At
1	2024-09	2024-11	CONFIRMED	Amah	Ama@gmail.com	1	2024-09	2024-09
2	2024-09	2024-10	CONFIRMED	yusuf	y@gmail.com	1	2024-09	2024-09
3	2024-09	2024-10	CONFIRMED	abu	abu@gmail.com	1	2024-09	2024-09
4	2024-09	2024-12	CONFIRMED	Rabia	rabia@gmail.com	5	2024-09	2024-09

Showing **1-10** of **32** products

Figure 4.6: Booking/payment page

User's View



Create Account

Enter your details below to create your account.

Name

Email

Password

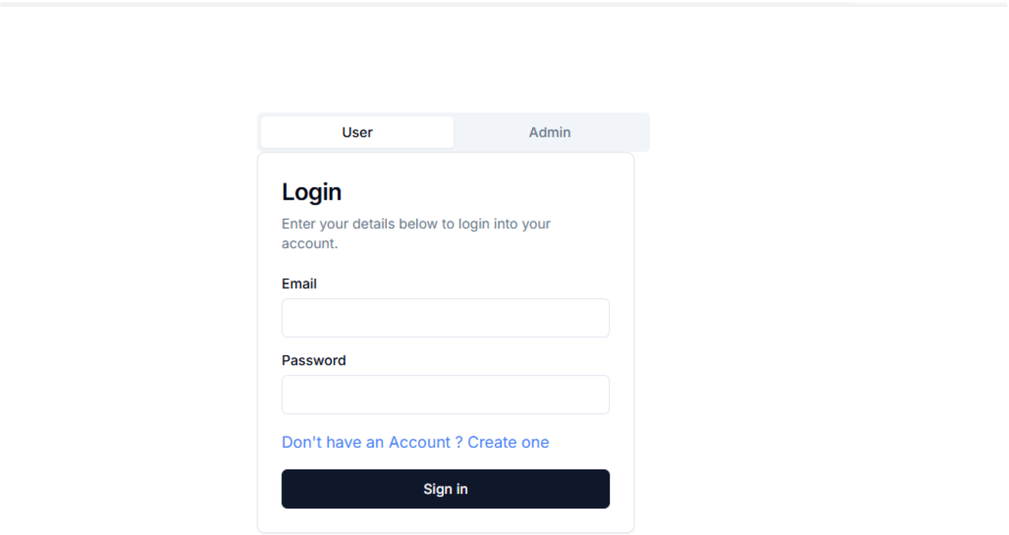
Gender

Select gender

Phone Number

Sign up

Figure 4.7: Sign up page(user)



User

Admin

Login

Enter your details below to login into your account.

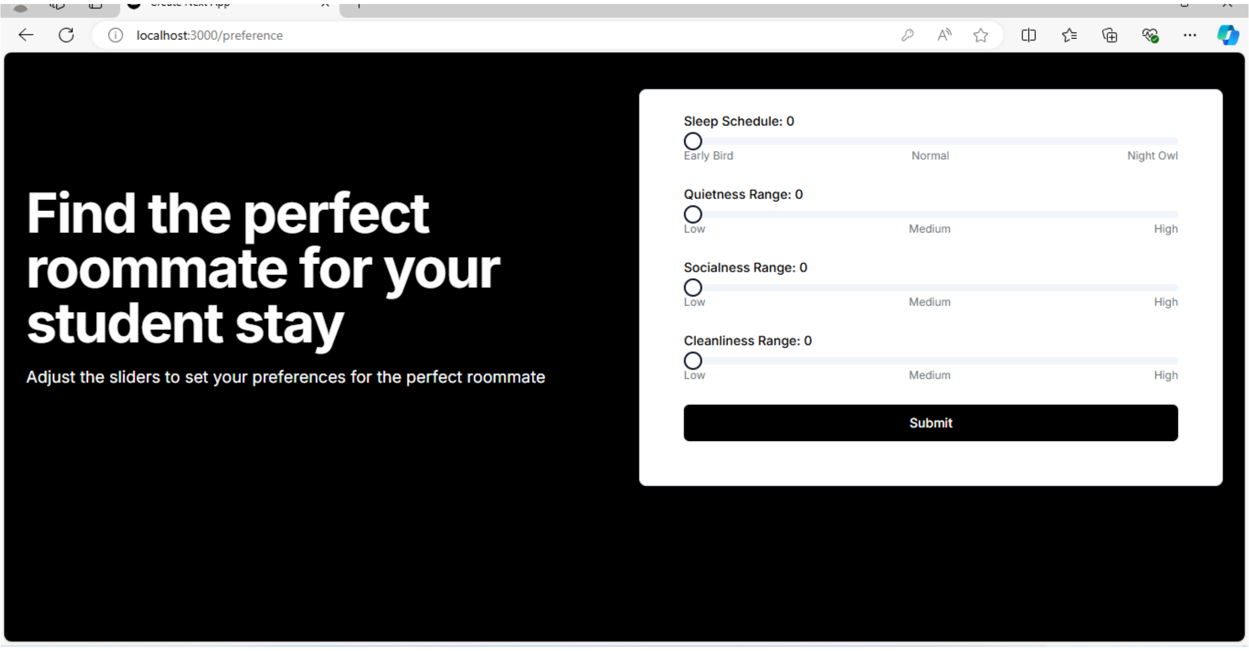
Email

Password

Don't have an Account ? Create one

Sign in

Figure 4.8: Login page(user)



Find the perfect roommate for your student stay

Adjust the sliders to set your preferences for the perfect roommate

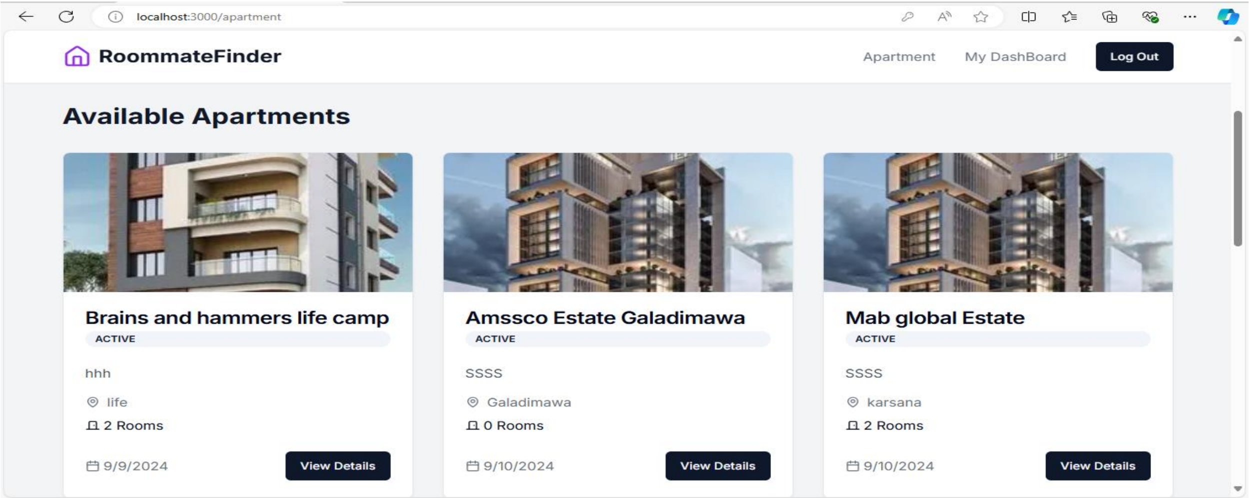
Sleep Schedule: 0

Quietness Range: 0

Socialness Range: 0

Cleanliness Range: 0

Figure 4.9: Preference page



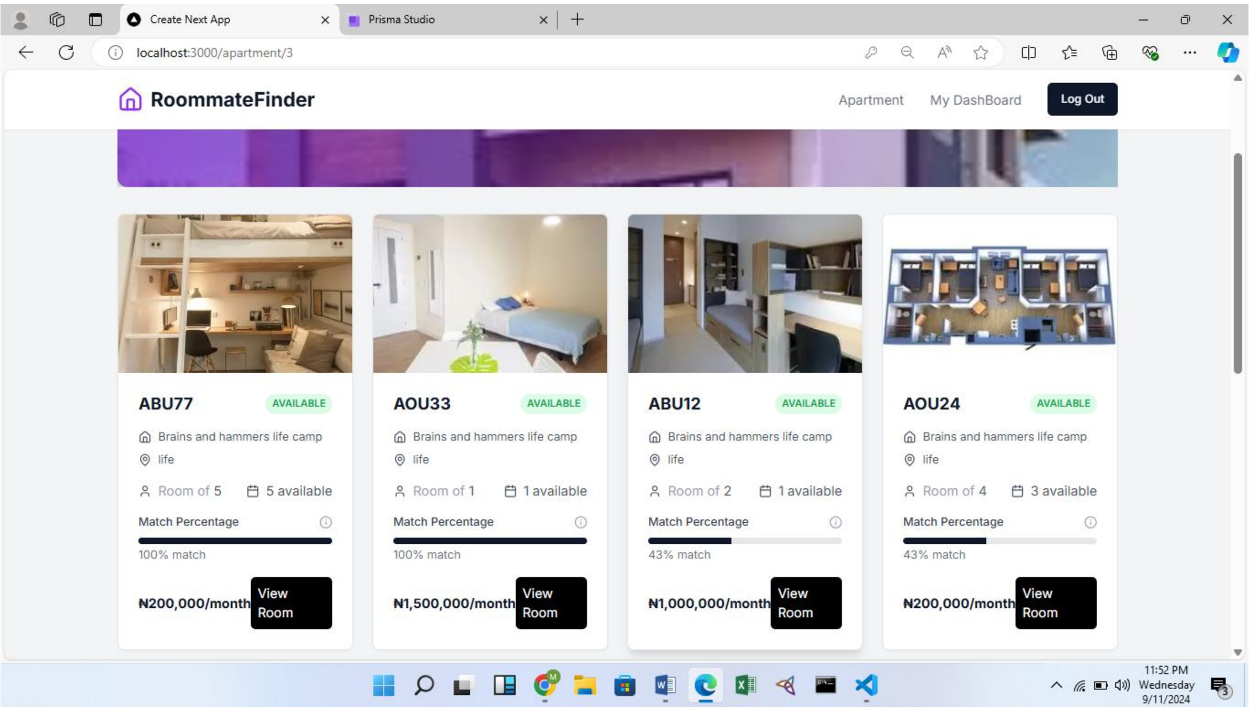
Available Apartments

Brains and hammers life camp
ACTIVE
hhh
life
2 Rooms
9/9/2024

Amssco Estate Galadimawa
ACTIVE
SSSS
Galadimawa
0 Rooms
9/10/2024

Mab global Estate
ACTIVE
SSSS
karsana
2 Rooms
9/10/2024

Figure 4.10: Apartment page



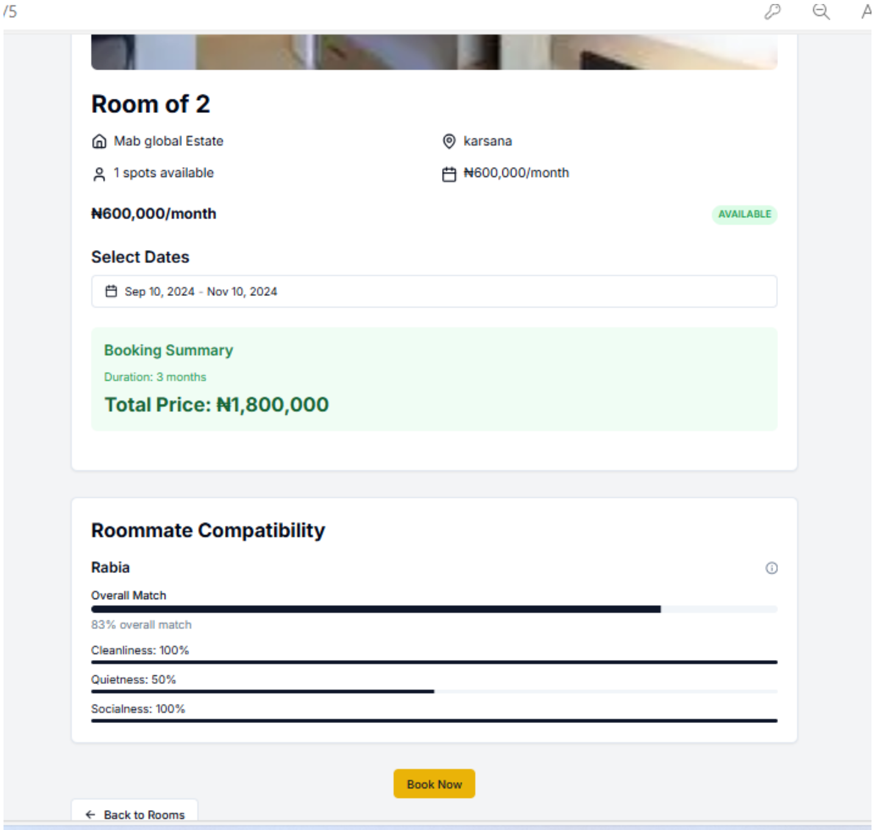
ABU77 AVAILABLE
Brains and hammers life camp
life
Room of 5 5 available
Match Percentage
100% match
₦200,000/month

AOU33 AVAILABLE
Brains and hammers life camp
life
Room of 1 1 available
Match Percentage
100% match
₦1,500,000/month

ABU12 AVAILABLE
Brains and hammers life camp
life
Room of 2 1 available
Match Percentage
43% match
₦1,000,000/month

AOU24 AVAILABLE
Brains and hammers life camp
life
Room of 4 3 available
Match Percentage
43% match
₦200,000/month

Figure 4.11: Room page



Room of 2

Mab global Estate

karsana

1 spots available

₦600,000/month

₦600,000/month

AVAILABLE

Select Dates

Sep 10, 2024 - Nov 10, 2024

Booking Summary

Duration: 3 months

Total Price: ₦1,800,000

Roommate Compatibility

Rabia

Overall Match

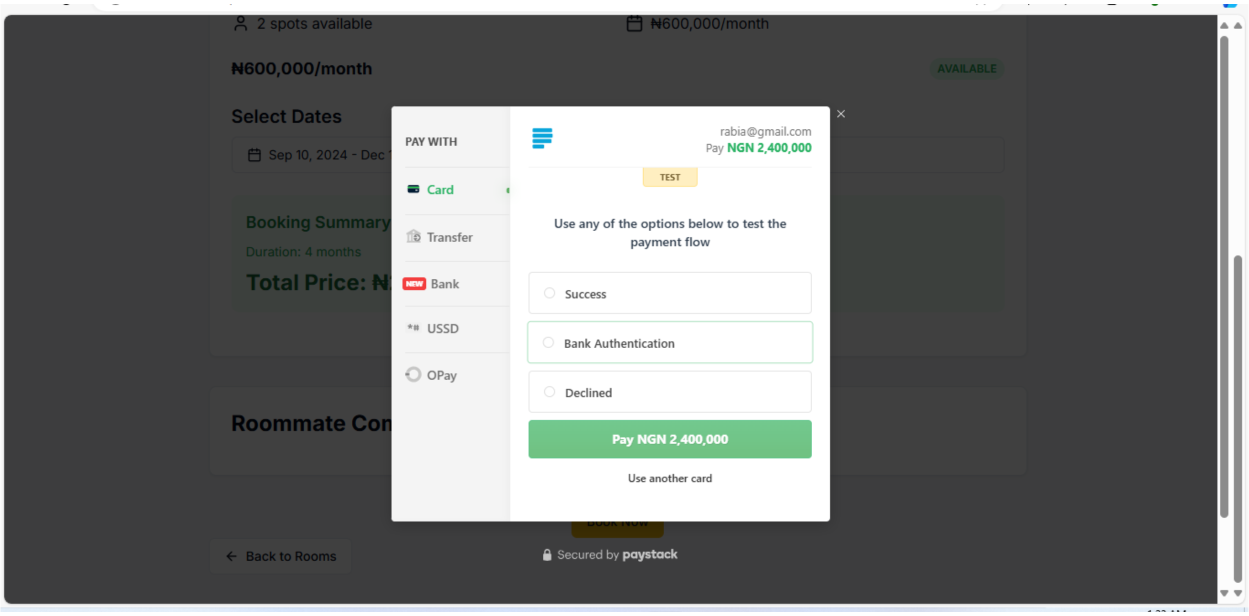
83% overall match

Cleanliness: 100%

Quietness: 50%

Socialness: 100%

Figure 4.12: Booking Page



2 spots available

₦600,000/month

₦600,000/month

AVAILABLE

Select Dates

Sep 10, 2024 - Dec

Booking Summary

Duration: 4 months

Total Price: ₦

Roommate Com

rabia@gmail.com

Pay **NGN 2,400,000**

TEST

Use any of the options below to test the payment flow

○ Success

○ Bank Authentication

○ Declined

Figure 4.13: Payment page

CHAPTER 5

DISCUSSION, CONCLUSION AND RECOMMENDATION

5.1 Overview

This concluding chapter provides a comprehensive reflection on the implementation of the student accommodation using preference Matching. It evaluates the achievements relative to the initial objectives of the project. Additionally, it highlights the main challenges encountered and the system's limitations. The chapter concludes with recommendations for future enhancements, along with final observations.

5.2 Objective Assessment

1. Automating the process of assigning accommodations to students.
2. Providing personalized housing suggestions based on each student's needs and preferences.
3. Bringing together all accommodation options in one place for easier browsing.
4. Enhancing the accuracy of housing recommendations through data analysis is important. The system built during this project effectively matches students' preferences with available housing options. This has significantly reduced the manual work required to manage student accommodation requests, making the process easier for both students and staff.

5.3 Limitations and Challenges

1. The mobile interface for the system is underdeveloped, leading to usability issues on smaller devices.
2. The recommendation engine requires more extensive data sources to improve its accuracy.
3. Customizable reports for staff and administrators are limited and require additional technical support.

4. Student self-service functionalities such as payment integration and room change requests remain limited.
5. The project has accumulated technical debt due to rapid development phases, and this will require dedicated resources for optimization, security, and future-proofing. Additionally, there are concerns regarding long-term maintenance and support, which will require a well-trained team to handle future updates.

5.4 Future Enhancements

1. Build mobile apps for iOS and Android to make the system easier to use.
2. Improve the recommendation engine by using more data sources, including outside housing databases.
3. Make reporting easier for staff by creating a simple tool that does not require technical help.

5.5 Recommendations

1. Set a reminder for renewal of payments
2. Set up a plan for feedback from users to keep improving the system.

5.6 Summary

The Student Accommodation System (Preference matching) is a big step toward making the process more efficient and tailored for students. However, to fully benefit from the system, ongoing support from leadership and stakeholders is needed. With regular updates, proper resources, and smart improvements, the system can boost student satisfaction and improve how things are run. By using modern technology, the institution can become a leader in student housing, setting the stage for future success.

REFERENCES

Aggarwal, C. C. (2016). *Recommender systems*. Springer International Publishing.

Breese, J. S., Heckerman, D., & Kadie, C. (1998). *Empirical analysis of predictive algorithms for collaborative filtering* (Technical Report MSR-TR-98-12). Microsoft Research.

Basavesh, D., Laharishree, S., Sthuthi, S., Tejaswini, N., & Vidya, R. (2023). Hostel finder: Location-based recommendation system for hostels and PGS with transit information. *International Journal of Research Publication and Reviews*. https://www.ijrpr.com

Burke, R. (2001). Knowledge-based recommender systems. In *Encyclopedia of library and information science*.

Burke, R. (2002). Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction, 12*(4), 331-370.

Carpenter, S. R., Pingali, P. L., Bennett, E. M., & Zurek, M. B. (Eds.). (2005). *Ecosystems and human well-being: Scenarios, volume 2*. Island Press.

Daly, E. M., Botea, A., Kishimoto, A., & Marinescu, R. (2014). Multi-criteria journey aware housing recommender system. In *Proceedings of the 8th ACM Conference on Recommender Systems* (pp. 325-328). ACM.

Desai, U. (2023, April 27). Recommendation systems explained: Understanding the basic to advance. *Medium*. https://utsavdesai26.medium.com/Apr 27, 2023

Ikejiaku, B. V. (2009). The relationship between poverty, conflict and development. *Journal of Sustainable Development, 2*(1), 15. https://doi.org/10.5539/jsd.v2n1p15

Isinkaye, F., Folajimi, Y., & Ojokoh, B. (2015). Recommendation systems: Principles, methods and evaluation. *Egyptian Informatics Journal, 16*, 100-105. https://doi.org/10.1016/j.eij.2015.06.005

Jain, A., & Gupta, C. (2018). Fuzzy logic in recommender systems. In *Fuzzy logic augmentation of neural and optimization algorithms: Theoretical aspects and real applications* (pp. 255-273). Springer.

Lorenzi, F., Fontanella, B., Prestes, E., & Peres, A. (2014). How to improve multi-agent recommendations using data from social networks? In *FLAIRS Conference*.

Omisore, M. O., & Samuel, O. W. (2014). Personalized recommender system for digital libraries. *International Journal of Web-Based Learning and Teaching Technologies*, *9*(1), 18-32. https://doi.org/10.4018/IJWLTT.2014010102

Omisore, M. O., Samuel, O. W., & Ogunniyi, T. O. (2013). A multi-technique approach for recommender systems. *International Journal of Advanced Trends in Computer Science and Engineering*, *2*(5), 112-118. https://doi.org/10.30534/ijatcse/2013/02252013

Ojokoh, B., Omisore, M., Samuel, O., & Ogunniyi, T. (2012). A fuzzy logic based personalized recommender system. *International Journal of Computer Science and Information Technology and Security*, *2*(5), 1008-1015.

Osgood, C. E., Suci, G. J., & Tannenbaum, P. H. (1957). *The measurement of meaning*. University of Illinois Press.

Pizzato, L., Rej, T., Chung, T., Koprinska, I., & Kay, J. (2010). RECON: A reciprocal recommender for online dating. In *Proceedings of the Fourth ACM Conference on Recommender Systems* (pp. 207-214). ACM.

Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th International World Wide Web Conference* (pp. 285-295). ACM.

Schafer, J. B., Konstan, J. A., & Riedl, J. (1999). Recommender systems in e-commerce. In *Proceedings of the 1st ACM Conference on Electronic Commerce* (pp. 158-166). ACM.

Schafer, J. B., Konstan, J. A., & Riedl, J. (2002). Meta-recommendation systems: User-controlled integration of diverse recommendations. In *Proceedings of the Eleventh International Conference on Information and Knowledge Management* (pp. 43-51). ACM.

Shanmuganathan, M., & Karthikeyan, R. (2016). A web-based recommendation system for housing selection: Design, implementation & evaluation. *Data Mining and Knowledge Engineering*, *8*(5), 134-137. https://doi.org/10.11648/j.dmke.2016.08.05.14

Terveen, L., Hill, W., Amento, B., McDonald, D., & Creter, J. (1997). PHOAKS: A system for sharing recommendations. *Communications of the ACM*, *40*(3), 59-62. https://doi.org/10.1145/253610.253623

Yuan, X., Lee, J. H., Kim, S. J., & Kim, Y. H. (2013). Toward a user-oriented recommendation system for real estate websites. *Information Systems*, *38*(2), 231-243. https://doi.org/10.1016/j.is.2012.10.003

Zain, J., & Khurshed, T. (2020). *Hostel recommender system* [Unpublished final year project]. Department of Computer Science and IT, Bahria University Lahore Campus.

APPENDICES

Appendix A – Project Document

IN-DEPTH PROJECT DOCUMENTATION

Full Candidate Name: Maryam Abba Yusuf

Student ID: BU/22B/IT/6965

Title: Student Accommodation System (Preference Matching)

Course of Study: B.Sc. Software Engineering

Background and Motivation

Student accommodation is a critical aspect of university life, ensuring that students have safe, affordable, and convenient places to live while pursuing their studies. In Abuja, the capital city of Nigeria, the demand for student housing has surged with the growing number of higher education institutions and students. Options range from university to private student housing complexes, shared apartments. However, finding the right accommodation that meets the diverse needs and preferences of students remains a challenge in Abuja's dynamic real estate market.

Initially, the process of finding student accommodation was manual, relying on physical visits, word of mouth, and simple listings. This approach was time-consuming, and often resulted in mismatches between students' needs and available housing options (Johnson, 2019). As student populations grew and the variety of housing options expanded, these manual processes became increasingly inadequate.

The system developed for this project leverages modern algorithms to match students based on their stated preferences. This approach is both personalized and data-driven, ensuring that users are matched with roommates and rooms that align with their living style.

The main goal of this system is to improve the student living experience by minimizing potential conflicts between roommates and promoting positive interactions. Students can specify their detailed preferences, which the system uses to suggest compatible rooms and roommates. This process is efficient, benefiting both students and accommodation managers by saving time and effort.

Additionally, the system gives students a sense of empowerment, knowing their living preferences are considered, which can result in higher satisfaction and increased retention for housing providers.

Statement of the Problem

Students often struggle to find suitable accommodation due to limited knowledge of available options and difficulty matching their preferences with available housing and roommates. This project addresses these challenges by proposing a **Student Accommodation System Using Preference Matching**, which helps students find accommodation that aligns with their specific needs, simplifying the search process and improving satisfaction

Appendix B – Interview

I conducted an informal interview with some of the students.

Questions asked are:

How do you currently search for student accommodation?

How would you describe your experience in finding accommodation as a student?

What factors do you prioritize when looking for student accommodation?

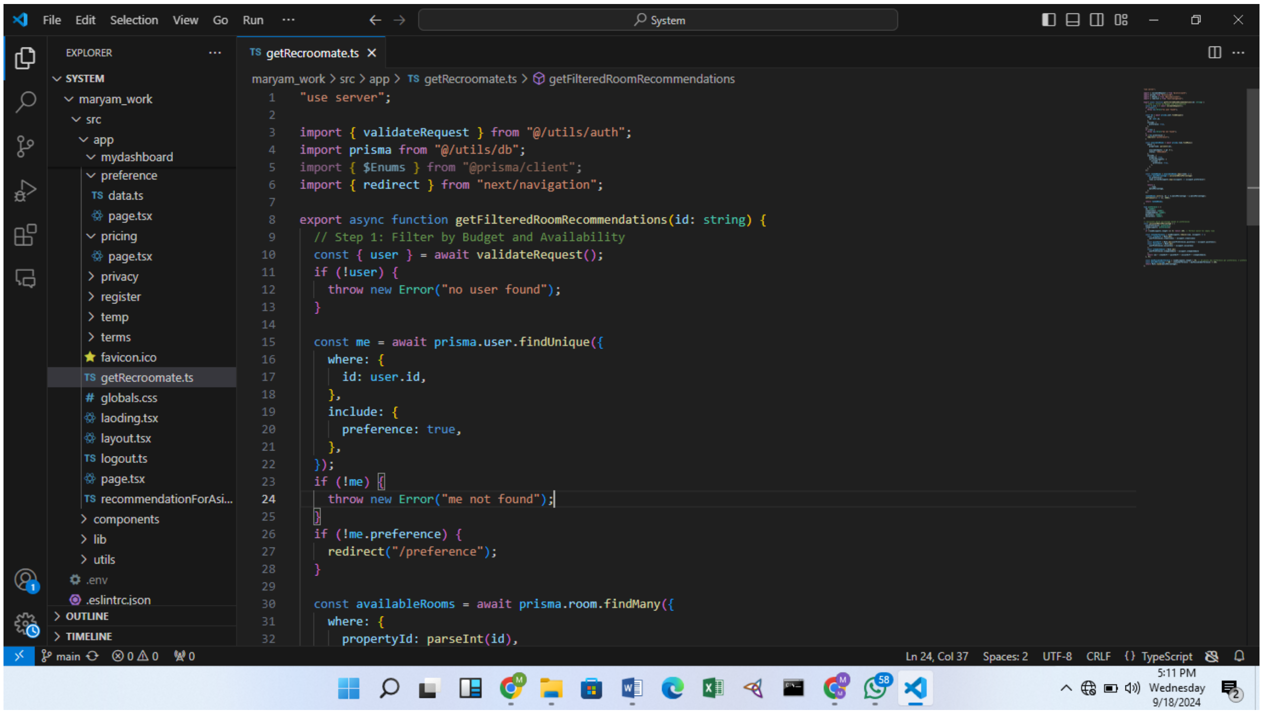
Would you find it useful if a system recommended accommodation based on your preferences?

What features would you like to see in a recommendation system for student accommodation?

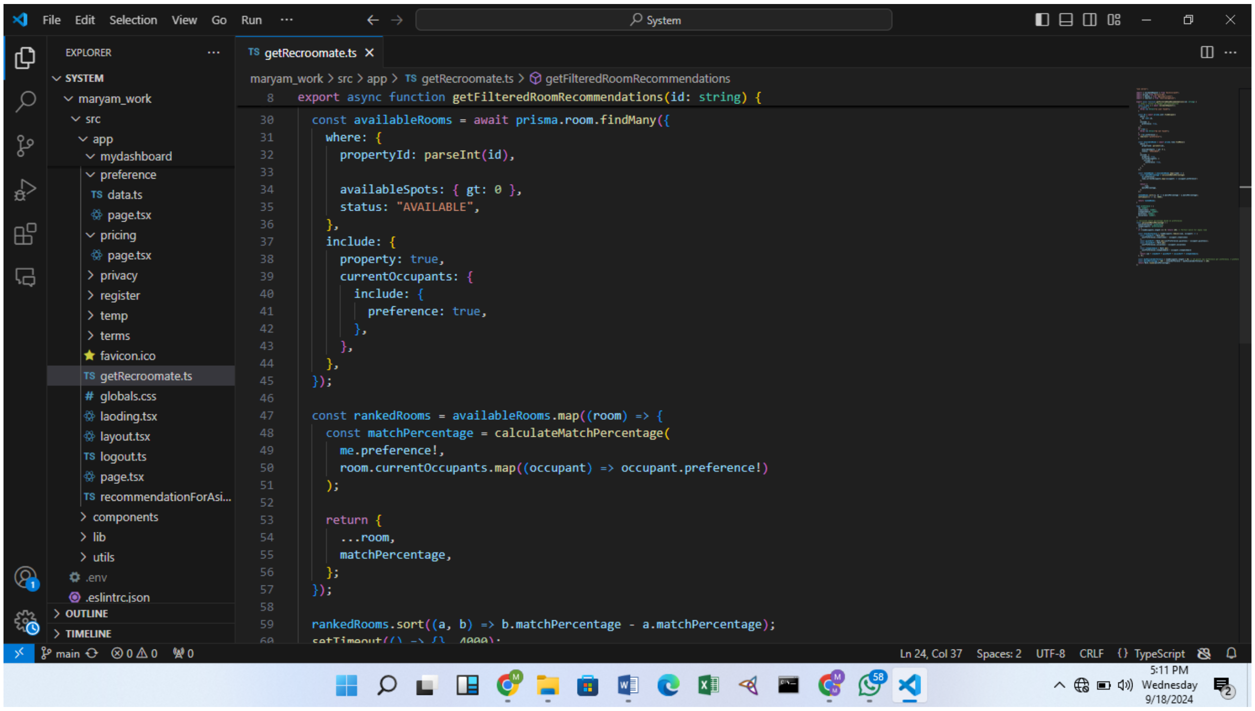
What are the common deal-breakers for you when selecting accommodation?

what could be improved in the current methods of searching for student accommodation?

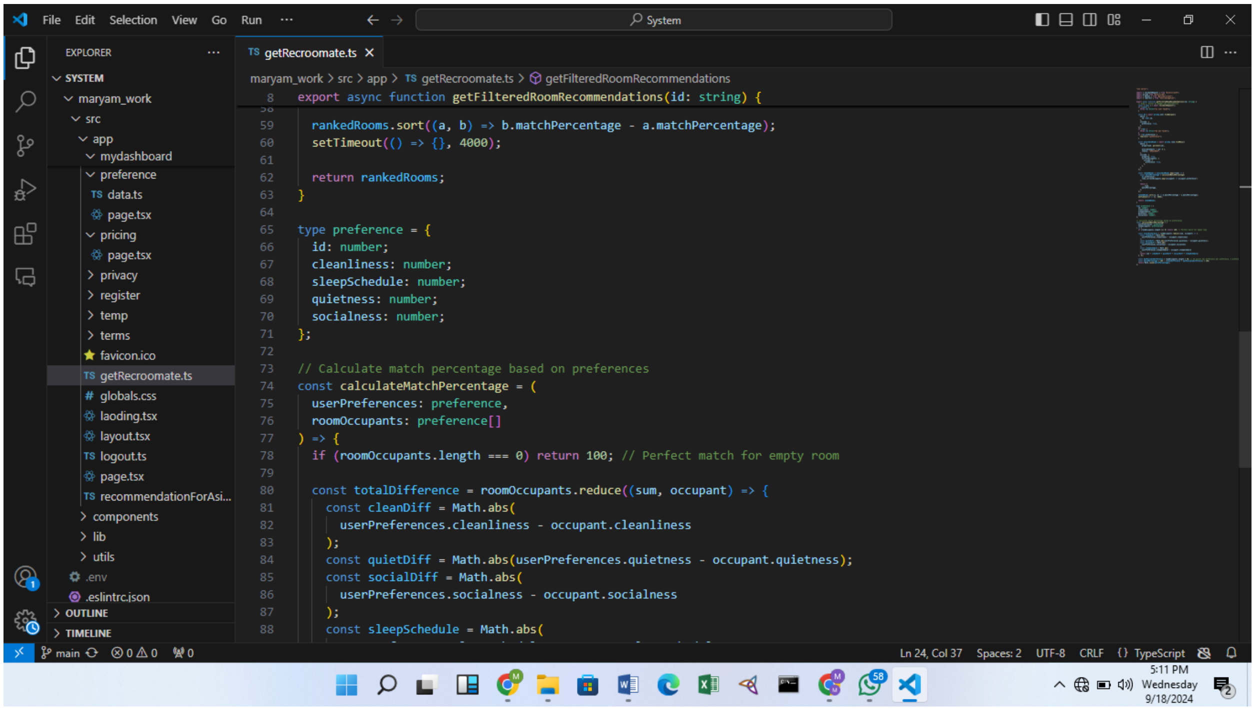
Appendix C – Source Codes



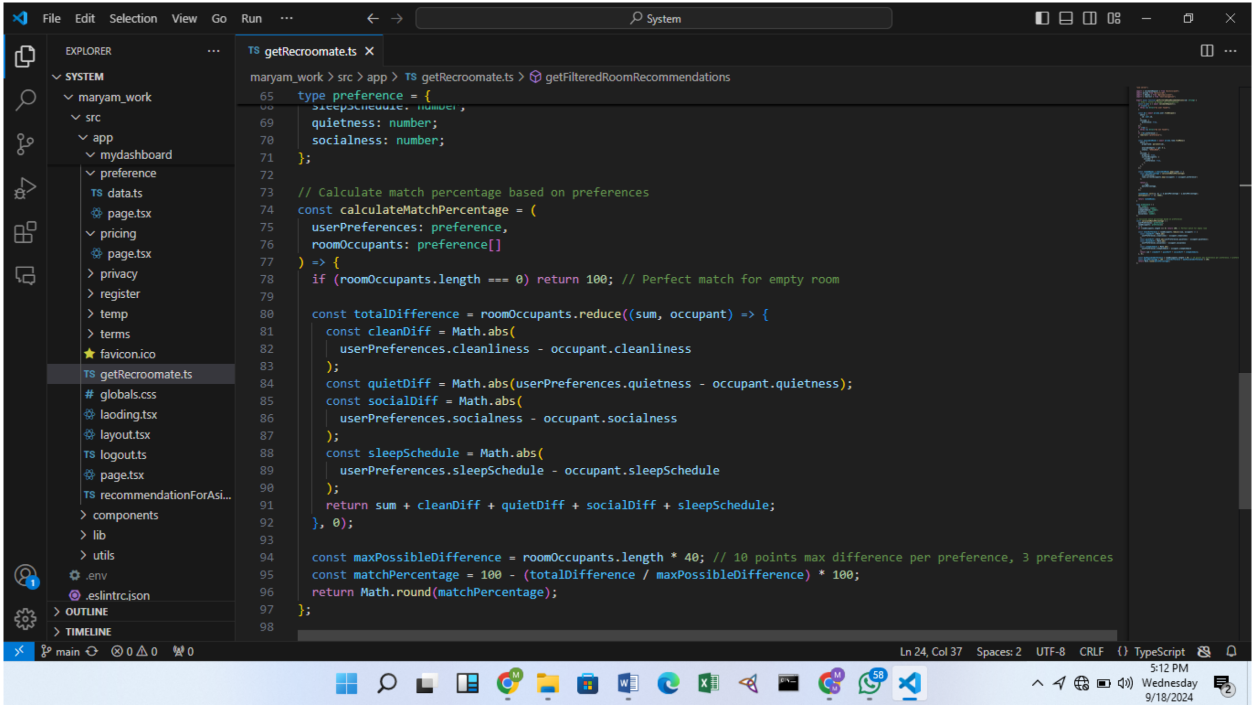
```
"use server";

import { validateRequest } from "@/utils/auth";
import prisma from "@/utils/db";
import { $Enums } from "@prisma/client";
import { redirect } from "next/navigation";

export async function getFilteredRoomRecommendations(id: string) {
  // Step 1: Filter by Budget and Availability
  const { user } = await validateRequest();
  if (!user) {
    throw new Error("no user found");
  }

  const me = await prisma.user.findUnique({
    where: {
      id: user.id,
    },
    include: {
      preference: true,
    },
  });
  if (!me) {
    throw new Error("me not found");
  }
  if (!me.preference) {
    redirect("/preference");
  }

  const availableRooms = await prisma.room.findMany({
    where: {
      propertyId: parseInt(id),
```



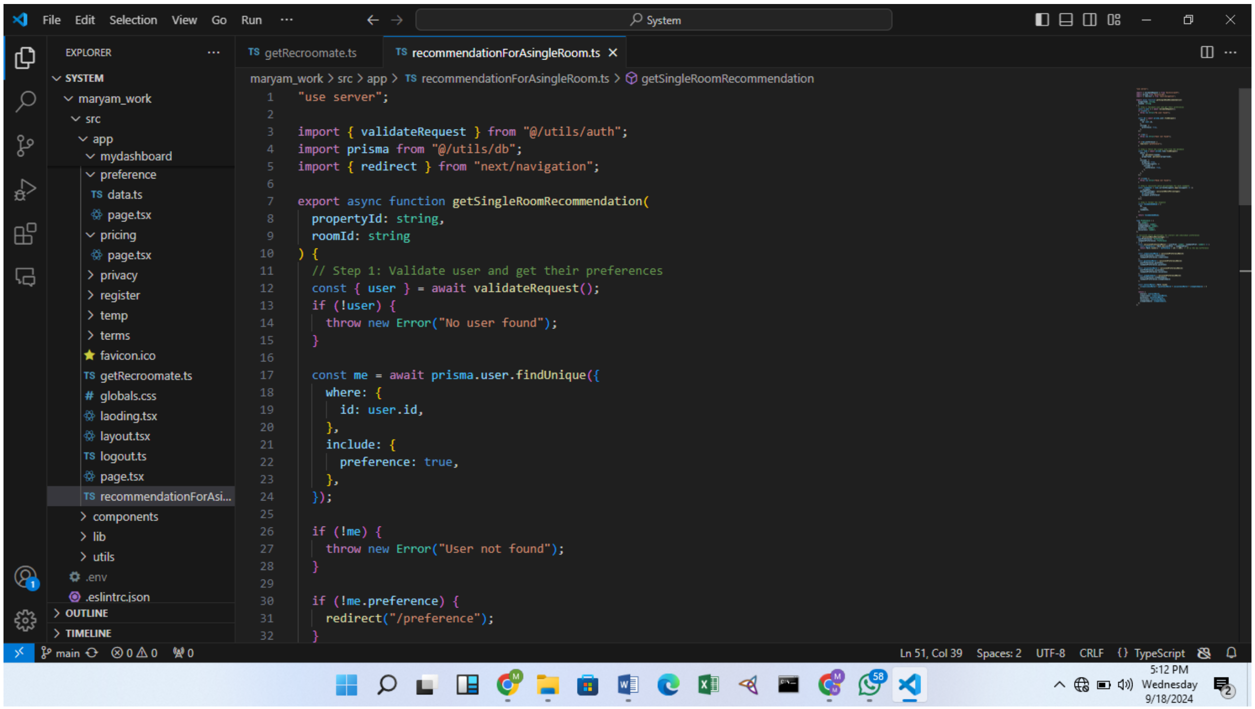
```
export async function getFilteredRoomRecommendations(id: string) {
  const availableRooms = await prisma.room.findMany({
    where: {
      propertyId: parseInt(id),

      availableSpots: { gt: 0 },
      status: "AVAILABLE",
    },
    include: {
      property: true,
      currentOccupants: {
        include: {
          preference: true,
        },
      },
    },
  });

  const rankedRooms = availableRooms.map((room) => {
    const matchPercentage = calculateMatchPercentage(
      me.preference!,
      room.currentOccupants.map((occupant) => occupant.preference!)
    );

    return {
      ...room,
      matchPercentage,
    };
  });

  rankedRooms.sort((a, b) => b.matchPercentage - a.matchPercentage);
  setTimeout(() => {}, 4000);
```



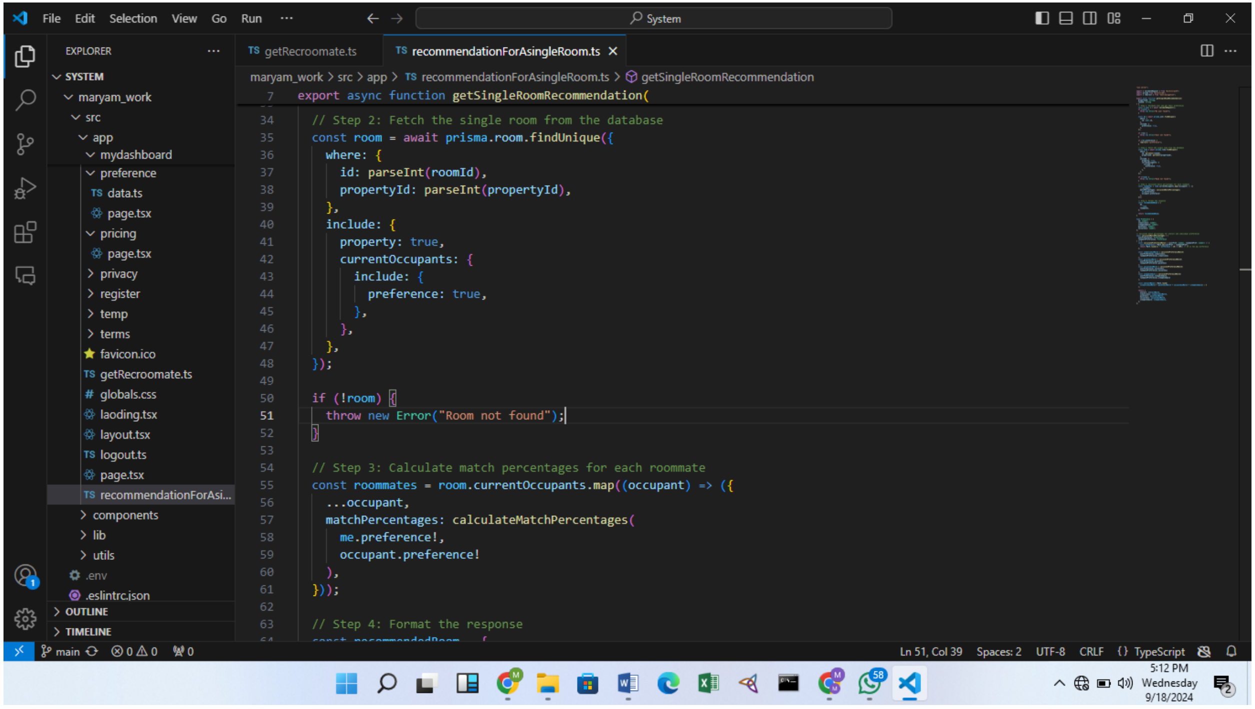
```
export async function getFilteredRoomRecommendations(id: string) {
  rankedRooms.sort((a, b) => b.matchPercentage - a.matchPercentage);
  setTimeout(() => {}, 4000);

  return rankedRooms;
}

type preference = {
  id: number;
  cleanliness: number;
  sleepSchedule: number;
  quietness: number;
  socialness: number;
};

// Calculate match percentage based on preferences
const calculateMatchPercentage = (
  userPreferences: preference,
  roomOccupants: preference[]
) => {
  if (roomOccupants.length === 0) return 100; // Perfect match for empty room

  const totalDifference = roomOccupants.reduce((sum, occupant) => {
    const cleanDiff = Math.abs(
      userPreferences.cleanliness - occupant.cleanliness
    );
    const quietDiff = Math.abs(userPreferences.quietness - occupant.quietness);
    const socialDiff = Math.abs(
      userPreferences.socialness - occupant.socialness
    );
    const sleepSchedule = Math.abs(
```



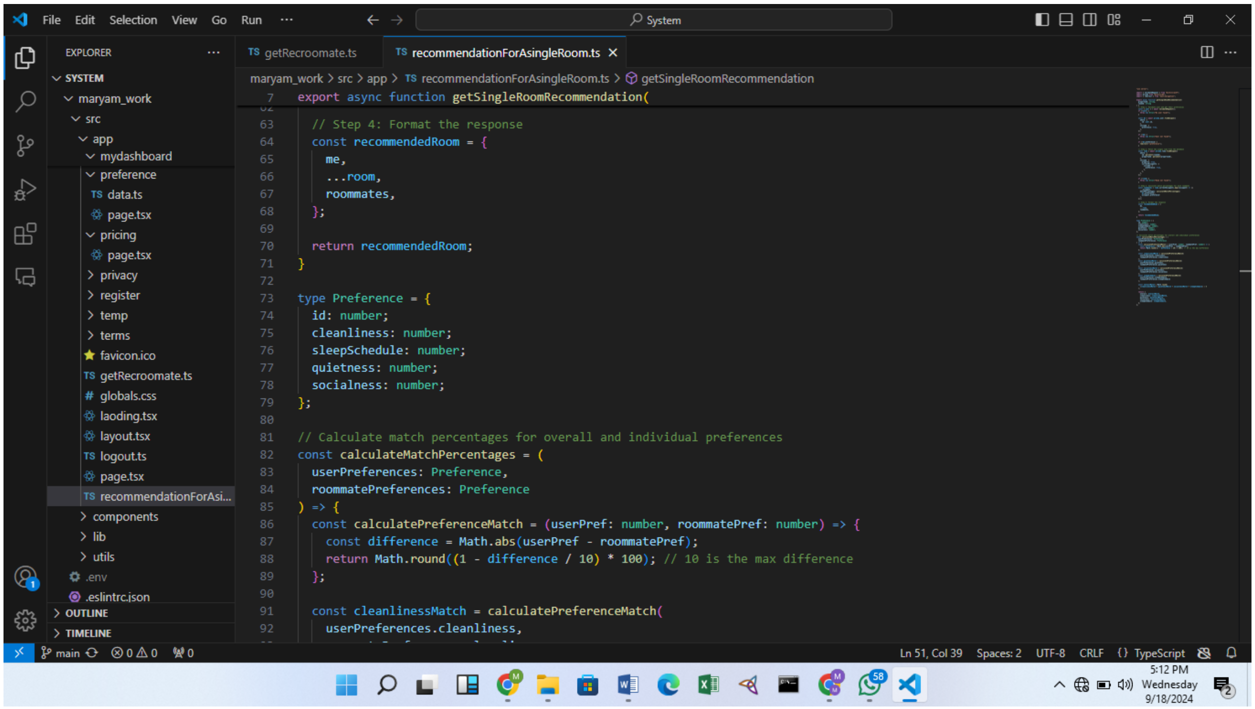
```
type preference = {
  quietness: number;
  socialness: number;
};

// Calculate match percentage based on preferences
const calculateMatchPercentage = (
  userPreferences: preference,
  roomOccupants: preference[]
) => {
  if (roomOccupants.length === 0) return 100; // Perfect match for empty room

  const totalDifference = roomOccupants.reduce((sum, occupant) => {
    const cleanDiff = Math.abs(
      userPreferences.cleanliness - occupant.cleanliness
    );
    const quietDiff = Math.abs(userPreferences.quietness - occupant.quietness);
    const socialDiff = Math.abs(
      userPreferences.socialness - occupant.socialness
    );
    const sleepSchedule = Math.abs(
      userPreferences.sleepSchedule - occupant.sleepSchedule
    );
    return sum + cleanDiff + quietDiff + socialDiff + sleepSchedule;
  }, 0);

  const maxPossibleDifference = roomOccupants.length * 40; // 10 points max difference per preference, 3 preferences
  const matchPercentage = 100 - (totalDifference / maxPossibleDifference) * 100;
  return Math.round(matchPercentage);
};
```



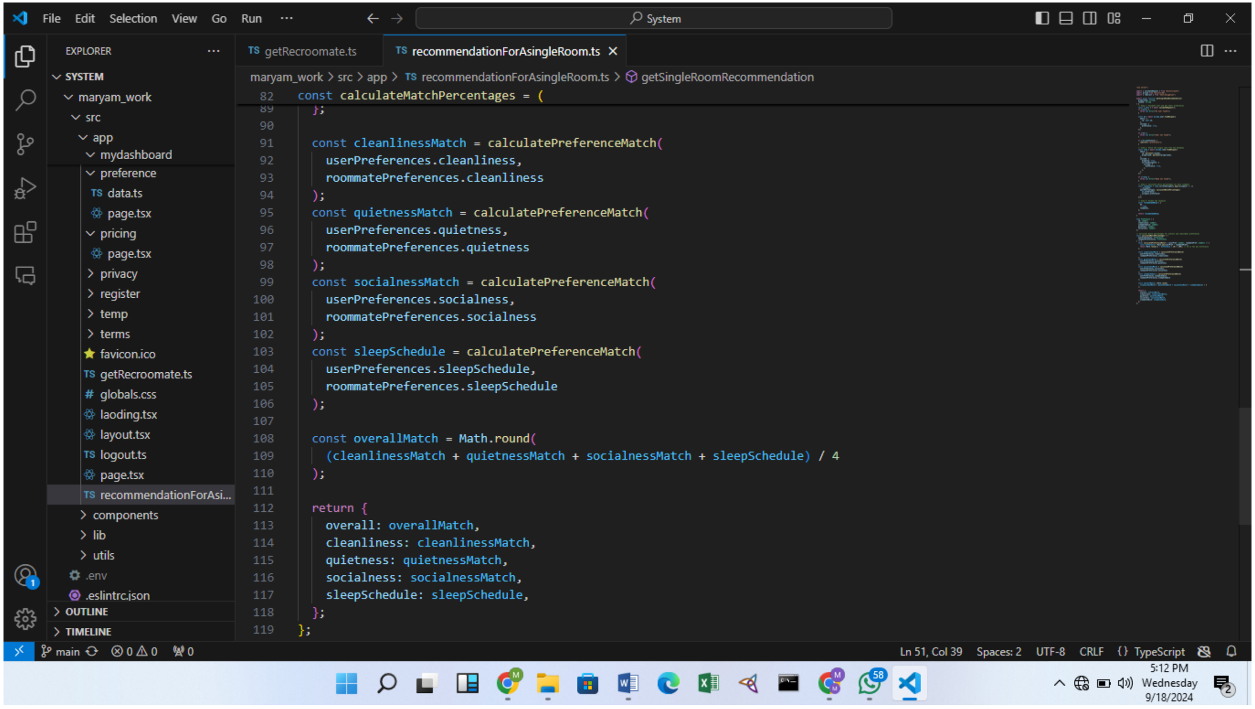
```
"use server";

import { validateRequest } from "@/utils/auth";
import prisma from "@/utils/db";
import { redirect } from "next/navigation";

export async function getSingleRoomRecommendation(
  propertyId: string,
  roomId: string
) {
  // Step 1: Validate user and get their preferences
  const { user } = await validateRequest();
  if (!user) {
    throw new Error("No user found");
  }

  const me = await prisma.user.findUnique({
    where: {
      id: user.id,
    },
    include: {
      preference: true,
    },
  });

  if (!me) {
    throw new Error("User not found");
  }

  if (!me.preference) {
    redirect("/preference");
  }
```



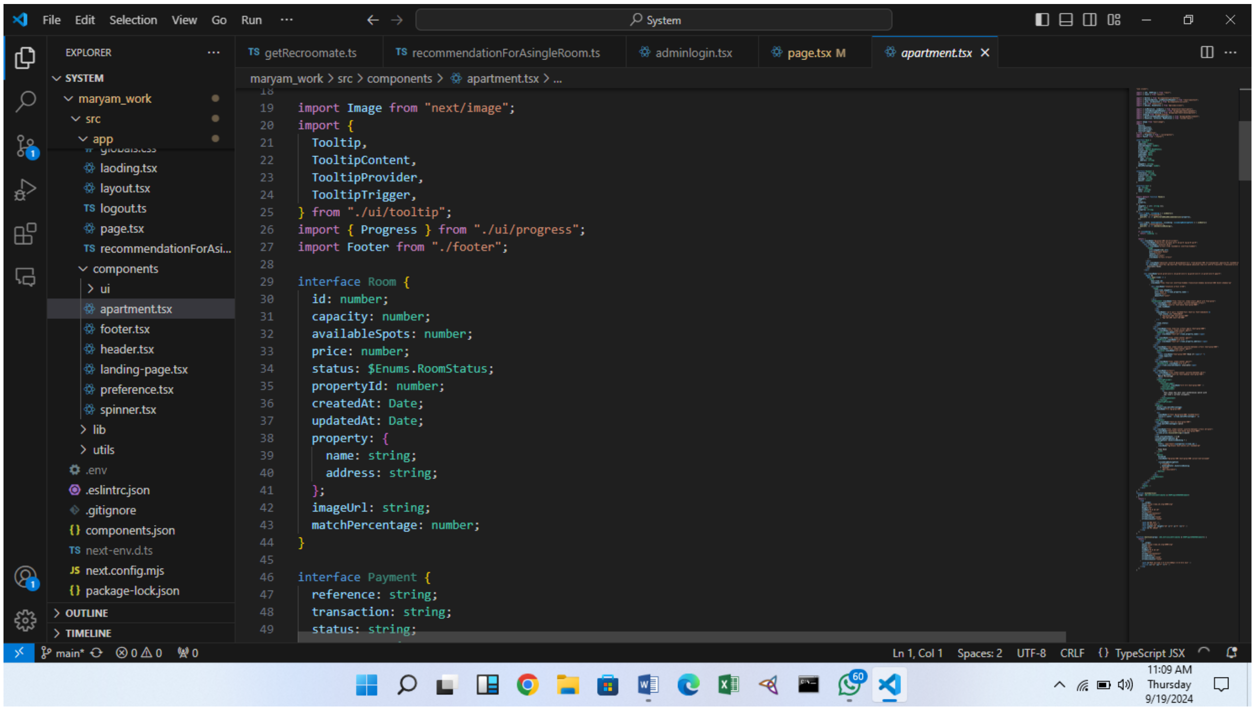
```
export async function getSingleRoomRecommendation(
  // Step 2: Fetch the single room from the database
  const room = await prisma.room.findUnique({
    where: {
      id: parseInt(roomId),
      propertyId: parseInt(propertyId),
    },
    include: {
      property: true,
      currentOccupants: {
        include: {
          preference: true,
        },
      },
    },
  });

  if (!room) {
    throw new Error("Room not found");
  }

  // Step 3: Calculate match percentages for each roommate
  const roommates = room.currentOccupants.map((occupant) => ({
    ...occupant,
    matchPercentages: calculateMatchPercentages(
      me.preference!,
      occupant.preference!
    ),
  }));

  // Step 4: Format the response
  const recommendedRoom = {
```



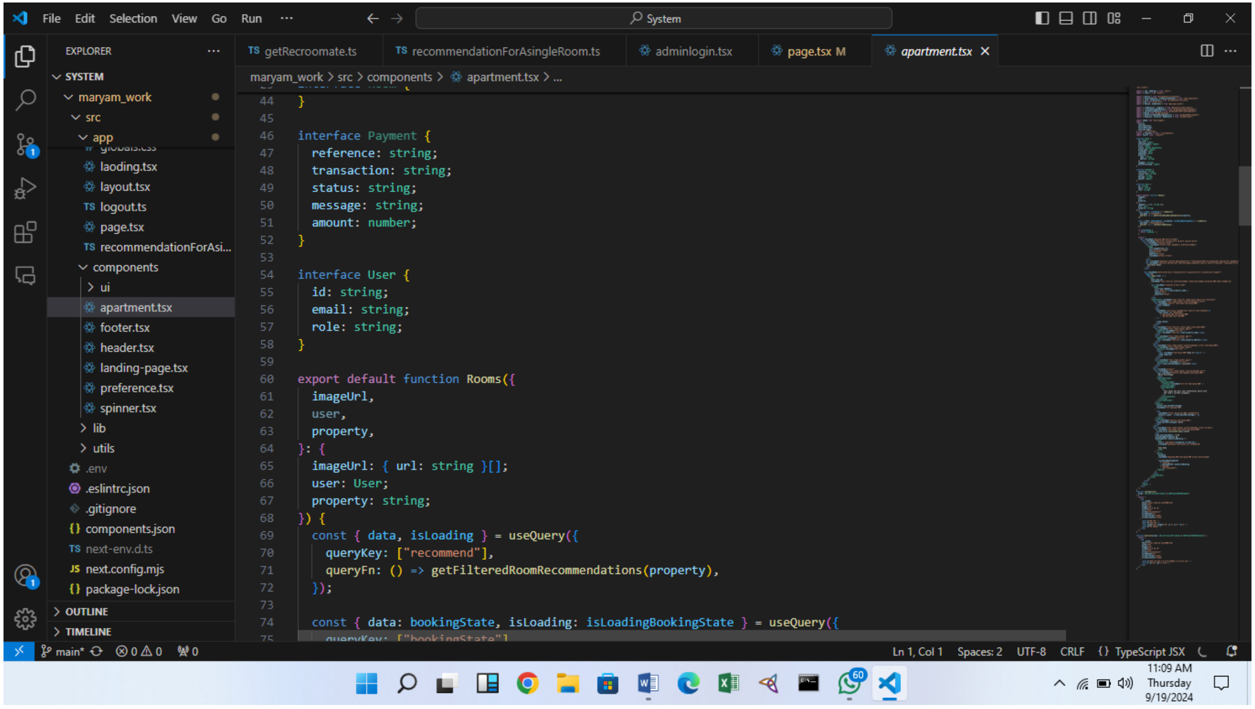
```
export async function getSingleRoomRecommendation(
  // Step 4: Format the response
  const recommendedRoom = {
    me,
    ...room,
    roommates,
  };

  return recommendedRoom;
}

type Preference = {
  id: number;
  cleanliness: number;
  sleepSchedule: number;
  quietness: number;
  socialness: number;
};

// Calculate match percentages for overall and individual preferences
const calculateMatchPercentages = (
  userPreferences: Preference,
  roommatePreferences: Preference
) => {
  const calculatePreferenceMatch = (userPref: number, roommatePref: number) => {
    const difference = Math.abs(userPref - roommatePref);
    return Math.round((1 - difference / 10) * 100); // 10 is the max difference
  };

  const cleanlinessMatch = calculatePreferenceMatch(
    userPreferences.cleanliness,
```



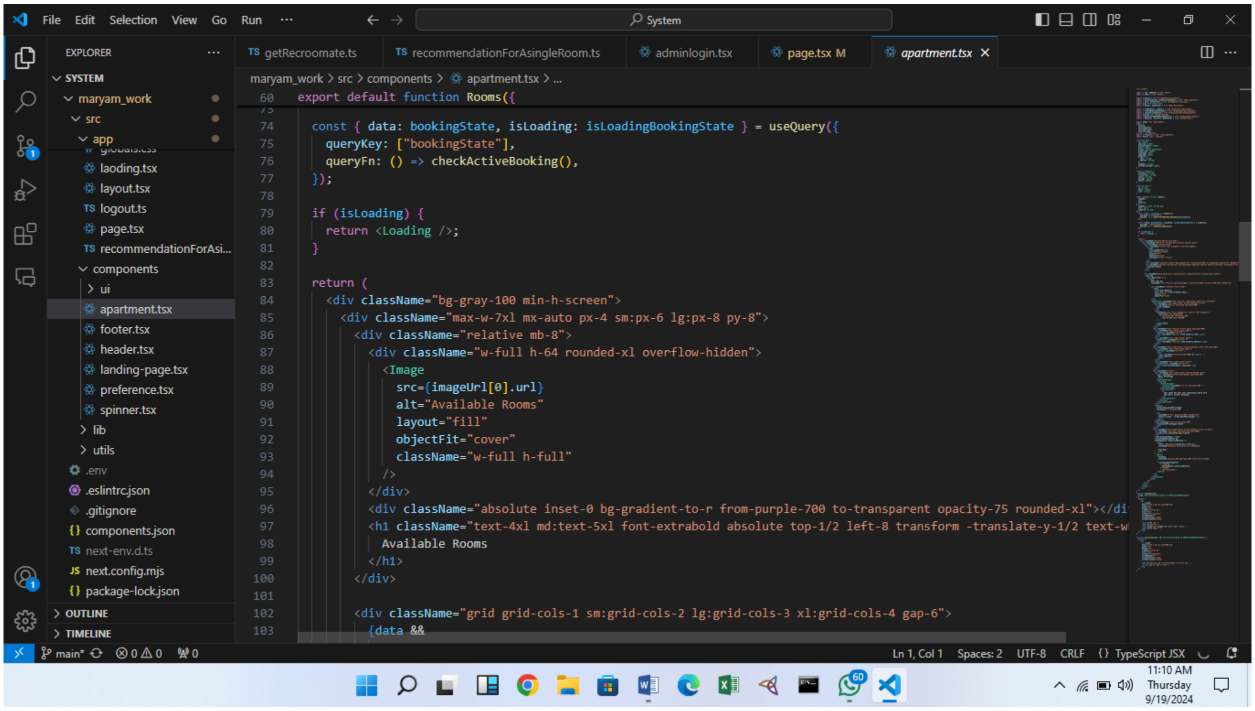
```
const calculateMatchPercentages = (
  };

  const cleanlinessMatch = calculatePreferenceMatch(
    userPreferences.cleanliness,
    roommatePreferences.cleanliness
  );
  const quietnessMatch = calculatePreferenceMatch(
    userPreferences.quietness,
    roommatePreferences.quietness
  );
  const socialnessMatch = calculatePreferenceMatch(
    userPreferences.socialness,
    roommatePreferences.socialness
  );
  const sleepSchedule = calculatePreferenceMatch(
    userPreferences.sleepSchedule,
    roommatePreferences.sleepSchedule
  );

  const overallMatch = Math.round(
    (cleanlinessMatch + quietnessMatch + socialnessMatch + sleepSchedule) / 4
  );

  return {
    overall: overallMatch,
    cleanliness: cleanlinessMatch,
    quietness: quietnessMatch,
    socialness: socialnessMatch,
    sleepSchedule: sleepSchedule,
  };
};
```



```
import Image from "next/image";
import {
  Tooltip,
  TooltipContent,
  TooltipProvider,
  TooltipTrigger,
} from "./ui/tooltip";
import { Progress } from "./ui/progress";
import Footer from "./footer";

interface Room {
  id: number;
  capacity: number;
  availableSpots: number;
  price: number;
  status: $Enums.RoomStatus;
  propertyId: number;
  createdAt: Date;
  updatedAt: Date;
  property: {
    name: string;
    address: string;
  };
  imageUrl: string;
  matchPercentage: number;
}

interface Payment {
  reference: string;
  transaction: string;
  status: string;
```



```
}

interface Payment {
  reference: string;
  transaction: string;
  status: string;
  message: string;
  amount: number;
}

interface User {
  id: string;
  email: string;
  role: string;
}

export default function Rooms({
  imageUrl,
  user,
  property,
}: {
  imageUrl: { url: string }[];
  user: User;
  property: string;
}) {
  const { data, isLoading } = useQuery({
    queryKey: ["recommend"],
    queryFn: () => getFilteredRoomRecommendations(property),
  });

  const { data: bookingState, isLoading: isLoadingBookingState } = useQuery({
    queryKey: ["bookingState"]
```



```
export default function Rooms({
  const { data: bookingState, isLoading: isLoadingBookingState } = useQuery({
    queryKey: ["bookingState"],
    queryFn: () => checkActiveBooking(),
  });

  if (isLoading) {
    return <Loading />;
  }

  return (
    <div className="bg-gray-100 min-h-screen">
      <div className="max-w-7xl mx-auto px-4 sm:px-6 lg:px-8 py-8">
        <div className="relative mb-8">
          <div className="w-full h-64 rounded-xl overflow-hidden">
            <Image
              src={imageUrl[0].url}
              alt="Available Rooms"
              layout="fill"
              objectFit="cover"
              className="w-full h-full"
            />
          </div>
          <div className="absolute inset-0 bg-gradient-to-r from-purple-700 to-transparent opacity-75 rounded-xl"></di
          <h1 className="text-4xl md:text-5xl font-extrabold absolute top-1/2 left-8 transform -translate-y-1/2 text-w
            Available Rooms
          </h1>
        </div>

        <div className="grid grid-cols-1 sm:grid-cols-2 lg:grid-cols-3 xl:grid-cols-4 gap-6">
          {data &&
```