



Rentomate

FINAL YEAR PROJECT REPORT 2022

**FACULTY OF COMPUTER SCIENCE AND
ENGINEERING (FCSE)**

**GHULAM ISHAQ KHAN INSTITUTE OF ENGINEERING
SCIENCES AND TECHNOLOGY (GIKI)**

Group Members

M.Umais Khan (2018341)

Ali Usama Nawaz (2018064)

Hassaan Ahmad (2018141)

Hamza Asad (2018128)

Advisor:

Ahsan Shah

Certificate of Approval

It is certified that the work presented in this report was performed by **Ali Usama Nawaz, Hassaan Ahmad, Muhammad Umais Khan and Hamza Asad** under the supervision of **Ahsan Shah**. The work is adequate and lies within the scope of the BS degree in Computer Science/Computer Engineering at Ghulam Ishaq Khan Institute of Engineering Sciences and Technology.

Ahsan Shah

(Advisor)

Dr. Ahmar Rashid

(Dean)

ABSTRACT

Rentomate is a home rental system, it is a web-based platform that allows users to either rent a home or put their home up for rent. The main goal of this project is to automate the complete process of renting a home. System has an interactive and user-friendly front-end and it uses role-based authentication. There are three stakeholders in the system i.e. Tenant, Landlord and Administrator. The system lists properties that are available for renting and shows it to tenants, system also incorporates a recommender system (Content Based Filtering) that shows relevant property listings respective to the tenant. Landlords can put up their property up for rent and can use the prediction analysis provided by the system to predict the approximate rent of their home. The system administrator acts as a mediator between landlord and tenant to avoid any fraudulent activities and introduce complete transparency between the concerned parties.

This document discusses the design, development, and detailed description of our project which consists of a machine learning model and a web app to visualize the results and implementation of the developed system.

ACKNOWLEDGEMENT

First and foremost, we express our gratitude to Allah Almighty for blessing us and providing us with the strength and intelligence necessary to complete this undertaking. We would like to convey our heartfelt gratitude to everyone who made it possible for us to complete this project. Ahsan Shah, our final year project advisor, deserves special thanks for his stimulating recommendations and support, which enabled us to effectively organize our project. Finally, many thanks go to our whole faculty of department of Computer Science and Engineering who have invested their full effort in guiding the team in achieving the goal. We appreciate the guidance given by the panel members in the form of comments and advice during our project evaluations.

TABLE OF CONTENTS

<i>Certificate of Approval</i>	2
<i>ABSTRACT</i>	3
<i>ACKNOWLEDGEMENT</i>	4
<i>TABLE OF CONTENTS</i>	5
<i>LIST OF FIGURES</i>	7
<i>LIST OF TABLES</i>	8
<i>CHAPTER I: INTRODUCTION</i>	9
Motivation	9
Project Perspective	9
Project Scope	10
Product Functioning	10
User Characteristics	11
<i>CHAPTER II: LITERATURE SURVEY</i>	12
<i>CHAPTER III: DESIGN</i>	14
Product Functions	14
Constraints	14
Assumptions and Dependencies	14
User Interfaces	15
Hardware Interfaces	15
Software Interfaces	15
Communication Interfaces	15
Functional Requirements	16
Functional Requirements with traceability information	17
Non-Functional Requirements	21
4. Performance Requirements: Workload	21
5. Performance Requirements: Scalability	21
6. Performance Requirements: Platform	22
7. Safety and Security Requirements	22
Software Quality Attributes	22
2. Availability	23
3. Maintainability	23
4. Business Rules	24
5. Accuracy	24
<i>CHAPTER IV: PROPOSED SOLUTION</i>	25

Architectural Design	25
Why MVC Architecture?	27
UML Diagrams	28
1. Use Case Diagram.....	28
2. Class Diagram	31
3. Component Diagram	33
4. Activity Diagram	34
5. Deployment Diagram	37
CHAPTER V: RESULTS AND DISCUSSION	38
Data Preprocessing:	38
Dataset before Cleaning:	39
Dataset after Cleaning:	39
Data Visualization	40
Result Using XGBRegression:	41
Result Using Linear Regression:	41
Result Using Random Forest Regression:	42
Rent Predictor using Random Forest Regression:	42
CHAPTER VI: CONCLUSION AND FUTURE WORK	47
REFERENCES	49

LIST OF FIGURES

Figure 1 MVC Architectural Model	25
Figure 2 Use case 1	29
Figure 3 Use Case 2	30
Figure 4 Use Case 3	31
Figure 5 Class Diagram	32
Figure 6 Component Diagram	33
Figure 7 Activity Diagram	34
Figure 8 Activity Diagram	35
Figure 9 Activity Diagram	36
Figure 10 Deployment Diagram	37
Figure 11 Home Page	43
Figure 12 Property Listing Page	44
Figure 13 Landlord Dashboard	45
Figure 14 Login Page	45
Figure 15 Registration Page	46

LIST OF TABLES

Table 1 - Functional Requirements	16
Table 2 FR1	17
Table 3 FR2	18
Table 4 FR3	18
Table 5 FR4	19
Table 6 FR5	19
Table 7 FR6	20
Table 8 FR7	20

CHAPTER I: INTRODUCTION

Motivation

According to a survey done by Gallup Pakistan in 2019, around 22% of Pakistan's population is living in rented homes which almost accounts to 48.4 million people and all of these people lack a proper platform that provides all of the features mentioned above collectively. Real Estate has been neglected by technology for a long time especially home renting sector as there is not a single platform available that streamlines home renting process. However, with the paradigm shift in technology it is about time that this changes so that the renting process becomes effective, and people included in this process can go through it smoothly and hassle free.

Project Perspective

The main purpose of this project is to provide end-users (tenants and homeowners) with a platform that caters to their needs properly by automating the whole process due to which time and physical effort look for a home or put a home up for rent is greatly reduced as well as introduce a mediator to resolve any conflicts that may arise before or after going through either of the processes. It also provides a platform to homeowners who are looking to put their houses up on rent by giving them a rent approximation feature where they can add features of their houses and according to those features an approximate rent will be generated which would give them an idea which they could use to their benefit while negotiating with potential tenants.

Project Scope

Rentomate is a home rental system, it is a web-based platform that allows users to either rent a home or put their home up for rent. This system basically has two end users i.e., tenants and homeowners. It basically provides tenants a platform where they could go through an extended list of homes to rent from. It also includes a filtering mechanism through which tenants can easily look for what they are interested in rather than going through the whole list. The aim of this system is not only to give exclusivity to the end users but also automate the home-renting process completely and securely but also make the process efficient, effective and hassle free.

Product Functioning

First off when the users access the website for the first time, they must sign up for which they have to provide all the required information like CNIC, email, Name etc. System also provides the users to manage their accounts where they can update their information. System not only incorporates a filtration mechanism which helps going through the house listings in an efficient manner but also a Recommender System that operates on Content-based filtering which shows relevant houses to the user based on their preferences. System also provides homeowners with a feature where they can manage their properties as well as use a prediction analysis algorithm based on machine learning that will help them approximate the rent of their property. Chatting platform is another function integrated into the system through which both end users can communicate and strike a deal that best suits them. Safe payment gateways have also been incorporated into the system for secure transactions with alerts which notifies both end users if a payment is either due or has been paid depending on the end user.

User Characteristics

This system consists of two main end users i.e., Tenants and Homeowners.

Product's perspective from both point of views is as following:

1. **Tenants** access the system as a platform where they can search for houses according to their preferences and after finding a house, they can negotiate the price with the homeowner and finalize the deal.
2. **Homeowners** access the system as a platform where they can put their houses up for rent and when a tenant is interested in their house, they can negotiate the terms and finalize the deal

CHAPTER II: LITERATURE SURVEY

In the most straightforward terms, Linear Regression is an administered Machine Learning model that distinguishes the best fit direct line between the free and subordinate factors, for example it finds the straight connection between the two factors. There are two types of straight relapse: straightforward and different. Just a single autonomous variable is available in straightforward direct relapse, and the model should lay out a direct connection among it and the reliant variable. Different Linear Regression, then again, utilizes more than one free factor to track down a relationship. The aim of model based on linear regression is to find a linear best-fit line that results in minimizing error.

Linear Regression is not the only model that was used to train the data, but two other models were also trained i.e., Extreme Gradient Boosting Regression and Random Forest Regression. Gradient boosting is a subset of ensemble machine learning methods for solving classification and regression predictive modelling problems. Ensembles are created using decision tree models. Trees are introduced to the ensemble one by one and fitted to correct previous model prediction errors one by one. XGB Regression is an ensemble machine learning model i.e., it uses multiple models and simultaneously trains and test them on the given dataset and choose the best model based on various ensemble techniques. XGB model uses gradient descent optimization technique and differentiable loss function. It is like neural network as when the model is fitted, it decreases the loss gradient.

Random Forest Regression is also an ensemble machine learning model that is based on supervised learning part of machine learning. The main aim of ensemble learning is that it joins multiple machine learning algorithms to

generate a more precise and accurate model than a standalone model. It basically works by constructing multiple decision trees/regression trees while training and gives the mean of the classes as the prediction of all combined algorithms (trees). This technique greatly improves accuracy but can sometime also result in over-fitting which is not a desirable attribute.

All the above-mentioned algorithms have been used to train and test the dataset. These models have also been evaluated using different metrics for regression like R-Squared Error, Mean Absolute Error, Adjusted R-Squared Error, and the model that got the maximum score or in other words most accurate among the above-mentioned algorithm was Random Forest Regression and the house rent predictor module is based upon this algorithm.

CHAPTER III: DESIGN

Product Functions

This product performs three major functions:

- Automation of rent collection
- Price Prediction
- Recommender System

Constraints

The only core constraint to access the system is stable internet connection along with a computer that can access the web is required.

Assumptions and Dependencies

This system basically automates the home renting process which has some legal aspect attached to it so one of the dependencies is that a deal between homeowner and tenant cannot go through unless they provide a signed affidavit along with a police report as currently there are no APIs available. Another dependency is of verifying the user when they sign-up on the website since NADRA hardly provides access to the verification API to personal businesses.

User Interfaces

User Interfaces will be divided into three main categories, namely the administrator who has the privileged access to the system functions, the landlord who wishes to manage their listings of property and tenants who want to browse the listing of the property and communicate with the landlord. The admin dashboard allows access of the database. The homepage will show the listings along with their details. Finally, the Property Listing page will show the properties listed by tenants and their status.

Hardware Interfaces

The product will require a use of a computing device such as PC, mobile phone tablet etc. An active internet connection is required.

Software Interfaces

The product will use machine learning models to process data. The output of will be saved onto the database and users can interact with the system via the web dashboard.

Communication Interfaces

All communications between users will be done via the web dashboard

Functional Requirements

The main functional requirements required for the system to operate are mentioned below in table 3.1, however, each ID is fully explained in the following section.

Table 1 - Functional Requirements

FR1	Authentication
FR2	Account Management
FR3	Filtration System
FR4	Property Management
FR5	Communication & Deal
FR6	Payment Management
FR7	Admin Support & Alerts

Functional Requirements with traceability information

Table 2 FRI

Requirement ID	FR1		Requirement Type		Functional		Use Case #		1
Status	<i>New</i>		<i>Agreed-to</i>	-	<i>Baselined</i>	-	<i>Rejected</i>	-	
Parent Requirement #									
Description	Registration and Authentication is required prior to using the system								
Rationale	It is necessary to prevent unauthorized access to the data								
Source					Source Document		-		
Acceptance/Fit Criteria	The user will have access to their dashboard								
Dependencies									
Priority	<i>Essential</i>	X	<i>Conditional</i>		-	<i>Optional</i>	-		
Change History									

Table 3 FR2

Requirement ID	FR2		Requirement Type	Functional		Use Case #		1
Status	New		Agreed-to	-	Baselined	-	Rejected	-
Parent Requirement #	Authentication							
Description	Management of all information related to account							
Rationale	This is necessary to keep track of information and keeping in up to date							
Source					Source Document	-		
Acceptance/Fit Criteria	The user can see the status of the property and invoices							
Dependencies								
Priority	Essential	x	Conditional	-	Optional	-		
Change History								

Table 4 FR3

Requirement ID	FR3		Requirement Type	Functional			Use Case #	3
Status	<i>New</i>		<i>Agreed-to</i>	-	<i>Baselined</i>	-	<i>Rejected</i>	-
Parent Requirement #								
Description	Filtration of properties according to defined attributes							
Rationale	It is important for users to see filtered properties which are relevant to them							
Source					Source Document	-		
Acceptance/Fit Criteria	Users can see properties in which they could be interested and make decisions							
Dependencies								
Priority	<i>Essential</i>	X	<i>Conditional</i>	-	<i>Optional</i>	-		
Change History								

Table 5 FR4

Requirement ID	FR4		Requirement Type	Functional		Use Case #		2	
Status	<i>New</i>		<i>Agreed-to</i>	-	<i>Baselined</i>	-	<i>Rejected</i>	-	
Parent Requirement #									
Description	Property Management System for maintaining the System								
Rationale	It is necessary to keep record of properties up to date								
Source					Source Document		-		
Acceptance/Fit Criteria	Allowing users to add/delete properties and perform various other actions								
Dependencies									
Priority	<i>Essential</i>	X	<i>Conditional</i>	-	<i>Optional</i>	-			
Change History									

Table 6 FR5

Requirement ID	FR5		Requirement Type	Functional		Use Case #		2	
Status	<i>New</i>		<i>Agreed-to</i>	-	<i>Baselined</i>	-	<i>Rejected</i>	-	
Parent Requirement #									
Description	Communication System between landlord and tenant								
Rationale	It is necessary for users to establish means of contact in case of any query								
Source					Source Document	-			
Acceptance/Fit Criteria	Allows users to chat with each other and assisting in making a deal								
Dependencies									
Priority	<i>Essential</i>	X	<i>Conditional</i>	-	<i>Optional</i>	-			
Change History									

Table 7 FR6

Requirement ID	FR6	Requirement Type		Functional		Use Case #		3
Status	<i>New</i>		<i>Agreed-to</i>	-	<i>Baselined</i>	-	<i>Rejected</i>	-
Parent Requirement #								
Description	Payment system to collect rent and miscellaneous fees							
Rationale	It is necessary for to streamline the process of collecting dues							
Source					Source Document		-	
Acceptance/Fit Criteria	Users can receive or pay rent and other fees							
Dependencies								
Priority	<i>Essential</i>	X	<i>Conditional</i>	-	<i>Optional</i>	-		
Change History								

Table 8 FR7

Requirement ID	FR7		Requirement Type	Functional		Use Case #	3	
Status	<i>New</i>		<i>Agreed-to</i>	-	<i>Baselined</i>	-	<i>Rejected</i>	-
Parent Requirement #								
Description	Administration support carries the duties of alerting concerned users with information and responding to user queries							
Rationale	It is necessary to inform users regarding their status on various things							
Source					Source Document	-		
Acceptance/Fit Criteria	Users shall be able to get informed when rent has been paid or arrival of due date.							
Dependencies								
Priority	<i>Essential</i>	X	<i>Conditional</i>	-	<i>Optional</i>	-		
Change History								

Non-Functional Requirements

3. Performance Requirements: Response Time

Databases use real time processing to handle workloads causing the data transactions to be quick and response time to be less. They also lay out secure, static, and production-grade hosting for developers. Performance here means the time it takes to handle a request. The app should be performant throughout the life. There max time for serving request should never be greater than one second.

4. Performance Requirements: Workload

A few workload requirements for the database and system are listed below. These Requirements define the capacity of the system and database to deal with transactions of data. The number of reads and writes that can be performed by the database are also listed below.

5. Performance Requirements: Scalability

It has been ensured that the web nodes which is running the Django code are independent from your persistence layer (database, caching, session storage etc) thus we can scale then independently.

Since the Django web nodes have no stored state, they scale horizontally. Due to the large open-source community of Django, most of the tools that is needed to scale application already exists. For example, Amazon Web Services (AWS) S3 is being used and Database indexes also make it much more

efficient to look up records by date or other indexed criteria.

6. Performance Requirements: Platform

The platform will use HTML, CSS, Bootstrap and react for the front-end of the application. PostgreSQL will be used for the database services. Django will be used for back-end services.

7. Safety and Security Requirements

This refers to the measures taken to protect the privacy of the user. Another important consideration is to avoid ransomware attack such as database encryption. To ensure this we are using following techniques:

- JWT tokens for Authentication and Authorization
- SSL to avoid eavesdrop attack
- API endpoints require authorization
- Ports remapping to avoid attack on components of the software.
- Minimum ports are exposed to public. Private networks are used to communicate between database and server.

Software Quality Attributes

1. Reliability

The created software will be dependable, bug-free, and error-free. Users can obtain various information about the properties. When you use any of the app's features, services will be delivered fast and correctly. In the event of an issue, the user will be able to provide feedback and will receive a prompt response from the administration staff.

2. Availability

If the user has a reliable internet connection, this application/website will be available 24 hours a day, 7 days a week. Canary deployment method to be used to avoid downtime. This deployment method deploys new code to just few servers at first and if there is breaking change the other servers would still work lowering the risk of downtime

3. Maintainability

Our app should be architecture such that it allows easy addition of new features without affecting the core app. Instead of monolithic architecture in which all functionality is written as a single app we would split app into components which will communicate through APIs. This technique will enable us to scale the product better as well as new features can easily be added without disturbing the current setup.

4 Business Rules

Rentomate application requires personal data of the stakeholders which is a confidential data to commercialize the application, different requirements by NADRA, police will need to be completed.

5 Accuracy

The server should be accurate in sense that it should correctly identify the roles and allow only authorized persons to access things. Furthermore, the data of the users must be accurate as to avoid fraudulent transactions in app.

CHAPTER IV: PROPOSED SOLUTION

Architectural Design

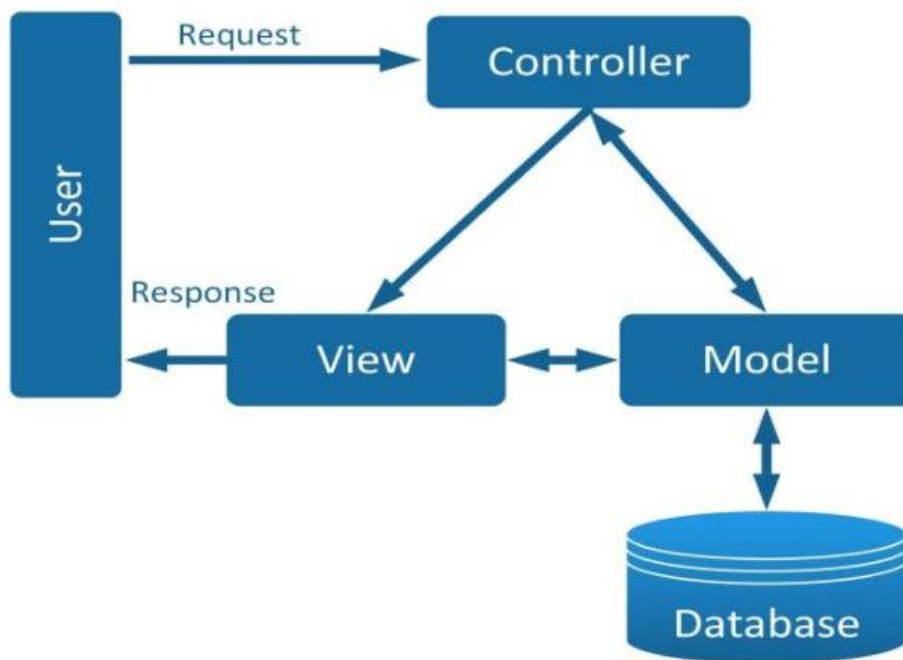


Figure 1 MVC Architectural Model

Model View Controller is an architectural approach that splits an application into three logical components: Model, View, and Controller. Each architectural component addresses a certain aspect of application development. MVC separates the display layer from the business logic layer. It was created with graphical user interfaces in mind for desktop computers (GUIs). In recent years, MVC design has become popular in web technology for developing online applications.

Each logic of the MVC architecture pattern has a specific role and responsibility within the application. For example:

View is the component of an application that represents the data display. The Data obtained from the model data is used to construct views. A view asks the model for information so that the user may see the resulting presentation.

Controller is the element of the software that controls user interaction. The controller interprets the user inputs, causing the model and view to alter accordingly. To update the model's state, a Controller sends orders to it (E.g., Saving a specific document). The controller also sends orders to its associated view, changing the display of the view (For example scrolling a particular document).

Model component is where data and logic are stored. It represents data that is sent between controller components or any other business logic. A Controller object, for example, will retrieve customer information from the database. It manipulates data before sending it back to the database or rendering the same data.

It updates itself in response to requests from the views as well as

commands from the controller. It is also the pattern's lowest level, responsible for data management.

Each logic of the architecture forms an abstraction of the work to be done to meet a specific request.

Why MVC Architecture?

- Simultaneous development – Multiple developers can work on the model, controller, and views at the same time.
- High cohesiveness – MVC allows for the logical grouping of similar controller operations. A given model's views are also grouped together.
- Little coupling — The MVC framework is designed to have low coupling between models, views, and controllers.
- Ease of modification – The separation of duties makes future development or modification easier, resulting in enhanced product scalability.
- Several model views — Models can have multiple perspectives.

UML Diagrams

Different views of the system that will be used for the development of the application, are modelled in Unified Modeling Language (UML) and are discussed in this section. These views include a use case diagram, class diagram, component diagram, activity and deployment diagram to give an overview of how the internal system will work and how the users will interact with the system.

1. Use Case Diagram

Use cases are developed from two user perspectives. Tenants and landlords can login and each would have a different homepage which caters to their requirements. Tenants and landlord can update and change account information when required. Properties can be searched by providing filtration criteria and they can be managed by landlords. Stakeholders can set up the deal as illustrated in the use cases below

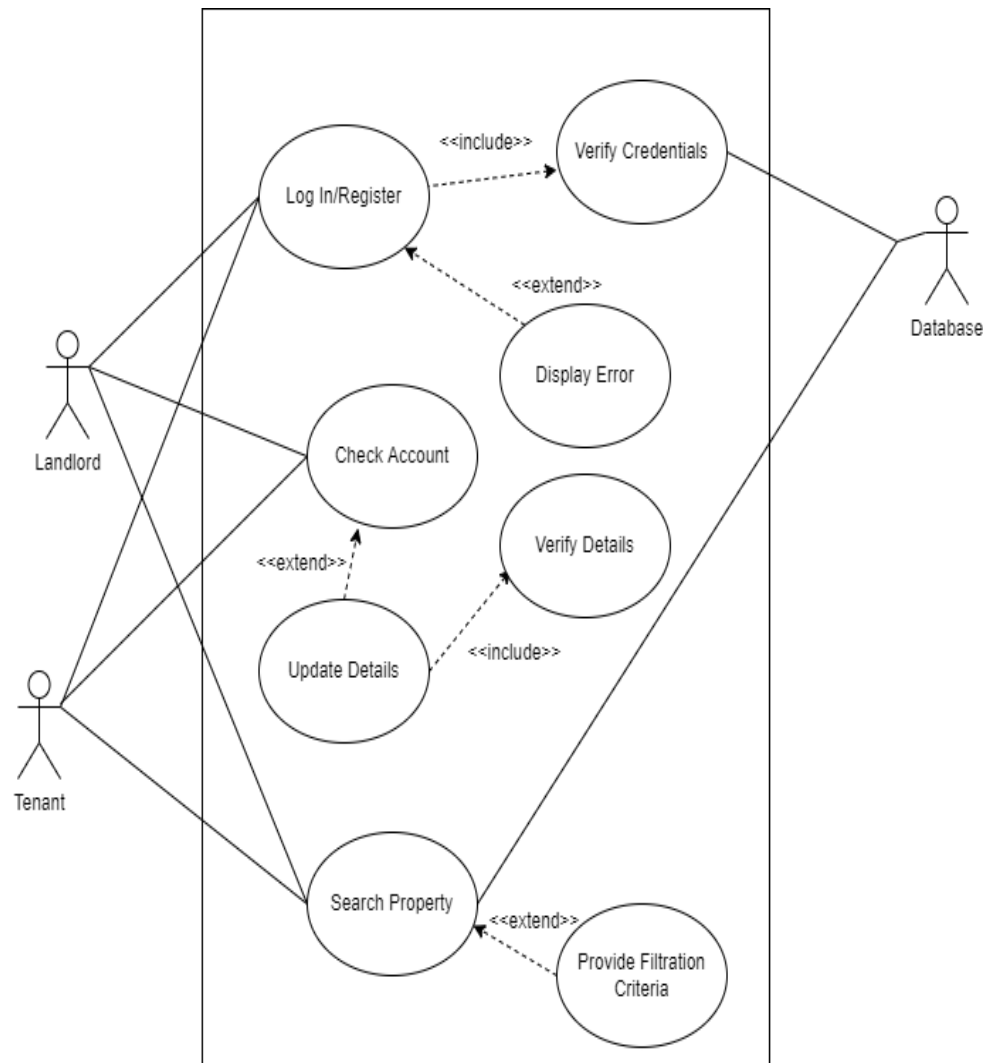


Figure 2 Use case 1

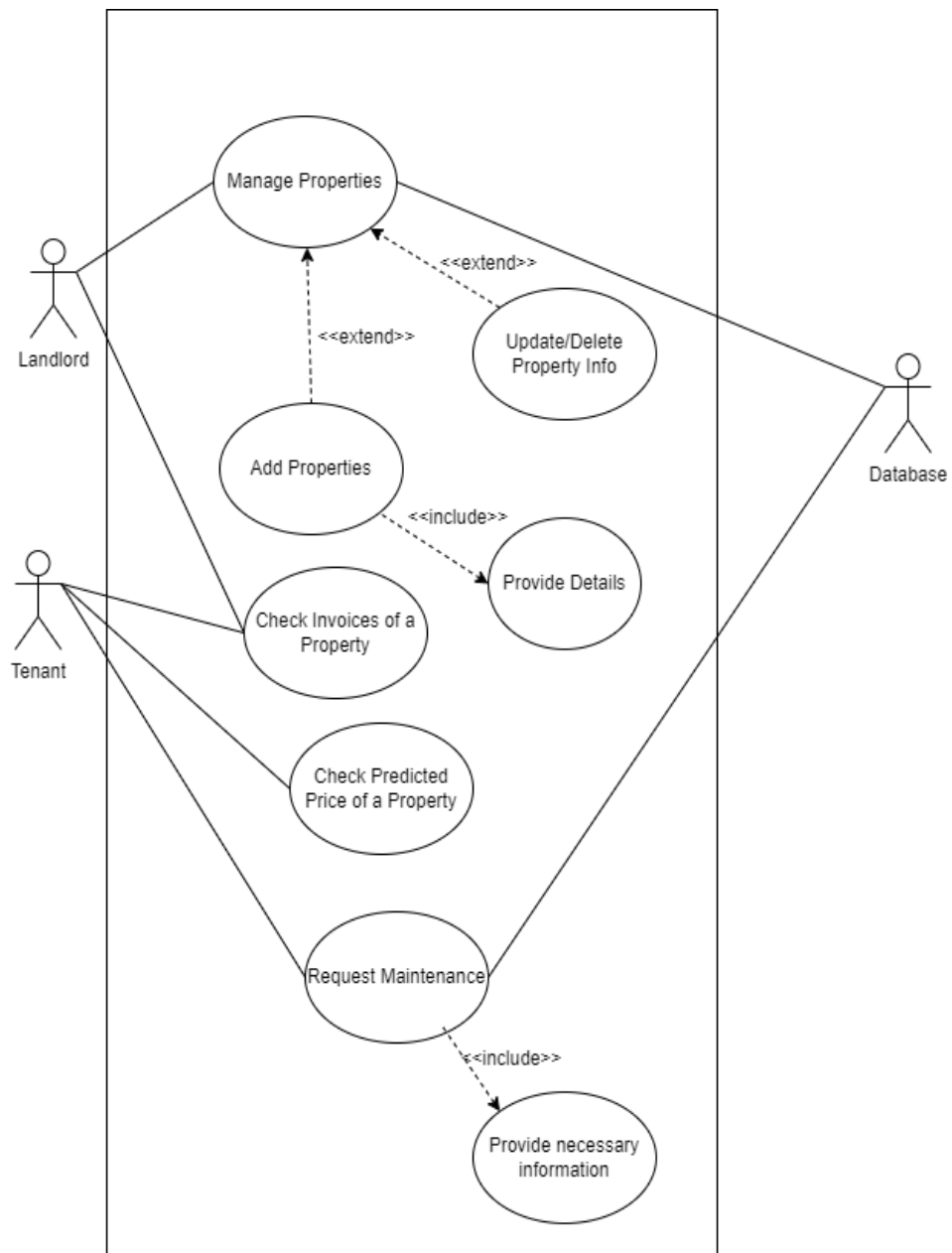


Figure 3 Use Case 2

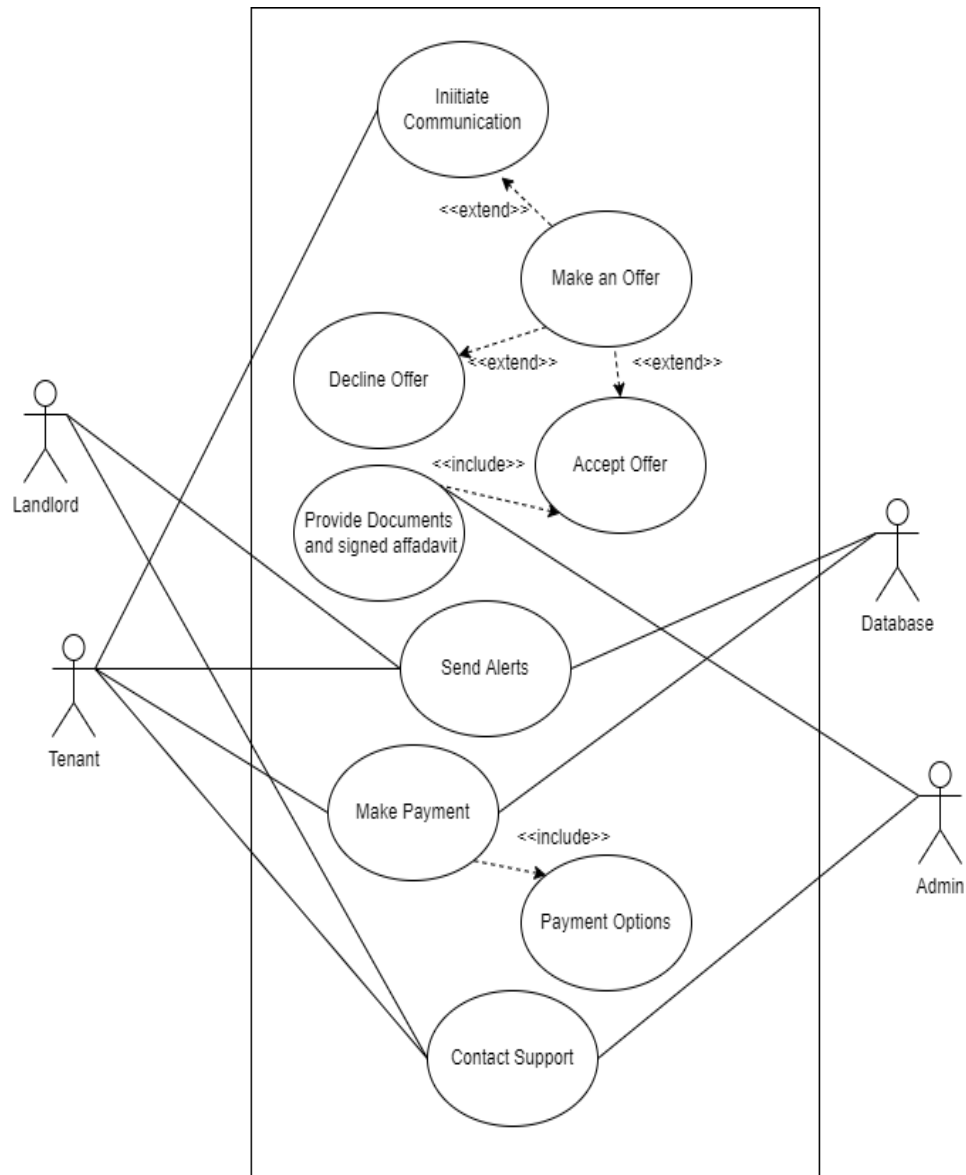


Figure 4 Use Case 3

2. Class Diagram

A class diagram is a type of static structural diagram in the Unified Modeling Language (UML) that represents the structure of a system by presenting the system's classes, attributes, operations (or methods), and connections among objects.



Figure 5 Class Diagram

3. Component Diagram

Deployment diagrams show the hardware components/nodes on which the software components are installed. Software tools are used to simulate complex business processes. Different components may be seen in our system's deployment diagram. UI components are responsible for user interface functionality. Access to data will be granted to the control unit and other components such as the time limit optimizer based on their requirements from this infrastructure.

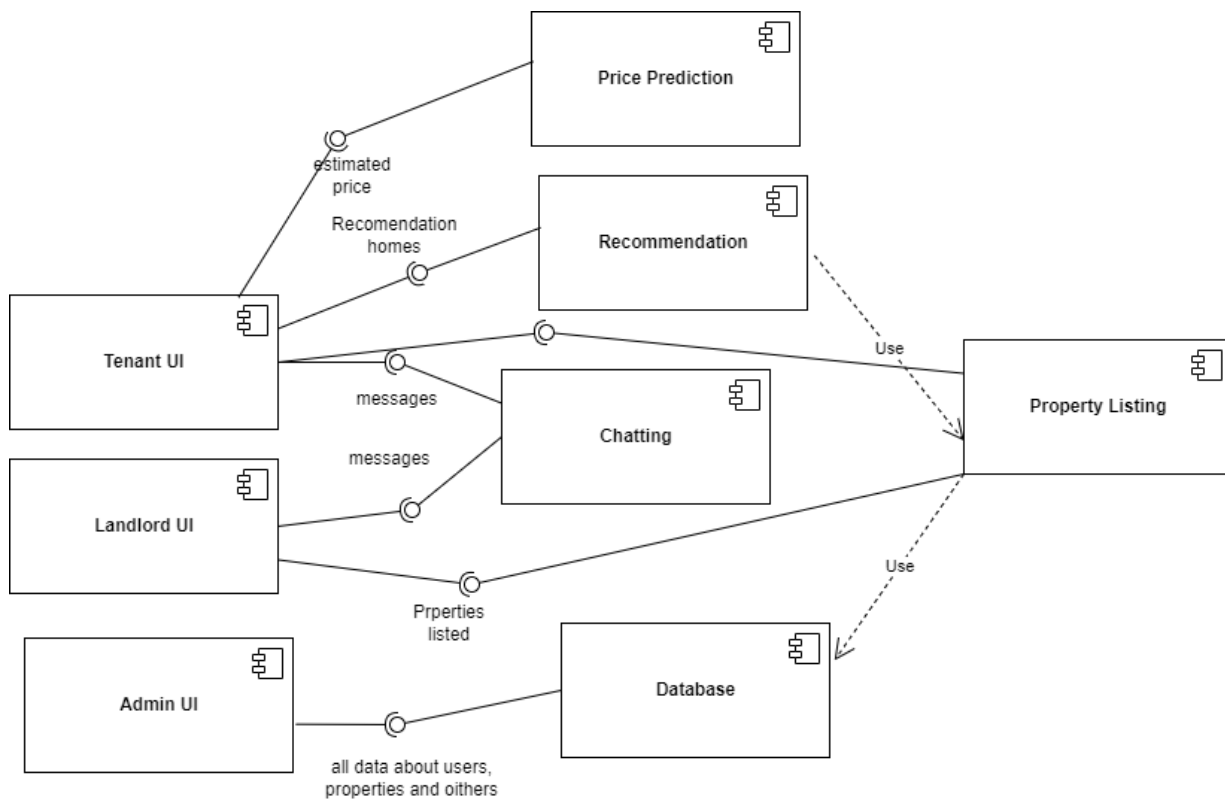


Figure 6 Component Diagram

4. Activity Diagram

An activity diagram is a behavioral diagram, which depicts the behavior of a system. An activity diagram depicts the control flow from a start point to an end point, as well as the multiple decision pathways that exist while the activity is being conducted.

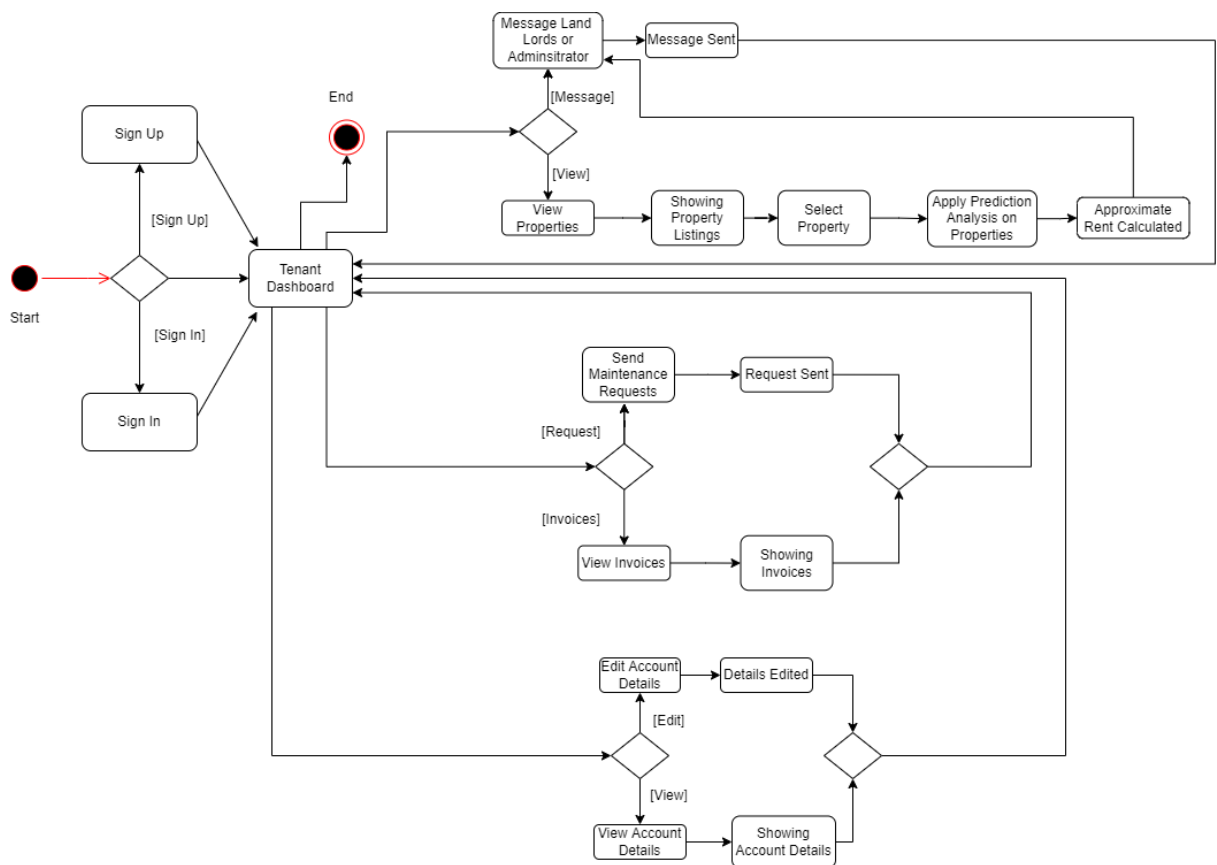


Figure 7 Activity Diagram

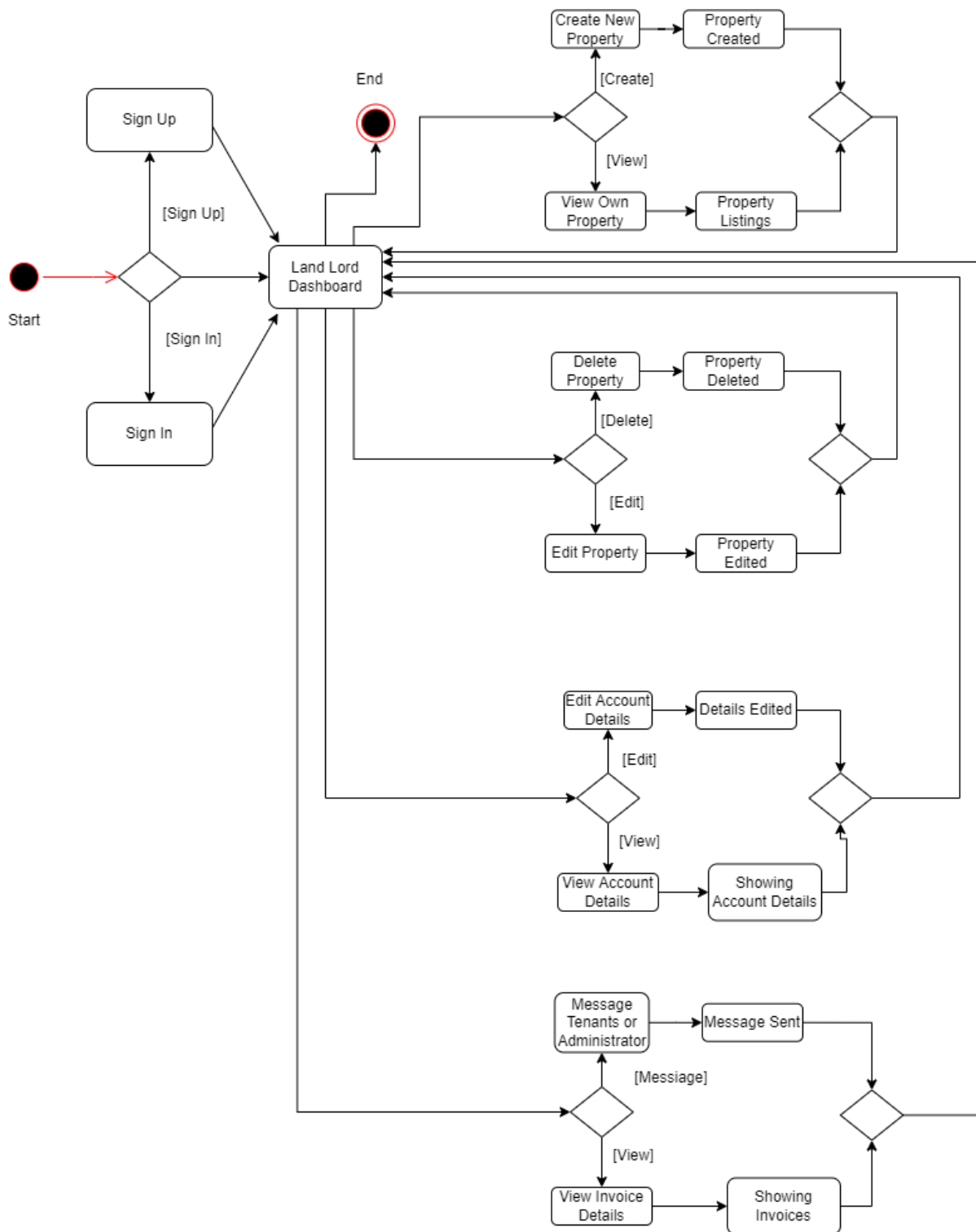


Figure 8 Activity Diagram

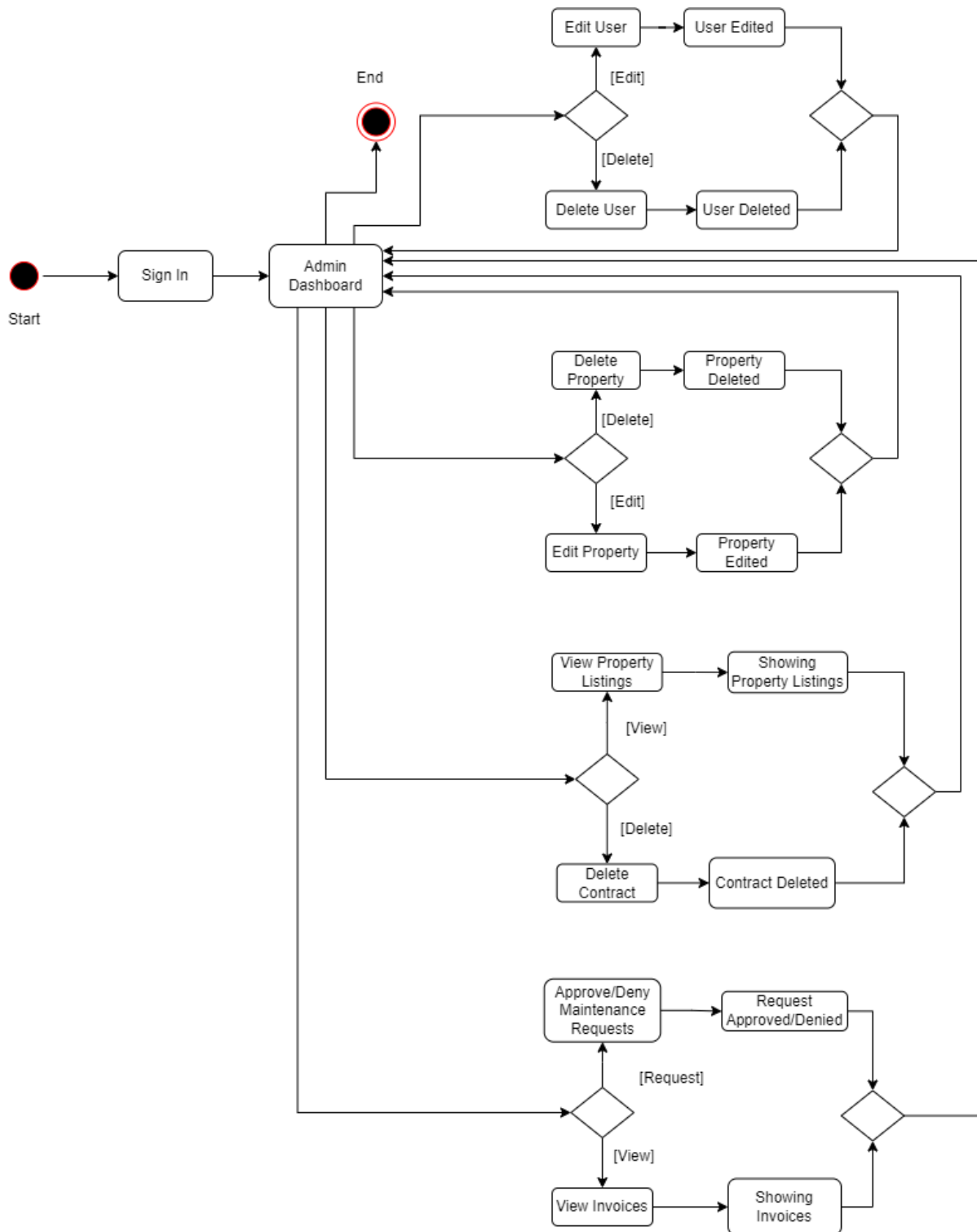


Figure 9 Activity Diagram

5. Deployment Diagram

A deployment diagram is a type of UML diagram that illustrates the execution architecture of a system, with nodes representing hardware or software execution environments and the middleware that links them. Deployment diagrams are often used to describe the physical hardware and software of a system. The end user will interact with the web application to search for properties, and the web app will communicate with the backend via APIs to present the results.

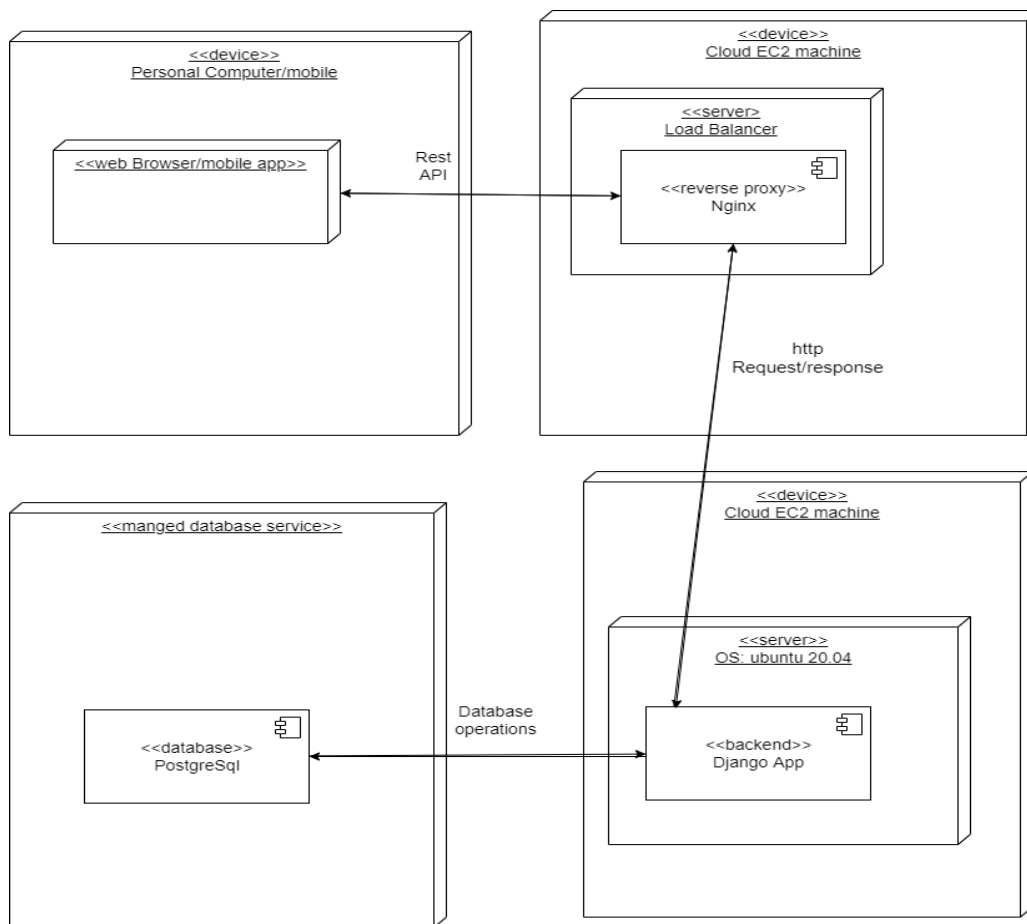


Figure 10 Deployment Diagram

CHAPTER V: RESULTS AND DISCUSSION

House Rent Prediction Algorithm Module is based on Supervised Learning part of Machine Learning. The dataset used for training the model belongs to zameen.com and was taken from kaggle.com. Initially the data was not clean, so the dataset was pre-processed and still model were not giving good accuracy which led to application of feature engineering to the dataset, which resulted in improved accuracy of the model.

Data Preprocessing:

```
[5] df.isnull().sum()
```

```
property_id      0
location_id      0
page_url         0
property_type    0
price            0
price_bin        0
location         0
city             0
province_name    0
locality         0
latitude         0
longitude        0
baths            0
area             0
area_marla       0
area_sqft        0
purpose          0
bedrooms         0
date_added       0
year             0
month            0
day              0
agency           47379
agent            47380
dtype: int64
```

Dataset before Cleaning:

property_id	location_id	page_url	property_type	price	price_bin	location	city	province_name	locality	...	area_marla	area_sqft	purpose	bedro	
0	347795	8	https://www.zameen.com/Property/lahore_model_t...	House	220000000	Very High	Model Town	Lahore	Punjab	Model Town, Lahore, Punjab	...	120.0	32670.12	For Sale	
1	482892	48	https://www.zameen.com/Property/lahore_multan_...	House	40000000	Very High	Multan Road	Lahore	Punjab	Multan Road, Lahore, Punjab	...	20.0	5445.02	For Sale	
2	555962	75	https://www.zameen.com/Property/eden_eden_aven...	House	9500000	Low	Eden	Lahore	Punjab	Eden, Lahore, Punjab	...	9.0	2450.26	For Sale	
3	562843	3821	https://www.zameen.com/Property/gulberg_2_gulb...	House	125000000	Very High	Gulberg	Lahore	Punjab	Gulberg, Lahore, Punjab	...	20.0	5445.02	For Sale	
4	686990	3522	https://www.zameen.com/Property/allama_iqbal_t...	House	21000000	High	Allama Iqbal Town	Lahore	Punjab	Allama Iqbal Town, Lahore, Punjab	...	11.0	2994.76	For Sale	

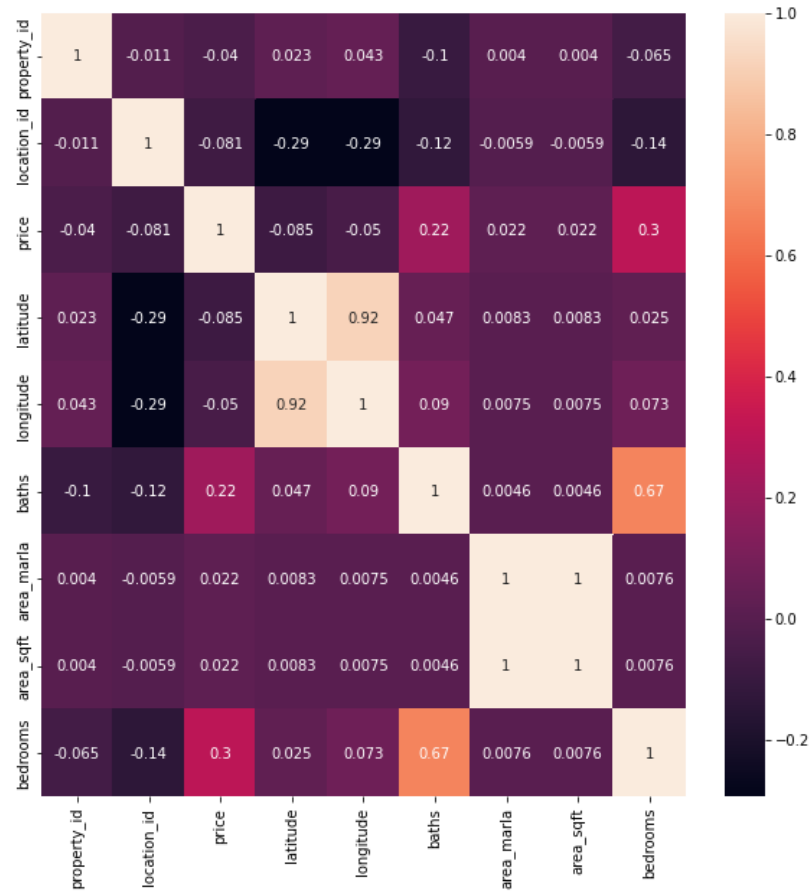
5 rows x 24 columns



Dataset after Cleaning:

	property_id	location_id	price	latitude	longitude	baths	area_marla	area_sqft	bedrooms
0	347795	8	220000000	31.483869	74.325686	0	120.0	32670.12	0
1	482892	48	40000000	31.431593	74.179980	5	20.0	5445.02	5
2	555962	75	9500000	31.499348	74.416959	0	9.0	2450.26	3
3	562843	3821	125000000	31.522069	74.355512	7	20.0	5445.02	8
4	686990	3522	21000000	31.506483	74.286017	5	11.0	2994.76	6

Data Visualization



After the data processing and visualization, 3 models were trained using the above-mentioned dataset by dropping the label column. Models were based on Linear Regression, Extreme Gradient Boosting Regression and Random Forest Regression. After training and testing the models, they were evaluated using metrics like R-Squared Error and Mean Absolute Error. Among the models, Random Forest Regression gave the highest accuracy due to which it was used to build the House Rent Prediction Algorithm. Dataset that was used for this part contained the actual price of property not rent so a ratio between house price and rent is calculated and appropriately applied to the input to generate appropriate output. Results are as follows:

Result Using XGBRegression:

```
# Training and Testing the Model using XGBRegressor (Based on Ensemble Regression Trees)

# Loading the Model
model = XGBRegressor()

# Training the Model
model.fit(X_train, Y_train)

# Evaluating the Model
model_test_prediction = model.predict(X_test)

# R-Squared Error
score_1 = metrics.r2_score(Y_test, model_test_prediction)

print("R-Squared Error (Testing): ", score_1)
```

[12:42:18] WARNING: /workspace/src/objective/regression_obj.cu:152: reg:linear is now deprecated in favor of reg:squarederror.
R-Squared Error (Testing): 0.45502449014488966

Result Using Linear Regression:

```
# Training and Testing the Model using Linear Regression

# Loading the Model
model_1= LinearRegression()

# Training the Model
model_1.fit(X_train, Y_train)

# Testing the Model
model_1_test_prediction = model_1.predict(X_test)

# Evaluating the Model Using R-Squared Error
score_2 = metrics.r2_score(Y_test, model_1_test_prediction)

print("R-Squared Error (Testing): ", score_2)
```

R-Squared Error (Testing): 0.10471121550583595

Result Using Random Forest Regression:

```
# Training and Testing the Model using Random Forest Regressor

# Loading the Model
model_2= RandomForestRegressor()

# Training the Model
model_2.fit(X_train, Y_train)

# Testing the Model
model_2_test_prediction = model_2.predict(X_test)

# Evaluating the Model Using R-Squared Error
score_3 = metrics.r2_score(Y_test, model_2_test_prediction)

print("R-Squared Error (Testing): ", score_3)

R-Squared Error (Testing): 0.5386980109910116
```

Rent Predictor using Random Forest Regression:

```
▶ # Building a Predictive System


input_data = (555962,75,31.499348,74.416959,0,9.0,2450.26,3)

# Converting to a Numpy Array
input_data = np.asarray(input_data)

# Reshaping Input Data
input_data = input_data.reshape(1,-1)

prediction = model_2.predict(input_data)
act = 9500000
diff = int(prediction - act)
pe = (diff/act) * 100
print('Actual Price: ', act)
print('Predicted Price: ', int(prediction))
print('Difference b/w Actual and Predicted : ', diff)
print('Percentage Error : ', pe)

📄 Actual Price: 9500000
Predicted Price: 9853040
Difference b/w Actual and Predicted : 353040
Percentage Error : 3.7162105263157894
```


[Home](#)
[View Properties](#)
[How it Works](#)
[Login](#)
[Sign Up](#)

Easy way to find a perfect property

We provide a complete service for the sale, purchase or rental of real estate

Location


Select your city

Property Type


Choose property type

Rent Range

Choose a price range




Featured Properties



DHA, Phase 6
Lahore, Punjab

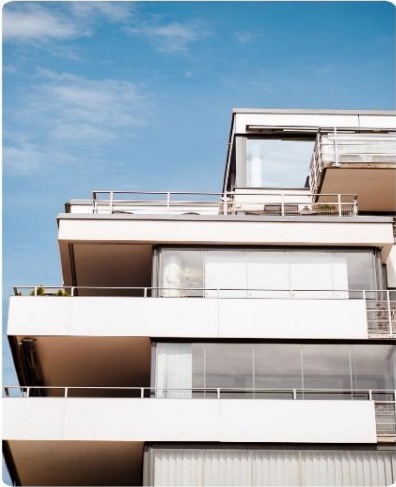
PKR 50000/month




Model Town
Lahore, Punjab

PKR 80000/month


Why you should rent your dream house with us?






Document Management

From lease documents to invoices, keep track of all documents on RentoMate.



Rental Payments

Let's eliminate hectic process of managing rentals. Pay & Collect rents online through RentoMate.



Maintenance Requests

Rather than a call when you're at dinner with family, learn about maintenance issues in RentoMate.

Get your dream house

Family is number one, and comfortable is number two. That two thing must be together. Let's start it.


[Join Us Now](#)


Figure 11 Home Page

Property Type

House Commercial

Apartment Plot

Bedrooms

1 2 3 4+

Bathrooms

1 2 3 4+

Price Range

PKR 12K-15k, 5k

Furnishing

Furnished UnFurnished

Amenities

☒ Water Supply

☐ Shared Accomodation

☒ Supply of Gas

☒ Garage

Reset Apply



PKR 25000/month

Recommended

Block C, Model Town

Lahore, Punjab

3 Beds 4 Baths 07 Marlas

Look no further as we have listed the best Bungalow just for you! This could be your chance to buy such a precious real estate asset. Don't miss out on these 7 Marla units.



Block F, Johar Town

Lahore, Punjab

5 Beds 4 Baths 06 Marlas

PKR 21000/mo



Cavalry Ground, Gulberg

Lahore, Punjab

4 Beds 4 Baths 10 Marlas

PKR 30000/mo



Phase 1, Ghazi Road

Lahore, Punjab

5 Beds 7 Baths 15 Marlas

PKR 55000/mo



Gulshan-e-Ravi

Lahore, Punjab

4 Beds 5 Baths 08 Marlas

PKR 27000/mo

Figure 12 Property Listing Page

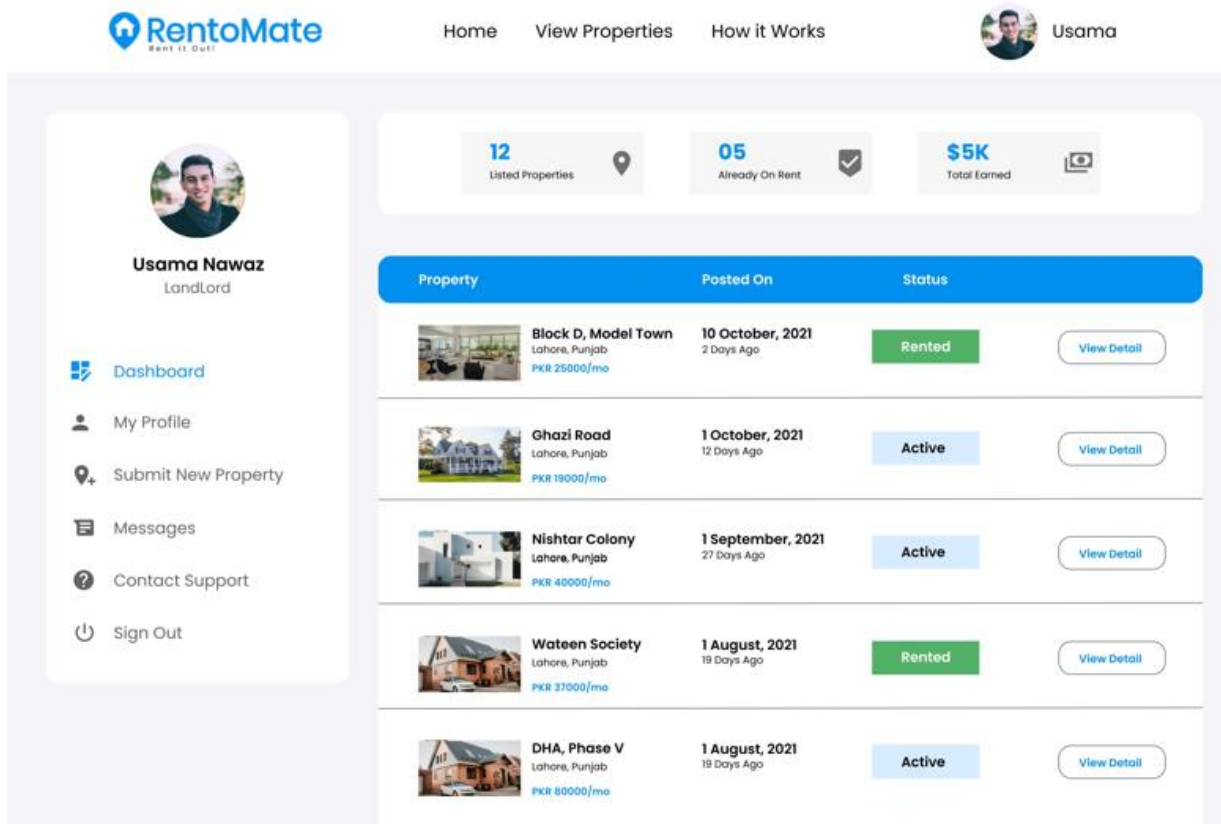


Figure 13 Landlord Dashboard



Login

Landlords grow rich in their sleep without working, risking or economizing

Username

Password

☒ Remember me

[Reset Password?](#)

[Log In to your account](#)

Don't have an account yet? [Join RentoMate Now](#)

Figure 14 Login Page



Register

A few clicks away from renting your dream house

First Name

Last Name

Email

Role

Password

Confirm Password

☐ I agree to all Terms, Privacy Policy & Fees

Create Account

Already, have an Account? [Log in](#)

Figure 15 Registration Page

CHAPTER VI: CONCLUSION AND FUTURE WORK

The main goal of this project was dedicated platform that provides automated and hassle-free medium to either collect and manage rentals and hunt for property in a more efficient and effective manner.

We were successfully able to train the machine learning model that was successful in estimating the price of the property or the price of the rent of a particular property. Its accuracy was reliable. It was tested on various properties having differences in many properties such as number of rooms, size of the properties, locations, and various other properties. By training our AI model, we were able to accurately predict the price of any property which was uploaded on our system based on the market trends. This helped to promote fair comparison between properties and significantly increase transparency of the process.

Our future goal for this project is to extend the features of the system to be able to work outside of the Pakistan if their government allows us to provide the dataset which is essential for the working of our price prediction model. The system was originally intended to be designed as a microservice but due to limitation of the time, it couldn't be done. Implementation of the micro service would have allowed for scalability and easier integration of the new features in the future. After a lot of research and brainstorming, we decided that the model and architecture described above met our requirements. The following technologies were used:

1. Machine Learning Model
2. Database on PostgreSQL, Backend on Django
3. Website on React

Before putting all the components together, each module was coded and tested independently. The data was gathered, and the machine learning model was trained in the first step. The backend of the application was developed in the second phase. Online resources and communication APIs were used to get reliable data. After fine tweaking the method and preparing the data numerous times, reasonable results were obtained. After then, the predictions were saved in a database. The database design was step four, but because the data had to be exported to the database, the database was built concurrently with phase three. After all these steps had been finished and assessed, the website was constructed to serve as the graphical user interface. were separately programmed and evaluated before joining all the components together. The first phase was to gather data and train the machine learning model. The second phase was to develop the application backend. Reliable data was collected from online resources and APIs developed for communication. Desirable results were achieved after fine tuning the algorithm and preprocessing the data several times. The predictions were then exported to the database. The database design was phase four, however, the data was to be exported to the database, due to which the database was created in parallel with phase 3. After all these phases were completed and evaluated, the website was created to act as the graphical user interface, such that it was easy to use. The model performed well after evaluation, yielding 90% accuracy in price prediction

REFERENCES

(<https://www.analyticsvidhya.com/blog/2022/02/linear-regression-with-python-implementation/>)

(<https://medium.com/@theclickreader/random-forest-regression-explained-with-implementation-in-python-3dad88caf165>)

(<https://towardsdatascience.com/gradient-boosting-classification-explained-through-python-60cc980eeb3d?gi=34376551f33d>)

Rentomate

ORIGINALITY REPORT

8%

SIMILARITY INDEX

2%

INTERNET SOURCES

1%

PUBLICATIONS

7%

STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to King's Own Institute Student Paper	1%
2	Submitted to University of Aberdeen Student Paper	1%
3	simonwillison.net Internet Source	1%
4	Submitted to Coventry University Student Paper	1%
5	Submitted to De Montfort University Student Paper	1%
6	Submitted to The Robert Gordon University Student Paper	1%
7	Submitted to University of Westminster Student Paper	<1%
8	www.coursehero.com Internet Source	<1%
9	Submitted to The Post Oak School Student Paper	<1%

10

Submitted to British Institute of Technology and
E-commerce

Student Paper

<1%

11

www.independentevents.com.au

Internet Source

<1%

