FACULTY OF COMPUTERS SCIENCE
AND INFORMATION TECHNOLOGY
DAMANHOUR UNIVERSITY

Maskank App

GRADUATION PROJECT DOCUMENTATION

Submitted By

- Ali Mohamed Sharaf
- Ahmed Said Abdo
- Mostafa Abdallah Ahmed
- Mohamed Salah elhosary
- Mahmoud Mohamed Abu Salim
- Eman Khalil Hanafy
- Donia Gaber Ahmed
- Kholoud Osama Abdul-Hafez
- Mariam Edwar Ebrahem
- Amira Ashraf Abdul Gany

Supervisor By

Dr. Noura Shoaib

Eng. Eman Anwar

2024

Graduation Project - English Abstract

Project Code:	
Project Title (in English):	**Maskank App**
Project Title (in Arabic):	**تطبيق مسكنك**
Scientific Department:	Computer Science – Information Technology
Supervisor(s):	Dr. Noura Shoaib - Eng. Eman Anwar
Project Team:	1- Ali Mohamed Sharaf 2- Ahmed Said Abdo 3- Mostafa Abdallah Ahmed 4- Mohamed Salah elhosary 5- Mahmoud Mohamed Abu Salim 6- Eman Khalil Hanafy 7- Donia Gaber Ahmed 8- Kholoud Osama Abdul-Hafez 9- Mariam Edwar Ebrahem 10- Amira Ashraf Abdul Gany

Project Abstract

The primary goal of the Maskank App is to revolutionize the apartment rental experience for students, vacationers, and individuals seeking accommodation. By leveraging cutting-edge technologies, the app aims to provide a seamless and personalized rental experience. Key objectives include enhancing user experience through an intuitive interface and personalized recommendations. The app targets a broad audience, focusing on short and long-term rentals. It integrates advanced technologies like Flutter for UI development and AI for intelligent filtering, recommendations, and price predictions. Continuous innovation and integration with external services ensure the app remains competitive and meets evolving user needs.

الملخص العربي لمشروع التخرج

كود المشروع:	
عنوان المشروع باللغة العربية:	**تطبيق مسكنك**
عنوان المشروع باللغة الإنجليزية:	**Maskank App**
القسم العلمي:	**علوم الحاسب – تكنولوجيا المعلومات**
اسم المشرف على المشروع:	**د. نورا شعيب – د. إيمان أنور**
أسماء فريق العمل من الطلاب:	1. **علي محمد شرف** 2. **أحمد سعيد عبده** 3. **مصطفى عبد الله أحمد** 4. **محمد صلاح الحصري** 5. **محمود محمد أبو سليم** 6. **إيمان خليل حنفي** 7. **دنيا جابر أحمد** 8. **خلود أسامة عبد الحفيظ** 9. **مريم إدوار إبراهيم** 10. **أميرة أشرف عبد الغني**

ملخص المشروع Project Abstract

الهدف الرئيسي من تطبيق مسكنك هو إحداث ثورة في تجربة استئجار الشقق للطلاب والمصطافين والأفراد الباحثين عن سكن. من خلال الاستفادة من التقنيات المتقدمة، يهدف التطبيق إلى توفير تجربة استئجار سلسة وشخصية. تشمل الأهداف الرئيسية تحسين تجربة المستخدم عبر واجهة بديهية وتوصيات مخصصة. يستهدف التطبيق جمهورًا واسعًا، مع التركيز على الإيجارات القصيرة والطويلة الأجل. يدمج التطبيق تقنيات متقدمة مثل فلاتر لتطوير واجهة المستخدم والذكاء الاصطناعي للتصفية الذكية والتوصيات والتنبؤ بالأسعار. الابتكار المستمر والتكامل مع الخدمات الخارجية يضمنان بقاء التطبيق تنافسيًا وتلبية احتياجات المستخدم المتطورة.

نموذج تقييم ومناقشة مشروع التخرج

اسم المشروع باللغة الانجليزية	
القسم العلمي	○ Computer Science ○ Information Systems ○ Information technology ○ Multimedia

م	اسم الطالب	هل المشروع يحقق الهدف المطلوب؟	Documentation (التنظيم العام، المحتوى، مستوى الكتابة، الأسلوب، مراعاة الاخلاقيات)	Presentation (درجة الإلمام، الوضوح، الإتقان).	استيعاب وتمكن الطالب من المشروع والرد على استفسارات وأسئلة اللجنة	اجمالي الدرجات
		5 درجات	5 درجات	5 درجات	25 درجة	40 درجة
1						
2						
3						
4						
5						
6						
7						
8						
9						
10						

الممتحنين **وكيل الكلية لشئون التعليم والطلاب**

عميد الكلية

نموذج وضع درجات أعمال السنة لمشروع التخرج (من قبل المشرف)

اسم المشروع باللغة الانجليزية	
القسم العلمي	○Computer Science ○ Information Systems ○ Information technology ○Multimedia

م	اسم الطالب	مدى التزام الطالب بحضور اللقاءات والمتابعة المستمرة مع المشرف	المشاركة في النقاشات والاجابة على الأسئلة المطروحة اثناء اللقاءات الدورية	استيعاب وتمكن الطالب من الأدوات التي استخدمت في تطوير المشروع	قدرة الطالب على تنفيذ المهام المكلف بها	قدرة الطالب على العمل ضمن فريق عمل	اجمالي درجات أعمال السنة
		10 درجة	10 درجة	10درجة	20 درجة	10 درجة	60 درجة
1							
2							
3							
4							
5							
6							
7							
8							
9							
10							

المشرف على المشروع **وكيل الكلية لشئون التعليم والطلاب**

عميد الكلية

TABLE OF CONTENTS

Table of Figures

Frequency of floor level	
Frequency of size	
Distribution of Houses per city	
Prices per House Type	
House size & price	
Bedrooms & price	
Bathrooms & price	
Floor level & price	
Price with furnished	
Rent price per region	
Extract a new features	
Label Encoding	
Dropped the houses isn't for rent features	
Splitting data into train and test data	
Choosing the best model	
Preprocess the data before enter the model	
Using flask to generate the local API	
Test API using postman	
Mobile Application	
Logic of signup with Getx	
Logic owner with Getx controller	
Code to show if network connection or not	
API to send data to backend	
API method helper to share use	
API method helper to share	
Logic controller of Getx by GetBuilder	
Delete method API	
Link UI with GetBuilder of logic Getx	
Post method	
Backend	
Register, login, logout for owner	
Register, login, logout for renter	
Verification email	
Add post && Approved	
Reset password	
Update password	

Chapter 1
Introduction

1.1 Introduction

The real estate market can be a complex and intimidating environment for new and experienced buyers, sellers, and investors. The sheer volume of information, the high stakes involved in transactions, and the multitude of choices can make the process overwhelming. Navigating through property listings, understanding market trends, dealing with legal paperwork, and negotiating prices require a considerable amount of expertise and time. First-time homebuyers often find themselves in uncharted territory, unsure of where to start and what to consider. Similarly, sellers need to strategize on pricing, marketing, and negotiating to get the best value for their property.

Moreover, real estate agents, while knowledgeable, also face the challenge of managing multiple clients, properties, and transactions simultaneously. They must stay updated with market changes, provide accurate information, and offer tailored advice to each client. This multifaceted process can lead to stress and errors if not managed properly.

An efficient and comprehensive real estate application could be a game-changer in this scenario. By consolidating property listings, market data, and essential tools into one platform, such an app can significantly streamline the real estate experience. It can offer features like real-time alerts for new listings, personalized recommendations based on user preferences, and tools for scheduling viewings and consultations. Additionally, providing educational resources and a platform for buyers and sellers to ask questions can demystify the process and empower them with the knowledge they need.

This application would not only simplify the buying and selling process but also enhance transparency and trust in real estate transactions. For agents, it would serve as a robust tool to manage their workload more efficiently and provide better service to their clients. In essence, this real estate application would act as a comprehensive guide and assistant, making the entire real estate journey smoother, more informed, and less stressful for everyone involved.

1.2 Problem Definition

The real estate market is a complex and dynamic environment where buyers and sellers interact with agents and agencies to facilitate property transactions. Traditional methods of real estate transactions are often time-consuming, lack transparency, and can be inefficient due to the reliance on physical paperwork and face-to-face interactions. With the increasing adoption of digital technology, there is a need for a comprehensive platform to streamline the real estate process, making it more accessible, transparent, and efficient for all parties involved.

Current Challenges:

1. Lack of Centralized Information:

   Buyers and sellers often struggle to find a centralized source of reliable property information. Information is scattered across various platforms, making it difficult to compare and make informed decisions.

2. Inefficient Communication:

   Communication between buyers, sellers, and agents can be slow and fragmented, leading to delays and misunderstandings in the transaction process.

3. Limited Access to Market Data:

   Buyers and sellers lack access to real-time market data, such as property prices, market trends, and neighborhood statistics, which are crucial for making informed decisions.

4. Manual and Paper-Based Processes:

   The traditional real estate process relies heavily on manual paperwork, which is time-consuming, prone to errors, and lacks transparency.

5. Difficulty in Finding Trustworthy Agents:

   Buyers and sellers often find it challenging to identify and connect with trustworthy and competent real estate agents.

6. Inadequate Property Visualization:

   Limited property visualization options (e.g., photos, videos, virtual tours) hinder buyers from getting a comprehensive view of the properties they are interested in.

Key Features:

1. User Accounts and Profiles:

   Allow users (buyers, sellers, agents) to create and manage their profiles with personalized dashboards.

2. Property Search and Filters:

   Implement robust search functionality with filters for location, price range, property type, and other relevant criteria.

3. Interactive Maps:

   Integrate interactive maps to display property locations and nearby amenities.

4. Favorites and Alerts:

   Enable users to save favorite properties and set up alerts for new listings that match their criteria.

5. Secure Messaging:

   Provide a secure messaging system for users to communicate directly within the platform.

6. Appointment Scheduling:

   Allow users to schedule property viewings and consultations with agents directly through the app.

7. Multi-Language Support:

   The application supports multiple languages to cater to a diverse user base.

1.3 Project Objective

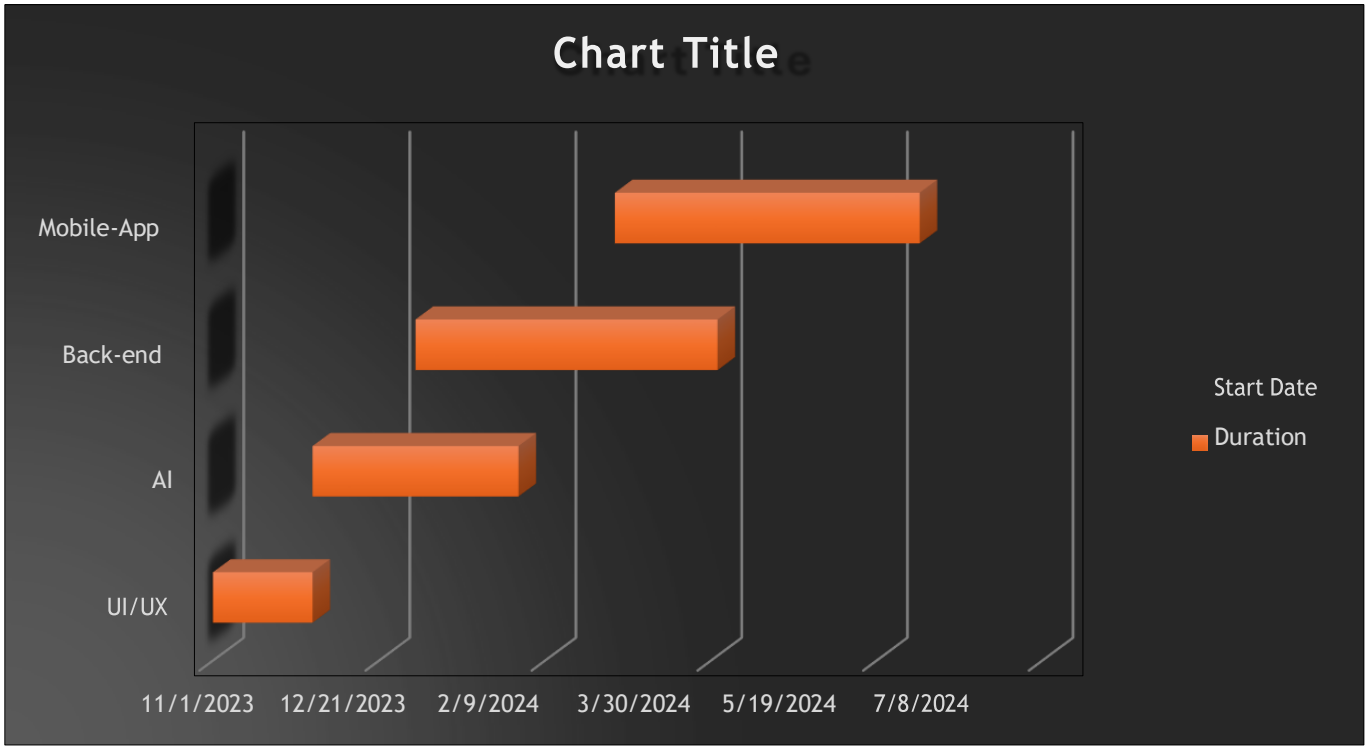
Our project offers:

- simplify and enhance the real estate experience for buyers, sellers, and agents by providing a comprehensive and user-friendly platform.
- Real-time Property Listings: Instant updates for new property listings tailored to user preferences.
- Advanced Search Filters: Comprehensive search tools to find properties based on specific criteria such as location, price range, property type, and amenities.
- Interactive Maps: Detailed maps showcasing property locations, nearby amenities, and neighborhood statistics to aid in decision-making.
- Virtual Tours: High-quality virtual tours provide a realistic sense of the property without needing an in-person visit.
- In-app Scheduling and Booking: Tools for scheduling property viewings, inspections, and meetings with real estate agents directly through the app.
- Property Management Features: Tools for scheduling viewings, managing listings, and tracking transaction progress.
- Personalized Recommendations: AI-driven suggestions based on user preferences and search history.
- We save the user time and effort for both the tenant and the owner of the apartment.
- Attractive User Interface: Provide an easy-to-use and visually appealing user interface for users to browse and book apartments seamlessly.
- User Registration and Account Management: Enable users to register and manage their accounts effortlessly
- customized Recommendations using AI: Utilize artificial intelligence techniques to recommend apartments based on user preferences and previous search history
- Apartment and Advertisements Management: Offer the ability to add and manage apartments available for rent, ensuring accuracy and quality of advertisement.

1.4 Project Scope

- Implement advanced search and filtering options, allowing users to find properties based on specific criteria such as location, price range, property type, number of rooms, amenities, etc. This will help users narrow down their options and find properties that meet their requirements.
- Provide comprehensive details about each property, including high-resolution images, floor plans, property size, amenities, nearby facilities (schools, hospitals, transportation), and any unique selling points. This will help users make informed decisions about the properties they are interested in.
- Provide information about the neighborhood where each property is located. This can include details about nearby schools, healthcare facilities, shopping centers, parks, crime rates, and public transportation options. Users can gain a better understanding of the area and its suitability for their needs.

1.5 Project Timeline



1.6 Document Organization

Our project documentation consists of six parts in addition to three appendices. These parts provide a comprehensive understanding of our MASKANK application, defining the problems, proposed solutions, and project objectives.

A brief description of the contents of each part:

- Part 1: Introduction

This section introduces the project objectives, the motivation behind the project, the approach used, and the scope of the project. It outlines the need for a user-friendly real estate application and the benefits it will bring to users in finding and managing property transactions efficiently.

- Part 2: Project Techniques and Technologies

This section covers the techniques and technologies employed in the project, as well as related works in the field. It highlights the advantages and disadvantages of existing real estate applications and explains how our application will combine the best features while addressing common shortcomings.

- Part 3: Project Analysis

This part delves into the project analysis process, detailing the functional requirements and features of the application. It includes diagrams such as use case diagrams, sequence diagrams, and system architecture to provide a clear understanding of the application's structure and functionality.

- Part 4: Design Specifications

This section focuses on the design aspects of the project, including the Entity-Relationship Diagram (ERD), class diagram, and detailed screen designs of the application. It emphasizes the importance of each component in the overall system design.

- Part 5: Implementation

This part demonstrates the implementation of the system, mapping the design into actual code and software components. It also discusses the testing process and presents the achieved results, highlighting the system's performance and reliability.

- Part 6: Conclusion and Future Work

This section summarizes the entire project, reflecting on the goals achieved and the challenges encountered. It also outlines potential improvements and future work to enhance the application's functionality and user experience.

Chapter 2
Literature Review

2.1 Background

Maskank App is designed to transform the apartment rental experience, making it more efficient and user-centric. Aimed at a diverse audience, including students, vacationers, and individuals seeking both short and long-term rentals, the app leverages advanced technology to simplify and enhance the entire rental process.

Key Objectives

1. Enhanced User Experience: The app focuses on providing an intuitive and user-friendly interface. It ensures smooth navigation and interactive features, making the search and rental process straightforward and efficient. Personalized recommendations help users find their ideal living spaces effortlessly.
2. Targeted Audience Reach: Maskank App addresses the needs of various user segments, understanding their unique preferences and requirements. This targeted approach helps in effectively catering to a wide demographic, including students, vacationers, and regular renters.
3. Technology Integration: The app utilizes cutting-edge technologies to enhance functionality and user experience. Built with Flutter for a robust user interface, it also integrates AI technology for intelligent apartment filtering, recommendations, and price prediction, ensuring a high-tech solution for users.
4. Seamless Integration: A key goal is to streamline the rental journey from search to lease agreement. The app integrates various external services, such as payment gateways and communication platforms, to enhance utility and convenience for users.
5. Personalized Recommendations: By leveraging AI algorithms and machine learning models, Maskank App offers personalized recommendations based on user preferences, budget, and location. Analyzing user behavior and historical data, the app tailors suggestions to enhance satisfaction.
6. Predictive Analytics: The app employs AI-driven predictive analytics to anticipate user needs and preferences, facilitating proactive decision-making and
7. customized offerings. Predictive pricing models help users make informed decisions by estimating rental prices based on market trends and property attributes.
8. Continuous Innovation: Maskank App is committed to ongoing innovation and improvement. Regular updates, feature enhancements, and user feedback-driven iterations ensure the app remains competitive and adapts to evolving market dynamics and user expectations.

Scope:

The scope of this plan encompasses the Software Quality Assurance (SQA) activities undertaken during the development and maintenance of the Maskank App.

Technologies Used

- Flutter: For user interface and front-end development.
- PHP Laravel: For back-end development.
- MySQL: For database management.
- Python: For machine learning and AI functionalities.
- Agile Methodology: Employed throughout the project to ensure flexibility and responsiveness to change.

This comprehensive approach ensures that the Maskank App delivers a seamless, innovative, and user-focused rental experience, leveraging the latest technologies and continuous improvement practices to meet the diverse needs of its users.

2.2 Programming Language and IDE:

2.2.1 Flutter:

1. **Visual studio code:** Visual Studio Code (VS Code) is a free, open-source code editor developed by Microsoft, which is available for Windows, macOS, and Linux. It is designed to be lightweight yet powerful, supporting a wide range of programming languages and development activities. Since its release in 2015, VS Code has gained significant popularity among developers due to its performance, flexibility, and extensive feature set.
2. **Dart:** Dart is an open-source, general-purpose programming language developed by Google. Designed for building web, server, and mobile applications, Dart is optimized for UI development and powers the popular Flutter framework. Its goal is to provide a productive, easy-to-learn language that offers high performance across various platforms.
3. **Flutter:** Flutter stands out as a powerful and flexible toolkit for building cross-platform applications. Its ability to deliver native performance, combined with a rich set of customizable widgets and a fast development cycle through Hot Reload, makes it an ideal choice for developers aiming to create beautiful, high-performance applications across mobile, web, and desktop platforms. The growing ecosystem and active community further enhance its appeal, making Flutter a robust solution for modern app development.

2.2.2 Backend:

1. **Visual studio code:** Visual Studio Code (VS Code) is a free, open-source code editor developed by Microsoft, which is available for Windows, macOS, and Linux. It is designed to be lightweight yet powerful, supporting a wide range of programming languages and development activities. Since its release in 2015, VS Code has gained significant popularity among developers due to its performance, flexibility, and extensive feature set.
2. **Php:** PHP (Hypertext Preprocessor) is a widely-used open-source server-side scripting language designed specifically for web development. Created in 1994 by Rasmus Lerdorf, PHP has evolved into a robust programming language that powers a significant portion of websites on the internet, including major platforms like Facebook, WordPress, and Wikipedia.
3. **Laravel:** Laravel is a popular open-source PHP framework designed for building modern web applications. Created by Taylor Otwell in 2011, Laravel aims to simplify development by providing elegant syntax, robust features, and tools that streamline common web development tasks.

2.2.3 Machine learning:

1. **Pycharm:** PyCharm stands out as a robust and versatile IDE tailored for Python developers. Its intelligent features, comprehensive tools, and strong support for various development workflows make it an essential tool for enhancing productivity and efficiency in Python programming. Whether for general development, web projects, or data science, PyCharm provides the capabilities needed to streamline coding, debugging, and project management processes
2. **Jupyter**: Jupyter is a powerful and versatile tool for interactive computing. Its ability to integrate code, visualizations, and narrative text in a single document makes it an invaluable resource for data scientists, researchers, and educators. The extensive support for multiple programming languages, coupled with its rich ecosystem and active community, ensures that Jupyter remains at the forefront of data analysis and scientific computing workflows.
3. **Python:** Python is a high-level, interpreted programming language known for its simplicity, readability, and versatility. Created by Guido van Rossum and first released in 1991, Python has become one of the most popular programming languages in the world. It is widely used in various domains, including web development, data science, artificial intelligence, scientific computing, automation, and more.

Machine Learning:

• Machine learning is a branch of artificial intelligence (AI) and computer science which focuses on the use of data and algorithms to imitate the way that humans learn, gradually improving its accuracy.

• And in our application, We use the data set to train the model so that it is able to predict the price of the apartment by collecting information about the apartment, such as the type of apartment, its size, the number of bathrooms, the number of bedrooms, floor, whether it is furnished or not, the city, the district, and it will be. Through these features, he is able to determine the price of the apartment, but we faced difficulty in that we found a dataset related to renting apartments, and after a lot of research, we found that the dataset is for apartments for sale and rent, and we used it and made a drop the rows for apartments for sale.

Model:

We used a linear regression model to determine the price of the apartment. This model is based on analyzing the relationship between the features and the price of the apartment. The model is trained on a dataset of apartments that have been sold or rented in the past.

Machine Learning Algorithms:

We tried a number of different algorithms to determine the best algorithm for determining the price of the apartment. We found that linear regression is the best algorithm for this project.

Future:

We plan to continue to develop and improve this application. We also plan to explore the use of other machine learning techniques to more accurately determine the characteristics of apartments.

2.3 Review of Relevant Work:

Due to the importance of the apartment rental , there are many applications and websites to help users with this

1-Property finder application:

Pros

- Coverage in Multiple countries
- Comprehensive property details
- User-friendly interface
- Advanced search options

Cons

- Data update challenges
- Costs for some services
- Variability in image quality

2- Aqarmap Egypt

Pros

- Wide range of properties
- User-friendly interface
- Comprehensive property information
- Advanced search options

Cons

- Data update challenges
- Localization of services
- Quality variation in images
- Market-specific user experience

3- Bayut Egypt app

Pros

- Wide range of properties
- Comprehensive property information
- User-friendly interface
- Advanced search options

Cons

- Data accuracy challenges
- Intense competition
- Some services may be paid

2.4 Relationship Between the Relevant Work and Our Own Work:

The relation between our work and relevant work is to apartment rental, aims to provide a seamless and personalized rental experience for users, deliver an intuitive, user-friendly interface that simplifies the apartment search and rental process, There is a choice for the owner to upload posts and The owner has his own interface, and the user has his own interface.

In addition to all of that, we added some distinguishing features that are suitable for users

So, the added work is:

- AI technology plays a pivotal role in enhancing the user experience by facilitating intelligent apartment filtering, recommendation, and price prediction processes.
- Admin has the ability to control the Dashboard
- We have the ability to verify the identity of the owner by uploading an identity card

Summary:

The Maskank App is designed to transform the apartment rental experience by addressing the diverse needs of students, vacationers, and regular renters. By employing advanced technologies and innovative features, the app aims to offer a seamless and personalized rental process.

Chapter 3
System Analysis

3.1 Introduction

We gathered extensive information about our users and competitors, it is time to analyze our system to precisely meet the needs of our target users. For this analysis, we started classifying them as information about our users extracted from our Expert interviews, users' interviews, and surveys. Other information was extracted about the technologies we used to give us the best performance.

3.2 Analysis of Existing System

Our project is essentially software that includes:

- Property Listings and Management: Enables users to list, search, and manage properties easily. Users can view detailed information, photos, and virtual tours of properties.
- Smart Recommendation System: Utilizes machine learning algorithms to analyze user preferences and provide personalized property recommendations.
- Market Analysis Tools: Provides tools for analyzing market trends, property values, and investment opportunities using advanced data analytics.
- Secure Transaction Processing: Ensures secure handling of financial transactions and documentation through encrypted and verified processes.
- Appointment Scheduling: Allows users to schedule property visits and meetings with real estate agents directly through the app.
- Neighborhood Insights: Provides detailed information about neighborhoods, including schools, crime rates, amenities, and public transportation.
- Price Estimator: Uses historical data and current market trends to estimate property prices and rental values.
- Agent Finder: Helps users find and connect with real estate agents based on their location, ratings, and specialties.
- Tenant Management: Offers landlords tools to manage tenants, including rent collection, maintenance requests, and lease agreements.
- User Profiles and Dashboards: Provide users with personal dashboards to manage their account information, saved properties, search preferences, and alerts. Users should be able to update their profiles and settings.

3.3 System Requirements

System requirements encompass broad and specific details defining what the customer expects from the application. These requirements should clearly articulate the customer's needs, specifying exactly what they want and how they envision achieving it. A customer's requirements aim to fulfill contractual obligations, solve specific problems, achieve business objectives, comply with industry standards, or meet other project guidelines.

3.3.1 Functional Requirements

A functional requirement outlines the tasks, actions, or activities that must be accomplished to provide operational capabilities and satisfy user requirements. These requirements are essential for delivering a functional and effective application. Some functional requirements can be identified from the needed operational capabilities, while others may emerge from diligent systems engineering and detailed analysis.

Description of feature	Functional requirements
1. User Login	- Users can log in using their email - The system must only allow users with valid email and password to enter. - The user must be able to log out after they finish using the system.
2. User Register	- Users must be able to register using email. - The system must be able to verify information.
3. Owner Login	- Owners can log in using their email. - The system must only allow owners with valid email and password to enter. - The owner must be able to log out after they finish using the system.
4. Owner Register	- Owners must be able to register using email and with their ID for high security. - The system must be able to verify information.
5. Add Property	- Owners can add any number of properties - Owners must fill out required fields such as property type, location, price, and description - Owners can upload photos and videos of the property.
6. Property Search	- Users can search for properties using various filters such as location, price range, property type, and amenities. - Users can save their searches for future reference.
7. Favorites/Bookmarks	- Users can bookmark or favorite properties they are interested in. - Users can view their list of favorited properties at any time.
8. Market Analysis	- Users can access market trends, price changes, and neighborhood statistics - Users can compare properties based on various parameters.

3.3.2 Non-Functional Requirements

- Performance Requirements:

Requirement ID	Requirement Description
1-PR	The system must respond to the business operation in less than 3 seconds for the user
2-PR	The program will work on the Android and Ios operating system
3-PR	The system should be reliable
4-PR	The system should respond to the operation messages to the users within 2 seconds

- Safety and Security Requirements:

The system cannot affect, harm, or damage the user. It also cannot damage the user's computer while accessing the system over a network

Requirement ID	Requirement Description
1-SSR	The system must handle safe login and logout through a session.
2-SSR	Hashing technology should be used to handle the secure login for users.
3-SSR	The database should be secured from SQL injection to prevent leaks or loss of information.

- Safety and Security Requirements:

The system cannot affect, harm, or damage the user. It also cannot damage the user's computer while accessing the system over a network

Requirement ID	Requirement Description
1-SSR	The system must handle safe login and logout through a session.

2-SSR	Hashing technology should be used to handle the secure login for users.
3-SSR	The database should be secured from SQL injection to prevent leaks or loss of information.

3.4 User Requirements

- The main users of our application are property buyers, sellers, agents, and landlords who need an efficient platform to manage their real estate transactions.
  The user will be able to:

1. Users can quickly find their way around the app thanks to a user-friendly interface that simplifies their experience.
2. Sign up, log in, and log out effortlessly:
   The app provides a straightforward process for users to create accounts, access their profiles, and securely log out when needed.

3. Create and manage customized profiles:
   Users can set up detailed profiles that reflect their preferences, saved searches, and listings they are interested in.
4. Search for properties with advanced filters:
   Buyers and renters can search for properties using a variety of filters such as location, price range, property type, and amenities.
5. Sellers and landlords can easily list their properties, including uploading photos, and descriptions, and setting prices or rent amounts.
6. The app allows users to schedule viewings directly through the platform and communicate with real estate agents via messaging.

3.5 Tools and Languages

Developing our software application can be divided into main two parts, which are the design part, the implementation part. The design part involves designing diagrams and designing the user interface of the mobile application. The implementation part involves programming languages, IDEs, frameworks, and libraries. The following list shows the
needed tools for software development

- Design Part
  1. Figma:

Purpose User Interface (UI) and User Experience (UX) design.

Description: Figma is a web-based design tool used for creating UI designs, wireframes, and prototypes. It allows real-time collaboration among team members.

2. Lucidchart:

Purpose: Diagramming and flowchart creation.

Description: Lucidchart is an online diagramming tool used to create flowcharts, wireframes, UML diagrams, and other visual representations of software architecture.

- Implementation Part

3. Python:

Purpose: Programming language for AI models and machine learning.

Description: Python is widely used for machine learning and AI due to its simplicity and extensive libraries like TensorFlow, Keras, and scikit-learn.

4. Dart (with Flutter):

Purpose: Programming language and framework for Android development.

Description: Dart is used with the Flutter framework to develop high-performance, cross-platform mobile applications, including Android.

5. PHP with Laravel:

Purpose: Backend development framework.

Description PHP is a widely-used server-side scripting language, and Laravel is a powerful MVC framework for PHP that simplifies the development of robust web applications with elegant syntax and built-in tools for routing, authentication, and database management.

6. Windows 10 (Server):

Purpose: Operating system for server deployment.

Description: Windows 10 can be used for development and server purposes, providing a stable environment for running backend services, databases, and AI models.

3.6 Summary

- Design Tools:

1. Figma: UI/UX design.

2. Lucidchart: Diagramming and flowchart creation.

- Implementation Tools and Languages:
    3. Python AI model and machine learning programming.
    4. Dart with Flutter: Android mobile application development.
    5. PHP with Laravel: Backend web application development.
    6. Windows 10: Server operating system.

These tools and languages cover the essential aspects of designing and implementing your software application, ensuring a comprehensive development workflow that includes mobile application development, AI integration, and backend development.

Chapter 4
System Design

4.1 Introduction

Software design sits at the technical kernel of the software engineering process and is applied regardless of the development paradigm and area of application. Design is the first step in the development phase for any engineered product or system. The designer's goal is to produce a model or representation of an entity that will later be built. Beginning, once system requirement has been specified and analyzed, system design is the first of the three technical activities design, code and test that is required to build and verify software.

The importance can be stated with a single word "Quality". Design is the place where quality is fostered in software development. Design provides us with representations of software that can assess for quality. Design is the only way that we can accurately translate a customer's view into a finished software product or system. Software design serves as a foundation for all the software engineering steps that follow. Without a strong design we risk building an unstable system one that will be difficult to test, one whose quality cannot be assessed until the last stage.

During design, progressive refinement of data structure, program structure, and procedural details are developed reviewed and documented. System design can be viewed from either technical or project management perspective. From the technical point of view, design is comprised of four activities architectural design, data structure design, interface design and procedural design.

4.2 UML Diagram

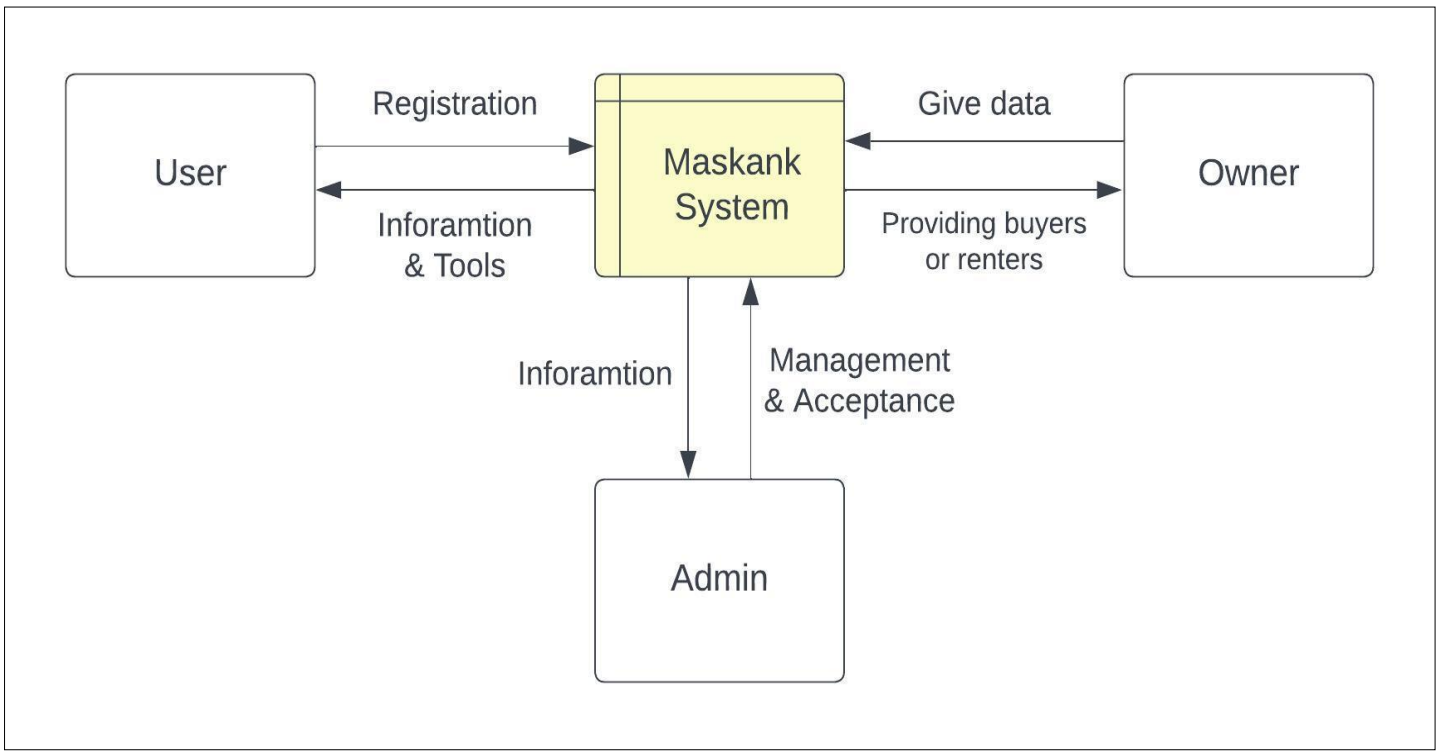
This Project will be represented in a Use-Case diagram, Action Diagram, Sequence diagram, and class diagram. The architecture that will be implemented is MVVM (Model-View-ViewModel), which will appear in sequence and class diagrams.

4.2.1 Context Diagram

A system context diagram in engineering is a diagram that defines the boundary between the system, or part of a system, and its environment, showing the entities that interact with it. This diagram is a high level view of a system. It is similar to a block diagram.

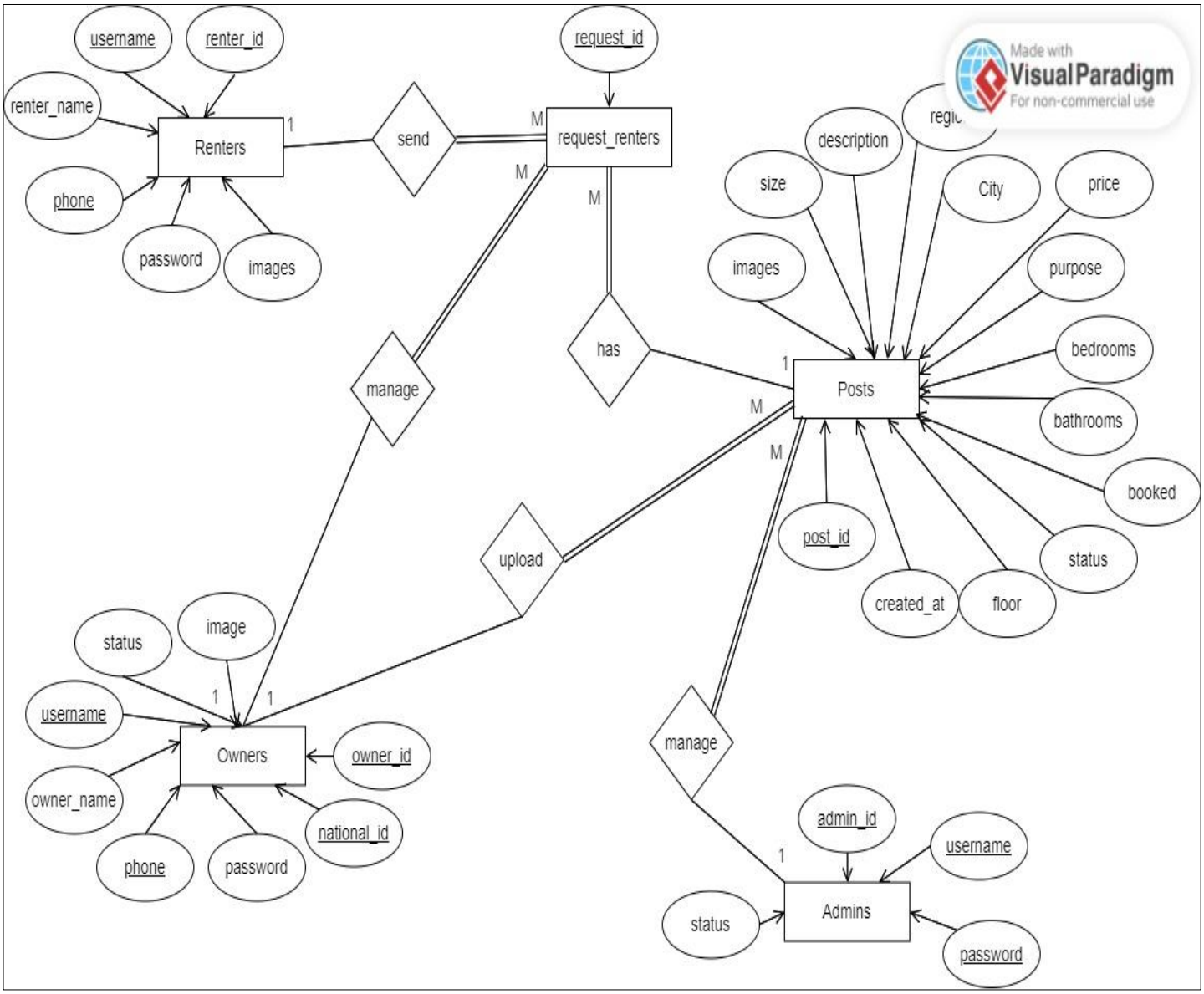
System context diagrams show a system, as a whole and its inputs and outputs from/to external factors. According to Kossiakoff and Sweet (2011):

> System Context Diagrams ... represent all external entities that may interact with a system ... Such a diagram pictures the system at the center, with no details of its interior structure, surrounded by all its interacting systems, environments and activities. The objective of the system context diagram is to focus attention on external factors and events that should be considered in developing a complete set of systems requirements and constraints.



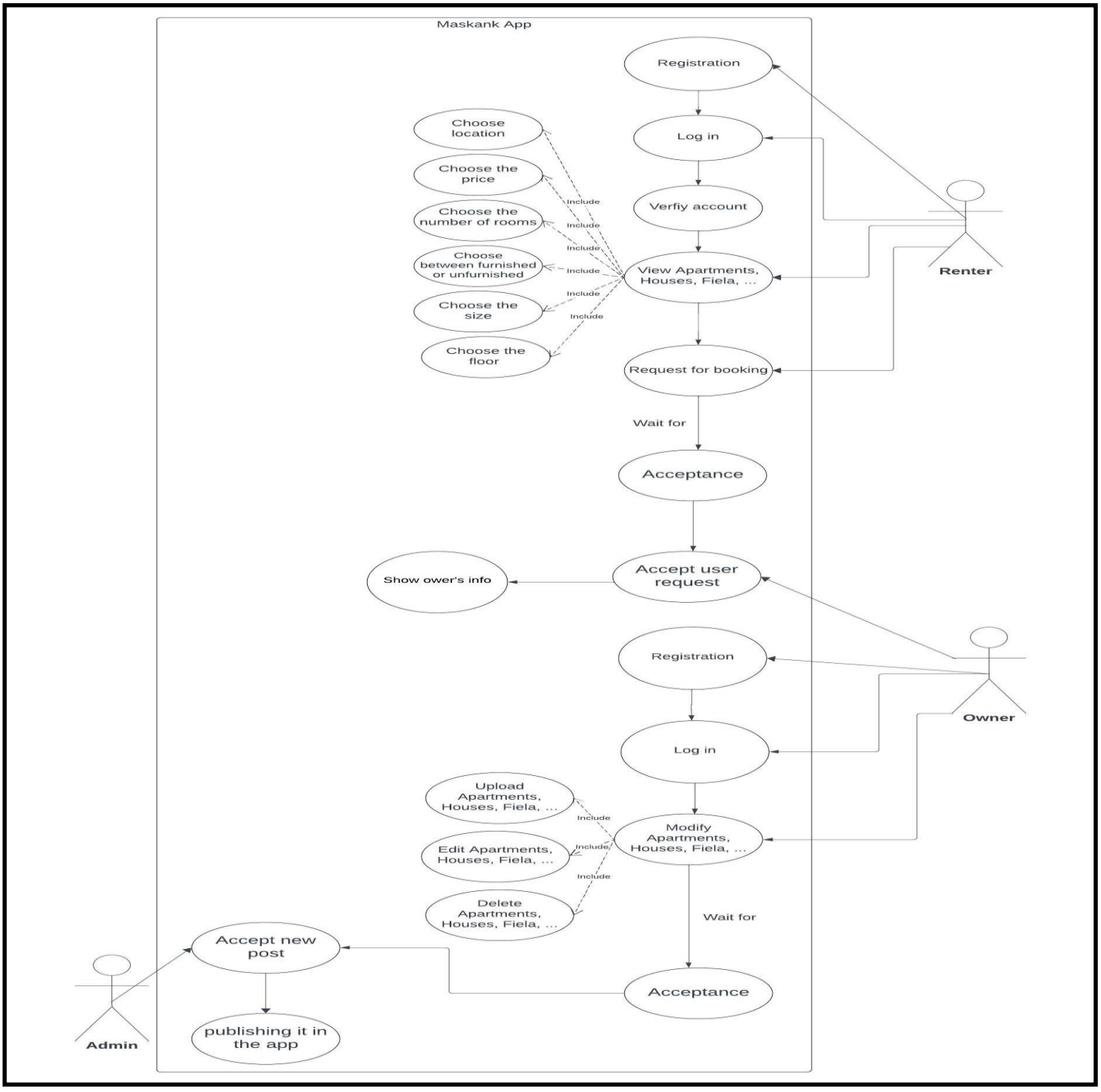
4.2.2 ER Diagram

An Entity Relationship (ER) Diagram is a type of flowchart that illustrates how “entities” such as people, objects or concepts relate to each other within a system. ER Diagrams are most often used to design or debug relational databases in the fields of software engineering, business information systems, education and research. Also known as ERDs or ER Models, they use a defined set of symbols such as rectangles, diamonds, ovals and connecting lines to depict the interconnectedness of entities, relationships and their attributes. They mirror grammatical structure, with entities as nouns and relationships as verbs.


username
renter_id
renter_name
Renters
phone
password
images
send
request_id
request_renters
description
size
City
price
images
purpose
bedrooms
has
manage
Posts
bathrooms
booked
post_id
status
created_at
floor
upload
image
status
username
Owners
owner_id
owner_name
national_id
phone
password
manage
admin_id
username
Admins
status
password
1
M
Made with VisualParadigm
For non-commercial use

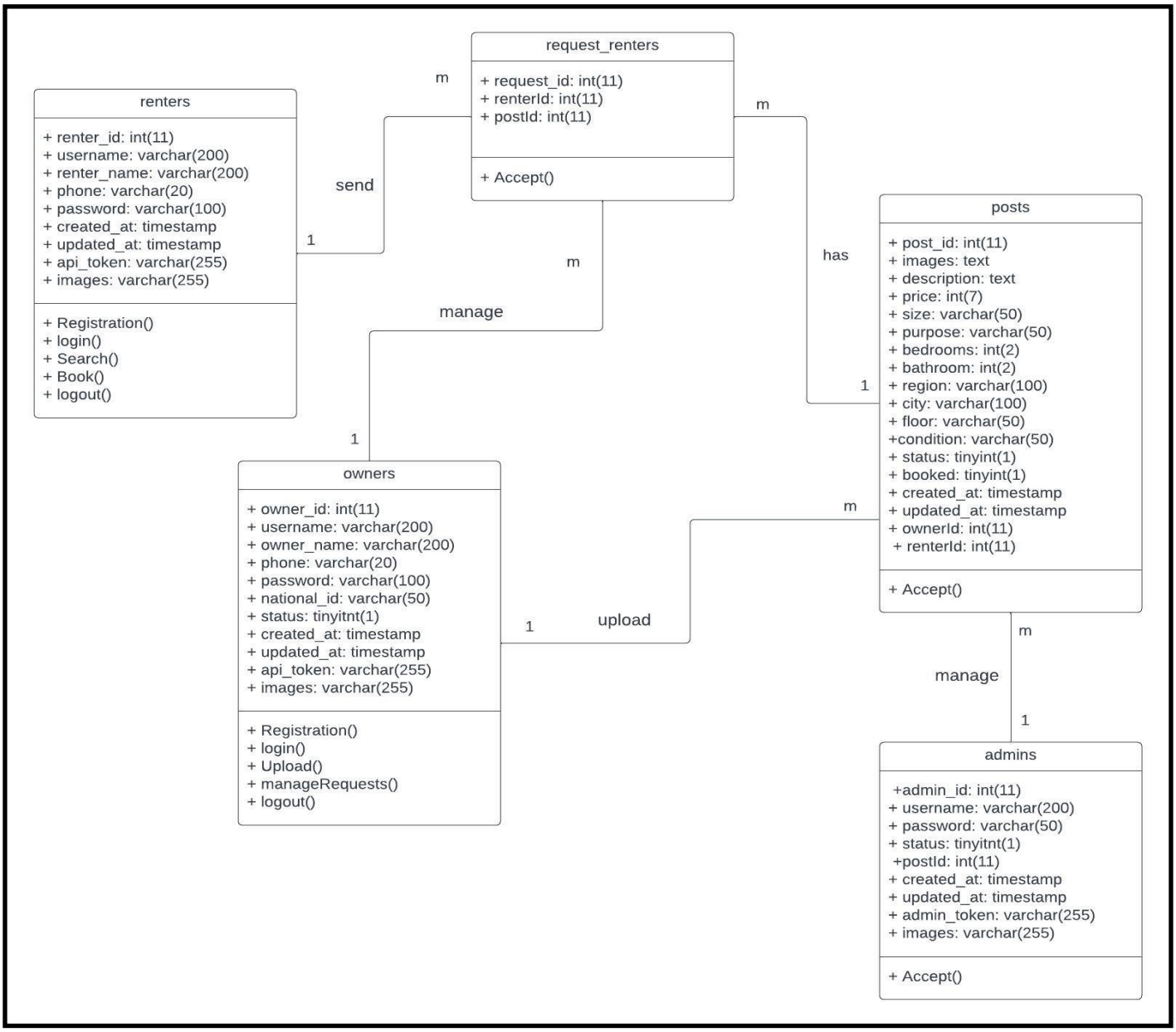
4.2.3 Use-Case Diagram

A use case diagram is a graphical depiction of a user's possible interactions with a system. A use case diagram shows various use cases and different types of users the system has and will often be accompanied by other types of diagrams as well. The use cases are represented by either circles or ellipses. The actors are often shown as stick figures.



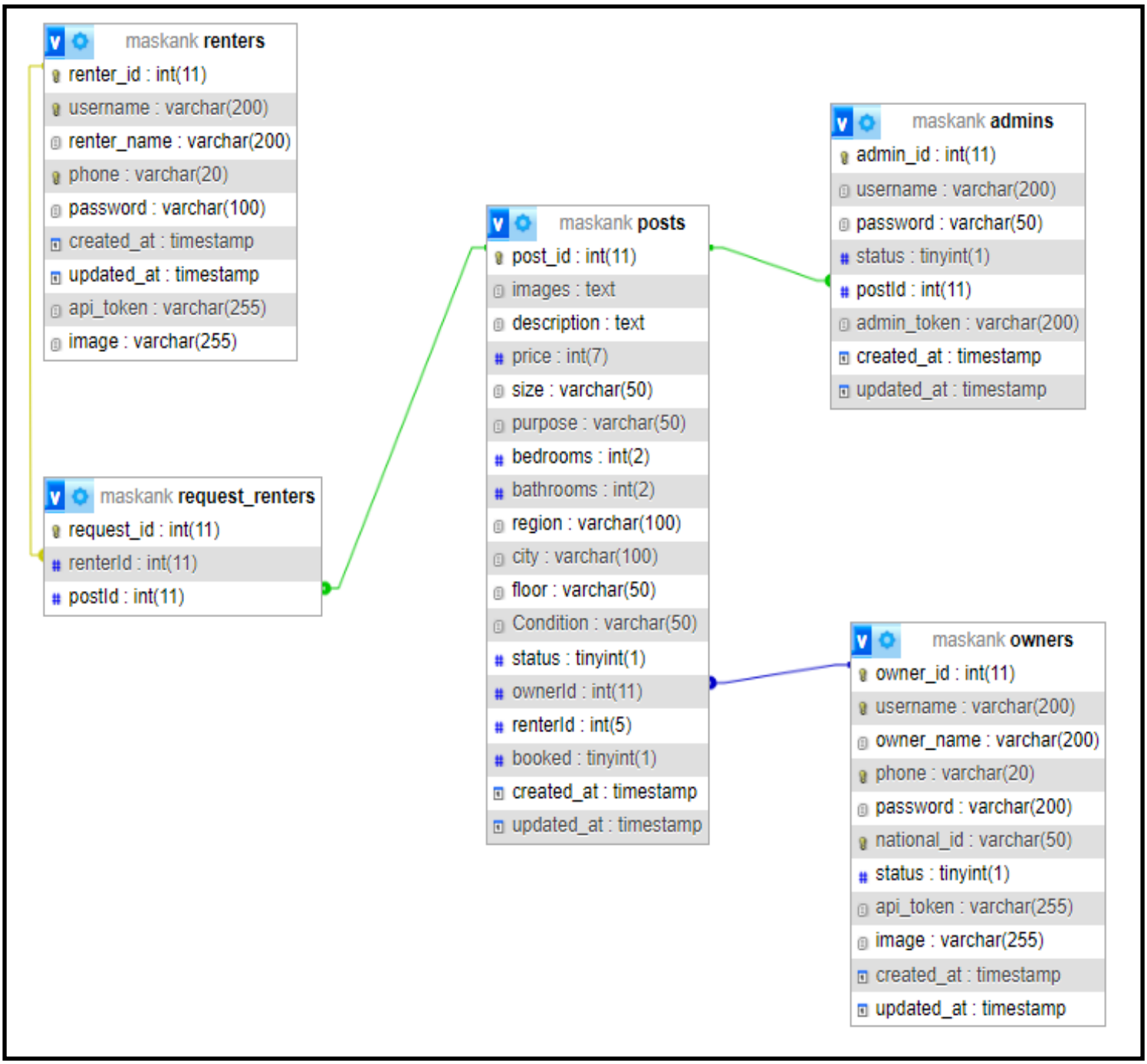
4.2.4 Class Diagram

A class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects. The class diagram is the main building block of object-oriented modeling. It is used for general conceptual modeling of the structure of the application, and for detailed modeling, translating the models into programming code. Class diagrams can also be used for data modeling.The classes in a class diagram represent both the main elements, interactions in the application, and the classes to be programmed.



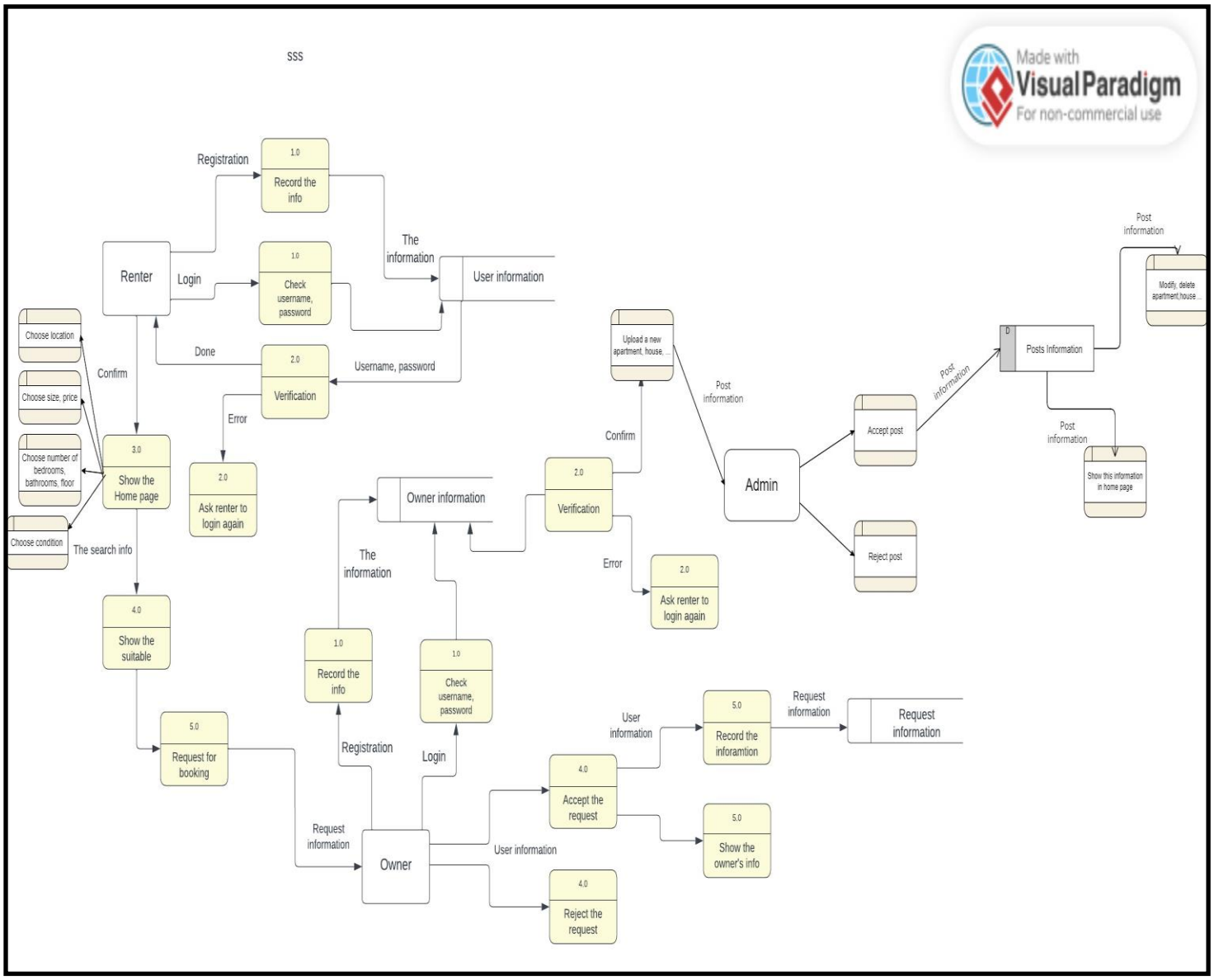
4.2.5 Schema Diagram

A database schema defines how data is organized within a relational database; this is inclusive of logical constraints such as, table names, fields, data types and the relationships between these entities.



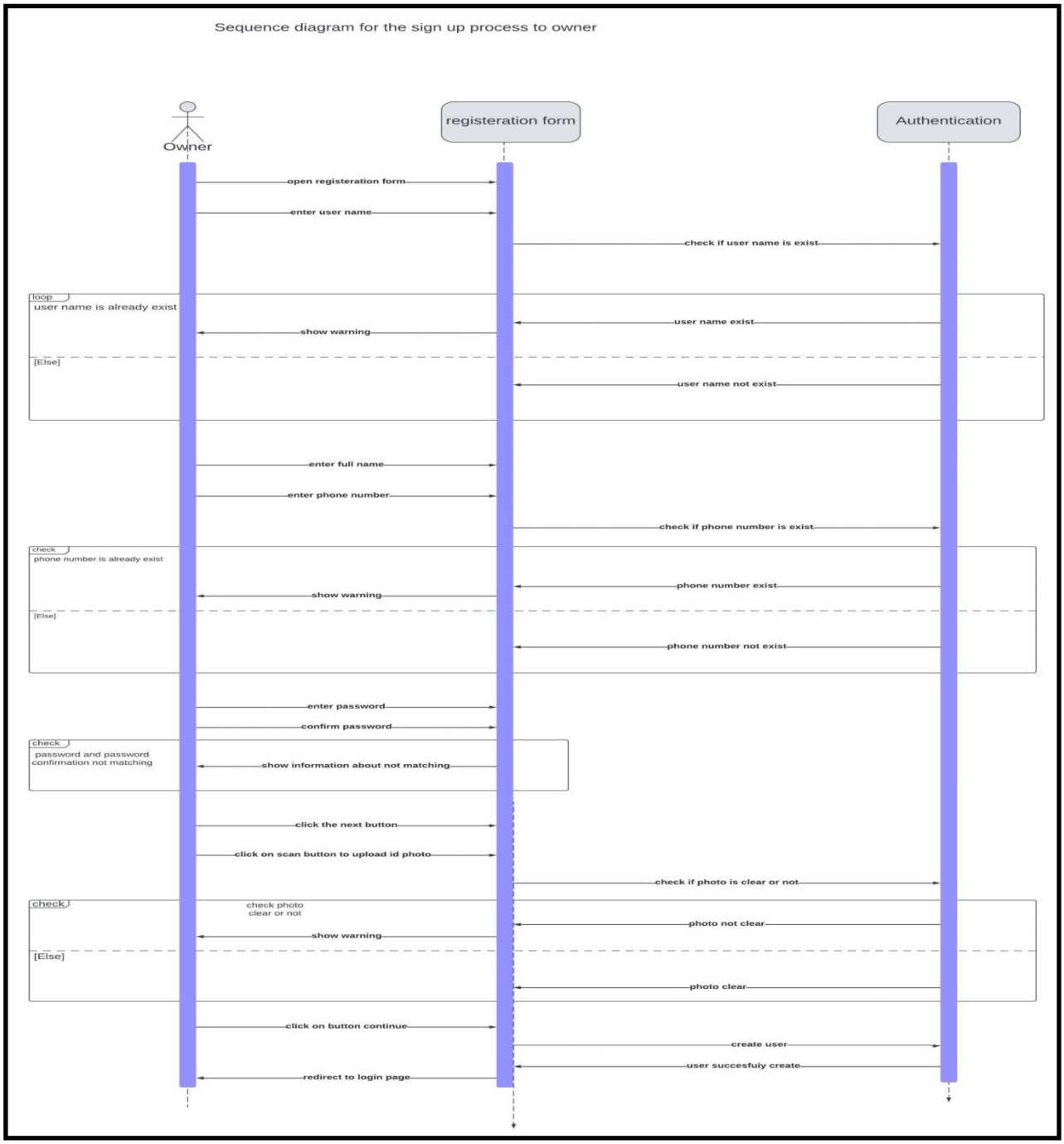
4.2.6 Data Flow Diagram

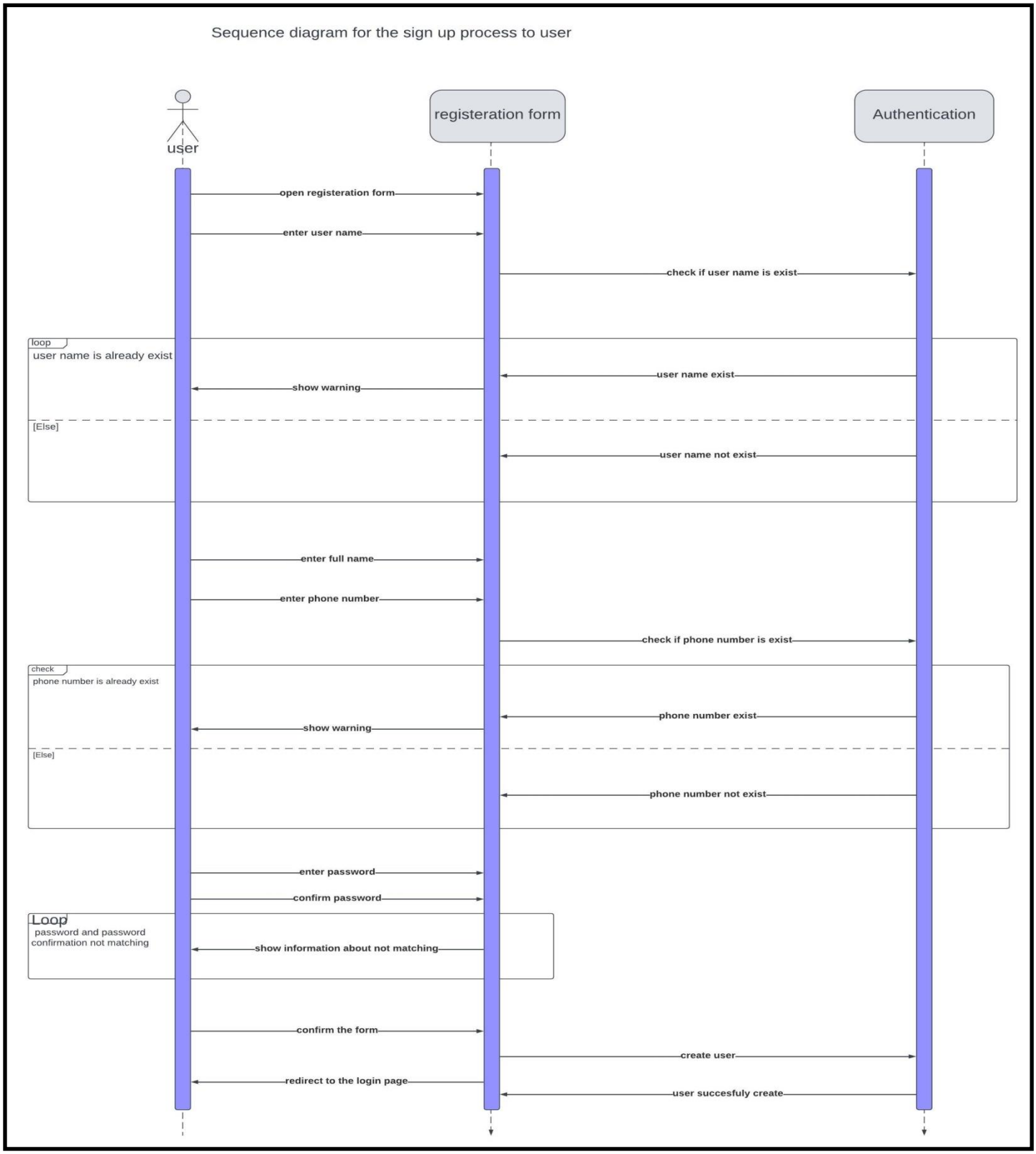
A data-flow diagram is a way of representing a flow of data through a process or a system (usually an information system). The DFD also provides information about the outputs and inputs of each entity and the process itself. A data-flow diagram has no control flow — there are no decision rules and no loops. Specific operations based on the data can be represented by a flowchart.

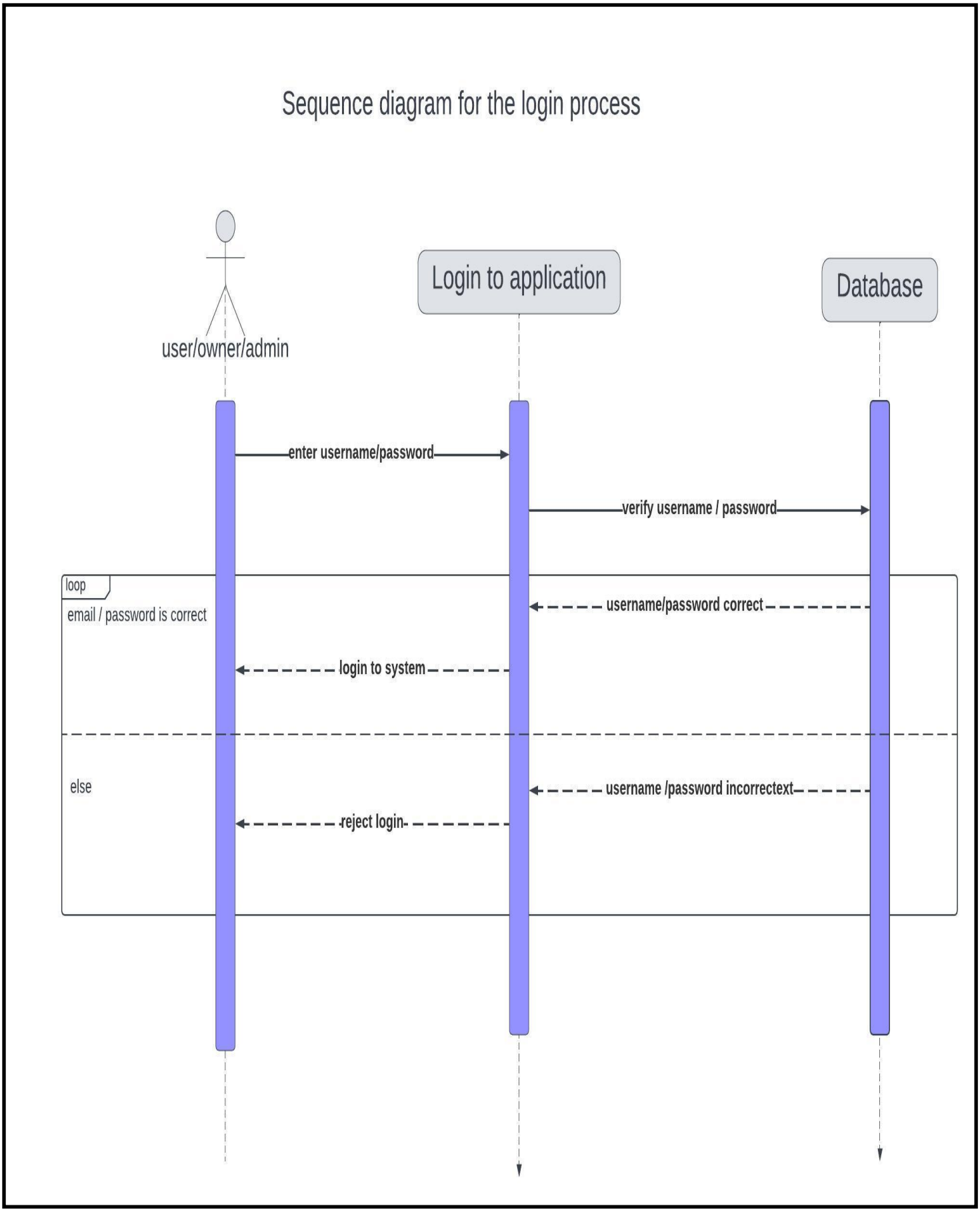


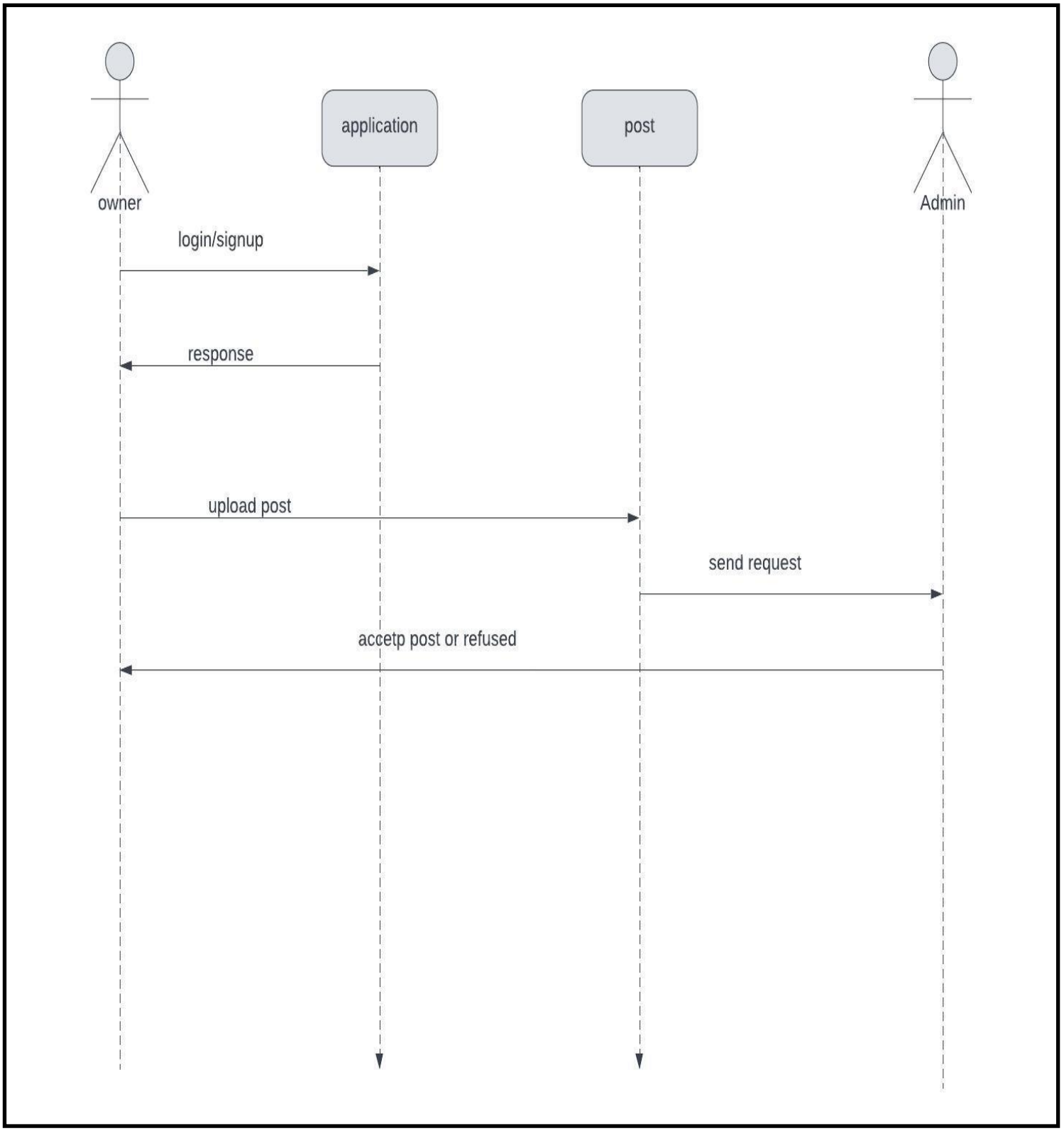
4.2.7 Sequence Diagram

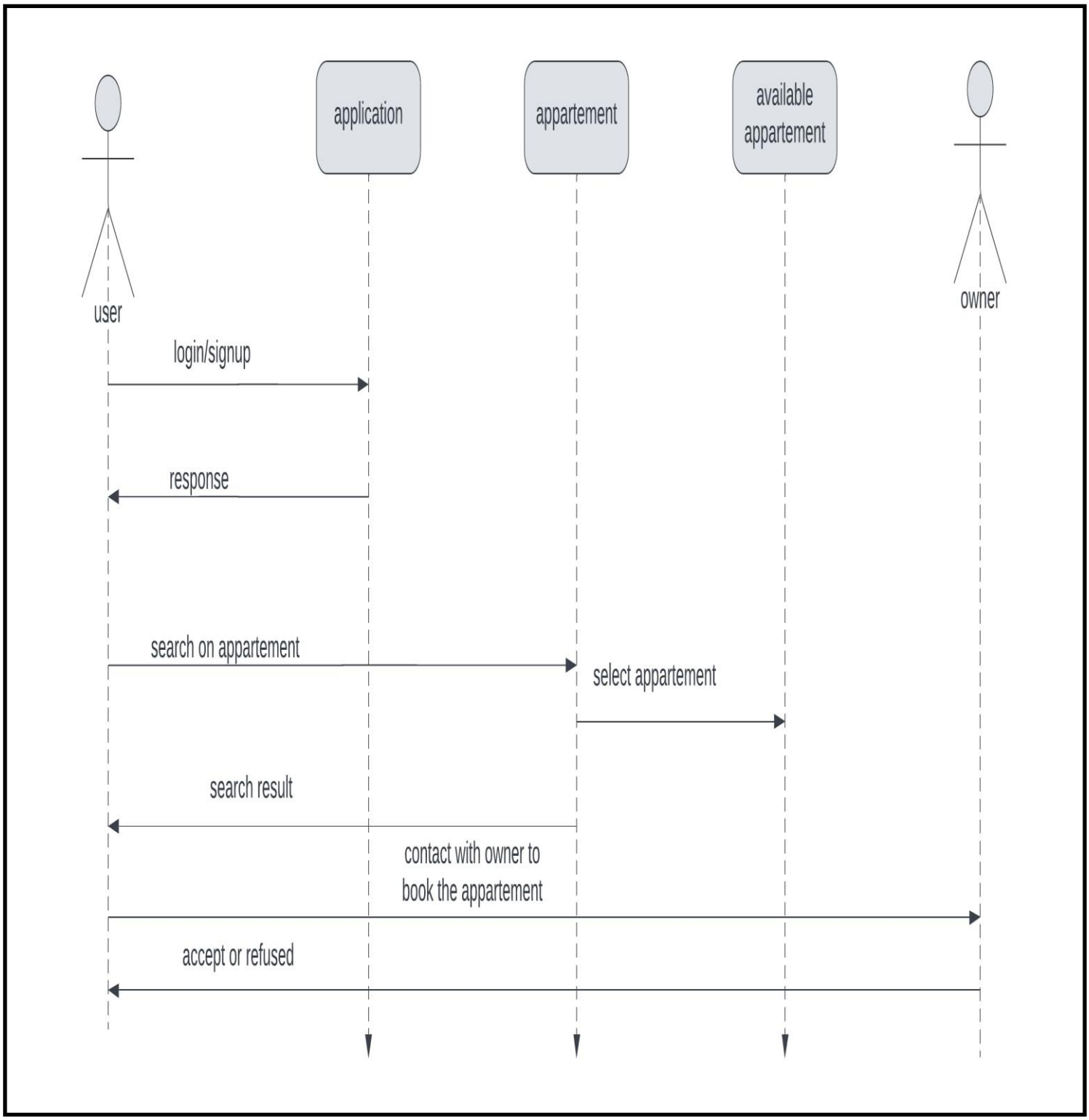
A sequence diagram shows process interactions arranged in time sequence. This diagram depicts the processes and objects involved and the sequence of messages exchanged as needed to carry out the functionality. Sequence diagrams are typically associated with use case realizations in the 4+1 architectural view model of the system under development. Sequence diagrams are sometimes called event diagrams or event scenarios.




Sequence diagram for the sign up process to user
user
registeration form
Authentication
open registeration form
enter user name
check if user name is exist
loop
user name is already exist
user name exist
show warning
[Else]
user name not exist
enter full name
enter phone number
check if phone number is exist
check
phone number is already exist
phone number exist
show warning
[Else]
phone number not exist
enter password
confirm password
Loop
password and password confirmation not matching
show information about not matching
confirm the form
create user
redirect to the login page
user succesfuly create

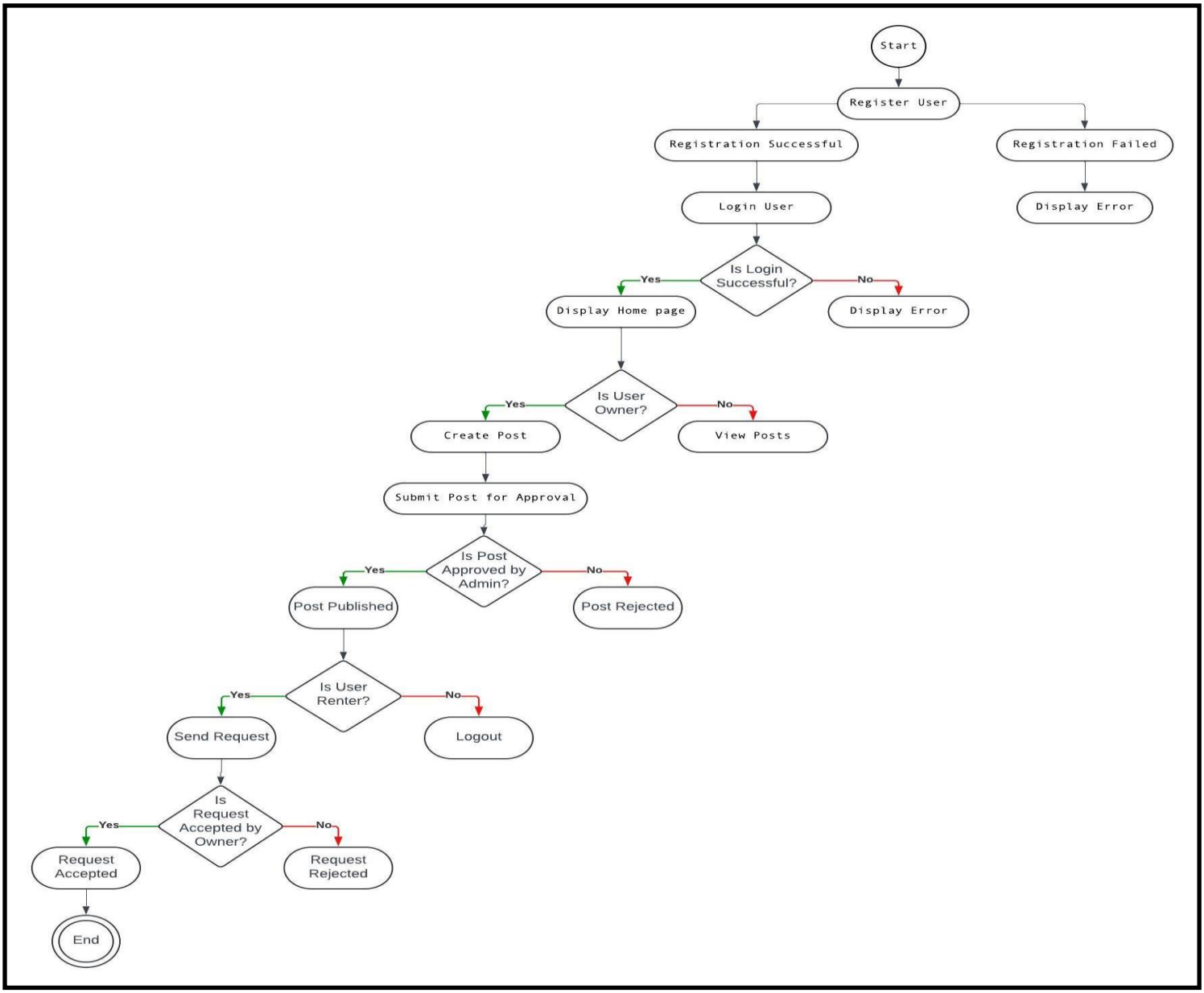

Sequence diagram for the login process
user/owner/admin
Login to application
Database
enter username/password
verify username / password
loop
email / password is correct
username/password correct
login to system
else
username /password incorrectext
reject login


owner
application
post
Admin
login/signup
response
upload post
send request
accetp post or refused


application
appartement
available appartement
user
owner
login/signup
response
search on appartement
select appartement
search result
contact with owner to book the appartement
accept or refused

4.2.8 Activity Diagram

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration, and concurrency. In the Unified Modeling Language, activity diagrams are intended to model both computational and organizational processes (i.e., workflows), as well as the data flows intersecting with the related activities. Although activity diagrams primarily show the overall flow of control, they can also include elements showing the flow of data between activities through one or more data stores.

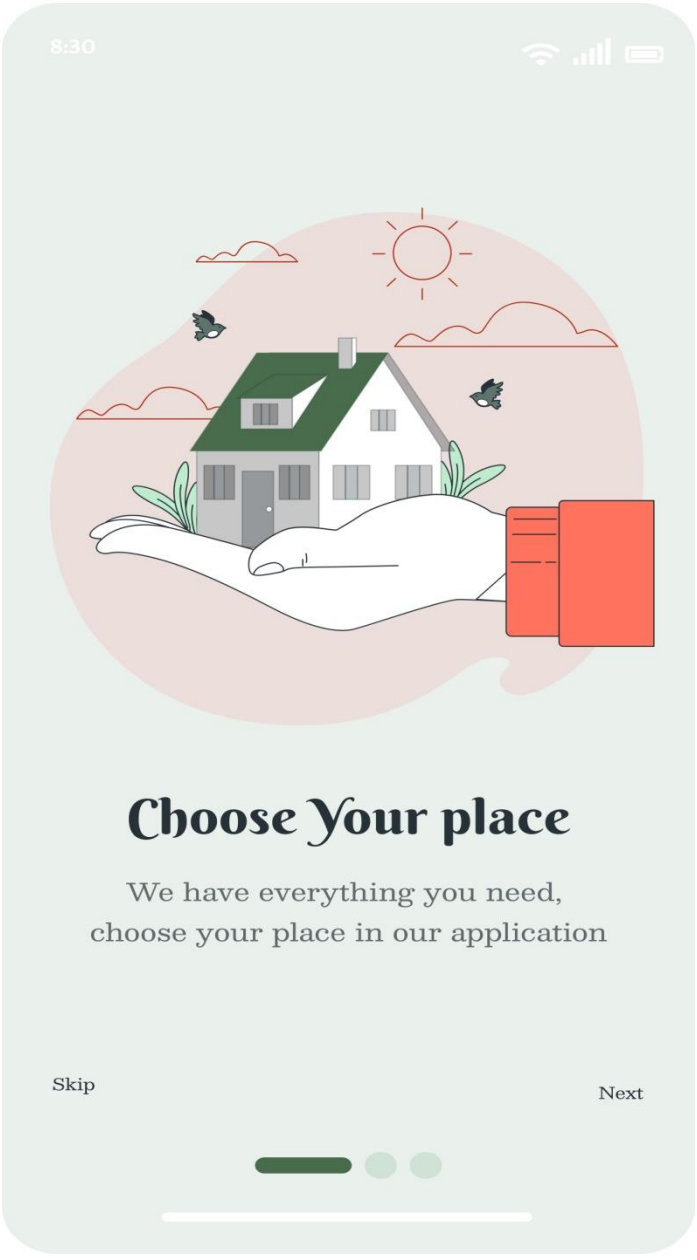


4.3 Interface Design

To design the UI of the application, we focused first on the UX process by creating a problem statement and user personas.

4.3.1 Welcome pages

These pages welcome the user and provide a little overview of the application before they start using it. They also explain why the user should use your **Maskank** application over other applications and what is special about it.



8:30

Easy and save your money

We can make it easy for you to find the place lowest possible price.

Skip

Next

8:30

Easy way to deal

Good communication Save your time with the click of a button.

Skip

Next

4.3.2 Choose Your Role page

After these pages, there is a page for you to choose What's your role:

If a user or owner is chosen: the same login and registration page will be displayed. However, if a owner is chosen, an additional page for him/her will be displayed, to scan his national card to increase security and ensure that this owner is real and prevent false data from entering the application.

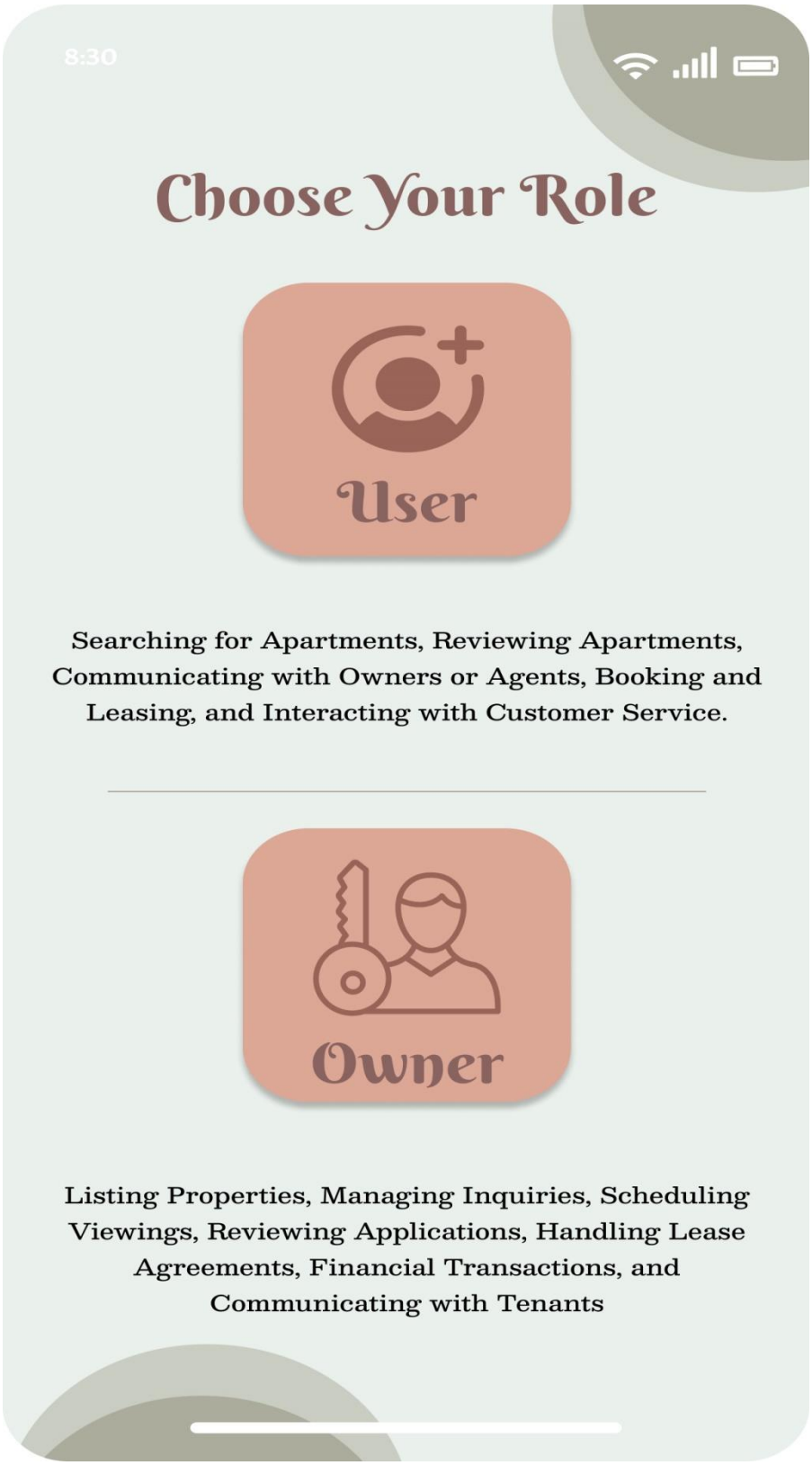
Users' features:

- Searching for Apartments, Reviewing Apartments, Communicating with Owners or Agents, Booking and Leasing, and Interacting with Customer Service.

Owners' features:

- Listing Properties, Managing Inquiries, Scheduling Viewings, Reviewing Applications, Handling Lease Agreements, Financial Transactions, and Communicating with Tenants.

After choosing between a user or a owner, the login page will be displayed.


8:30
Choose Your Role
User
Searching for Apartments, Reviewing Apartments, Communicating with Owners or Agents, Booking and Leasing, and Interacting with Customer Service.
Owner
Listing Properties, Managing Inquiries, Scheduling Viewings, Reviewing Applications, Handling Lease Agreements, Financial Transactions, and Communicating with Tenants

4.3.3 Login page

If an account has been previously created, the user or owner is required to enter:

1. username
2. password

Then press the login button.

We verify this data in the Database and

then log in successfully.

If he does not have an account,

click on “Sign up” at the bottom of the page.

The same login page appears for the user

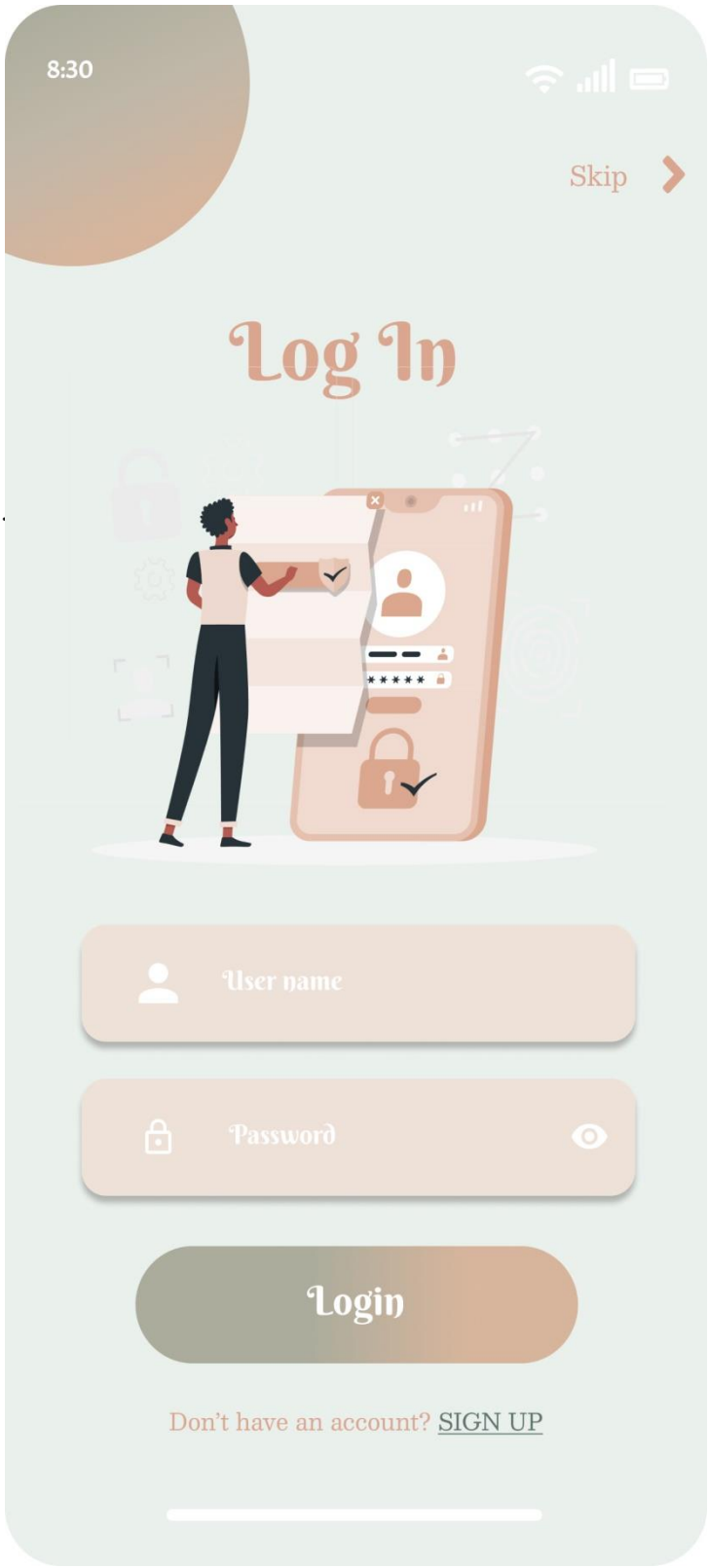
and the owner.

When you click the Skip button,

you will be directed directly to

the home page in the application

without the need to log in.



4.3.4 Registration page

If you do not already have an account and want to create a new account, you must enter this data:

1. username
2. full name
3. phone
4. password
5. confirm password

Then press the "Sign up" button.

- If you are a user: you will be taken directly to the home page.
- If you are a owner: you will be taken to the page for scanning your national card.

On the page of scanning the owner’s national card:

the national number will be obtained and then entered in our database. Then only real owners will be allowed to enter the application in order to increase security, and then all the data displayed within the application will be largely correct and will not contain fake data.

- After verifying the validity of the national ID, the verification success page will be displayed.
- If the national ID is incorrect, the registration process will fail.

8:30

Register

User name

Full name

Phone

Password

Confirm Password

Sign Up

Already have an account? LOG IN

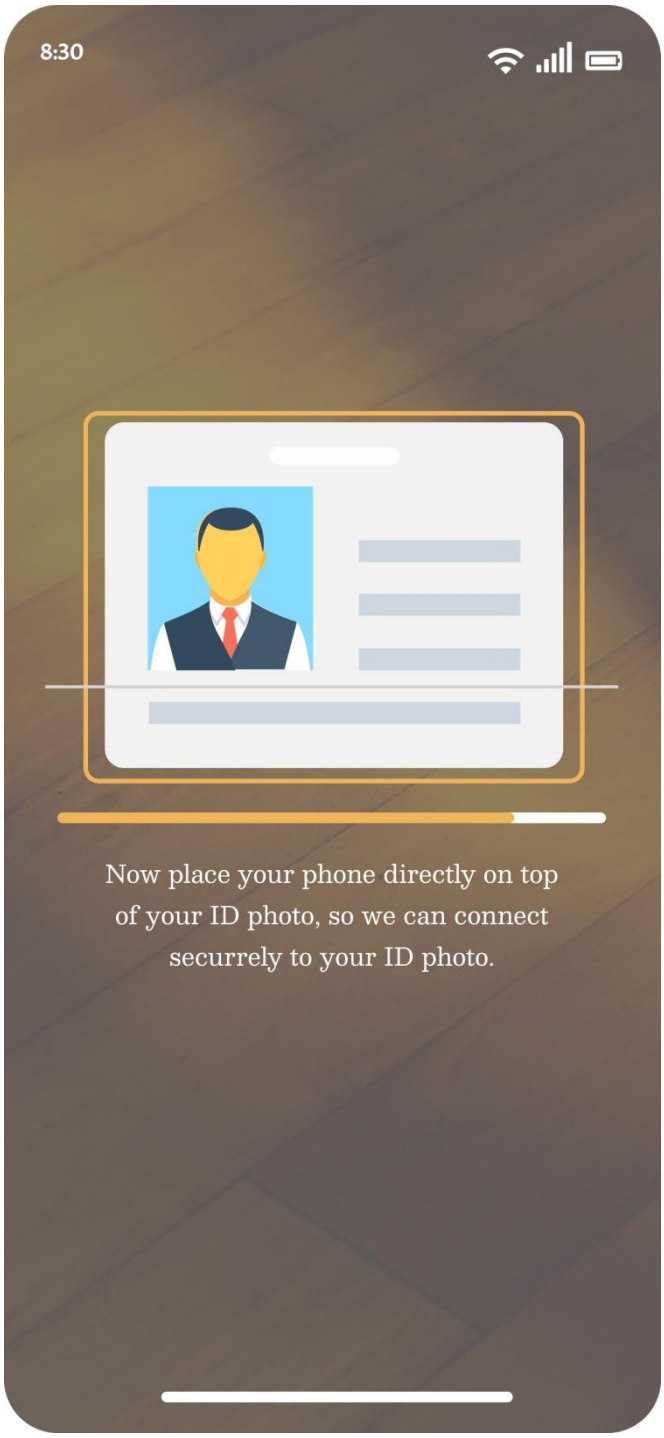
8:30

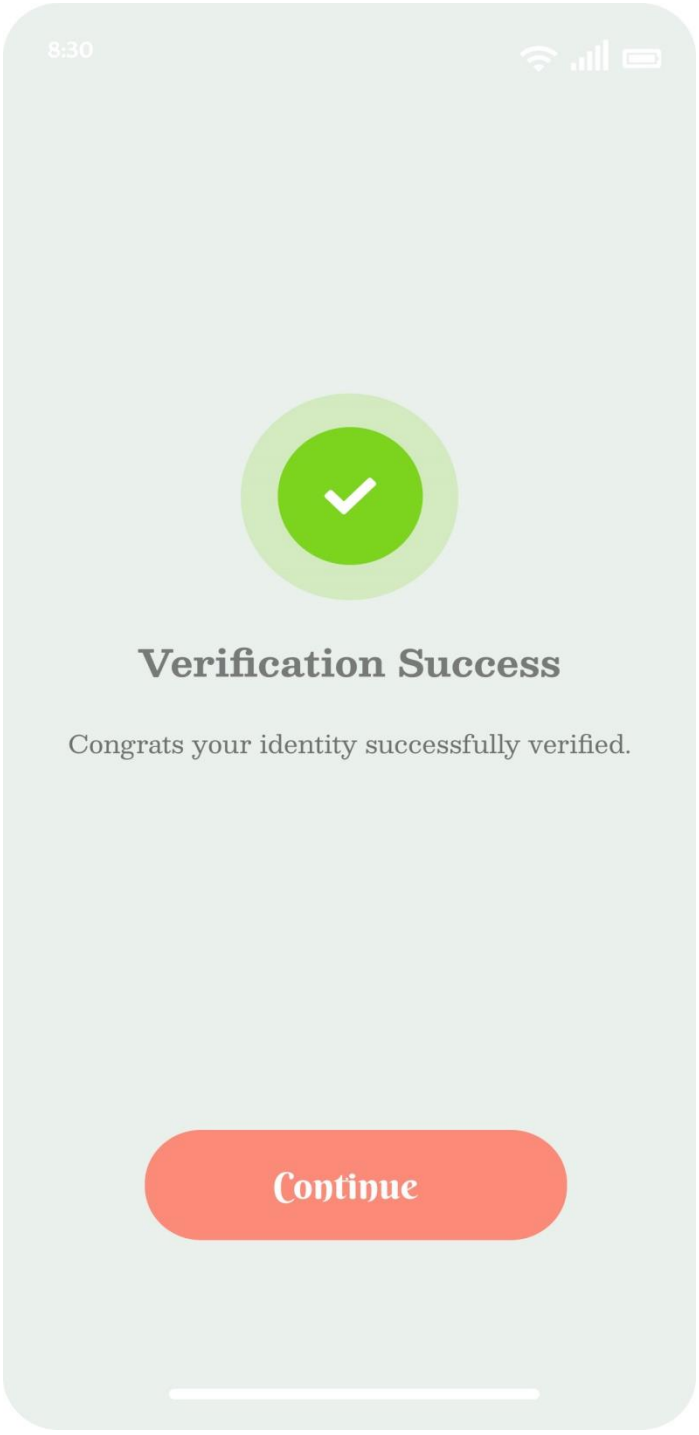
Scan Your ID

Now place your phone directly on top of your id photo so we can connect securely

Position your photo inside the box

Scan Now


8:30
Now place your phone directly on top of your ID photo, so we can connect securrely to your ID photo.


8:30
Verification Success
Congrats your identity successfully verified.
Continue

4.3.5 Home page (User)

This page serves as the starting point for users, providing them with easy access to key features and relevant information. The following description outlines the main components of the home user page

In this page there are two main section

- Recent Search Section:

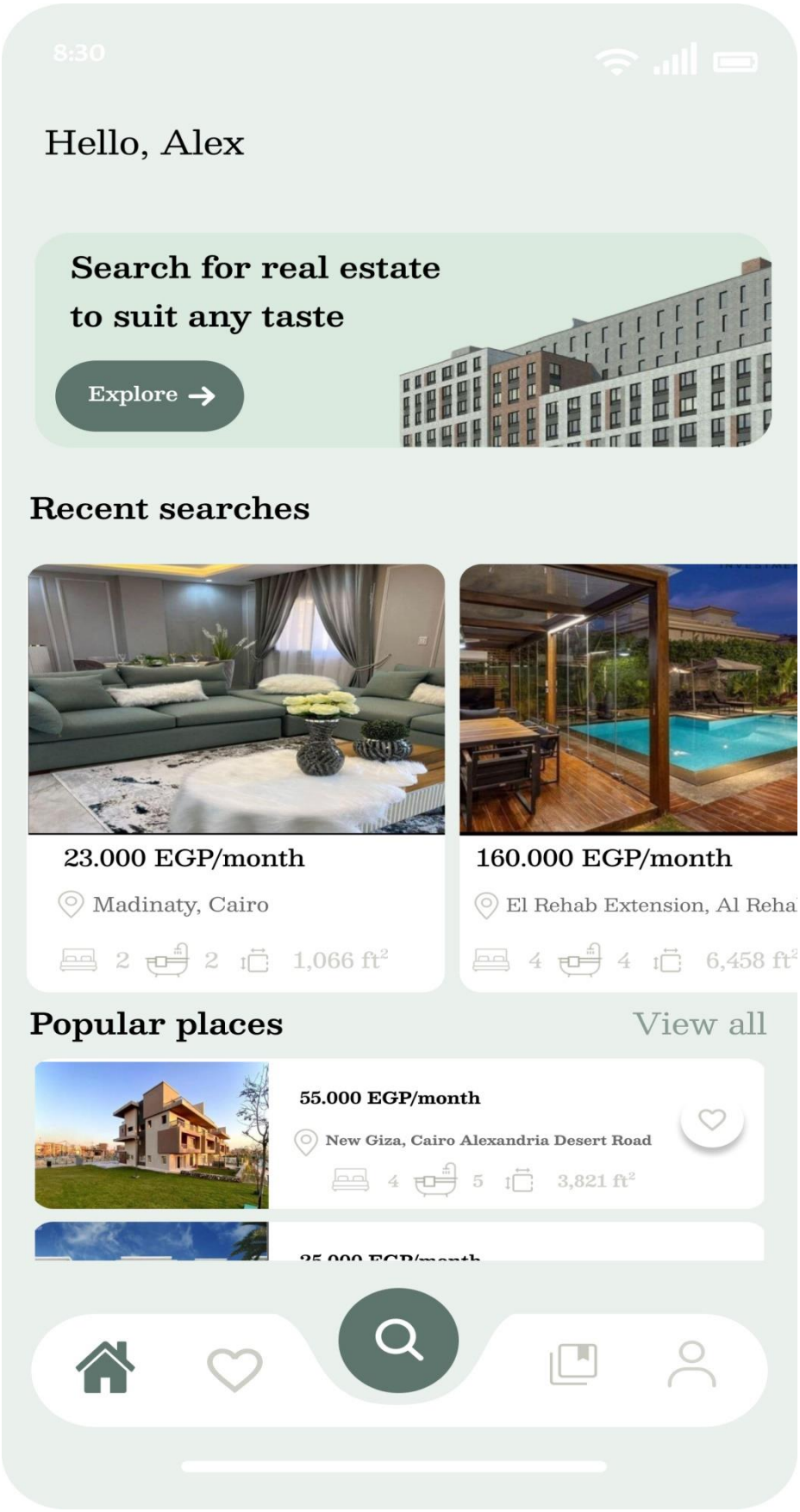
  This section displays the user's most recent apartment search queries, offering quick access to their past searches. Each recent search is presented in a visually appealing manner, showcasing relevant details such as location, price range, number of bedrooms , and number of bathrooms , and the size. Users can simply click on a recent search to revisit the search results, saving them time and effort.

- Popular Places:

  Adjacent to the recent search section, the home user page highlights a "Popular Places" area. This section showcases popular neighborhoods or areas that are frequently searched for by users. It provides users with a glimpse of the most sought-after locations.

- Navigation Bar:

  The home user page features a sleek and intuitive navigation bar at the bottom of the screen. This navigation bar includes essential options such as "Home," "Search," "Saved Apartments," " favorite," and "User Profile." It allows users to seamlessly navigate between different sections of the app, ensuring a smooth user experience.


8:30
Hello, Alex
Search for real estate
to suit any taste
Explore
Recent searches
23.000 EGP/month
Madinaty, Cairo
2 2 1,066 ft²
160.000 EGP/month
El Rehab Extension, Al Reha
4 4 6,458 ft²
Popular places
View all
55.000 EGP/month
New Giza, Cairo Alexandria Desert Road
4 5 3,821 ft²

4.3.6 Home page (Owner)

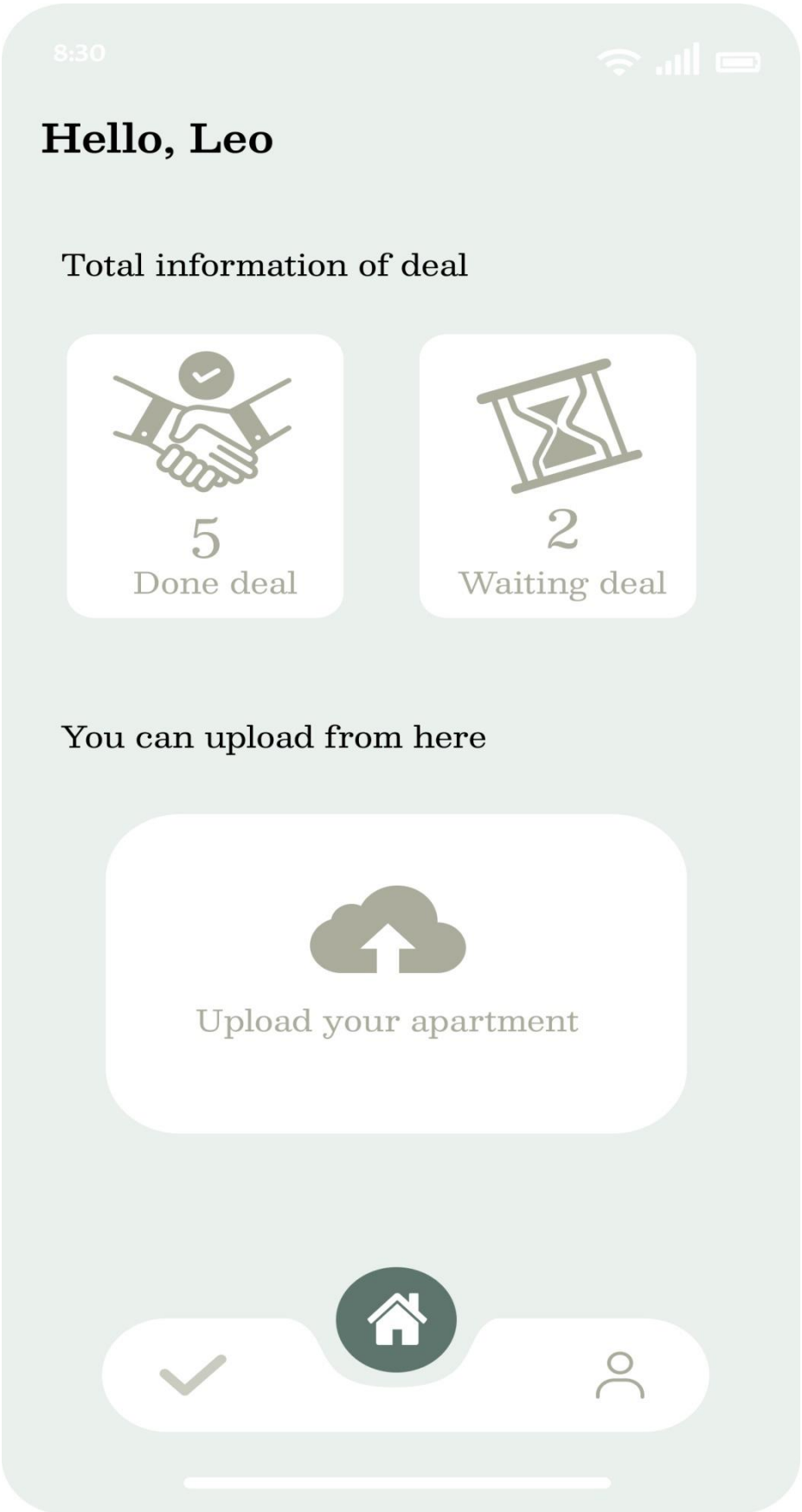
On this page, there are three main section

- Done Deals Section:
  This section presents a list of previously completed rental agreements.

- Waiting Deals Section:
  Adjacent to the done deals section, the Home (owner) page includes a "Waiting Deals" area. This section displays apartments that have received rental inquiries or have potential tenants interested in renting them. It allows homeowners to keep track of pending deals and manage communication with prospective tenants.

- Upload Apartment Button:
  Prominently positioned on the home (owner) page, there is a dedicated "Upload Apartment" button. This button enables homeowners to easily add a new apartment to their rental property portfolio. By clicking the button, homeowners can access a streamlined form where they can enter comprehensive details about the apartment, such as location, number of bedrooms and bathrooms, amenities, and photos. The upload process is designed to be intuitive and efficient, allowing homeowners to quickly list their available apartments.

- Navigation Bar:
  The home (owner) page features a sleek and intuitive navigation bar at the bottom of the screen. This navigation bar includes essential options such as "User Profile," " approved post," " Home,"


8:30
Hello, Leo
Total information of deal
5
Done deal
2
Waiting deal
You can upload from here
Upload your apartment

4.3.7 Favorite page

On this page, there are only section

The main section of the Favorite Page is dedicated to displaying the user's favorite apartments. Each apartment listing is presented as a visually appealing card, featuring a thumbnail image, basic details (such as location, number of bedrooms, and rental price), and icons representing key amenities. Users can easily scan through the grid to view their saved apartments at a glance.

Navigation Bar:

The favorite page features a sleek and intuitive navigation bar at the bottom of the screen. This navigation bar includes essential options such as:

- Home
- Search
- Saved Apartments
- Favorite
- User Profile

It allows users to seamlessly navigate between different sections of the app, ensuring a smooth user experience.

8:30

Favourite

4,691,004 EGP

Nyoum mostakbl, Mostakbal City Compounds

3 3 1,539 ft²

11,000,000 EGP

Badya Palm Hills, 6 October Compounds

4 5 2,153 ft²

4.3.8 Booked page

When the user clicks on the "Book" button at the bottom of any post (apartment - house - room...) to request communication and inform the owner that he wants to book this, then all the posts that have been reserved will be saved on this page to be returned to at any time he wants.

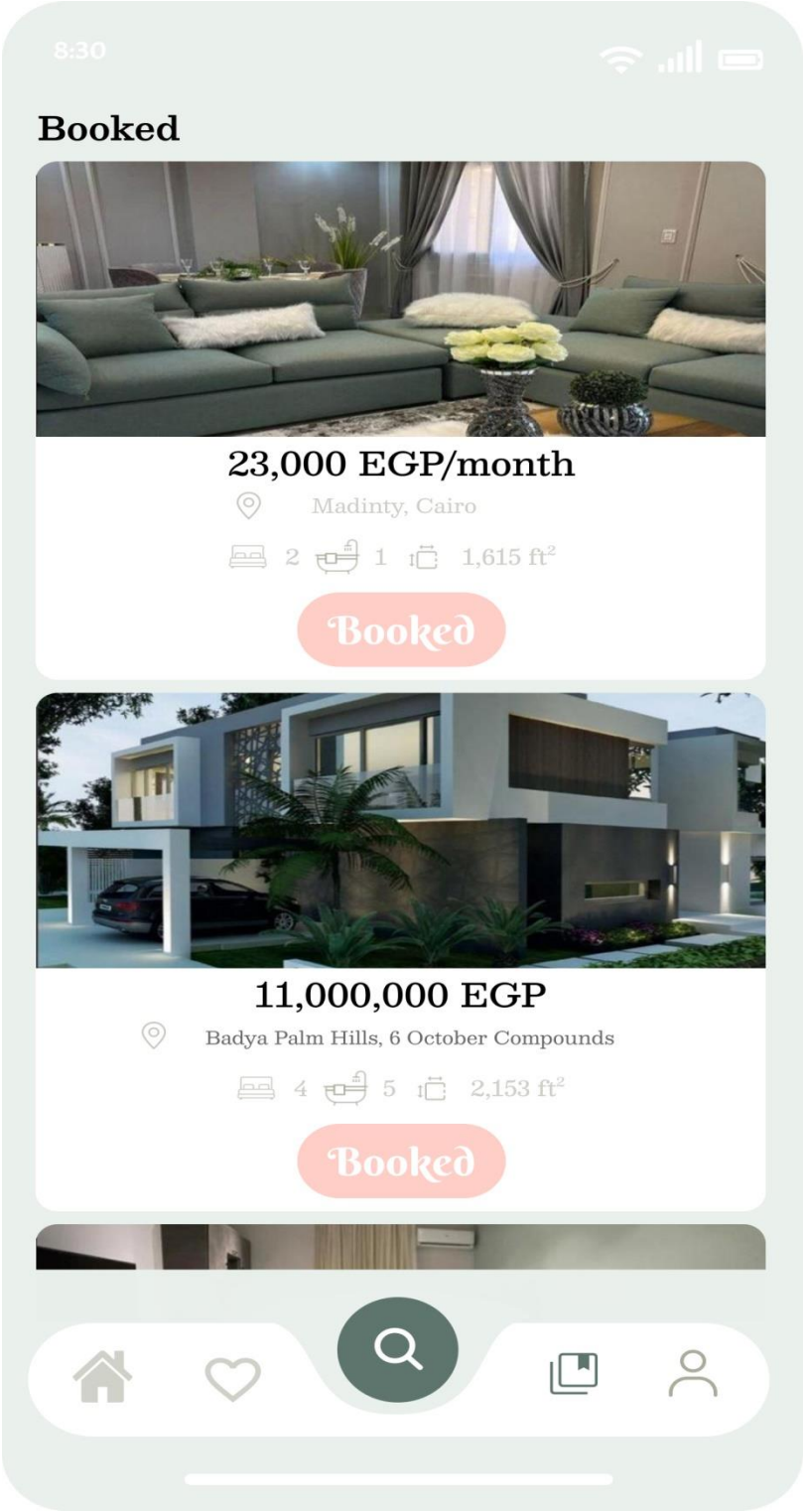
Also, when the owner approves

the reservation request,

the owner’s contact information

will be displayed to the user

to can connect with him/her.



4.3.9 Search page

On this page, the user searches for the specifications (apartment - house - room...) that he wants by typing in the search box at the beginning of this page.

Then what he wants will appear in the Recommendation section.

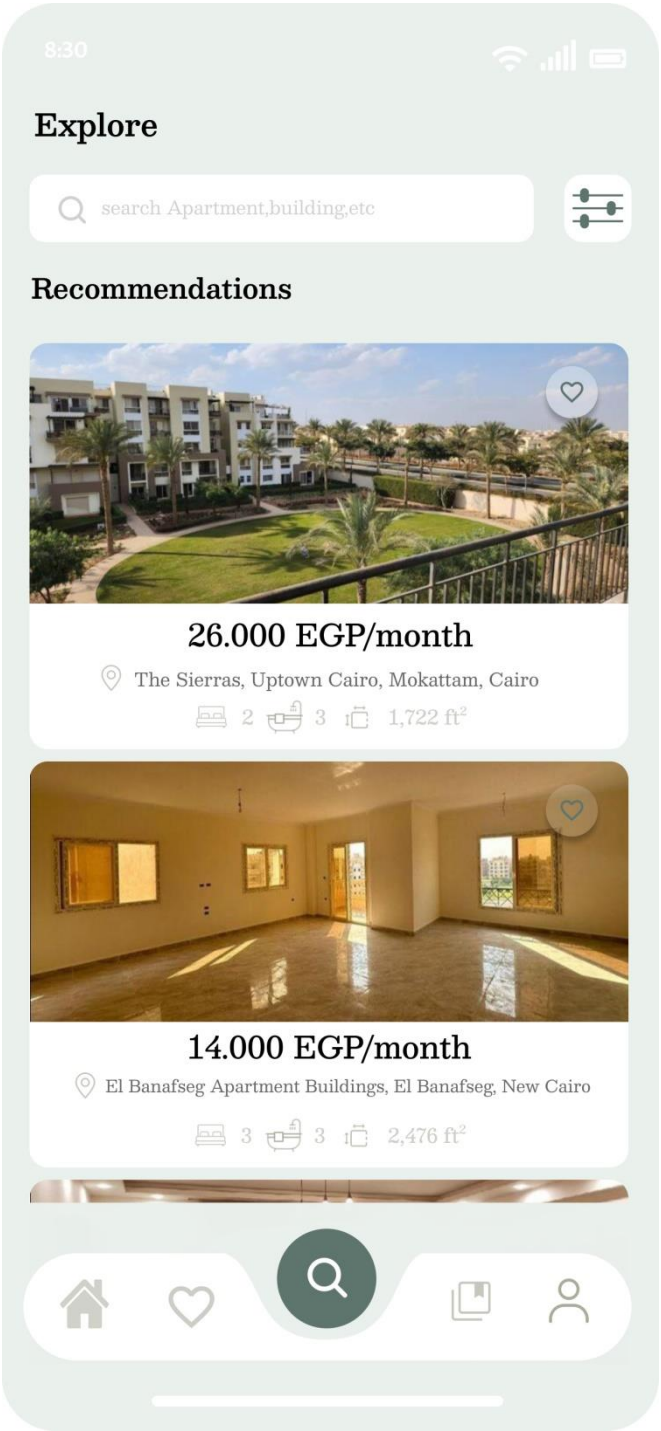
Data that will appear:

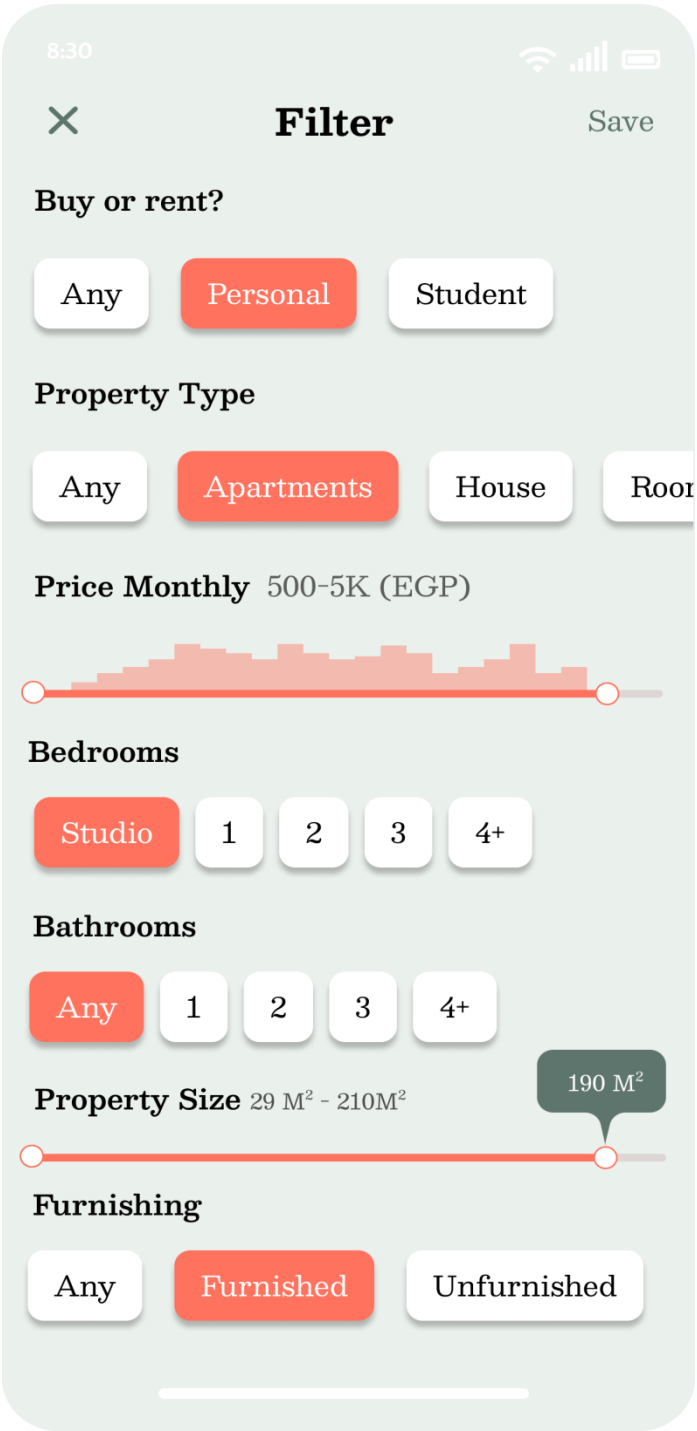
- Image
- the price
- Place
- Number of bedrooms
- Number of bathrooms
- Total size

If the user wants to use another method of searching other than writing, he will use the filter feature, through which he will only choose several options shown in front of him, and through them the application will show the best results based on his choices.

Including:

- the price
- the size
- bedrooms
- Bathrooms
- Furnished or not?
- the purpose

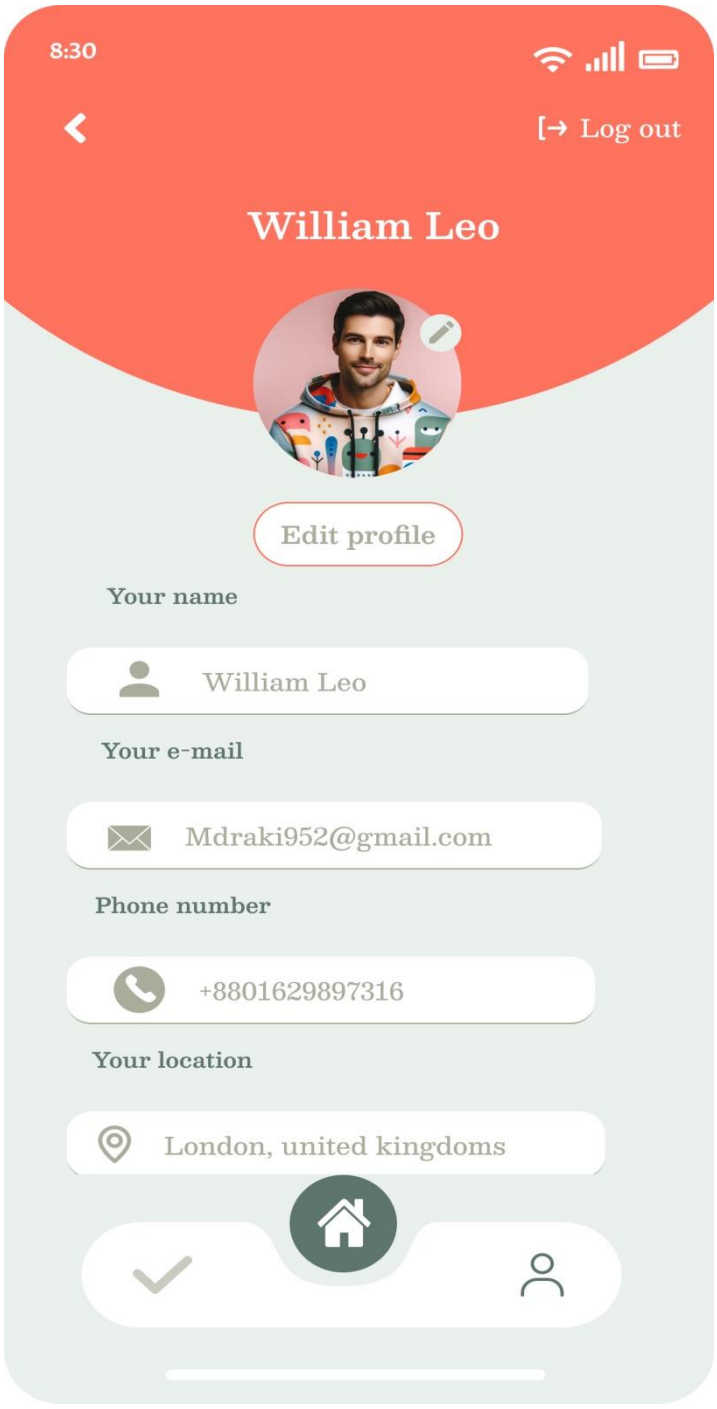

8:30
Explore
search Apartment,building,etc
Recommendations
26.000 EGP/month
The Sierras, Uptown Cairo, Mokattam, Cairo
2 3 1,722 ft²
14.000 EGP/month
El Banafseg Apartment Buildings, El Banafseg, New Cairo
3 3 2,476 ft²

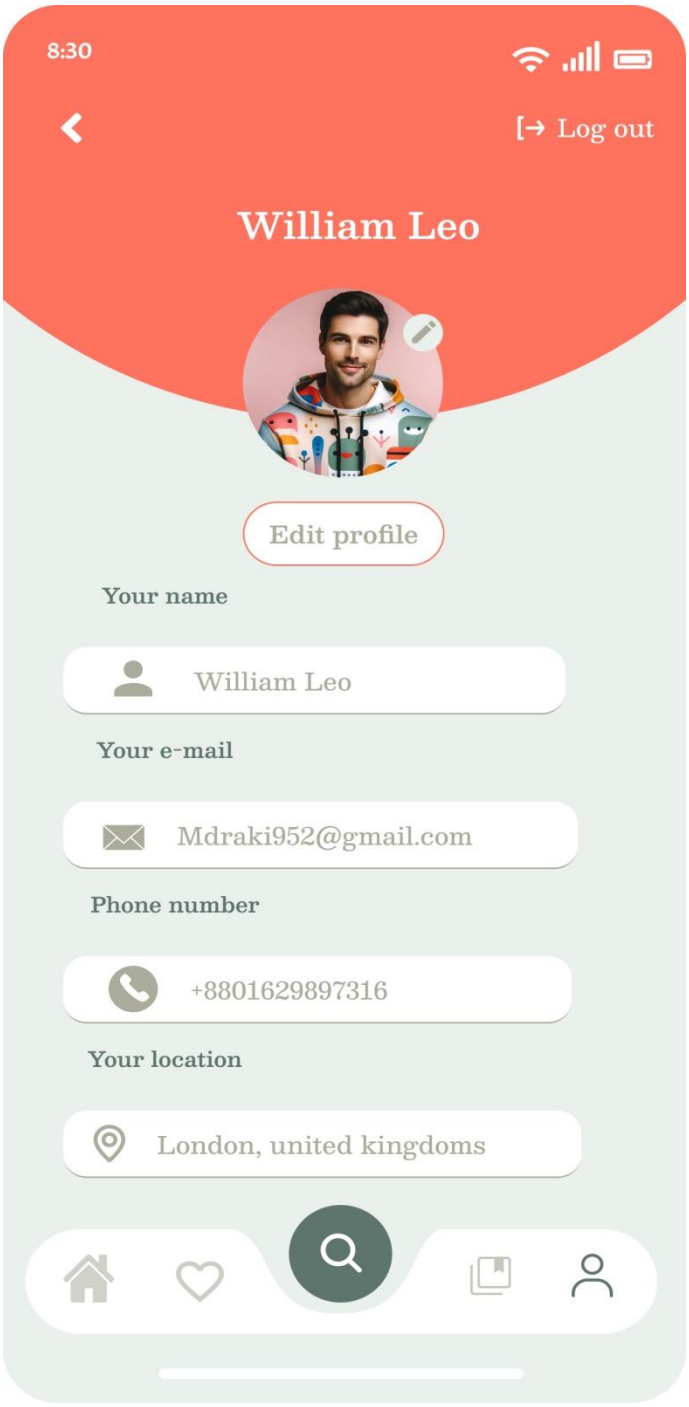

8:30
Filter
Save
Buy or rent?
Any
Personal
Student
Property Type
Any
Apartments
House
Price Monthly 500-5K (EGP)
Bedrooms
Studio
1
2
3
4+
Bathrooms
Any
1
2
3
4+
Property Size 29 M² - 210M²
190 M²
Furnishing
Any
Furnished
Unfurnished

4.3.10 Profile page

On this page, the user or owner can easily modify his data by clicking on the edit profile button, such as:

(Image - Full name – Username – Phone – Location)





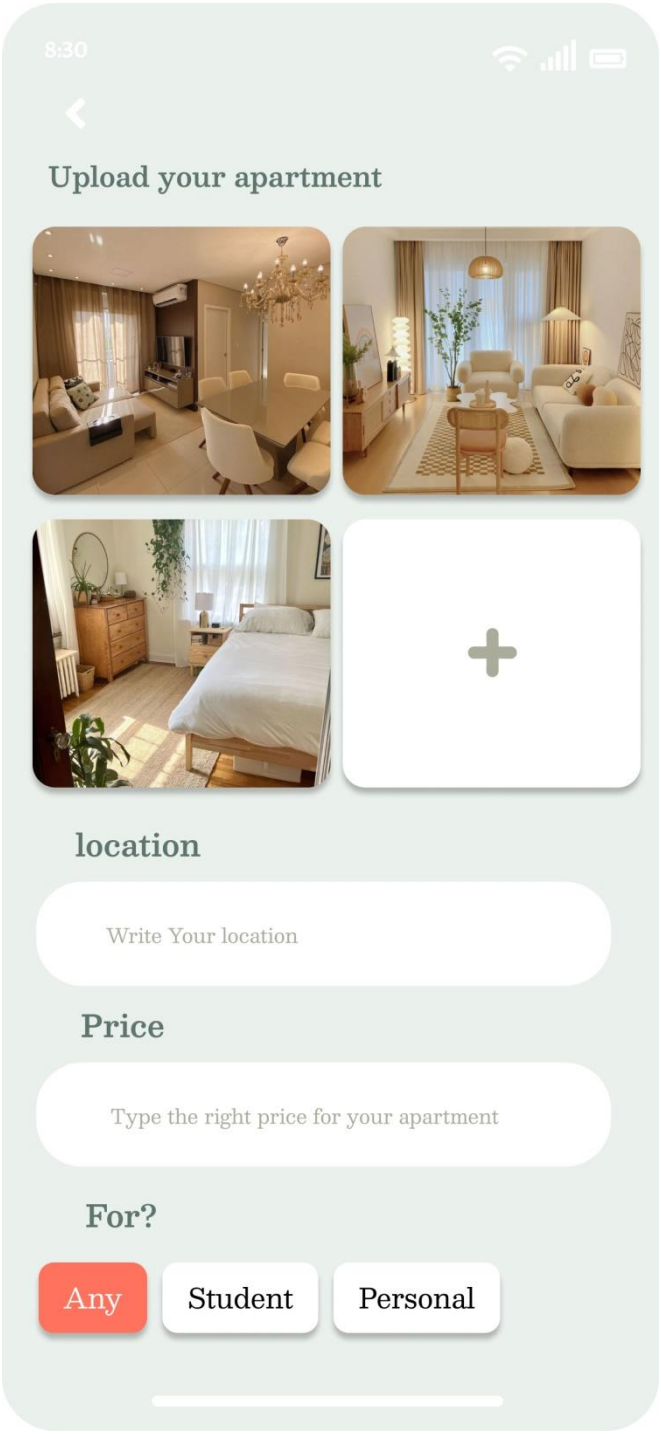
4.3.11 Upload Apartment

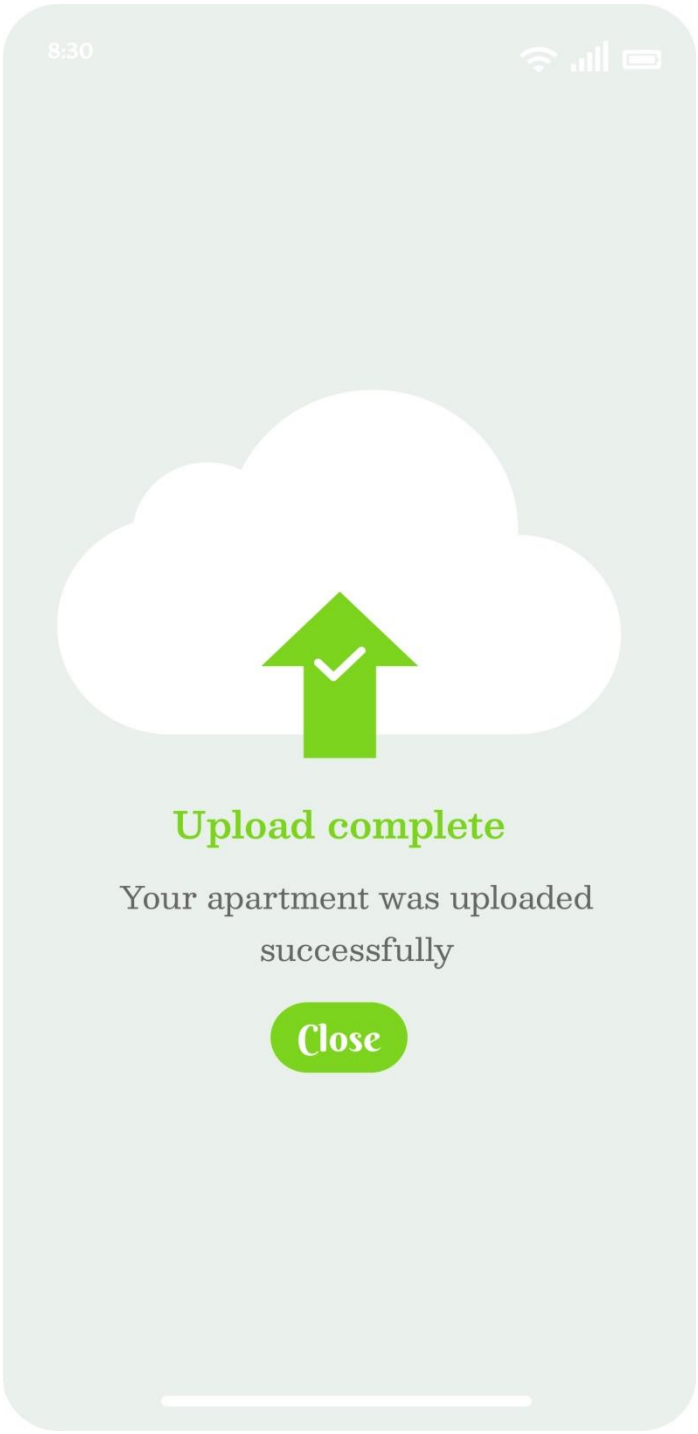
The main section of the Upload Apartment page consists of a comprehensive apartment information form. This form allows homeowners to enter specific details about their apartment for effective listing. The form typically includes the following fields:

1. Photos: Homeowners can upload multiple photos of their apartment, showcasing different areas and angles. This feature enables homeowners to present their apartment visually and attractively.
2. Location: Homeowners can input the address This helps potential renters understand the apartment's proximity to desired areas and amenities.
3. Type: Homeowners can choose the type of apartment, such as "Student," "Personal," or other relevant categories. This selection assists renters in finding apartments that align with their specific needs.
4. Bedrooms and Bathrooms: Homeowners can indicate the number of bedrooms and bathrooms in the apartment This information helps renters filter their search based on their desired space requirements.
5. Property Type: Homeowners can specify the property type, such as "Apartment," "House," "Duplex," or other relevant options.
6. Description: Homeowners can provide a detailed description of the apartment This description helps potential renters gain a deeper understanding of the apartment's unique qualities.

Upload Button:

After homeowners have entered all the necessary information in the apartment information form, an "Upload" button is prominently displayed. Clicking this button triggers the process of uploading the apartment listing to the app's database.


8:30
Upload your apartment
location
Write Your location
Price
Type the right price for your apartment
For?
Any
Student
Personal


8:30
Upload complete
Your apartment was uploaded successfully
Close

4.3.12 Post Details

This page provides users with comprehensive information about a selected apartment

- Apartment Photos:
    - The main section of the Apartment Details page showcases high-quality photos of the apartment.
    - Adjacent to the apartment photos, the page displays essential information about the apartment. This includes the property type, such as "Apartment," "House," " Duplex," or other relevant categories. Additionally, the page provides details about the apartment's size. Users can also view the number of bedrooms and bathrooms available in the apartment, helping them assess its suitability for their needs.
- Amenities and Features:
    - The Apartment Details page presents a list of amenities and features available in the apartment. This section highlights the apartment's unique selling points, such as air conditioning, in-unit laundry, parking availability, fitness facilities, or any other amenities that may be relevant to potential renters. Users can quickly scan this list to determine if the apartment meets their desired criteria.
- Description:
    - Below the amenities and features section, the page includes a detailed description of the apartment. This description is written by the property owner and provides additional information about the apartment's characteristics, layout, and any specific details that may be important to potential renters.
- Booking Button:
    - After reviewing the apartment details, users will find a prominently displayed "Book" button.

8:30

23,000 EGP/month

Furnished apartment for rent, first residence

Property Type:	Apartment
Property Size:	1,066 ft² (99m²)
Bedrooms:	2
Bathrooms:	2

Amenities

- Furnished
- Central A/C
- Built in Wardrobes
- Balcony
- Kitchen Appliances
- Security
- Lobby in Building
- Walk-in Closet

Book

8:30

23,000 EGP/month

Furnished apartment for rent, first residence

Property Type:	Apartment
Property Size:	1,066 ft² (99m²)
Bedrooms:	2
Bathrooms:	2

Amenities

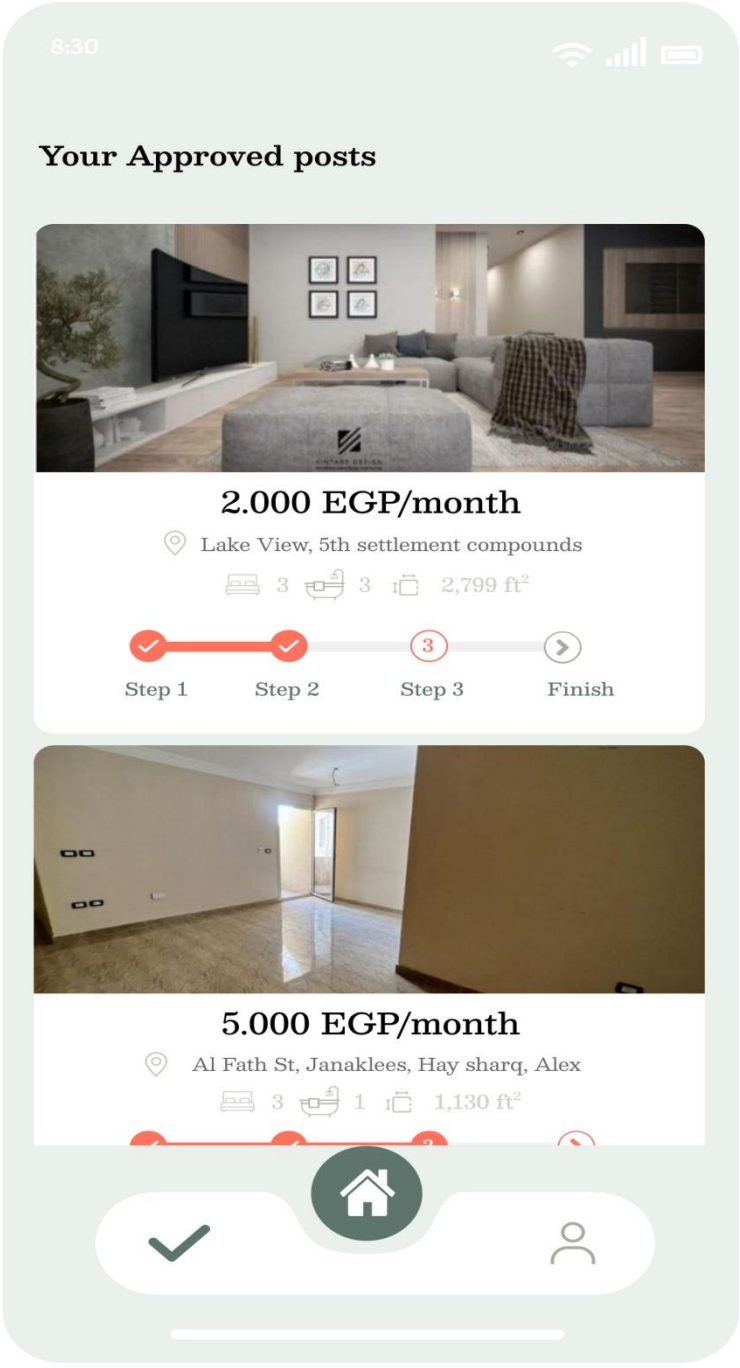
- Furnished
- Central A/C
- Built in Wardrobes
- Balcony
- Kitchen Appliances
- Security
- Lobby in Building
- Walk-in Closet

Book

4.3.13 Approved Posts

These This page allows property owners to review and control the acceptance or refusal of user bookings.

The main section of the Approved Post page displays a list of booking requests submitted by users



4.3.14 Admin Panel

on this page can the admin registered with us in Data Biz (only one admin) manage the entire application, as he has the authority to accept or reject posts that the owner uploads to the application and then display them in the application to the user.

The admin also has the right to delete the view of the entire number of posts on the application, the number of users, and the number of owners. You can also see the entire application's statistics on this page.

Chapter 5

Data Analysis

5.1 Introduction

In previous chapters, we have demonstrated the potential requirements and limitations of our system. In this chapter, we provide an implementation tool, and provide a sample of the most important codes we used to build our prototype and application

5.2 Predicting Rent Prices

5.2.1 **Predicting rent prices using AI** is a crucial feature for a house rental app. It provides valuable insights to both users and app owners, enabling informed decision-making and enhancing the overall user experience .

5.2.2 **Based the feature that we found** , we thought that it is possible to design a machine learning model capable predicting the price of the house more accurate than the human and more fair.

The process started by asking some questions:

I. What are the main feature for etermining the price of the house?
II. What is needed data?
III. What are the cases that I should consider?

5.2.3 Cases should be considered:

- **Limited data:** What happens if there's a limited amount of rental data available in a specific location? Can the model still provide reasonable estimates?
- **Data bias:** How will you address potential biases in the training data? For example, if the data primarily consists of high-end properties, the model might underestimate rents for more affordable ones.
- **Missing values:** How will you handle missing data points for certain property attributes? Can you impute missing values based on other available data or remove incomplete entries?
- **Rapid market changes:** How will the model adapt to sudden shifts in rent prices due to economic factors or new developments in an area?
- **Unique property features:** Can the model account for unique features that may not be explicitly included in the data, such as exceptional views or historical significance?

- **Seasonality:** Does the model consider seasonal variations in rent prices (e.g., higher rents during peak tourist seasons)?
- **Model accuracy:** What level of accuracy is considered acceptable for the rent predictions? How will you measure and communicate the model's accuracy to users?
- **Explainability:** Can you provide users with some explanation for the predicted rent price? This can build trust and transparency.
- **User feedback loop:** How will you incorporate user feedback on the prediction accuracy to improve the model over time?

5.2.4 In the future: What can I do to improve the model?

- **Regular data updates:** Regularly update your training data with fresh information to reflect changing market trends.
- **Model monitoring:** Monitor the model's performance over time and retrain it periodically to maintain accuracy.
- **Error handling:** Implement mechanisms to handle unexpected situations and provide informative messages to users if predictions seem unreasonable.

The steps followed:

- Data Collection and Preprocessing
    - Identify data sources
    - Collect property attributes
    - Clean and Preprocess Data
- Machine Learning Model Development
    - Choose an algorithm
- Data Splitting**:** Divide the data into two sets
    - Training set
    - Testing set
- Training the model
- Model Evaluation
- Testing the model
- Model validation

5.3 Dealing with dataset:

We searched a lot for the dataset to have data for apartments for rent in Egypt but we faced difficulty after that we were going to collect the data ourselves but we found that the number will also be small until we found the data but it has data for rent and sale so we used it and made a drop for the data that is for sale but this affected the number of data decreased and affected the accuracy of the model

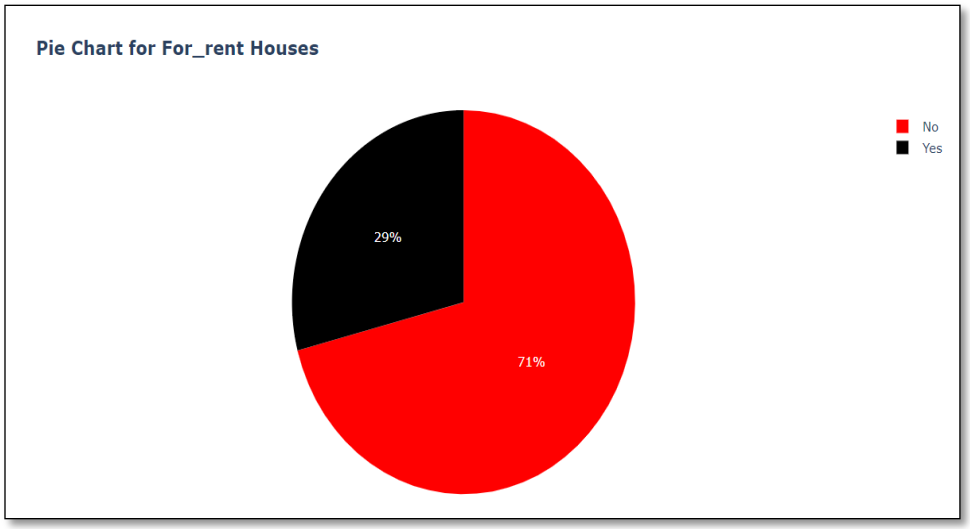
5.3.1 The dataset and the features:

```
df = pd.read_csv('houses_data_v2.csv')
df.head(10)
```

	House_Type	Size	Bedrooms	Bathrooms	Floor	Furnished	For_rent	Region	City	Price
0	Apartment	170	3	2	9	No	No	Zahraa Al Maadi	Cairo	1546400
1	Apartment	104	2	1	7	No	No	Nasr City	Cairo	950000
2	Apartment	160	3	2	1	No	No	Mostakbal City	Cairo	2100000
3	Apartment	160	3	3	Ground	No	No	New Cairo - El Tagamoa	Cairo	3994232
4	Apartment	145	3	2	3	No	No	New Cairo - El Tagamoa	Cairo	370000
5	Apartment	103	2	1	3	No	No	New Capital City	Cairo	3000000
6	Penthouse	140	3	3	3	No	No	New Cairo - El Tagamoa	Cairo	2952000
7	Apartment	118	2	2	2	No	No	New Capital City	Cairo	1327500
8	Apartment	200	4	3	4	No	No	Madinaty	Cairo	3300000
9	Apartment	88	2	2	6	No	No	Nasr City	Cairo	1235000

5.3.2 A Pie Chart for Apartments for sale versus apartments for rent :

Our dataset has apartments for rent and apartments for sale we made pie chart to show the ratio between them



5.3.3 Quick look at info about the data :

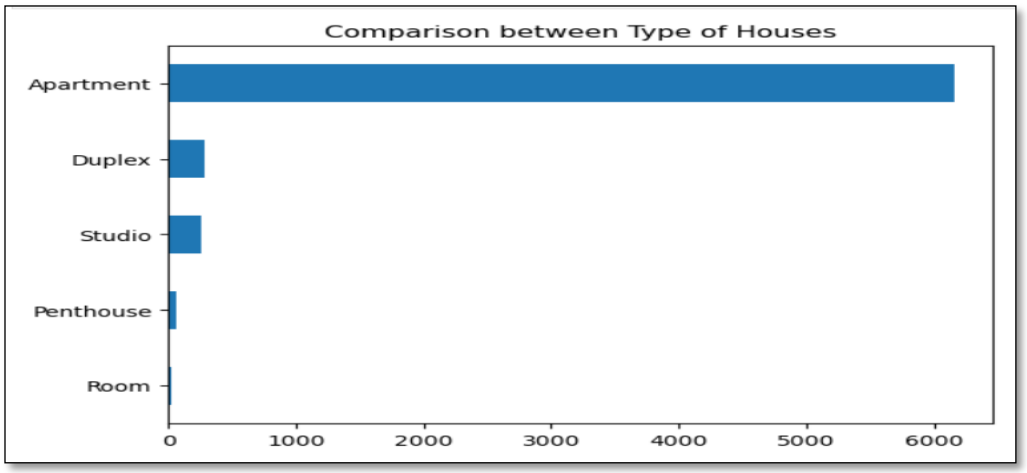
This DataFrame contains 23,347 entries with 10 columns describing rental properties. Each entry includes details like house type, size, number of bedrooms and bathrooms, floor level, furnishing status, region, city, and most importantly, the asking price.

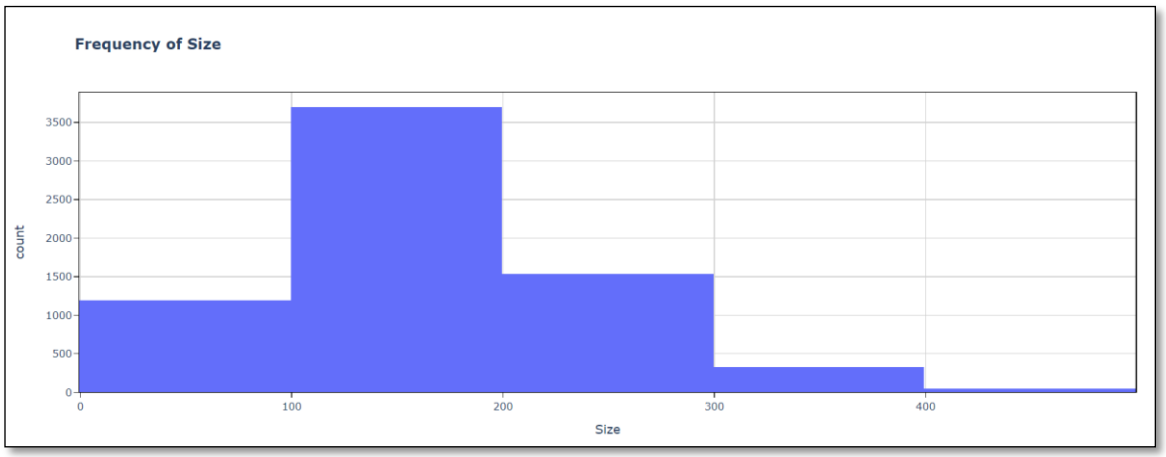
```
df.info()
```

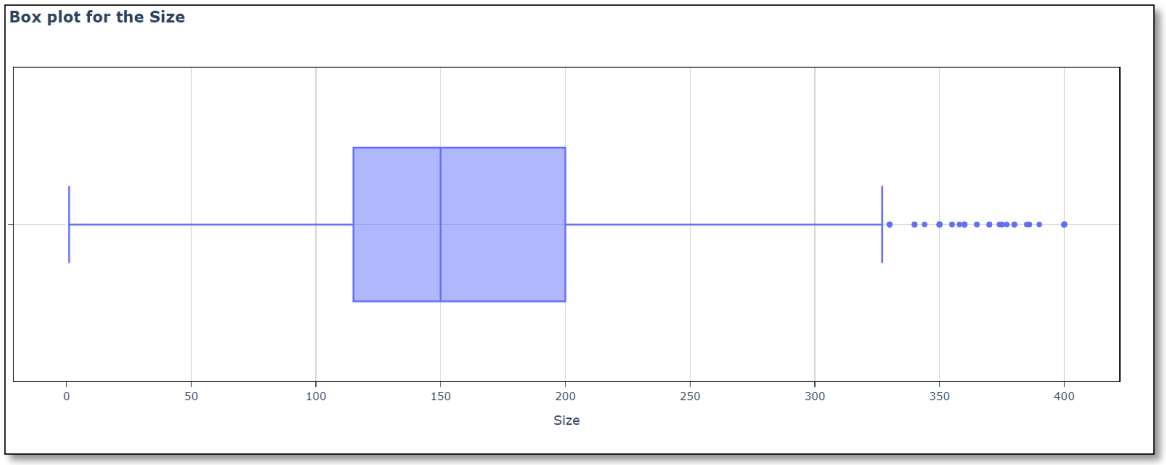
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 23347 entries, 0 to 23346
Data columns (total 10 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   House_Type  23347 non-null  object
 1   Size        23347 non-null  int64 
 2   Bedrooms    23347 non-null  int64 
 3   Bathrooms   23347 non-null  int64 
 4   Floor       23347 non-null  object
 5   Furnished   23347 non-null  object
 6   For_rent    23347 non-null  object
 7   Region      23347 non-null  object
 8   City        23347 non-null  object
 9   Price       23347 non-null  int64 
dtypes: int64(4), object(6)
memory usage: 1.8+ MB
```

5.3.4 Data visualization:

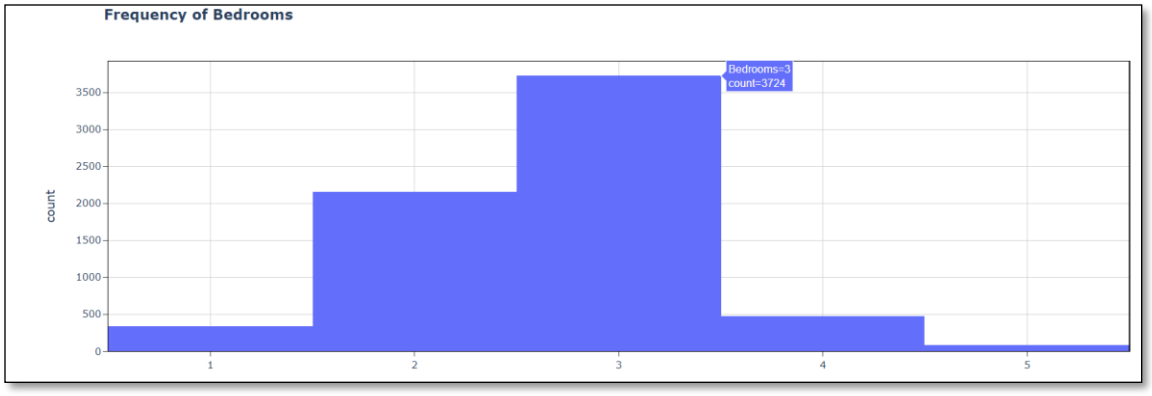
Clearly most common type of houses is Apartment

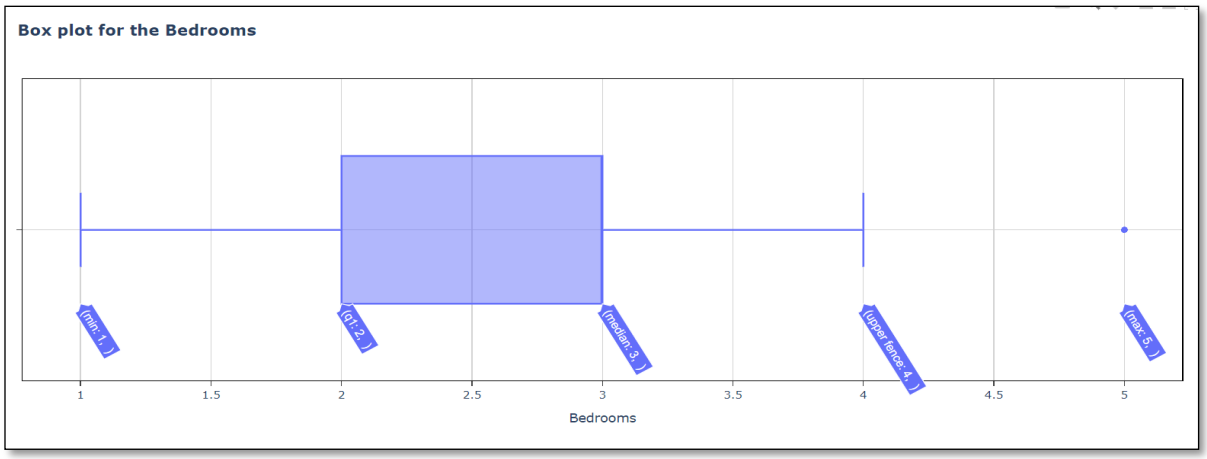






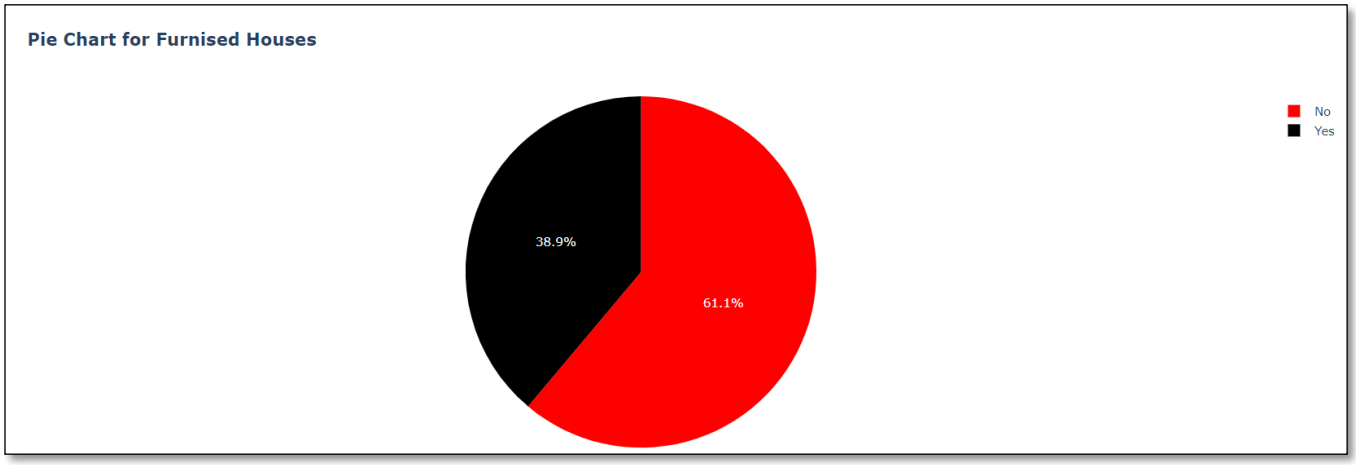
-The most common house size in range 100- 200

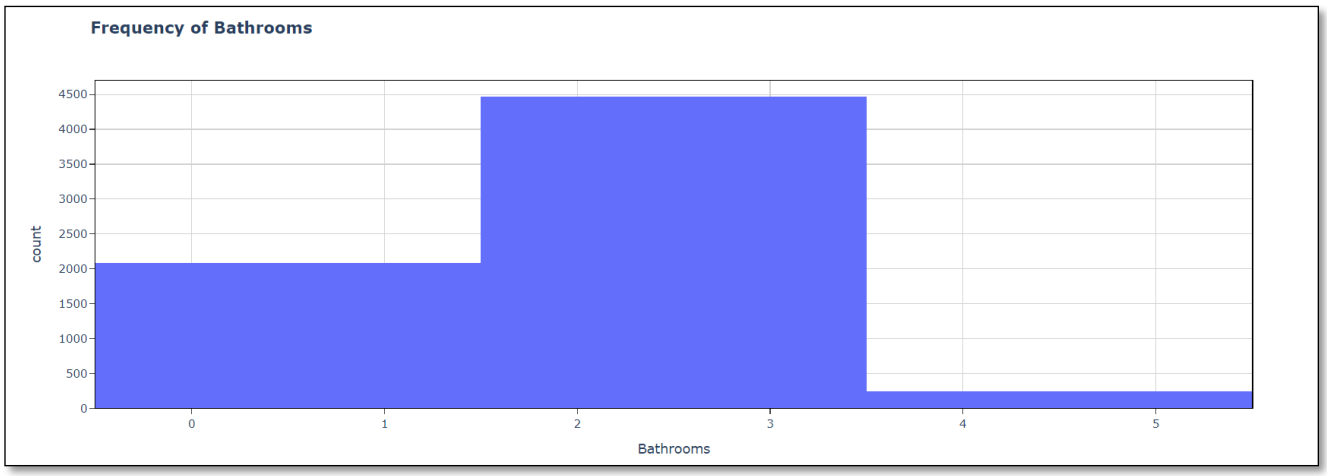


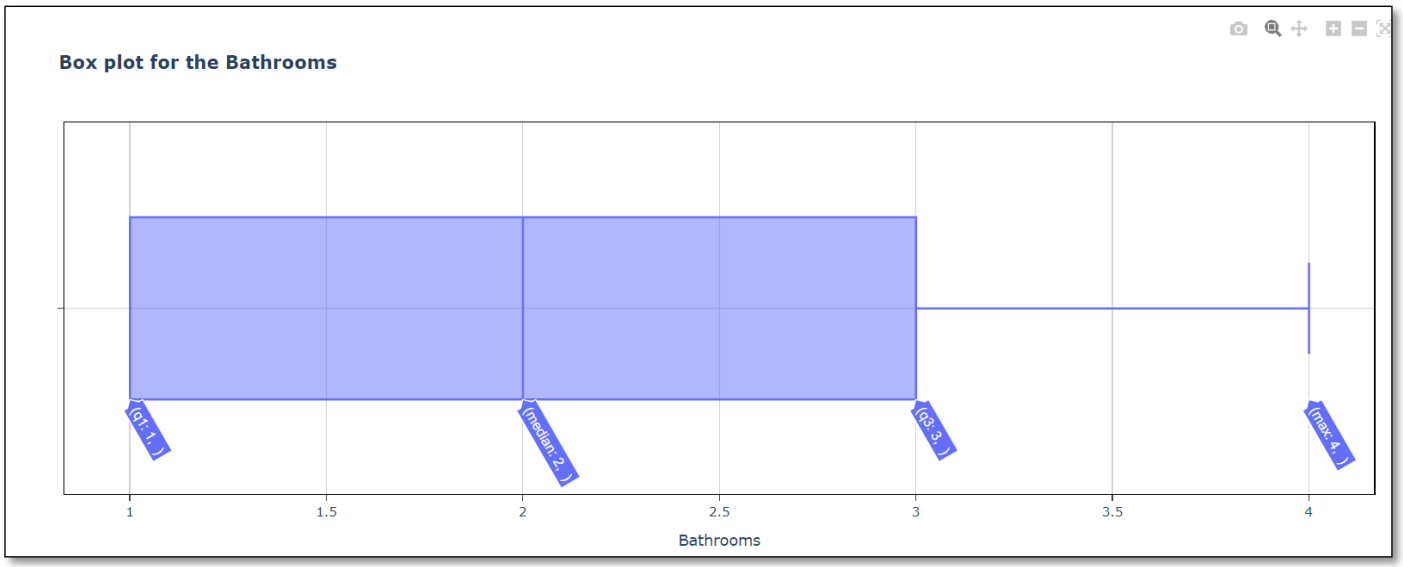


- Most common # of bedrooms 3 and 2

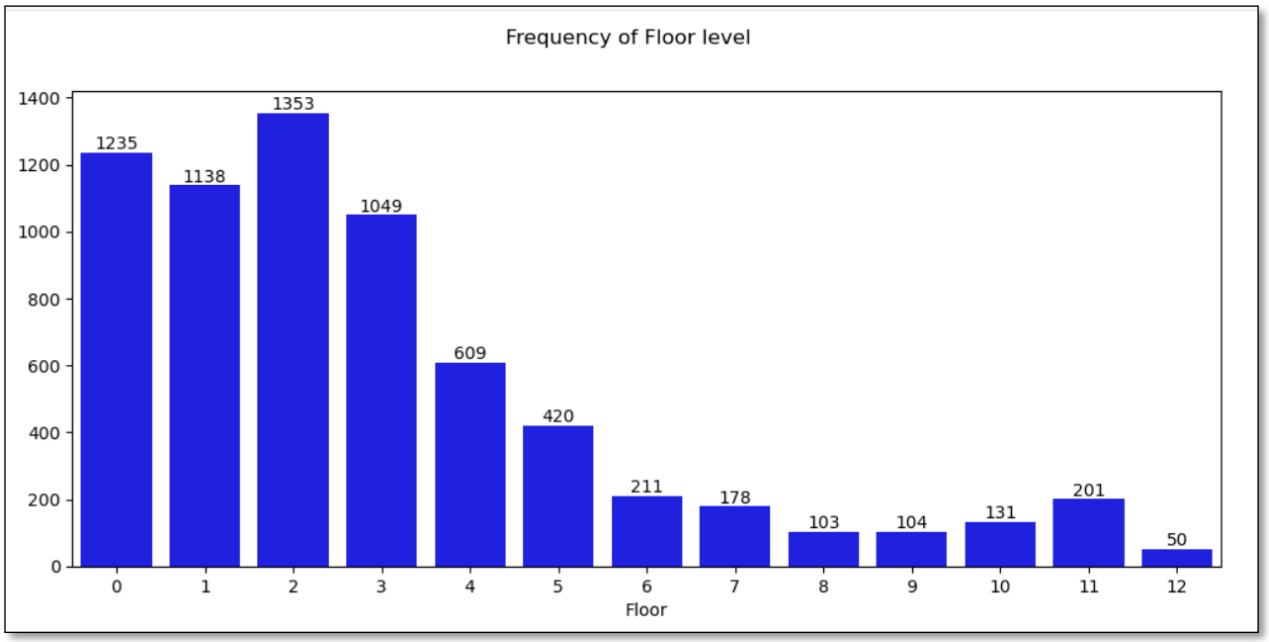
- About 61% of the dataset houses are not furnished



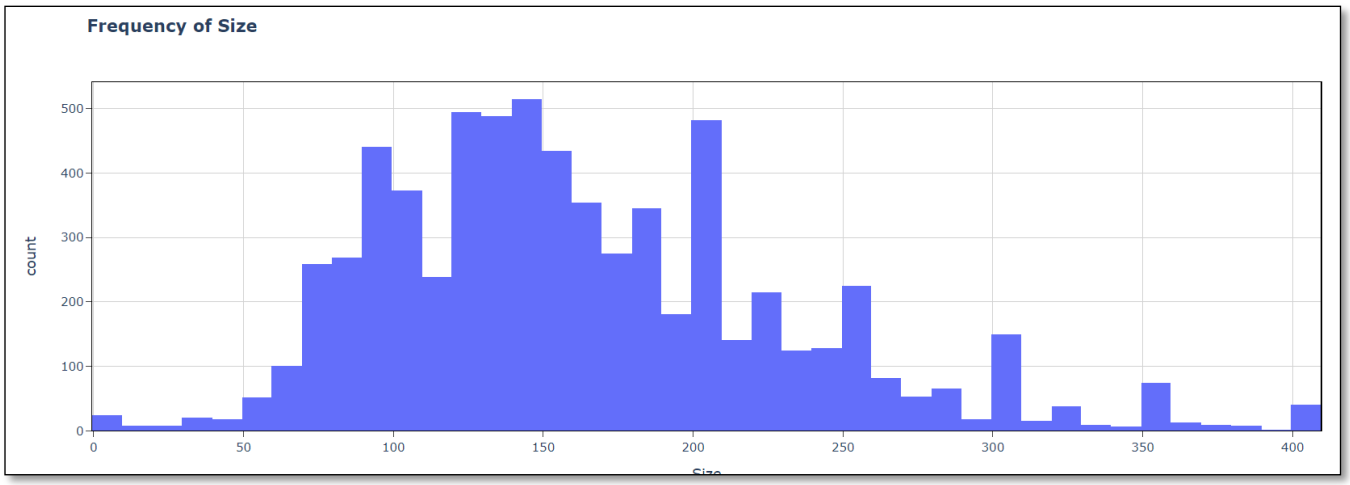


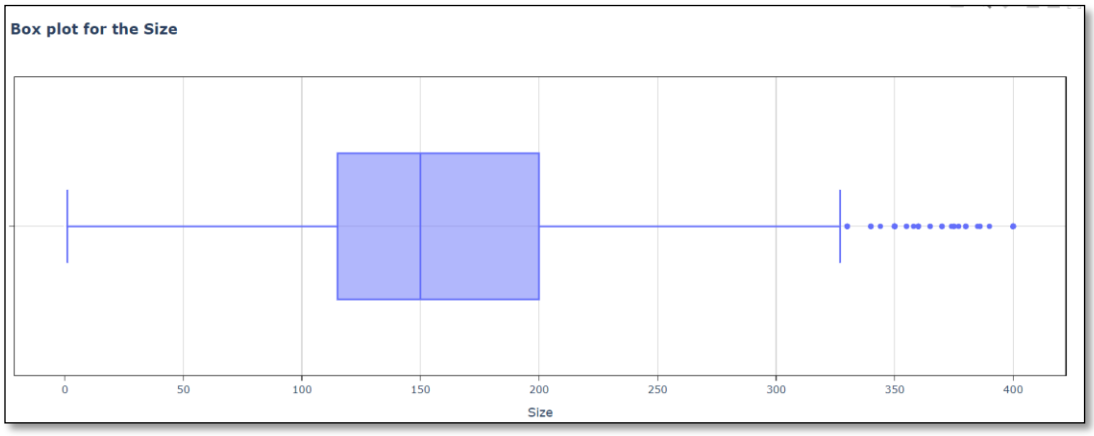


- Most common # of bathrooms 2 and 3

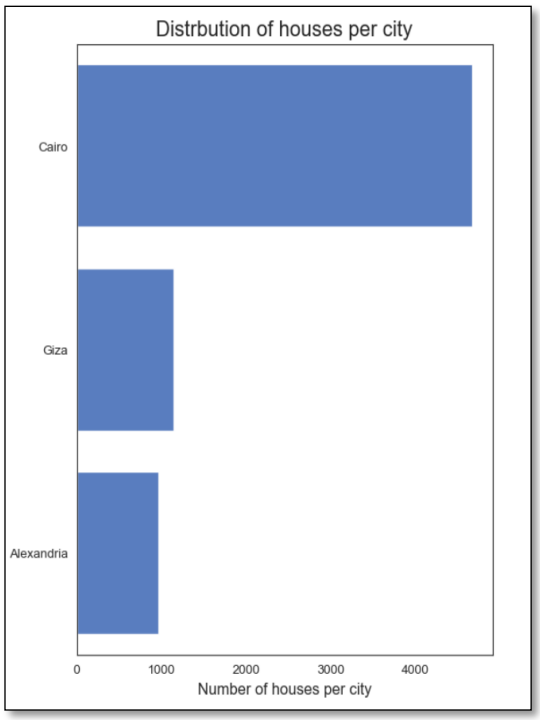


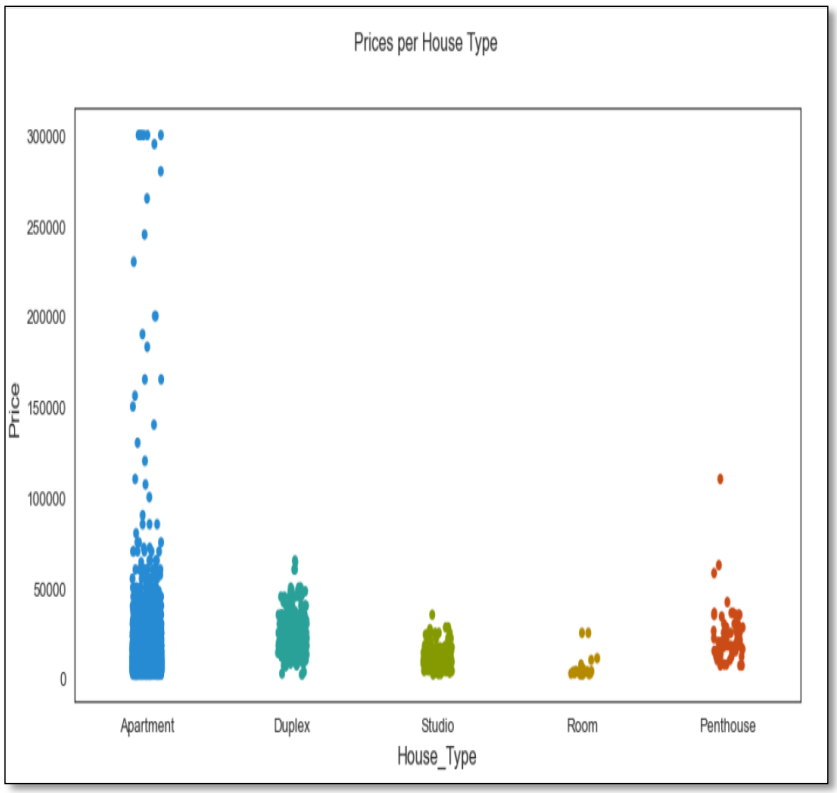
- The Most common floor levels 2nd , 0, 1st and 3rd





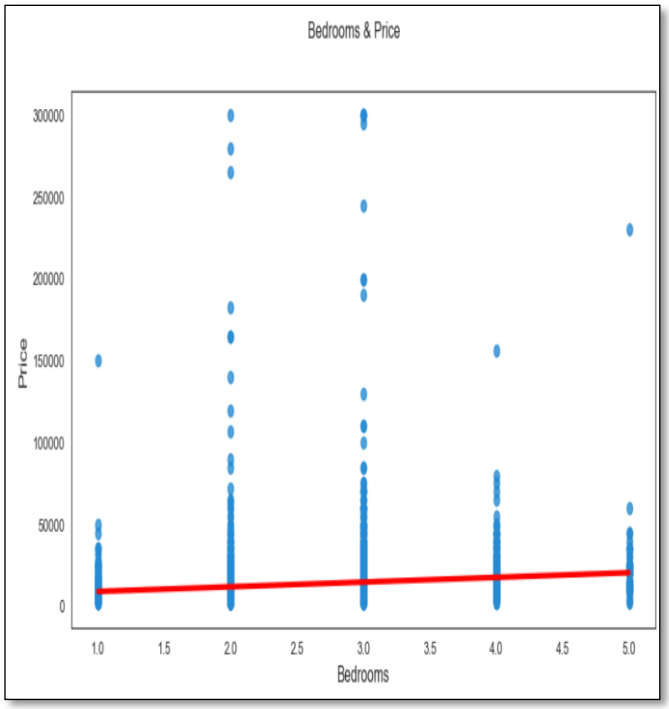
-The most common house size in range 120 - 160

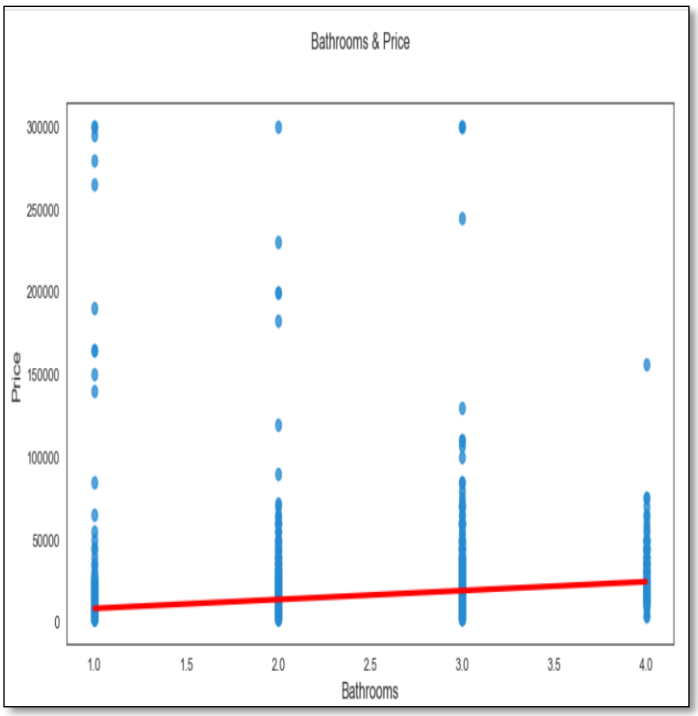


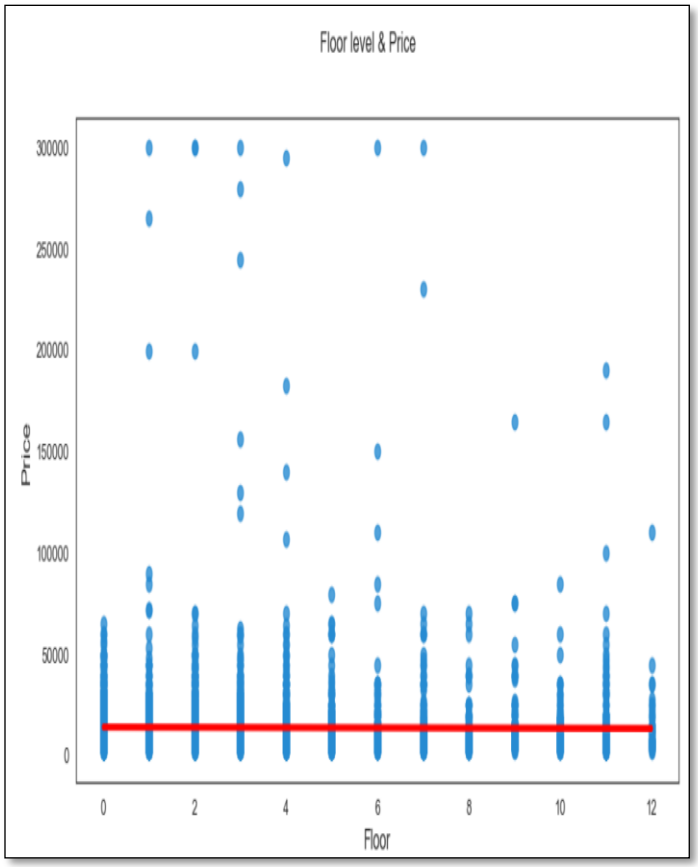


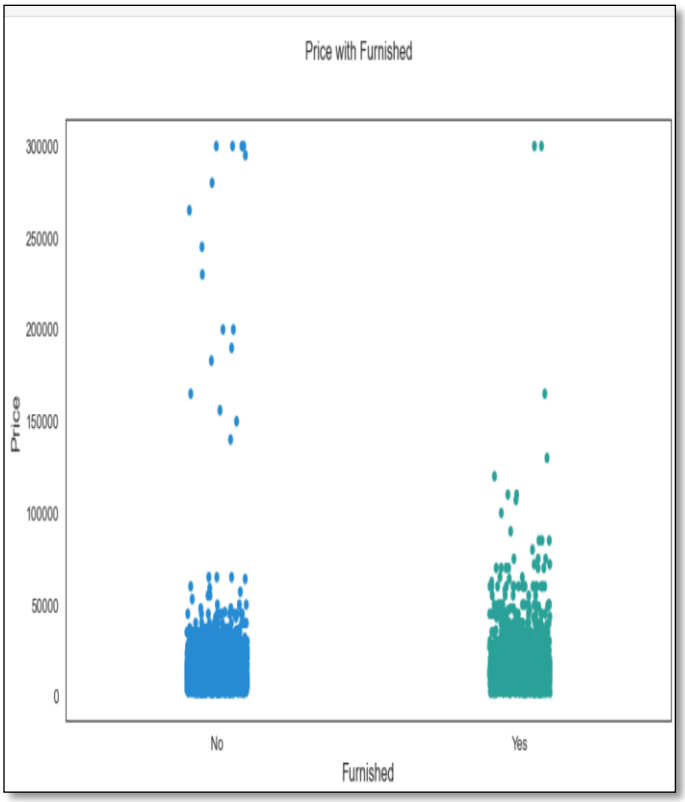
- The highest price is the Apartment

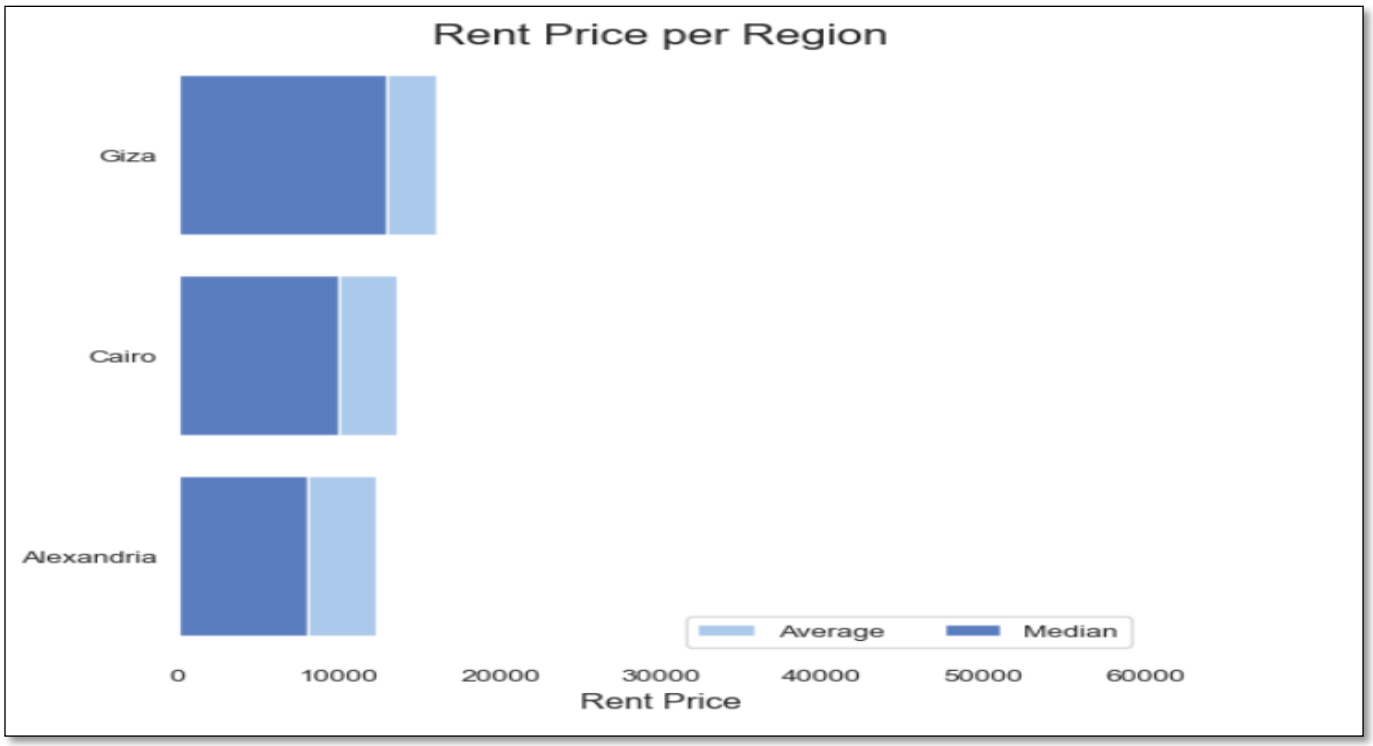
- The higher the size, the higher the price (Positive relationship)

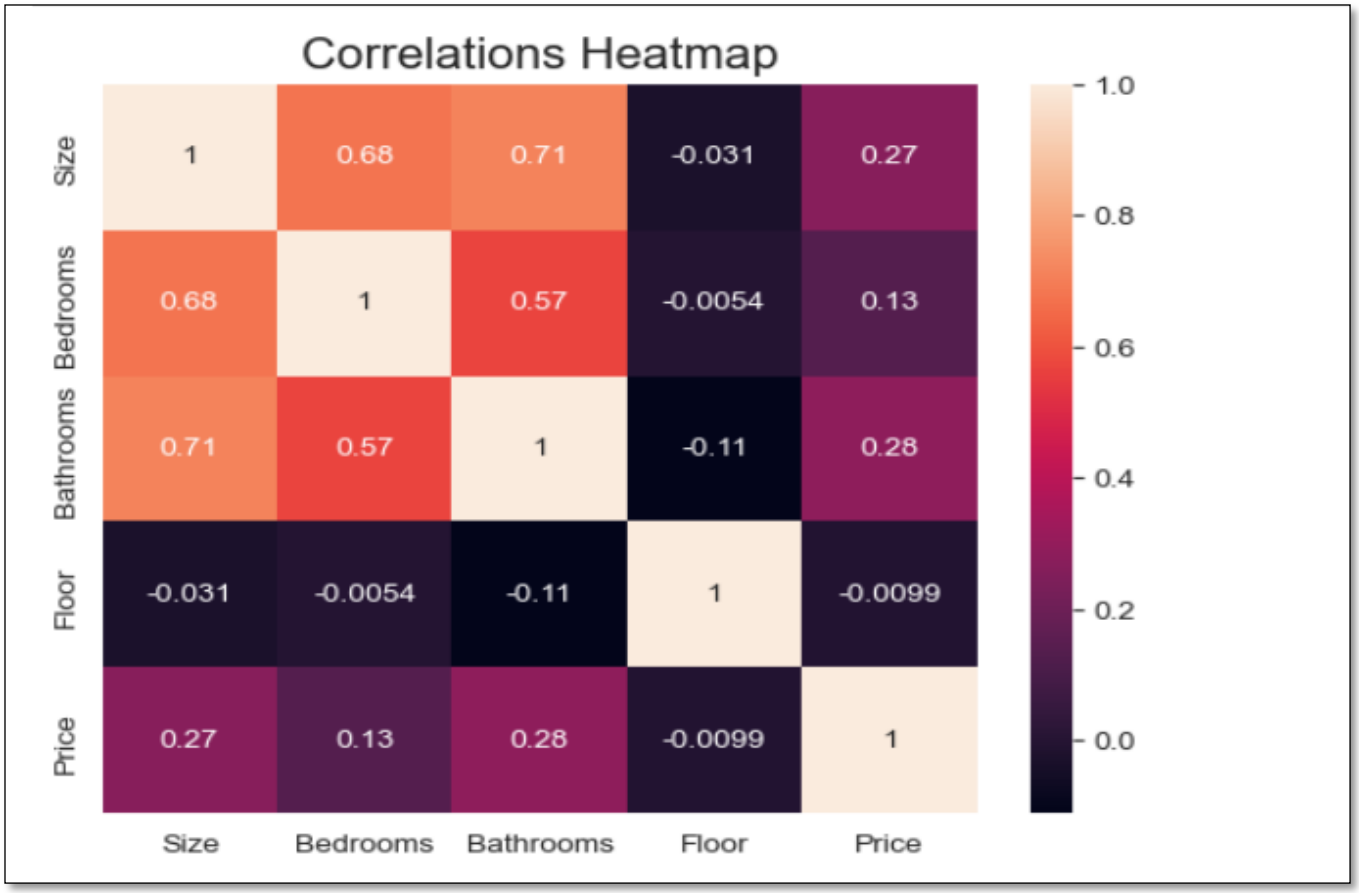

Bedrooms & Price
Price
300000
250000
200000
150000
100000
50000
0
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
Bedrooms

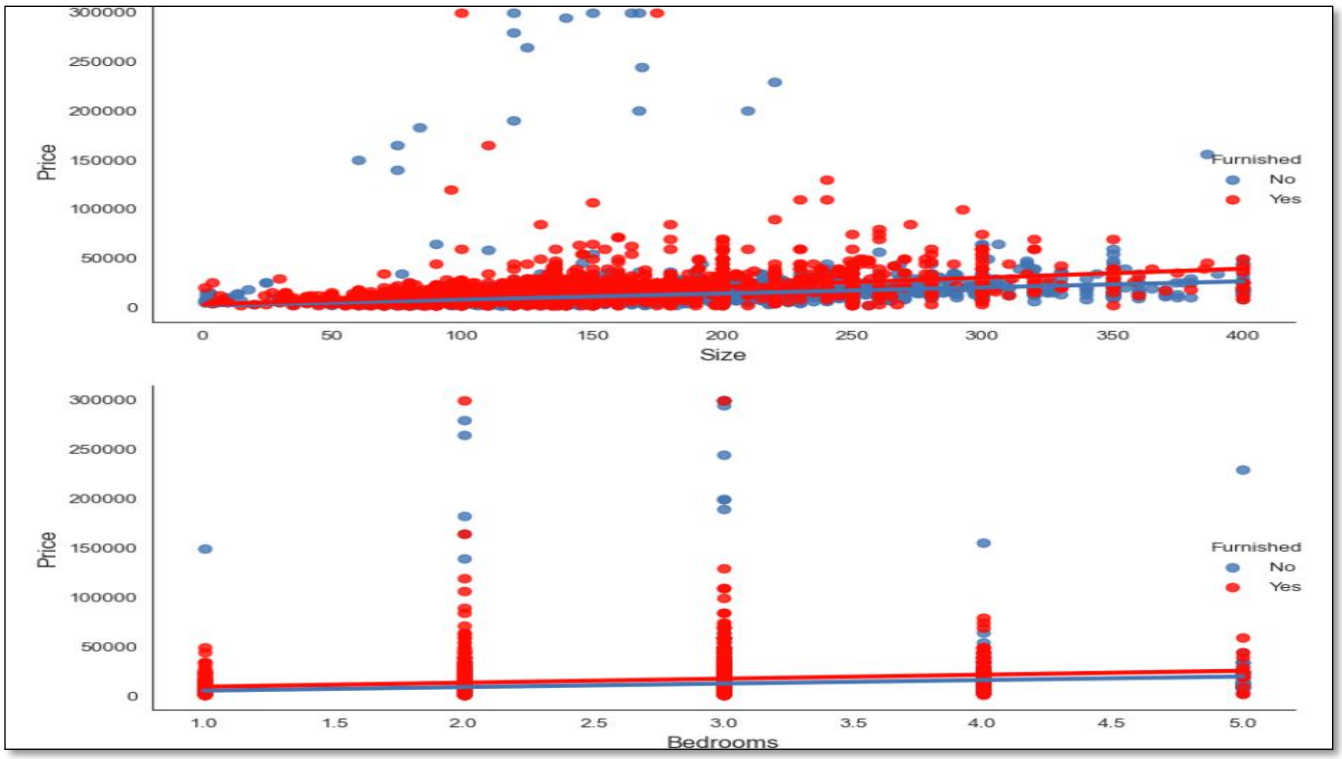

Bathrooms & Price
Price
300000
250000
200000
150000
100000
50000
0
1.0
1.5
2.0
2.5
3.0
3.5
4.0
Bathrooms

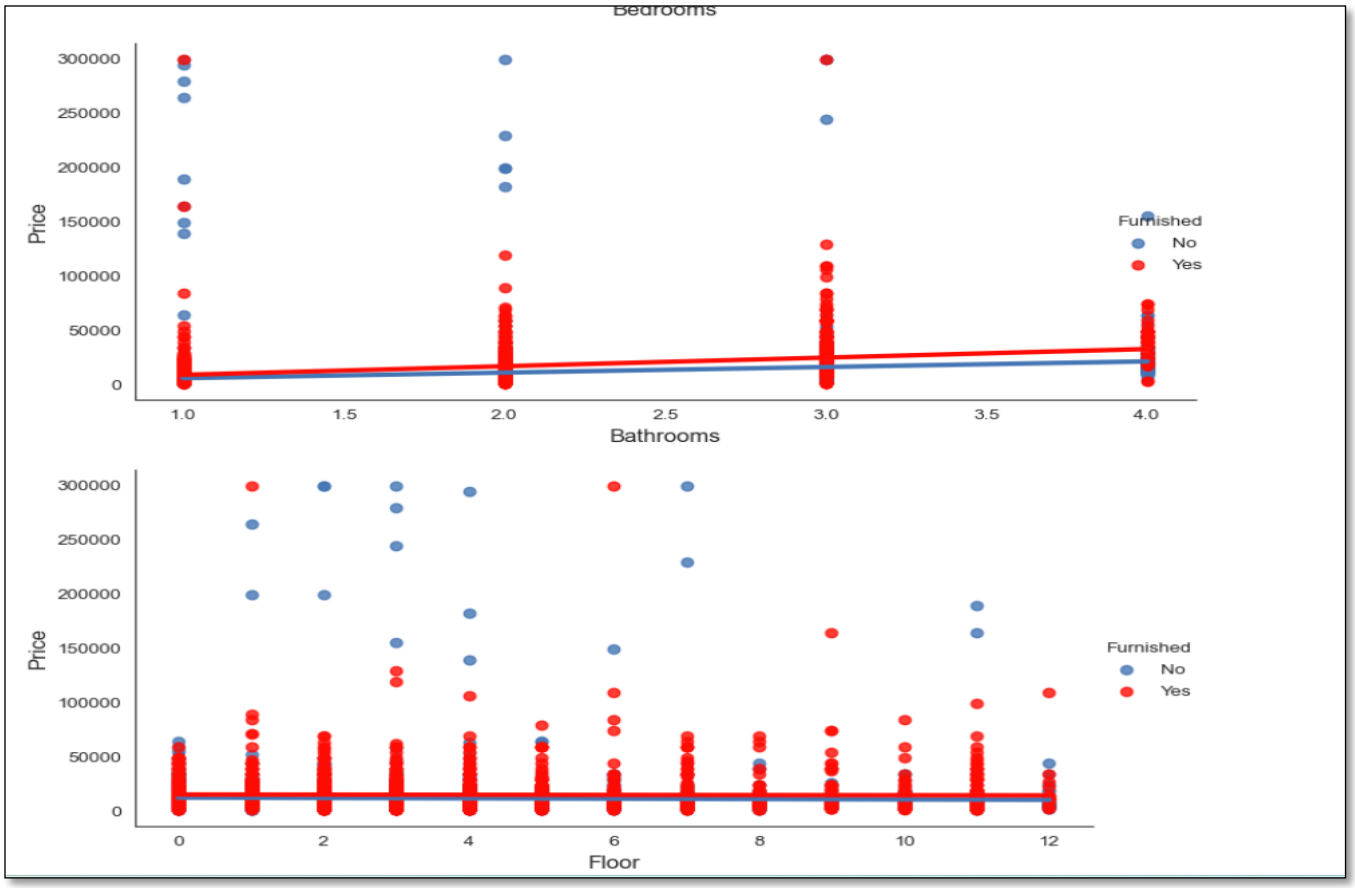

Floor level & Price
Price
300000
250000
200000
150000
100000
50000
0
0
2
4
6
8
10
12
Floor

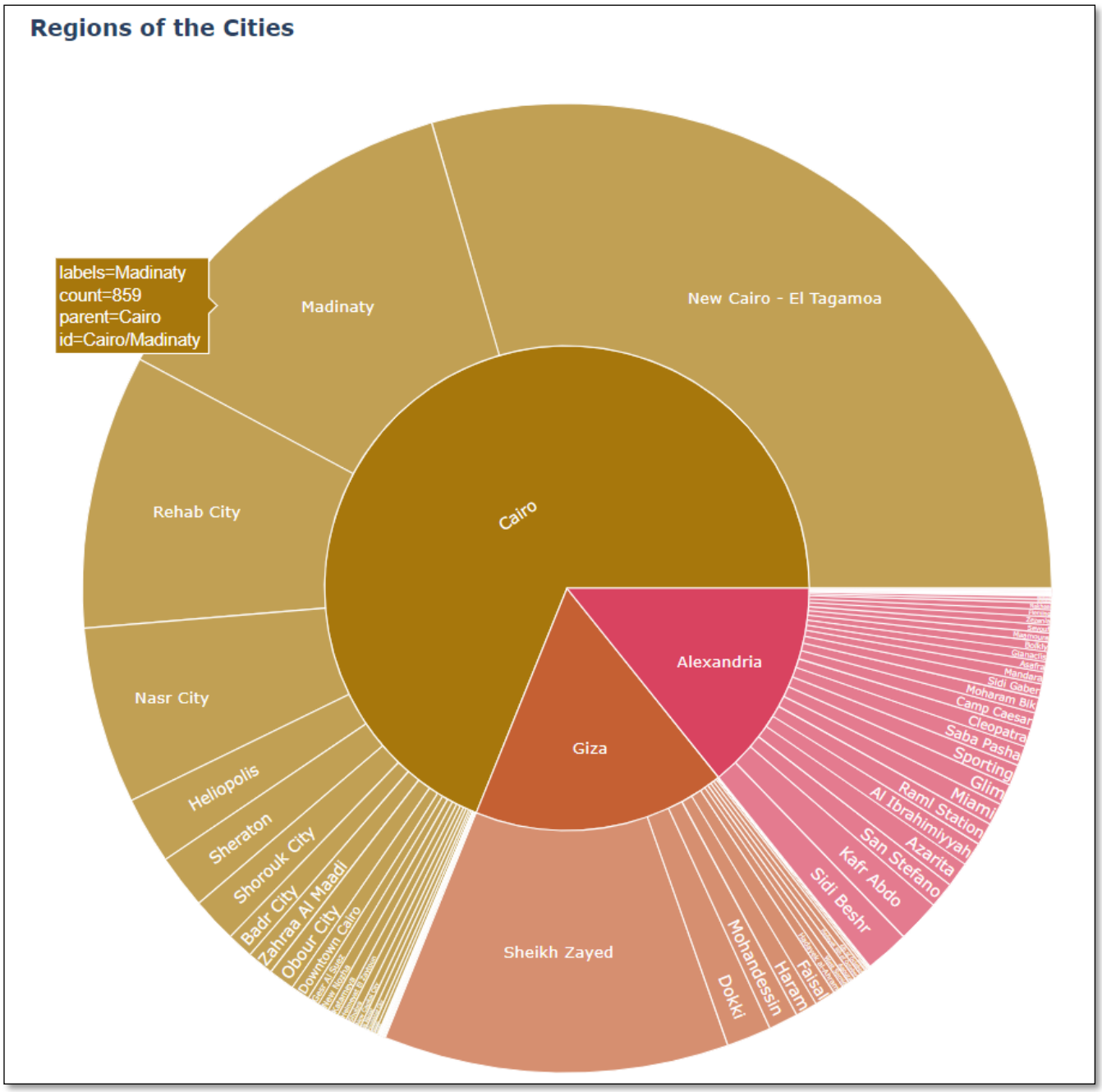

Price with Furnished
Price
300000
250000
200000
150000
100000
50000
0
No
Yes
Furnished


Rent Price per Region
Giza
Cairo
Alexandria
Average
Median
0
10000
20000
30000
40000
50000
60000
Rent Price


Correlations Heatmap
Size
Bedrooms
Bathrooms
Floor
Price
1
0.68
0.71
-0.031
0.27
0.68
1
0.57
-0.0054
0.13
0.71
0.57
1
-0.11
0.28
-0.031
-0.0054
-0.11
1
-0.0099
0.27
0.13
0.28
-0.0099
1
Size
Bedrooms
Bathrooms
Floor
Price
1.0
0.8
0.6
0.4
0.2
0.0


Price
Size
Furnished
No
Yes
Price
Bedrooms
Furnished
No
Yes


Bedrooms
Price
Bathrooms
Furnished
No
Yes
Price
Floor
Furnished
No
Yes


Regions of the Cities
labels=Madinaty
count=859
parent=Cairo
id=Cairo/Madinaty
Madinaty
New Cairo - El Tagamoa
Rehab City
Cairo
Nasr City
Alexandria
Giza
Heliopolis
Sheraton
Shorouk City
Badr City
Zahraa Al Maadi
Obour City
Downtown Cairo
Sheikh Zayed
Dokki
Mohandessin
Haram
Faisal
Kafr Abdo
Sidi Beshr
San Stefano
Azarita
Al Ibrahimiyyah
Raml Station
Miami
Glim
Sporting
Saba Pasha
Cleopatra
Camp Caesar
Moharam Bik
Sidi Gaber
Mandara
Asafra
Gianaclis
Bolkly
Mamoura
Smouha

Chapter 6
System Implementation

5.4 Machine Learning:

Dependencies:

Extracted a new feature from the product name using Python functions:

```
df['no_of_rooms'] = df['Bedrooms'] + df['Bathrooms']
df.head()
```

	House_Type	Size	Bedrooms	Bathrooms	Floor	Furnished	For_rent	Region	City	Price	no_of_rooms
0	0	200	3	3	1	0	Yes	55	1	12000	6
1	0	130	3	2	2	1	Yes	55	1	8500	5
2	0	130	2	1	3	0	Yes	55	1	5000	3
3	0	200	3	1	0	0	Yes	54	1	8500	4
4	0	160	3	2	0	0	Yes	55	1	7000	5

5.4.1 Label Encoding :

is a technique for converting categorical data into numerical data that machine learning algorithms can understand

```
types_of_apartments = ['Apartment', 'Duplex', 'Studio', 'Room', 'Penthouse']
```

```
label_encoder = LabelEncoder()
df['House_Type'] = label_encoder.fit_transform(df['House_Type'])
```

```
def encode_floor(floor):
    if floor == 'Ground':
        floor = 0
    if floor == '10+':
        floor = 11
    if floor == 'Highest':
        floor = 12
    return int(floor)

df['Floor'] = df['Floor'].apply(encode_floor)
```

```
# for i, apartment_type in enumerate(types_of_apartments):
#     print(apartment_type, '-->', i)
```

```
df['Furnished'] = label_encoder.fit_transform(df['Furnished'])
```

```
df['Region'] = label_encoder.fit_transform(df['Region'])
```

```
df['City'] = label_encoder.fit_transform(df['City'])
```

```
df_filtered = df[df['For_rent'] != 'No']
```

```
df_filtered.head()
```

5.4.2 Dropped the houses isn't for rent features.

```
df.drop(df[df['For_rent'] == 'No'].index, inplace=True)
```

5.4.3 The result:

```
print(df["For_rent"])
```

```
33        Yes
34        Yes
35        Yes
36        Yes
37        Yes
         ...
23342     Yes
23343     Yes
23344     Yes
23345     Yes
23346     Yes
Name: For_rent, Length: 6782, dtype: object
```

5.4.4 Splitting data into train and test data:

```
X = df.drop(['Price', 'For_rent'], axis=1)
y = df['Price']

# Split data into train and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

5.4.5 Choosing the best model:

```
model = LinearRegression()
model.fit(X_train, y_train)
```

```
LinearRegression()
```

```
y_pred = model.predict(X_test)
```

```
# Evaluate the model
rmse = mean_squared_error(y_test, y_pred, squared=False)

# Calculate R-squared (R2) score
r2 = r2_score(y_test, y_pred)

print(f"Root Mean Squared Error (RMSE): {rmse}")
print(f"R-squared (R2) Score: {r2}")
```

```
Root Mean Squared Error (RMSE): 15617.660354775175
R-squared (R2) Score: 0.12736498525918372
```

```
# Initialize Gradient Boosting Regressor
gb_regressor = GradientBoostingRegressor(n_estimators=100, learning_rate=0.1, random_state=42)

# Train the model
gb_regressor.fit(X_train, y_train)

# Make predictions
y_pred = gb_regressor.predict(X_test)

# Evaluate the model
mse = mean_squared_error(y_test, y_pred)
rmse = mse ** 0.5
r2 = r2_score(y_test, y_pred)

print(f"Root Mean Squared Error (RMSE): {rmse}")
print(f"R-squared (R2) Score: {r2}")
```

```
Root Mean Squared Error (RMSE): 15012.013796250443
R-squared (R2) Score: 0.1937335335269348
```

```
# Initialize XGBoost Regressor
xgb_regressor = XGBRegressor(n_estimators=100, learning_rate=0.1, random_state=42)

# Train the model
xgb_regressor.fit(X_train, y_train)

# Make predictions
y_pred = xgb_regressor.predict(X_test)

# Evaluate the model
mse = mean_squared_error(y_test, y_pred)
rmse = mse ** 0.5
r2 = r2_score(y_test, y_pred)

print(f"XGBoost Regression - Root Mean Squared Error (RMSE): {rmse}")
print(f"XGBoost Regression - R-squared (R2) Score: {r2}")
```

```
XGBoost Regression - Root Mean Squared Error (RMSE): 13999.144862634157
XGBoost Regression - R-squared (R2) Score: 0.29886168126816626
```

```
# Initialize Random Forest Regressor
rf_regressor = RandomForestRegressor(n_estimators=100, random_state=42)

# Train the model
rf_regressor.fit(X_train, y_train)

# Make predictions
y_pred = rf_regressor.predict(X_test)

# Evaluate the model
mse = mean_squared_error(y_test, y_pred)
rmse = mse ** 0.5
r2 = r2_score(y_test, y_pred)

print(f"Random Forest Regression - Root Mean Squared Error (RMSE): {rmse}")
print(f"Random Forest Regression - R-squared (R2) Score: {r2}")
```

```
Random Forest Regression - Root Mean Squared Error (RMSE): 14926.798082532536
Random Forest Regression - R-squared (R2) Score: 0.20286109853563805
```

5.4.6 preprocess the data before enter the model:

```
def preproces(type, fur, region, city):
    if fur == 'Yes':
        fur = 1
    if fur == 'No':
        fur = 0
    if city == 'Alexandria':
        city = 0
    if city == 'Cairo':
        city = 1
    if city == 'Giza':
        city = 2
    if type == 'Apartment':
        type = 0
    if type == 'Chalet':
        type = 1
    if type == 'Duplex':
        type = 2
    if type == 'Penthouse':
        type = 3
    if type == 'Room':
        type = 4
    if type == 'Stand Alone Villa':
        type = 5
    if type == 'Studio':
        type = 7
    if type =='Town House':
        type=8
    if type =='Twin House':
        type=9
```

```
    if region == '15 May City':
        region = 0
    if region == '6th of October':
        region = 1
    if region == 'Abasiya':
        region = 2
    if region == 'Abu Qir':
        region = 3
    if region == 'Abu Talat':
        region = 4
    if region == 'Agouza':
        region = 5
    if region == 'Al Hadrah':
        region = 6
    if region == 'Al Ibrahimiyyah':
        region = 7
    if region == 'Al Manial':
        region = 8
    if region == 'Ard El Lewa':
        region = 9
    if region == 'Asafra':
        region = 10
    if region == 'Awayed':
        region = 11
    if region == 'Azarita':
        region = 12
    if region == 'Badr City':
        region = 13
    if region == 'Bahray - Anfoshy':
        region = 14
s()  ›  if region == 'Bahray - Anfoshy'
```

```
    if region == 'Bahray - Anfoshy':
        region = 14
    if region == 'Basateen':
        region = 15
    if region == 'Bolkly':
        region = 16
    if region == 'Borg al-Arab':
        region = 17
    if region == 'Boulaq Dakrour':
        region = 18
    if region == 'Camp Caesar':
        region = 19
    if region == 'Cleopatra':
        region = 20
    if region == 'Dar al-Salaam':
        region = 21
    if region == 'Dokki':
        region = 22
    if region == 'Downtown Cairo':
        region = 23
    if region == 'Faisal':
        region = 24
    if region == 'Fleming':
        region = 25
    if region == 'Garden City':
        region = 26
    if region == 'Gesr Al Suez':
        region = 27
    if region == 'Gianaclis':
        region = 28
```

```
    if region == 'Giza District':
        region = 29
    if region == 'Glim':
        region = 30
    if region == 'Hadayek 6th of October':
        region = 31
    if region == 'Hadayek al-Ahram':
        region = 32
    if region == 'Hadayek al-Kobba':
        region = 33
    if region == 'Haram':
        region = 34
    if region == 'Heliopolis':
        region = 35
    if region == 'Helmeyat El Zaytoun':
        region = 36
    if region == 'Helwan':
        region = 37
    if region == 'Imbaba':
        region = 38
    if region == 'Kabbary':
        region = 39
    if region == 'Kafr Abdo':
        region = 40
    if region == 'Katameya':
        region = 41
    if region == 'Maamoura':
        region = 42
    if region == 'Madinaty':
        region = 43
```

```
    if region == 'Mandara':
        region = 44
    if region == 'Manshiyya':
        region = 45
    if region == 'Matareya':
        region = 46
    if region == 'Miami':
        region = 47
    if region == 'Mohandessin':
        region = 48
    if region == 'Moharam Bik':
        region = 49
    if region == 'Moneeb':
        region = 50
    if region == 'Montazah':
        region = 51
    if region == 'Mostakbal City':
        region = 52
    if region == 'Nakheel':
        region = 53
    if region == 'Nasr City':
        region = 54
    if region == 'New Cairo - El Tagamoa':
        region = 55
    if region == 'New Capital City':
        region = 56
    if region == 'New Nozha':
        region = 57
    if region == 'Obour City':
        region = 58
```

```
    if region == 'Qasr al-Nil':
        region = 59
    if region == 'Raml Station':
        region = 60
    if region == 'Ras El Tin':
        region = 61
    if region == 'Rehab City':
        region = 62
    if region == 'Saba Pasha':
        region = 63
    if region == 'Salam City':
        region = 64
    if region == 'San Stefano':
        region = 65
    if region == 'Schutz':
        region = 66
    if region == 'Seyouf':
        region = 67
    if region == 'Sheikh Zayed':
        region = 68
    if region == 'Sheraton':
        region = 69
    if region == 'Shorouk City':
        region = 70
    if region == 'Shubra':
        region = 71
    if region == 'Sidi Beshr':
        region = 72
    if region == 'Sidi Gaber':
        region = 73
```

```
if region == 'Sporting':
    region = 74
if region == 'West Somid':
    region = 75
if region == 'Zahraa Al Maadi':
    region = 76
if region == 'Zamalek':
    region = 77
if region == 'Zezenia':
    region = 78
return type, fur, region, city
```

5.4.7 Python :

rements.txt | preproces.py | app.py | final-final-model.ipynb

```python
from flask import Flask, request, jsonify
import joblib
import numpy as np
from preproces import preproces

# load the model from disk
model = joblib.load('final_model.pkl')

app = Flask(__name__)

@app.route('/house_pred', methods=['POST'])
def prediction():
    try:
        house_details = request.get_json()
        h_type = house_details['type']
        h_size = house_details['size']
        h_bedrooms = house_details['bedrooms']
        h_bathrooms = house_details['bathrooms']
        h_floor = house_details['floor']
        h_fur = house_details['fur']
        h_region = house_details['region']
        h_city = house_details['city']

        num_rooms = h_bathrooms+h_bedrooms

        # if h_city == "Cairo":
        #     h_city=1
```

```
        h_type, h_fur, h_region, h_city = preproces(h_type, h_fur, h_region, h_city)
        output = np.array([h_type, h_size, h_bedrooms, h_bathrooms, h_floor, h_fur, h_region, h_city, num_rooms])
        price = model.predict(output.reshape(1, -1))
        return jsonify(price[0])

    except Exception as e:
        # Handle exceptions and return an error message in JSON format
        return jsonify({'error': str(e)})


if __name__ == '__main__':
    # Run the Flask application in debug mode
    app.run(debug=True)
```

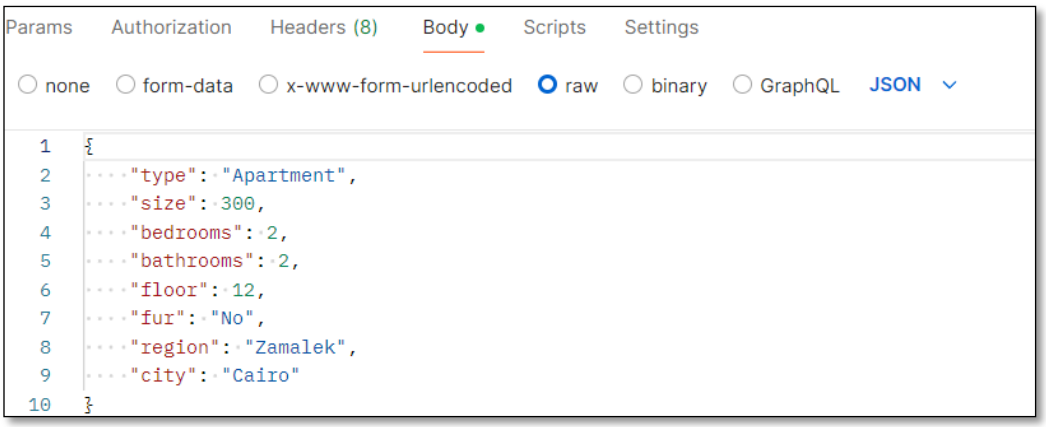
We saved the model using Python Joblib .

```
import joblib

# Assuming best_gb_regressor is your trained Gradient Boosting Regressor model

# Save the model to a file
joblib.dump(model, 'final_model.pkl')
```

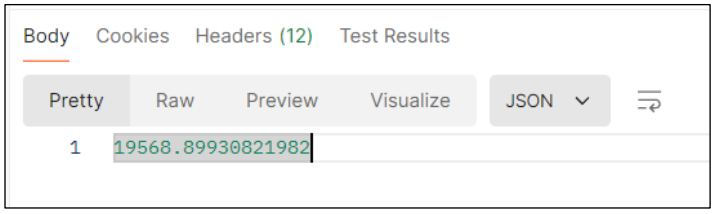
Using Flask to generate the local API.

```
# load the model from disk
model = joblib.load('final_model.pkl')

app = Flask(__name__)

@app.route('/house_pred', methods=['POST'])
```

Test the API Using Postman:



The result :



The Accuracy :

```
Root Mean Squared Error (RMSE): 15617.660354775175
R-squared (R2) Score: 0.12736498525918372
```

5.5 Mobile Application

5.5.1 Logic of signup with Getx

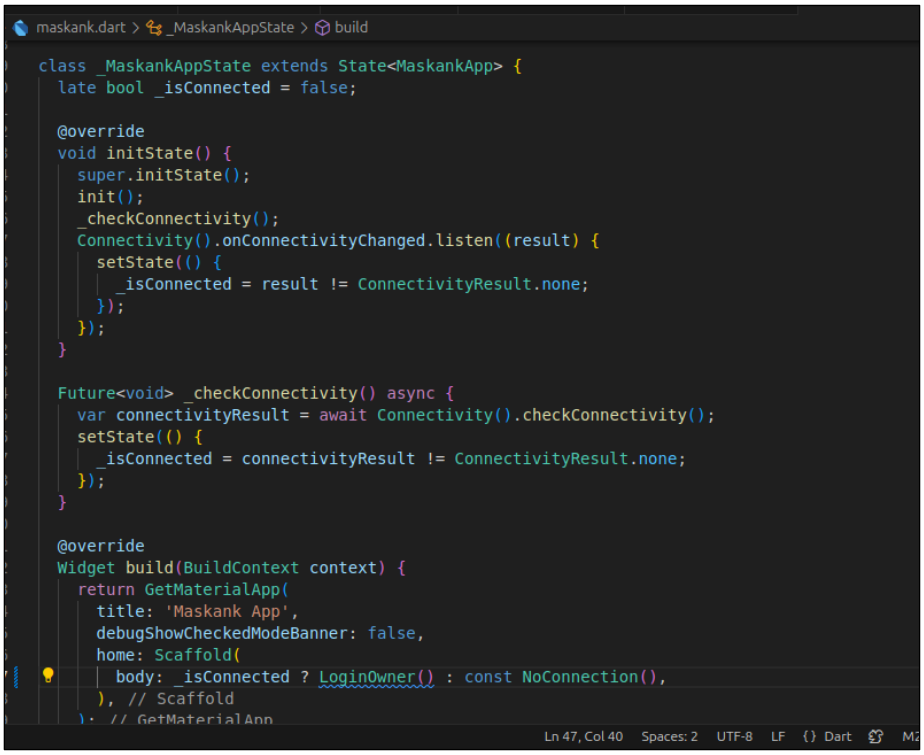
```dart
class RegesterOwnerController extends GetxController {
  final usernameController = TextEditingController();
  final fullnameController = TextEditingController();
  final phoneController = TextEditingController();
  final passwordController = TextEditingController();
  final confirmpasswordController = TextEditingController();
  GlobalKey<FormState> formkey = GlobalKey<FormState>();
  bool isRegister = false;
  bool isvisible = false;
  void changeVisible() {
    isvisible = !isvisible;
    update();
  }

  void registerUser() async {
    isRegister = true;

    update();
    SharedPreferences prefs = await SharedPreferences.getInstance();

    ApiClient apiClient = ApiClient(
      appBaseUrl: AppConstants.baseUrl,
      sharedPreferences: prefs,
    );

    Map<String, dynamic> userData = {
      'username': usernameController.text,
      'password': passwordController.text,
      'owner_name': fullnameController.text,
      'phone': phoneController.text,
      'image': 'test.jpg'
```

5.5.2 This login logic of owner with GetX controller :

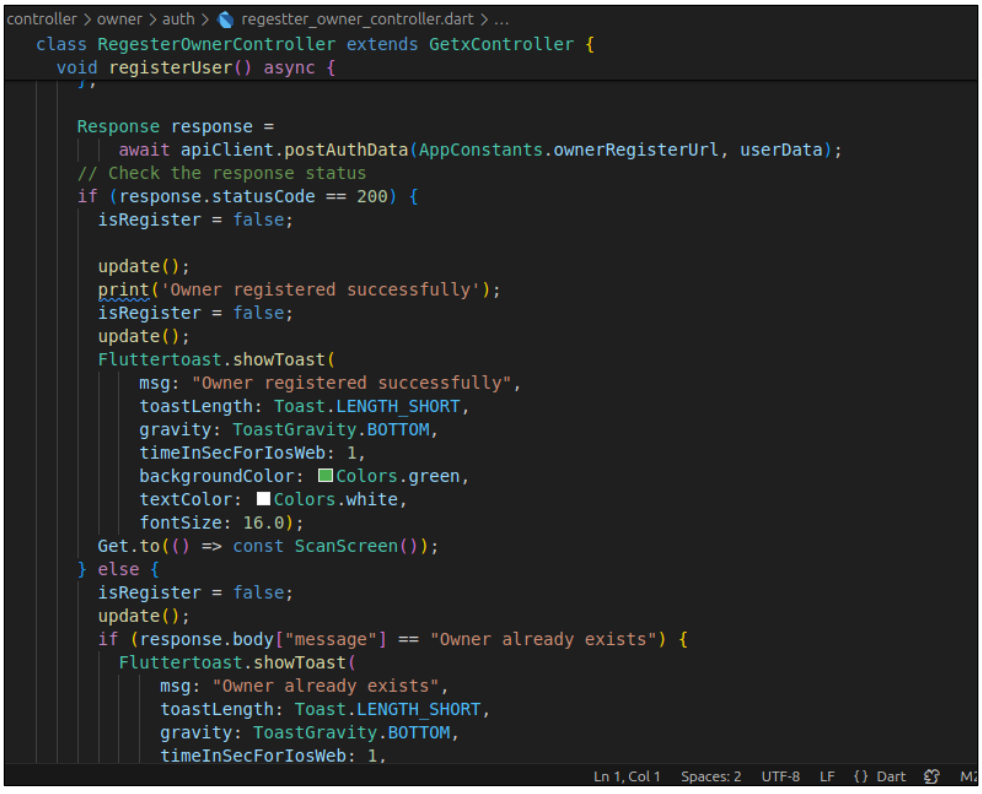
```dart
ntroller > owner > auth > login_owner_controller.dart > LoginOwnerController > isLogin

class LoginOwnerController extends GetxController {
  final username = TextEditingController();
  final password = TextEditingController();
  GlobalKey<FormState> formKey = GlobalKey<FormState>();
  bool isVisible = false;
  bool isLogin = false;

  void toggleVisibility() {
    isVisible = !isVisible;
    update();
  }

  void login() {
    if (formKey.currentState!.validate()) {
      loginOwner();
    }
  }

  void loginOwner() async {
    isLogin = true;
    update();
    SharedPreferences prefs = await SharedPreferences.getInstance();
    ApiClient apiClient = ApiClient(
      appBaseUrl: AppConstants.baseUrl,
      sharedPreferences: prefs,
    );

    Map<String, dynamic> userData = {
      'username': username.text,
      'password': password.text,
```

6

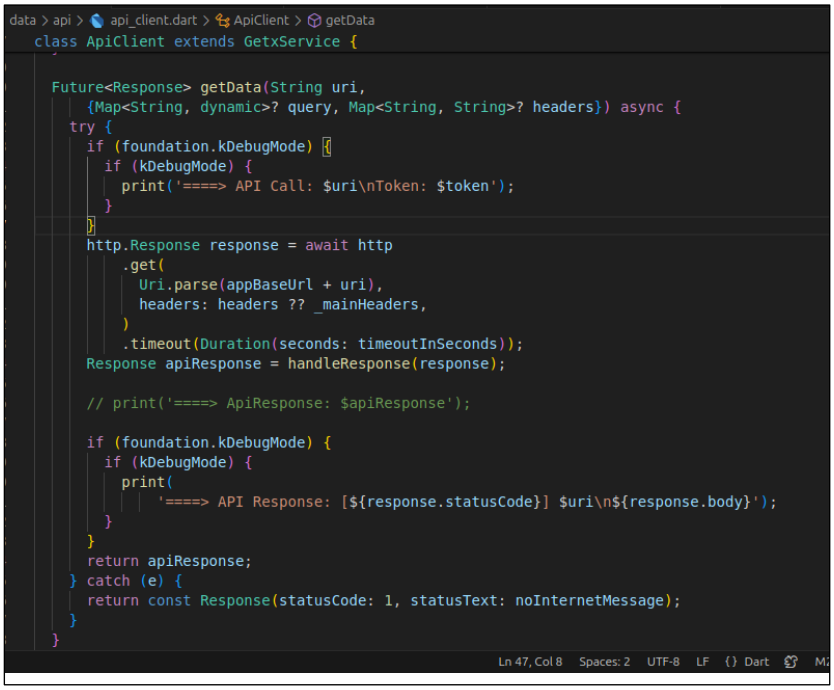
5.5.3 Here logic code to show if network connection or no:



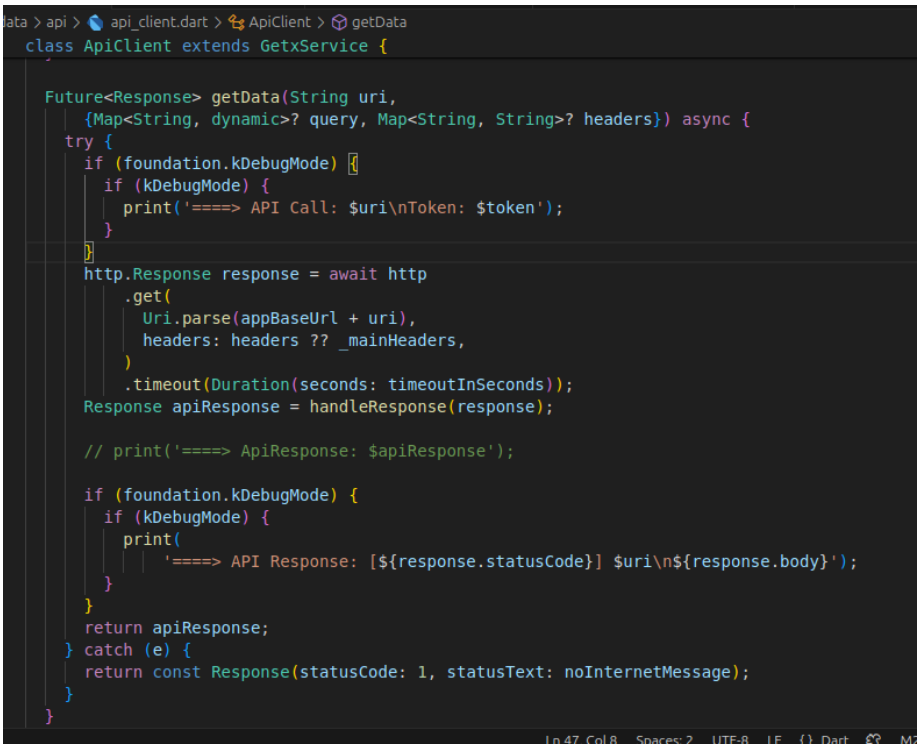
5.5.4 Here we used post method of API to send data to backend :



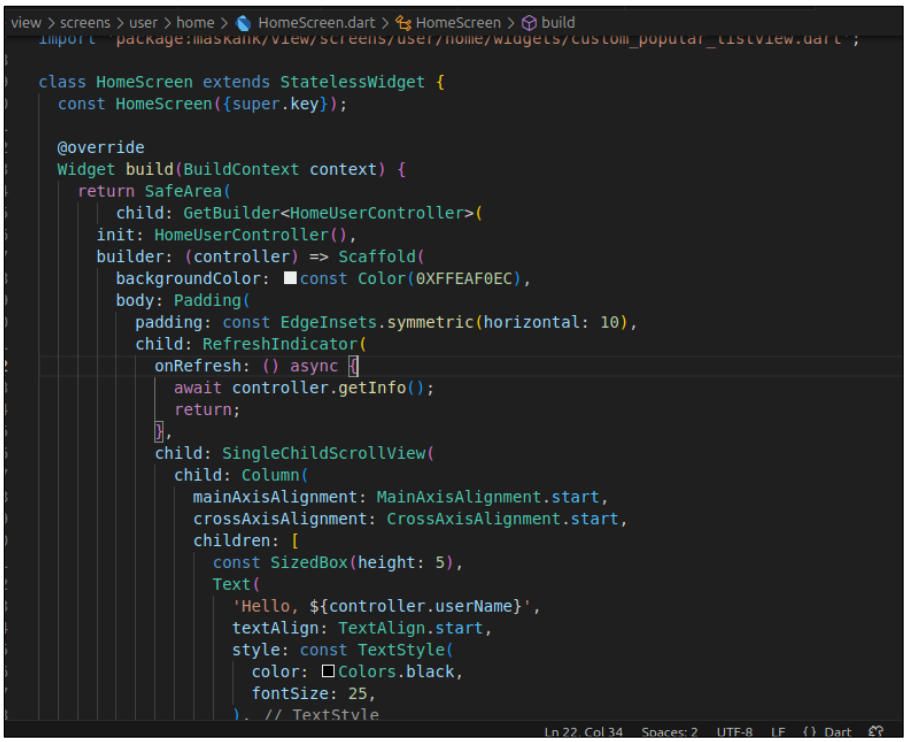
5.5.5 We make API method helper here like post and get && put to share use:



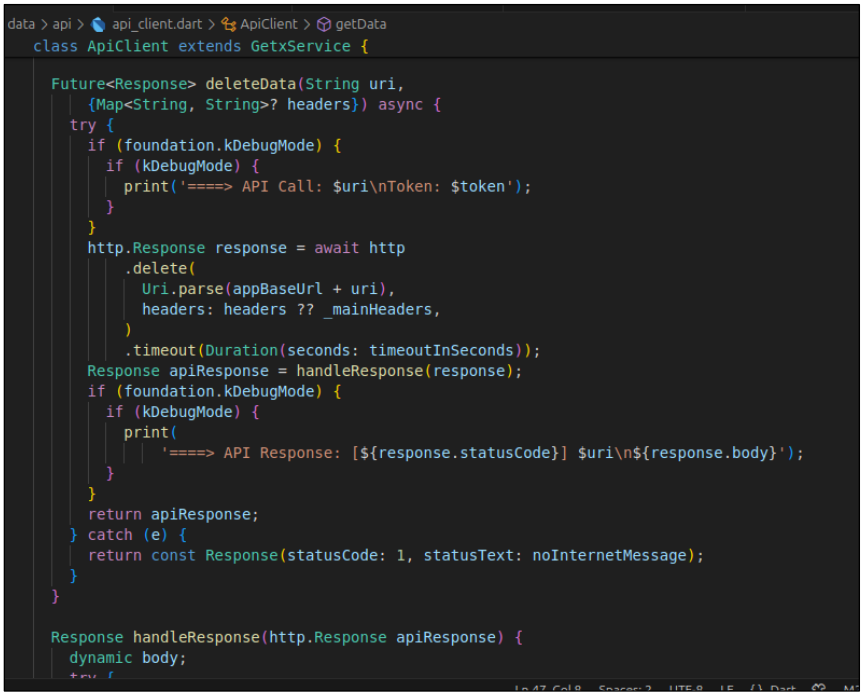
5.5.6Use this get method we make API method helper here like post and get && put to share :



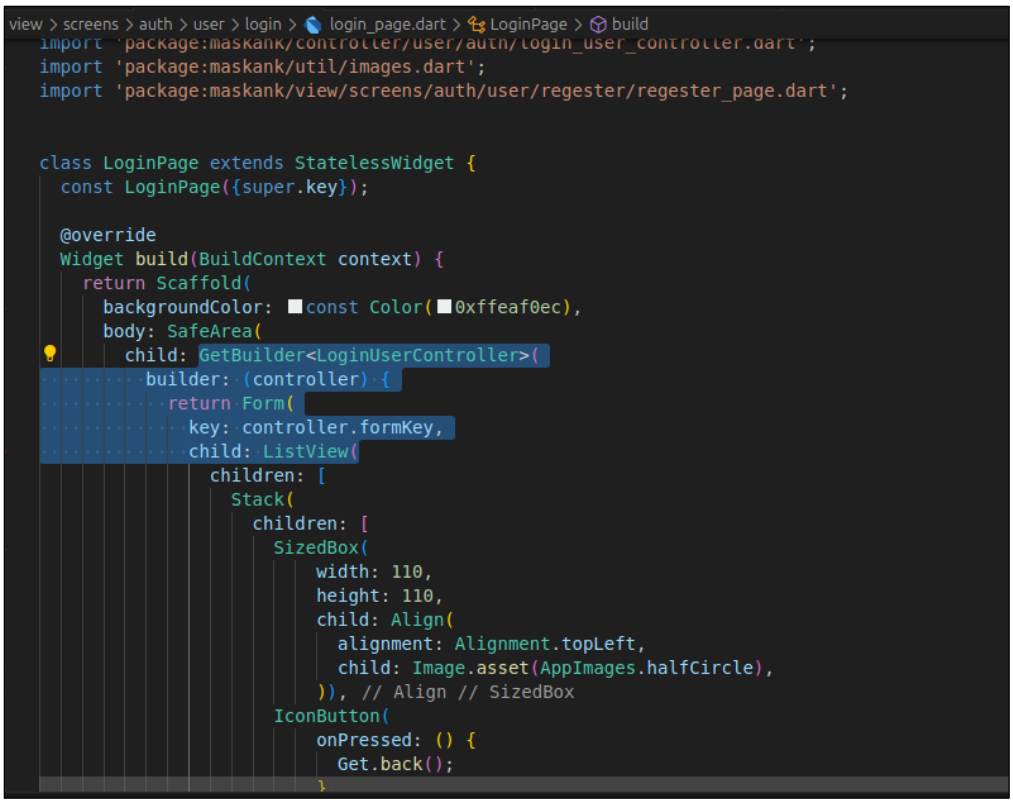
5.5.7 Here view we Used logic Controller of getX by GetBuilder :



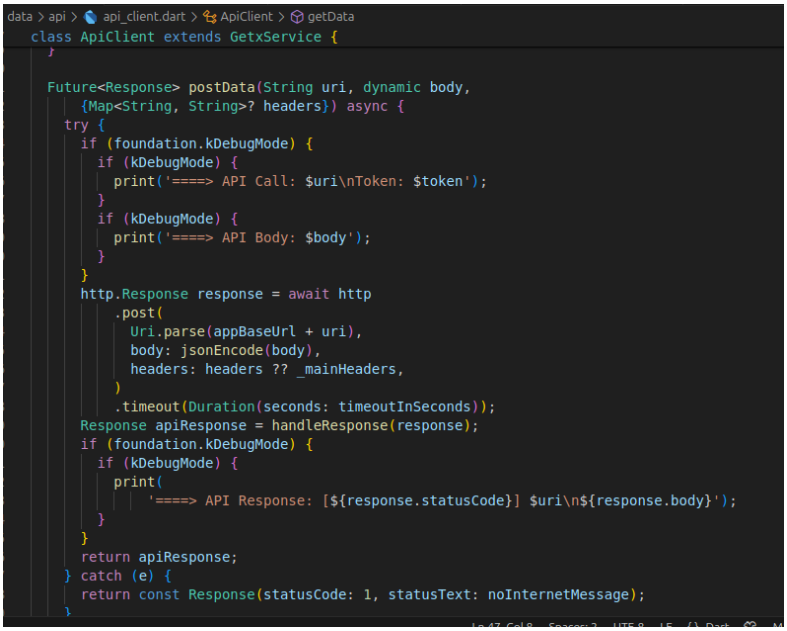
5.5.8 Delete method API:



5.5.9 To link UI with logic here getBuilder of logic Getx :

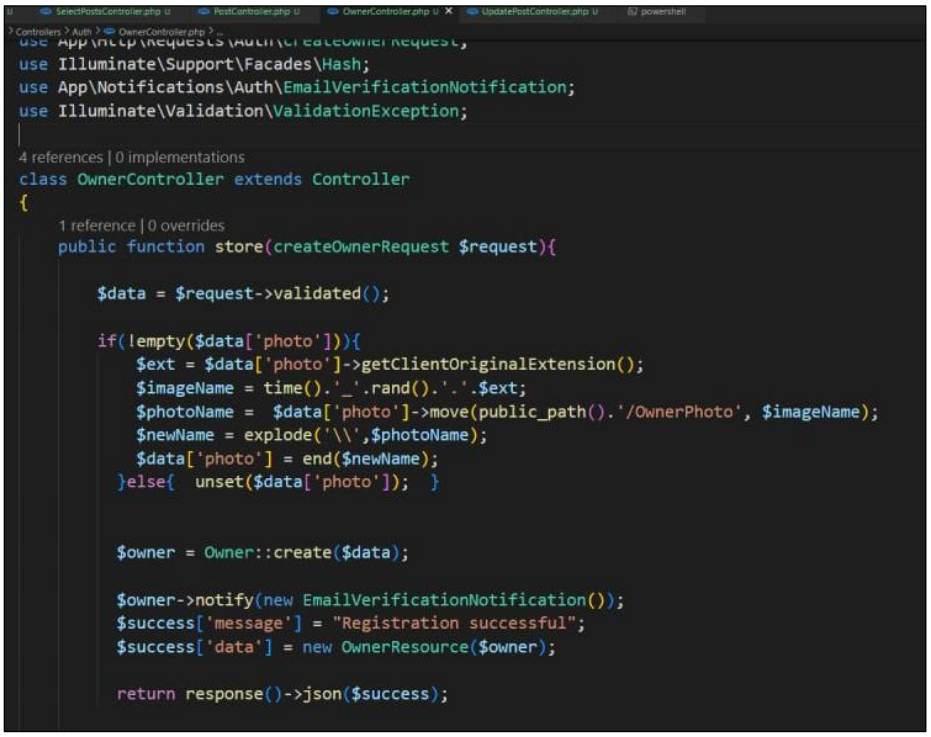


5.5.10 This post method :



5.6 Backend

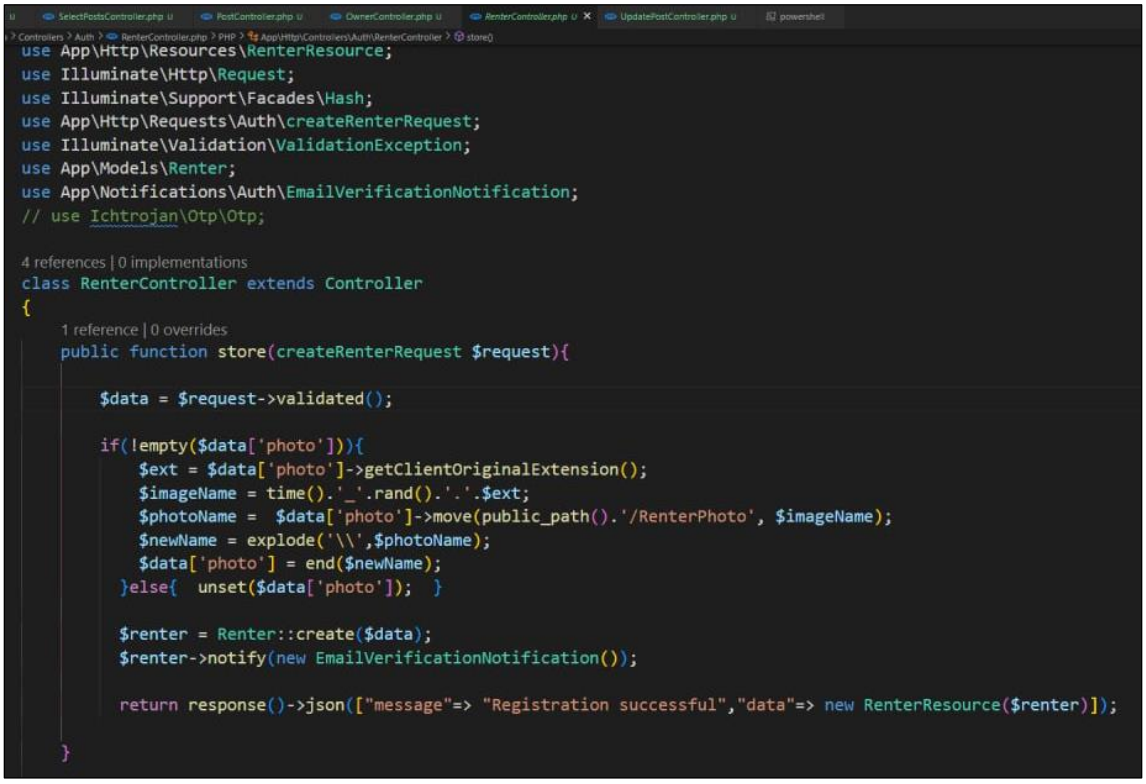
5.6.1 Register login logout for owner:



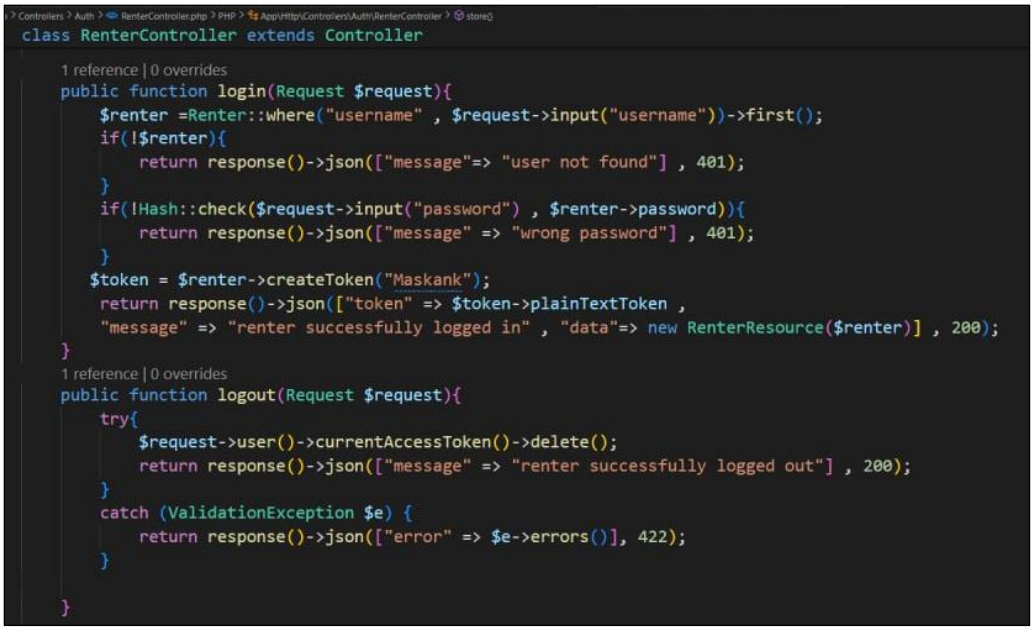
```php
use Illuminate\Support\Facades\Hash;
use App\Notifications\Auth\EmailVerificationNotification;
use Illuminate\Validation\ValidationException;

4 references | 0 implementations
class OwnerController extends Controller
{
    1 reference | 0 overrides
    public function store(createOwnerRequest $request){

        $data = $request->validated();

        if(!empty($data['photo'])){
            $ext = $data['photo']->getClientOriginalExtension();
            $imageName = time().'_'.rand().'.'.$ext;
            $photoName =  $data['photo']->move(public_path().'/OwnerPhoto', $imageName);
            $newName = explode('\\',$photoName);
            $data['photo'] = end($newName);
          }else{  unset($data['photo']);  }


          $owner = Owner::create($data);

          $owner->notify(new EmailVerificationNotification());
          $success['message'] = "Registration successful";
          $success['data'] = new OwnerResource($owner);

          return response()->json($success);
```

```php
class OwnerController extends Controller
{

    1 reference | 0 overrides
    public function login(Request $request){
        $owner =Owner::where("username" , $request->input("username"))->first();
        if(!$owner){
            return response()->json(["message"=> "owner not found"] , 401);
        }
        if(!Hash::check($request->input("password") , $owner->password)){
            return response()->json(["message" => "wrong password"] , 401);
        }
       $token = $owner->createToken("Maskank");
        return response()->json(["token" => $token->plainTextToken ,
        "message" => "owner successfully logged in", "owner" => new OwnerResource($owner)] , 200);
    }

    1 reference | 0 overrides
    public function logout(Request $request)
    {
        try{
            $request->user()->currentAccessToken()->delete();
            return response()->json(["message" => "owner successfully logged out"] , 200);
        }
        catch (ValidationException $e) {
            return response()->json(["error" => $e->errors()], 422);
        }
    }
```

5.6.2 Register login logout for renter :

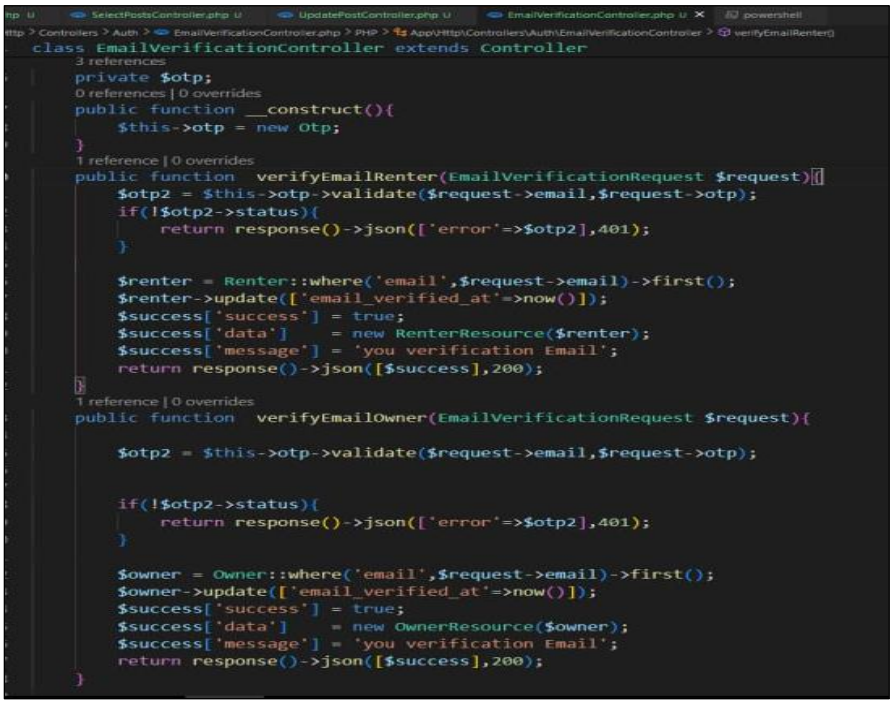


```php
use App\Http\Resources\RenterResource;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Hash;
use App\Http\Requests\Auth\createRenterRequest;
use Illuminate\Validation\ValidationException;
use App\Models\Renter;
use App\Notifications\Auth\EmailVerificationNotification;
// use Ichtrojan\Otp\Otp;

4 references | 0 implementations
class RenterController extends Controller
{
    1 reference | 0 overrides
    public function store(createRenterRequest $request){

        $data = $request->validated();

        if(!empty($data['photo'])){
            $ext = $data['photo']->getClientOriginalExtension();
            $imageName = time().'_'.rand().'.'.$ext;
            $photoName =  $data['photo']->move(public_path().'/RenterPhoto', $imageName);
            $newName = explode('\\',$photoName);
            $data['photo'] = end($newName);
         }else{  unset($data['photo']);  }

         $renter = Renter::create($data);
         $renter->notify(new EmailVerificationNotification());

         return response()->json(["message"=> "Registration successful","data"=> new RenterResource($renter)]);

    }
```



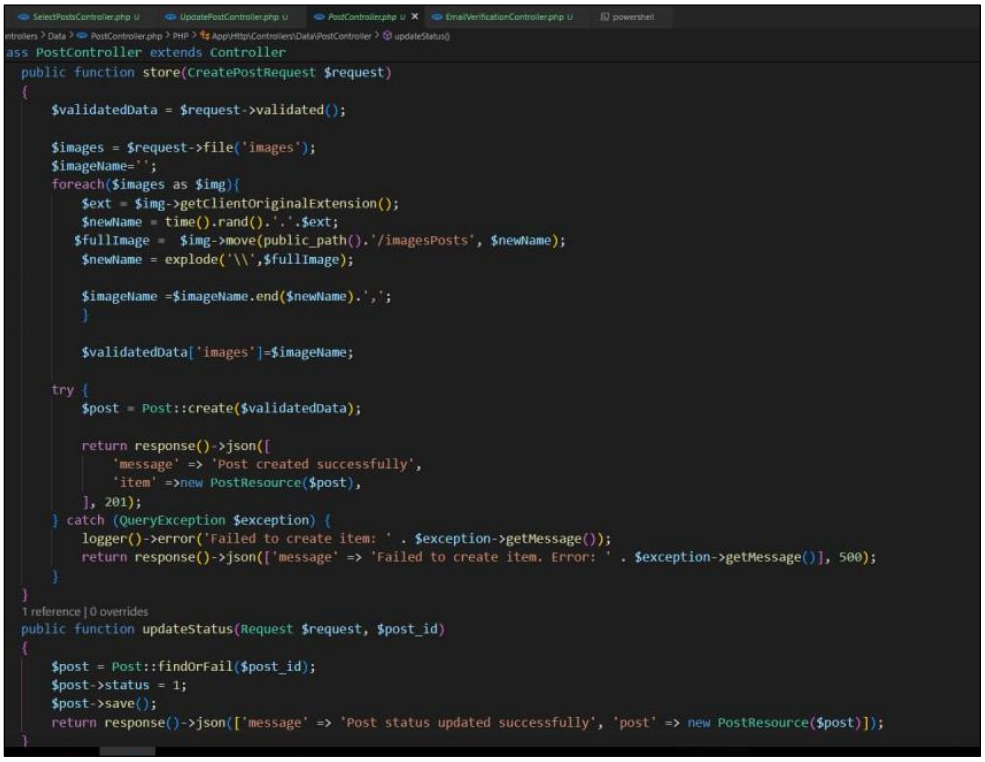
```php
class RenterController extends Controller

    1 reference | 0 overrides
    public function login(Request $request){
        $renter =Renter::where("username" , $request->input("username"))->first();
        if(!$renter){
            return response()->json(["message"=> "user not found"] , 401);
        }
        if(!Hash::check($request->input("password") , $renter->password)){
            return response()->json(["message" => "wrong password"] , 401);
        }
       $token = $renter->createToken("Maskank");
        return response()->json(["token" => $token->plainTextToken ,
        "message" => "renter successfully logged in" , "data"=> new RenterResource($renter)] , 200);
    }
    1 reference | 0 overrides
    public function logout(Request $request){
        try{
            $request->user()->currentAccessToken()->delete();
            return response()->json(["message" => "renter successfully logged out"] , 200);
        }
        catch (ValidationException $e) {
            return response()->json(["error" => $e->errors()], 422);
        }

    }
```

5.6.3 Verification email :



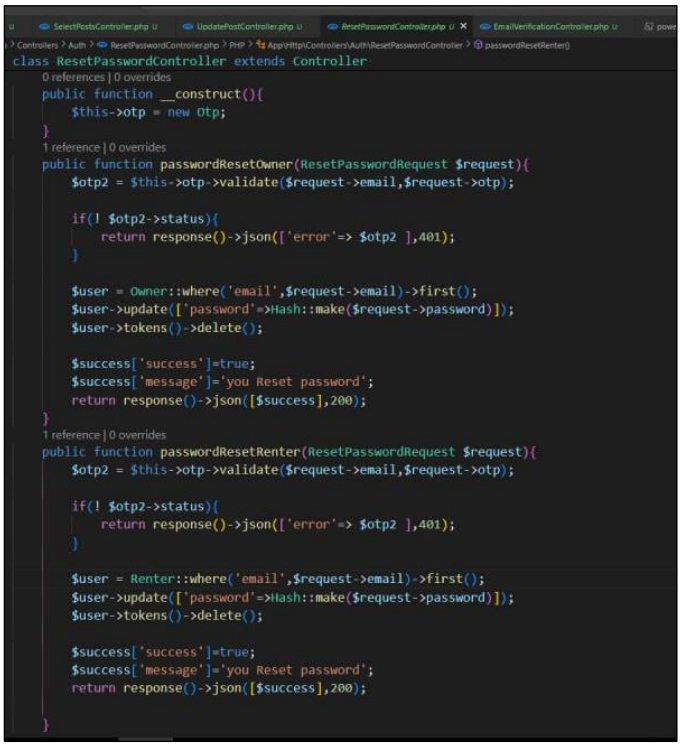
```php
class EmailVerificationController extends Controller
{
    private $otp;
    public function __construct(){
        $this->otp = new Otp;
    }
    public function  verifyEmailRenter(EmailVerificationRequest $request){
        $otp2 = $this->otp->validate($request->email,$request->otp);
        if(!$otp2->status){
            return response()->json(['error'=>$otp2],401);
        }

        $renter = Renter::where('email',$request->email)->first();
        $renter->update(['email_verified_at'=>now()]);
        $success['success'] = true;
        $success['data']    = new RenterResource($renter);
        $success['message'] = 'you verification Email';
        return response()->json([$success],200);
    }
    public function  verifyEmailOwner(EmailVerificationRequest $request){

        $otp2 = $this->otp->validate($request->email,$request->otp);


        if(!$otp2->status){
            return response()->json(['error'=>$otp2],401);
        }

        $owner = Owner::where('email',$request->email)->first();
        $owner->update(['email_verified_at'=>now()]);
        $success['success'] = true;
        $success['data']    = new OwnerResource($owner);
        $success['message'] = 'you verification Email';
        return response()->json([$success],200);
    }
```

5.6.4 Add post && Approved :



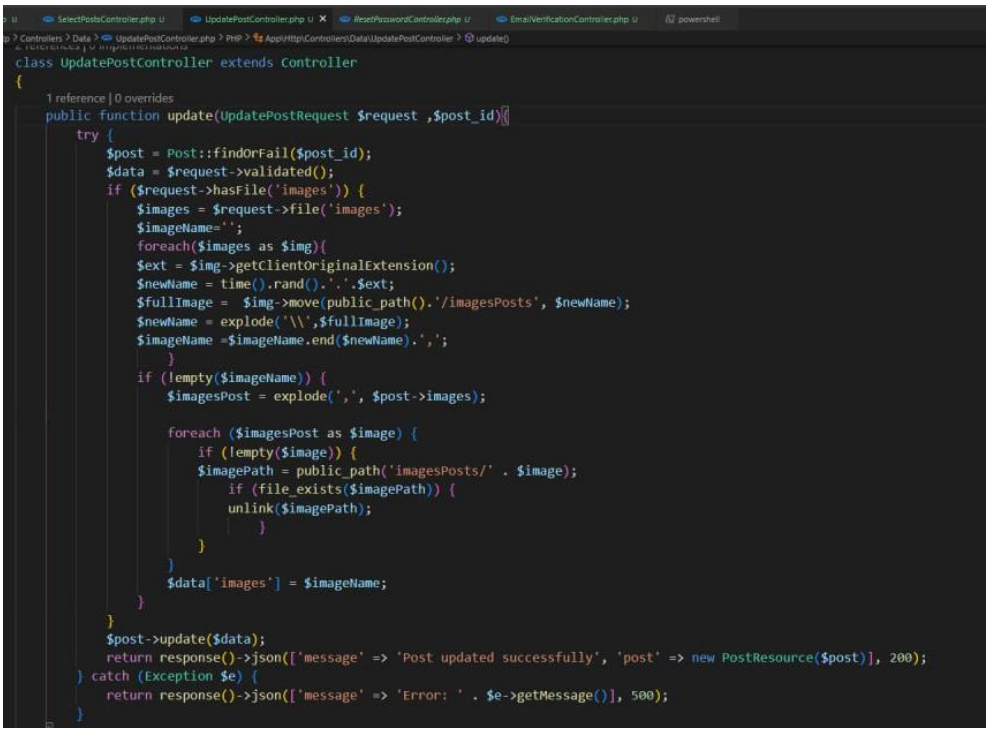
```php
class PostController extends Controller
{
    public function store(CreatePostRequest $request)
    {
        $validatedData = $request->validated();

        $images = $request->file('images');
        $imageName='';
        foreach($images as $img){
            $ext = $img->getClientOriginalExtension();
            $newName = time().rand().'.'.$ext;
           $fullImage =  $img->move(public_path().'/imagesPosts', $newName);
            $newName = explode('\\',$fullImage);

            $imageName =$imageName.end($newName).',';
            }

            $validatedData['images']=$imageName;

        try {
            $post = Post::create($validatedData);

            return response()->json([
                'message' => 'Post created successfully',
                'item' =>new PostResource($post),
            ], 201);
        } catch (QueryException $exception) {
            logger()->error('Failed to create item: ' . $exception->getMessage());
            return response()->json(['message' => 'Failed to create item. Error: ' . $exception->getMessage()], 500);
        }
    }
    public function updateStatus(Request $request, $post_id)
    {
        $post = Post::findOrFail($post_id);
        $post->status = 1;
        $post->save();
        return response()->json(['message' => 'Post status updated successfully', 'post' => new PostResource($post)]);
    }
```

5.6.5 Reset password :



```php
class ResetPasswordController extends Controller
    0 references | 0 overrides
    public function __construct(){
        $this->otp = new Otp;
    }
    1 reference | 0 overrides
    public function passwordResetOwner(ResetPasswordRequest $request){
        $otp2 = $this->otp->validate($request->email,$request->otp);

        if(! $otp2->status){
            return response()->json(['error'=> $otp2 ],401);
        }

        $user = Owner::where('email',$request->email)->first();
        $user->update(['password'=>Hash::make($request->password)]);
        $user->tokens()->delete();

        $success['success']=true;
        $success['message']='you Reset password';
        return response()->json([$success],200);
    }
    1 reference | 0 overrides
    public function passwordResetRenter(ResetPasswordRequest $request){
        $otp2 = $this->otp->validate($request->email,$request->otp);

        if(! $otp2->status){
            return response()->json(['error'=> $otp2 ],401);
        }

        $user = Renter::where('email',$request->email)->first();
        $user->update(['password'=>Hash::make($request->password)]);
        $user->tokens()->delete();

        $success['success']=true;
        $success['message']='you Reset password';
        return response()->json([$success],200);

    }
```

5.6.6 Update post :



```php
class UpdatePostController extends Controller
{
    1 reference | 0 overrides
    public function update(UpdatePostRequest $request ,$post_id){
        try {
            $post = Post::findOrFail($post_id);
            $data = $request->validated();
            if ($request->hasFile('images')) {
                $images = $request->file('images');
                $imageName='';
                foreach($images as $img){
                $ext = $img->getClientOriginalExtension();
                $newName = time().rand().'.'.$ext;
                $fullImage =  $img->move(public_path().'/imagesPosts', $newName);
                $newName = explode('\\',$fullImage);
                $imageName =$imageName.end($newName).',';
                    }
                if (!empty($imageName)) {
                    $imagesPost = explode(',', $post->images);

                    foreach ($imagesPost as $image) {
                        if (!empty($image)) {
                        $imagePath = public_path('imagesPosts/' . $image);
                            if (file_exists($imagePath)) {
                            unlink($imagePath);
                                }
                            }
                        }
                    $data['images'] = $imageName;
                }
            }
            $post->update($data);
            return response()->json(['message' => 'Post updated successfully', 'post' => new PostResource($post)], 200);
        } catch (Exception $e) {
            return response()->json(['message' => 'Error: ' . $e->getMessage()], 500);
        }
```

Chapter 7

Conclusion and Future Work

6.1 Introduction

This part summarizes all our research effort and the main challenges that we faced during our research. We will also discuss our future efforts and the main things we want to add in our application.

6.2 research effort

We focused all our effort to find an easy way and confident place for finding apartment to student specially to rent with good prices and dispensing with the broker by using AI to determines the price of the apartment and find a real database with real prices for rent.

6.3 Challenges

The main technical challenges that we faced are the following:

- Lack of data for rent in all governorates
- Lack of data for rent with real prices

6.4 Conclusion

This documentation summarizes our work at every stage of the project. When we started with our project, we had so many ideas and so many features that we wanted to add, we didn't have a clear idea on how we would implement them but we didn't stop searching and working to try and make our features as suitable for parents as possible. We had knowledge of machine learning subjects, we were learning more and more every day, we worked constantly for the short time period we had which was mainly 6 months. As required in every project, research is utmost important. So, we spent a lot of time going through the background 80 literature. We looked at various approaches for doing the models. Finally, we tried our best to overcome all the problems that we faced by applying new techniques every day.

6.5 Future work

This is not the end of our road, we will keep working on our application to release it to the public and increase the quality of the features, it still needs more research and development, and this is what we will continue to do to reach the best possible efficiency.

6.5.1 Enhance accuracy of models.

6.5.2 Find more database for all governorates.

6.6 related work

Bayut egypt	property finder	Our application (MASKNK)
(https://www.bayut.eg/en/)	(https://www.propertyfinder.eg/en)	(link the app)
Its an application is used to buy apartment, villa,town house,chalet….	Its an application used for buy and rent house,office,villa,apartment,studio……	Its an application used for rent house,apartment……
• Its talk to rich people • Its only used in buy not rent • The seller display the price that he want • Not comfortable in use	• Its so expensive • The seller display the price that he want	• Its almost talk to students and its our goal to help them to find apartments to rent for study • Its not expensive its talking to middle class • The best advantage is that the AI detect the price of the apartment and therefor we dispense with realtors . • The ease of ui that make it comfortable in using

6.7References

I. **Project link**

II. **Flutter**
https://pub.dev/
https://fluttergems.dev/

III. **Machine Learning**
http://scikit-learn.org
http://kaggle.com
https://docs.python.org/3/library/index.html https://gemini.google.com/app

IV. **Back End**
https://laravel.com https://www.php.net/manual/en/langref.php

V. **UI**
https://dribbble.com https://www.behance.net

ملخص المشروع باللغة العربية

الهدف الرئيسي من تطبيق مسكنك هو إحداث ثورة في تجربة استئجار الشقق للطلاب والمصطافين والأفراد الباحثين عن سكن. من خلال الاستفادة من التقنيات المتقدمة، يهدف التطبيق إلى توفير تجربة استئجار سلسة وشخصية. تشمل الأهداف الرئيسية تحسين تجربة المستخدم عبر واجهة بديهية وتوصيات مخصصة. يستهدف التطبيق جمهورًا واسعًا، مع التركيز على الإيجارات القصيرة والطويلة الأجل. يدمج التطبيق تقنيات متقدمة مثل فلاتر لتطوير واجهة المستخدم والذكاء الاصطناعي للتصفية الذكية والتوصيات والتنبؤ بالأسعار. الابتكار المستمر والتكامل مع الخدمات الخارجية يضمنان بقاء التطبيق تنافسيًا وتلبية احتياجات المستخدم المتطورة.

ويعطينا التطبيق عدة مزايا مثل:

واجهة مستخدم سلسة وبسيطة لسهولة استخدام التطبيق

نخاطب الطلاب بشكل كبير لتسهيل ايجاد مسكن

الاسعار واقعية تناسب الفئة المتوسطه

الاستغناء عن السمسار

استخدام الذكاء الاصطناعي للتنبوء بالسعر

المشكله المراد حلها

يعد سوق العقارات بيئة معقدة وديناميكية حيث يتفاعل المشترون والبائعون مع الوكلاء والوكالات لتسهيل المعاملات العقارية. غالبًا ما تستغرق الطرق التقليدية للمعاملات العقارية وقتًا طويلاً، وتفتقر إلى الشفافية، ويمكن أن تكون غير فعالة بسبب الاعتماد على الأوراق المادية والتفاعلات وجهًا لوجه. ومع تزايد اعتماد التكنولوجيا الرقمية، هناك حاجة إلى منصة شاملة لتبسيط العملية العقارية، وجعلها أكثر سهولة وشفافية وكفاءة لجميع الأطراف المعنية.

التحديات الحالية:

عدم وجود معلومات مركزية:

غالبًا ما يكافح المشترون والبائعون للعثور على مصدر مركزي لمعلومات الملكية الموثوقة.

المعلومات متناثرة عبر منصات مختلفة، مما يجعل من الصعب المقارنة واتخاذ قرارات مستنيرة.

التواصل غير الفعال:

يمكن أن يكون التواصل بين المشترين والبائعين والوكلاء بطيئًا ومجزأًا، مما يؤدي إلى التأخير وسوء الفهم في عملية المعاملة.

الوصول المحدود إلى بيانات السوق:

ويفتقر المشترون والبائعون إلى إمكانية الوصول إلى بيانات السوق في الوقت الفعلي، مثل أسعار العقارات، واتجاهات السوق، وإحصاءات الأحياء، والتي تعتبر ضرورية لاتخاذ قرارات مستنيرة.

العمليات اليدوية والورقية:

تعتمد العملية العقارية التقليدية بشكل كبير على الأعمال الورقية اليدوية، والتي تستغرق وقتًا طويلاً، وعرضة للأخطاء، وتفتقر إلى الشفافية.

صعوبة العثور على وكلاء جديرين بالثقة:

غالبًا ما يجد المشترون والبائعون صعوبة في التعرف على وكلاء العقارات الجديرين بالثقة والكفاءة والتواصل معهم.

تصور غير مناسب للملكية:

إن خيارات تصور العقارات المحدودة (مثل الصور ومقاطع الفيديو والجولات الافتراضية) تعيق المشترين من الحصول على رؤية شاملة للعقارات التي يهتمون بها.

الادوات المستخدمة

حسابات المستخدمين وملفاتهم الشخصية:

السماح للمستخدمين (المشترين والبائعين والوكلاء) بإنشاء ملفاتهم الشخصية وإدارتها باستخدام لوحات معلومات مخصصة.

البحث عن العقارات والتصفية:

قم بتنفيذ وظيفة بحث قوية باستخدام مرشحات للموقع والنطاق السعري ونوع العقار والمعايير الأخرى ذات الصلة.

الخرائط التفاعلية:

دمج الخرائط التفاعلية لعرض مواقع العقارات والمرافق القريبة.

المفضلة والتنبيهات:

تمكين المستخدمين من حفظ العقارات المفضلة وإعداد التنبيهات للقوائم الجديدة التي تتوافق مع معاييرهم.

المراسلة الآمنة:

توفير نظام مراسلة آمن للمستخدمين للتواصل مباشرة داخل المنصة.

جدولة المواعيد:

السماح للمستخدمين بجدولة عروض العقارات والتشاور مع الوكلاء مباشرة من خلال التطبيق.

دعم متعدد اللغات:

يدعم التطبيق لغات متعددة لتلبية احتياجات قاعدة مستخدمين متنوعة

هدف المشروع

يقدم مشروعنا:

تبسيط وتعزيز التجربة العقارية للمشترين والبائعين والوكلاء من خلال توفير منصة شاملة وسهلة الاستخدام.

قوائم العقارات في الوقت الفعلي: تحديثات فورية لقوائم العقارات الجديدة المصممة خصيصًا لتفضيلات المستخدم.

عوامل تصفية البحث المتقدم: أدوات بحث شاملة للعثور على العقارات بناءً على معايير محددة مثل الموقع ونطاق السعر ونوع العقار ووسائل الراحة.

الخرائط التفاعلية: خرائط تفصيلية تعرض مواقع العقارات والمرافق القريبة وإحصائيات الأحياء للمساعدة في اتخاذ القرار.

الجولات الافتراضية: توفر الجولات الافتراضية عالية الجودة إحساسًا واقعيًا بالملكية دون الحاجة إلى زيارة شخصية.

الجدولة والحجز داخل التطبيق: أدوات لجدولة عروض العقارات وعمليات التفتيش والاجتماعات مع وكلاء العقارات مباشرة من خلال التطبيق.

ميزات إدارة الممتلكات: أدوات لجدولة العروض وإدارة القوائم وتتبع تقدم المعاملات.

توصيات مخصصة: اقتراحات تعتمد على الذكاء الاصطناعي بناءً على تفضيلات المستخدم وسجل البحث.

نوفر على المستخدم الوقت والجهد لكل من المستأجر ومالك الشقة.

واجهة مستخدم جذابة: توفير واجهة مستخدم سهلة الاستخدام وجذابة بصريًا للمستخدمين لتصفح الشقق وحجزها بسلاسة.

تسجيل المستخدم وإدارة الحساب: تمكين المستخدمين من تسجيل وإدارة حساباتهم دون عناء

توصيات مخصصة باستخدام الذكاء الاصطناعي: استخدم تقنيات الذكاء الاصطناعي للتوصية بالشقق بناءً على تفضيلات المستخدم وسجل البحث السابق

إدارة الشقق والإعلانات: توفير إمكانية إضافة وإدارة الشقق المتاحة للإيجار، مع ضمان دقة وجودة الإعلان