HOUSE RENT PREDICTION

A Project Report

Submitted By

P. Chaitanya

2203031240984

G. Gokul Sai Ram

220303124021

R. H. S. V. Prudhvi

2203031241123

D. Krishna Chowdary

220303124053

in Partial Fulfilment for the Award of

the Degree of

BACHELOR OF TECHNOLOGY

COMPUTER SCIENCE &

ENGINEERING

Under the Guidance of

Lecturer. Binisa Makwana

VADODARA

November - 2025

PARUL UNIVERSITY

CERTIFICATE

This is to Certify that Project - II (203105400) of 7th Semester entitled "**HOUSE RENT PREDICTION**" of Group No. **AI- 1 6 4** has been successfully completed by

- P. CHAITANYA – 2203031240984
- G. GOKUL SAI RAM – 220303124021
- R. H. S. V. PRUDHVI – 2203031241123
- D. KRISHNA CHOWDARY - 220303124053

under my guidance in partial fulfillment of the Bachelor of Technology (B. Tech) in Computer Science & Engineering of Parul University in Academic Year 2024- 2025.

Date of Submission:_____

Lecturer. Binisa Makwana

Project Guide

Dr. Sanjay Agal,

Head of Department,

AIDS, PIET,
Parul University.

Project Coordinator: -

Dr. Mohammad Arif

ACKNOWLEDGEMENT

"Success is the sum of small efforts, repeated day in and day out."

-Robert Collier

We take this opportunity to express our sincere gratitude to all those who have supported and guided us throughout the successful completion of our project, **"House Rent Prediction."**

First and foremost, we extend our heartfelt thanks to our respected project guide, **Lecturer. Binisa Makwana**, Assistant Lectureressor, Department of Artificial Intelligence and Data Science, Parul University, for her continuous encouragement, valuable insights, and patient guidance. Her expert advice and constant motivation were instrumental in shaping our project and helping us overcome every challenge we encountered.

We would also like to express our sincere appreciation to **Dr. Sanjay Agal**, Head of Department (AI & DS), for providing us with the necessary facilities, valuable suggestions, and a supportive learning environment that greatly contributed to our academic growth. A special note of thanks to our Project Coordinator, **Dr. Mohammad Arif**, for her consistent coordination, timely reviews, and encouragement throughout the development process.

We extend our gratitude to all the faculty members of the **Department of Artificial Intelligence and Data Science** for their guidance, encouragement, and constructive feedback that helped us refine our ideas and improve the quality of our work. We would also like to acknowledge the contribution of our classmates, friends, and families for their continuous encouragement, moral support, and understanding during the entire course of this project.

Lastly, we thank each other as team members — **P. Chaitanya, G. Gokul Sai Ram, R. H. S. V. Prudhvi, and D. Krishna Chowdary** — for our cooperation, determination, and dedication that made this project possible.

P. Chaitanya-2203031240984

G. Gokul Sai Ram-220303124021

R. H. S. V. PRUDHVI – 2203031241123

D. Krishna Chowdary- 220313124053

CSE, PIET

Parul University,

Vadodara

ABSTRACT

The **“House Rent Prediction”** project focuses on developing a data-driven application capable of accurately estimating house rental prices based on multiple influencing factors such as location, area, number of bedrooms, furnishing type, and other amenities. With the rapid urbanization and dynamic housing markets in metropolitan and semi-urban areas, determining a fair rental price has become a complex task for both tenants and property owners. This project aims to bridge that gap by leveraging machine learning techniques to deliver precise, reliable, and transparent rent predictions.

The proposed system utilizes a machine learning model trained on diverse datasets containing real estate parameters and rental information collected from various online sources. By analyzing features such as geographic location, proximity to key facilities, property size, and furnishing status, the model predicts the expected rent value for a given property. The project employs algorithms such as Linear Regression, Random Forest, and XGBoost to identify patterns and relationships between these variables, ultimately selecting the most efficient model through performance evaluation metrics like Mean Absolute Error (MAE) and Root Mean Square Error (RMSE).

A key strength of this project lies in its **user-centric interface**, which allows users to input property details and instantly obtain a rent estimate. This enables property owners to price their listings competitively and assists tenants in making informed decisions while searching for rental homes. The system can also provide analytical insights into rent trends within a city or region, offering valuable information for investors, property managers, and policymakers.

Beyond its predictive capabilities, the project emphasizes **data visualization and interpretability**, ensuring that users can clearly understand how each factor influences rental pricing. The integration of machine learning models with an interactive web interface makes the system scalable, adaptable, and easily deployable for real-world applications.

In essence, **House Rent Prediction** demonstrates how the combination of data analytics and intelligent modeling can simplify decision-making in the housing sector. By providing accurate, real-time predictions and actionable insights, the system contributes to a more transparent, efficient, and fair rental ecosystem for all stakeholders.

Key Points: Machine Learning, Regression, Rent Prediction, Real Estate, Data Analytics.

Table of Contents

List of Tables

List of Figures

Chapter 1

INTRODUCTION

1.1 Problem Overview and Motivation

The increasing urbanization and migration of people to cities for employment, education, and better living conditions have significantly influenced the housing sector. As the number of tenants continues to grow, the demand for affordable rental properties has risen sharply. However, determining the appropriate rental value for a property remains a major challenge, as it depends on various factors such as location, property size, furnishing type, and surrounding amenities. The lack of a systematic approach often leads to inconsistent pricing across different regions.

To solve these problems, The **House Rent Prediction** project uses a data-driven machine learning approach to estimate fair rental prices based on key housing features such as city, area, bedrooms, and furnishing type. By analyzing historical data, it ensures greater accuracy and transparency in rent estimation, helping both tenants and owners make informed decisions. The project aims to leverage artificial intelligence to promote fair pricing and improve efficiency in the rental housing market.

1.2 Purpose and Scope of the Project

The main objective of the **House Rent Prediction** project is to build a machine learning model that can estimate rental prices accurately using key features such as city, area, number of rooms, and furnishing type. By analyzing historical housing data, the system minimizes human error and ensures transparent, data-driven rent prediction, helping both tenants and property owners make informed decisions efficiently. s

The project's scope extends beyond simple prediction to providing real-world value for tenants, landlords, and real estate agents. It enables fair pricing, better market analysis, and improved decision-making. With future enhancements like live data integration and wider geographical based

coverage, the system can evolve into a complete rental insight and forecasting platform.

1.3 Significance of Rent Prediction Models

Accurate rent prediction plays a vital role in promoting fairness and transparency in the real estate market. It helps tenants avoid overpaying and enables landlords to set competitive prices based on real data rather than assumptions. By leveraging machine learning, the system provides reliable estimates that reflect true market conditions, enhancing trust and efficiency in rental transactions.

Beyond individual benefits, rent prediction models contribute to broader market insights and policy planning. Real estate agents, investors, and analysts can use these predictions to understand regional trends and forecast price changes. This data-driven approach supports better resource allocation, smarter investment decisions, and a more balanced housing ecosystem overall.

Moreover, rent prediction models serve as valuable tools for real estate Lectureressionals and policymakers. They provide analytical insights into market trends, regional affordability, and property demand patterns. Such data-driven approaches can help shape housing policies, investment strategies, and urban development planning in a more informed and efficient way.

1.4 User-Centered Design Model

The **House Rent Prediction** system is developed using the Python programming language due to its simplicity and extensive library support for data science and machine learning. The project utilizes key Python libraries such as *pandas* and *NumPy* for data preprocessing, *Matplotlib* and *Seaborn* for visualization, and *scikit-learn* for implementing regression models. These tools enable efficient data handling, analysis, and accurate prediction of rental values.

The dataset, sourced from **Kaggle**, contains details like city, area, BHK, bathrooms, furnishing type, and rent. The data undergoes cleaning, encoding, and normalization before model training to ensure reliability and consistency Algorithms such as **Linear Regression**, **Random Forest Regressor**, and **XGBoost** are trained and evaluated using metrics like *MAE*, *RMSE*, and R^2 *score* to determine the most effective model for rent prediction.

Once the model is finalized, it is deployed through a **Streamlit-based web interface** that allows users to input property details and instantly receive a rent estimate. This integration bridges machine learning with user interaction, creating a simple yet powerful application. The modular architecture ensures scalability, maintainability, and adaptability for future enhancements or integration with live real estate data sources.

1.5 User-Centered Design and Use Case

The **House Rent Prediction** system is designed with a user-centered approach, emphasizing simplicity, accessibility, and ease of use. The goal is to ensure that users—whether tenants, landlords, or real estate Lectureressionals—can interact with the model effortlessly. The **Streamlit interface** provides a clean and intuitive layout where users can input details such as city, area, bedrooms, bathrooms, and furnishing type to get an instant rent prediction.

The interface focuses on responsiveness and clarity, minimizing technical complexity for end users. Once the data is entered, the trained machine learning model processes the input and displays the estimated rent within seconds. The prediction output is accompanied by visual cues or short explanations, helping users understand how key factors like location or furnishing status influence the result.

From a practical standpoint, the system benefits multiple user groups. Tenants can use it to compare rental options and plan budgets effectively, while landlords can determine competitive listing prices. For real estate agents and analysts, it serves as a quick analytical tool to assess property trends. The user-friendly design ensures that accurate, data-driven insights are accessible to everyone, regardless of technical expertise.

In conclusion, The **House Rent Prediction** project introduces a data-driven solution to one of the most common challenges in the real estate market—estimating fair and accurate rental prices. By integrating machine learning techniques with real-world housing data, the system brings transparency, consistency, and efficiency to the rental process. The chapter outlined the motivation behind the project, its objectives, scope, significance, and technical foundation, all of which form the basis for the development of a reliable predictive model.

Chapter 2

LITERATURE SURVEY

This chapter discusses prior research and developments in house rent and property price prediction. It explores the various machine-learning models, preprocessing techniques, and system frameworks that have been implemented in earlier studies. The objective is to understand existing methods, identify research gaps, and provide the foundation upon which this project has been designed.

2.1 Overview

The prediction of property rental prices has become a key research area due to the increasing urbanization and dynamic nature of housing markets. Traditional estimation techniques relied heavily on human judgment, real-estate agents, or basic statistical averages. However, these approaches often resulted in biased or inconsistent pricing due to limited consideration of multiple variables such as locality, property size, furnishing type, and amenities.

With the advancement of **Artificial Intelligence (AI)** and **Machine Learning (ML)**, researchers have developed predictive models capable of analyzing large datasets to determine fair rental prices. These models enable data-driven decision-making, improving transparency for both tenants and property owners.

2.2 Machine Learning Models for Price Prediction

Various researchers have proposed and compared different machine-learning algorithms to predict house rent and property prices.

- **Geerts (2023)** conducted a comparative analysis of linear and ensemble learning techniques, concluding that tree-based models such as Random Forest and Gradient Boosting yield higher predictive accuracy.
- **Sharma (2024)** introduced an optimized XGBoost algorithm that effectively handled large datasets and minimized overfitting through regularization and learning-rate adjus

- **Patel et al. (2022)** demonstrated that Random Forest and Neural Network models achieve robust predictions for structured rental data.
- **Bansal and Singh (2020)** emphasized that incorporating spatial features, such as location and distance from city centers, significantly improves rent prediction accuracy.
- **Kumar and Das (2023)** experimented with hybrid models that combine LightGBM and Random Forest, achieving an R^2 score of 0.93 with minimal training time.

These studies collectively highlight that ensemble learning algorithms, such as **XGBoost** and **Random Forest**, outperform traditional linear methods because they better capture nonlinear relationships and feature interactions.

2.3 Feature Engineering and Data Preprocessing

Effective data preprocessing is vital for achieving accurate rent predictions.

- **Patel et al. (2022)** stressed the importance of cleaning the dataset, handling missing values, and encoding categorical attributes for reliable model performance.
- **Bansal and Singh (2020)** demonstrated that normalizing features like area and number of bedrooms prevents data bias during training.
- **Nagaraju (2022)** developed a feature-selection framework that combined correlation analysis with recursive feature elimination, reducing redundant data and improving model efficiency.

Feature engineering also involves creating derived attributes, such as rent per square foot or amenity index, which help models capture complex patterns and improve prediction accuracy.

2.4 Comparative Analysis of Existing Systems

Existing platforms such as **NoBroker**, **99acres**, and **MagicBricks** provide property listings but lack intelligent rent-prediction mechanisms. These systems rely on manual entries from property owners and brokers, which often contain inconsistencies.

Recent academic systems have introduced automated valuation methods. For instance, **Raj and Thomas (2021)** developed a web-based rent estimation model that employed machine-learning regression techniques. However, their approach used limited datasets and lacked location-based demand prediction.

The proposed system in this project extends these ideas by incorporating advanced algorithms, a cleaner data pipeline, and a user-friendly Streamlit web interface for real-time prediction and visualization.

2.5 Research Gap

From the literature reviewed, the following gaps were identified:

1. Most studies utilized static or region-specific datasets, limiting the model's generalization capability.
2. Few systems provided an explainable or interactive deployment framework.
3. The integration of modern web technologies (such as Streamlit) with predictive models was largely unexplored.

The proposed system aims to overcome these challenges by combining **ensemble-based ML algorithms** with an **interactive web interface**, enabling accurate and transparent rent predictions across multiple cities.

2.6 Visual Comparison of Previous Works

To visually represent prior research findings, the following figure compares the accuracy of key machine-learning models commonly used in rent prediction.

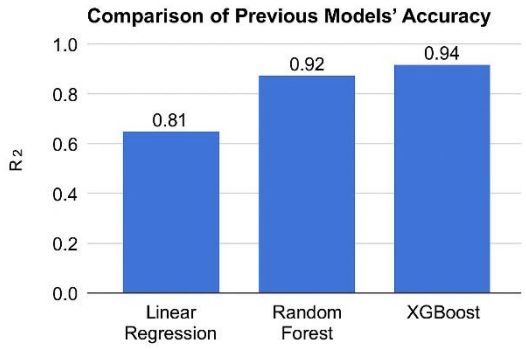

Fig 2.1: Comparison of Previous Models' Accuracy

2.7 Summary of Reviewed Studies

Author	Year	Method	Dataset	Outcome
Geerts et al.	2023	Random Forest, Gradient Boosting	European Housing Dataset	Ensemble models outperform linear method
Sharma & Mehta	2024	XGBoost Regression	Indian Rental Dataset	Improved accuracy using feature engineering.
Patel et al.	2022	Random Forest, Neural Network	Indian Cities	RF efficient for structured tabular data.
Bansal & Singh	2020	Spatial Analysis + ML	Urban Property Dataset	Location encoding improves prediction.
Kumar & Das	2023	LightGBM, Hybrid Models	Multi-City Data	Achieved R^2 = 0.93 using optimized features.
Raj & Thomas	2021	AI-Powered Prediction System	Rental Market Analysis	Introduced web-deployed ML prediction model.

Table 2.1: Summary of major research works in rent and property price prediction.

2.8 Summary

This chapter reviewed the existing literature and discussed the evolution of machine-learning methods for rental price prediction. It was observed that ensemble-based approaches such as Random Forest and XGBoost consistently outperform traditional linear models. The insights from these studies formed the basis for the model selection, preprocessing strategy, and deployment framework adopted in this project.

The next chapter, **System Analysis**, explains the overall system architecture, feasibility study, and the functional and non-functional requirements of the proposed *House Rent Prediction* system.

Chapter 3

ANALYSIS / SOFTWARE REQUIREMENTS SPECIFICATION (SRS)

3.1 Introduction

This chapter describes the detailed analysis of the proposed *House Rent Prediction System*, outlining its purpose, functionality, design considerations, and external interface requirements. It serves as a comprehensive technical and functional specification that defines the structure, components, and behavior of the system to ensure efficient implementation and evaluation.

3.1.1 Purpose

The purpose of this document is to define the system requirements and functional specifications of a web-based application designed to predict rental prices for residential properties using machine-learning techniques.

The system aims to provide a transparent, data-driven method for estimating rent based on parameters such as city, area (in square feet), number of bedrooms (BHK), furnishing type, and available amenities.

This document serves as a guideline for developers, analysts, and stakeholders involved in building, testing, and deploying the project.

3.1.2 Terms

This project document follows the **IEEE Software Requirements Specification (SRS)** standard. All requirements, features, and technical elements are articulated in clear and measurable terms.

3.1.3 Intended Audience and Readers

The primary readers of this report include:

- **Project Developers:** Responsible for system design, model training, and web deployment.
- **Project Guides / Reviewers:** Evaluate design, implementation, and results.
- **End Users:** Property owners, tenants, and real-estate analysts who will use the application for rent predictions.
- **Quality Assurance (QA) Teams:** Test and validate system accuracy and reliability.

This document also serves as a reference for future researchers or students working on AI-driven real-estate prediction systems.

3.1.4 Product Scope

The proposed system focuses on delivering a complete end-to-end rent prediction platform through a user-friendly web interface. The key components include:

- **Data Collection Module:** Gathers housing data (city, area, BHK, furnishing, rent).
- **Preprocessing Module:** Cleans, transforms, and encodes data for model training.
- **Prediction Engine:** Utilizes machine-learning algorithms such as Linear Regression, Random Forest, and XGBoost.
- **Visualization Module:** Displays results and model comparisons.
- **Web Interface:** Allows users to input property details and view predicted rent values.
- **Authentication Module:** Handles secure user logins and Lecturerile management.
- **Deployment Environment:** Streamlit web framework integrated with the trained model for real-time prediction.

3.1.5 References

- Kaggle Dataset: *House Rent Prediction Dataset* (Banerjee, 2022).
- Techniques for configuring cloud servers.
- Best practices for database architecture design.
- Strategies for securing user authentication in web applications.
- Methods for optimizing front-end performance.

3.2 General

3.2.1 Products

The *House Rent Prediction System* is a standalone web-based product designed to assist users in estimating rent prices accurately. It integrates data analytics, machine learning, and visualization into a single user interface.

Future expansions may include integration with GIS (Geographical Information Systems) for map-based predictions, as well as mobile app versions for broader accessibility.

3.2.2 Product Features

- Rent prediction based on multiple input parameters (city, area, BHK, furnishing type, amenities).
- Data preprocessing for handling missing and inconsistent entries.

- Machine-learning model training and performance comparison.
- Web interface for inputting details and displaying predicted rent.
- Graphical visualization of rent trends and accuracy metrics.
- User authentication for personalized data access.
- Admin access for dataset updates and retraining models.
- Real-time prediction results displayed via Streamlit.

3.2.3 User groups and characteristics

- **General User (Tenant):** Uses the web interface to enter property details and receive rent predictions.
- **Property Owner / Broker:** Compares predicted values with market trends for listing decisions.
- **Administrator:** Manages data uploads, monitors model performance, and retrains algorithms.
- **Researcher / Analyst:** Studies model accuracy, dataset trends, and explores new predictive features.

3.2.4 Operating environment

- **Server:** Cloud-hosted (AWS / Render / Localhost).
- **Programming Language:** Python 3.10+.
- **Frontend Framework:** Streamlit for web-based UI.
- **Libraries:** scikit-learn, pandas, numpy, matplotlib, XGBoost.
- **Database:** CSV/SQLite.
- **Operating System:** Windows 11.
- **Browser Support:** Google Chrome, Edge, Firefox.

3.2.5 Design and Functionality

The system is designed for high performance and scalability. The machine-learning pipeline handles large datasets efficiently through data preprocessing, feature encoding, and batch prediction. Caching mechanisms and optimized model serialization (using joblib) ensure that predictions load quickly.

The web interface allows users to input parameters dynamically and visualize results in real time, ensuring transparency and reliability.

3.2.6 User Information

User support and instructions include:

- How to register or log in to the platform.
- How to input property details for prediction.
- Understanding the output (predicted rent, comparison chart).
- Interpreting accuracy metrics such as R^2 and MAE.
- Reporting issues and contacting admin support.

A digital help guide and demo videos can be added to assist new users in navigating the web interface.

3.2.7 Assumptions and Dependencies

- Datasets are assumed to be accurate and regularly updated.
- Users must have a stable internet connection.
- The model assumes input parameters (e.g., area, furnishing type) are within realistic ranges.
- Future accuracy depends on dataset growth and retraining frequency.
- Application depends on Streamlit framework and Python environment compatibility.

3.3 External Interface Requirements

3.3.1 User Interface

The *RentRight* interface (developed in Streamlit) provides a clean and interactive layout.

- Input boxes for City, Area (sqft), BHK, Bathroom, and Furnishing Type.
- Predict button triggers ML model execution.
- Result section displays predicted rent in INR.
- Graphical representation of rent vs. actual data using bar and scatter charts.
- Navigation menu for Dataset Overview, Model Insights, and About pages.

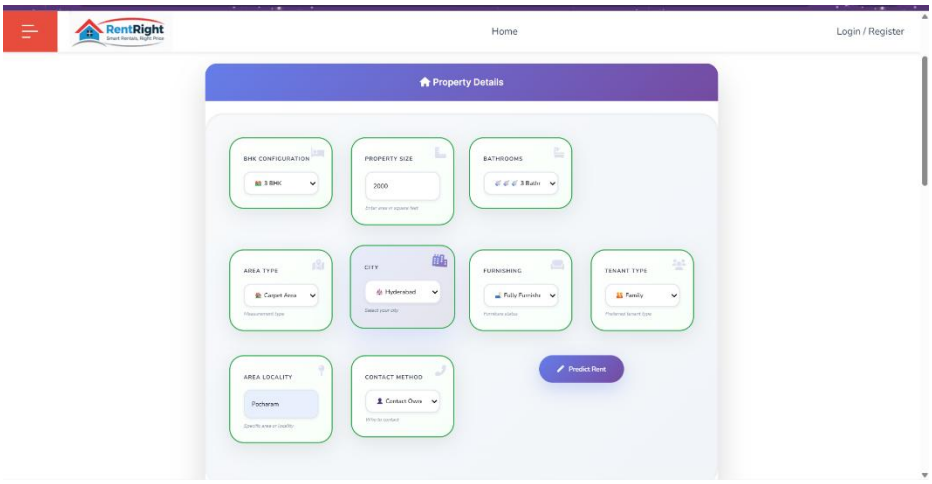

Figure 3.4: Streamlit User Interface

3.3.2 Hardware Interface

- Cloud-based virtual servers for data storage and model hosting.
- Client devices: laptops, desktops, or smartphones with browsers.
- System utilizes CPU/GPU resources for model inference.

3.3.3 Software Interface

- Python 3.10+, scikit-learn, pandas, matplotlib, XGBoost.
- Streamlit framework for frontend interaction.
- SQLite / CSV for backend data storage.
- API integration for future location-based prediction modules.

3.3.4 Communication

- HTTP/HTTPS requests between client and Streamlit server.
- Internal communication between ML model and UI through Streamlit callbacks.
- Data input and output managed via local or cloud-hosted scripts.
- Optional email module for feedback or updates.

3.4 Functions and requirements

3.4.1 Functional requirements

Administrator

- Manage dataset uploads and ensure data quality.
- Retrain models periodically for better accuracy.
- Monitor model performance metrics (R^2, MAE, RMSE).
- Approve and verify property listings submitted by owners.
- Manage user accounts (add, suspend, or remove users).
- Maintain application logs and ensure data security.
- Oversee overall system performance and resolve issues.

Owner

- Register and log in to the RentRight platform.
- Add, update, or delete property listings.
- Input details such as city, area, furnishing type, and BHK configuration.
- Use the rent prediction feature to estimate optimal rent values.
- View system-generated rent comparisons and trends.
- Contact potential tenants through the communication module.
- Access personal dashboard to monitor listed properties and prediction history.

Customers

- Register and log in to the RentRight application.
- Input property details to get predicted rent instantly.
- Compare rent estimates with current market listings.
- Search for properties based on city, price range, or amenities.
- Save or bookmark listings for future reference.
- Contact property owners for inquiries.

3.4.2 Non-Functional Requirements

- Usability: The interface should be simple, visually clear, and easy to navigate.
- Scalability: System can handle larger datasets by retraining models.
- Security: Secure user login and validation for dataset updates.
- Reliability: Model prediction should remain consistent within ±5% deviation range.
- Maintainability: The codebase should support updates and model improvements.

CHAPTER 4

SYSTEM DESIGN

This chapter presents the design of the House Rent Prediction System (RentRight), which focuses on predicting house rental prices using machine learning. It outlines the overall architecture, system workflow, and database structure, ensuring that each component works together to meet the project's goals.

System design translates user and functional requirements into a structured technical blueprint, defining the data flow, architecture, and interactions between different modules.

4.1 System Architecture

The architecture of the House Rent Prediction System defines how various components—such as the user interface, backend processing modules, database, and prediction model—interact with each other.

The design ensures seamless communication between user input and the machine-learning model that predicts rent values in real time.

4.1.1 Component Description

- **User Interface:** The front-end interface allows users to input property details (city, area, BHK, furnishing type, etc.) and view predicted rent results.
- **Data Preprocessing Module:** Cleans raw data, handles missing values, encodes categorical features (e.g., furnishing type), and scales numeric data before feeding it to the ML model.
- **Machine Learning Model:** The core engine that uses trained regression algorithms like **Linear Regression**, **Random Forest**, and **XGBoost** to predict rental prices based on the input attributes.

- **Database Storage:** Stores all relevant data, including user accounts, property information, and historical records of rent predictions.
- **Result Visualization:** Displays predicted rent, performance charts, and comparison graphs for user understanding and decision-making.

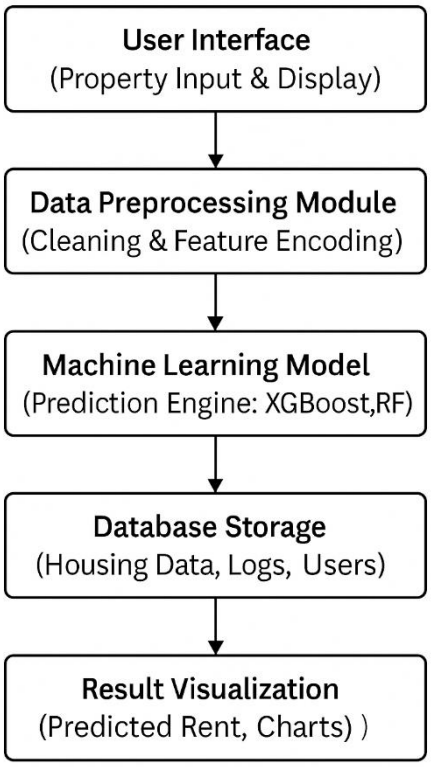

Figure 4.1: UML Diagram of the House Rent Prediction System

4.1.2 Data Stream (System Workflow)

- Users enter property details through the web interface.
- Data is sent to the backend preprocessing module.
- Preprocessing prepares the data for the trained ML model.
- The ML model predicts the rental price.
- The predicted result is stored in the database and displayed on the user interface.
- The system updates analytics and logs for model monitoring.

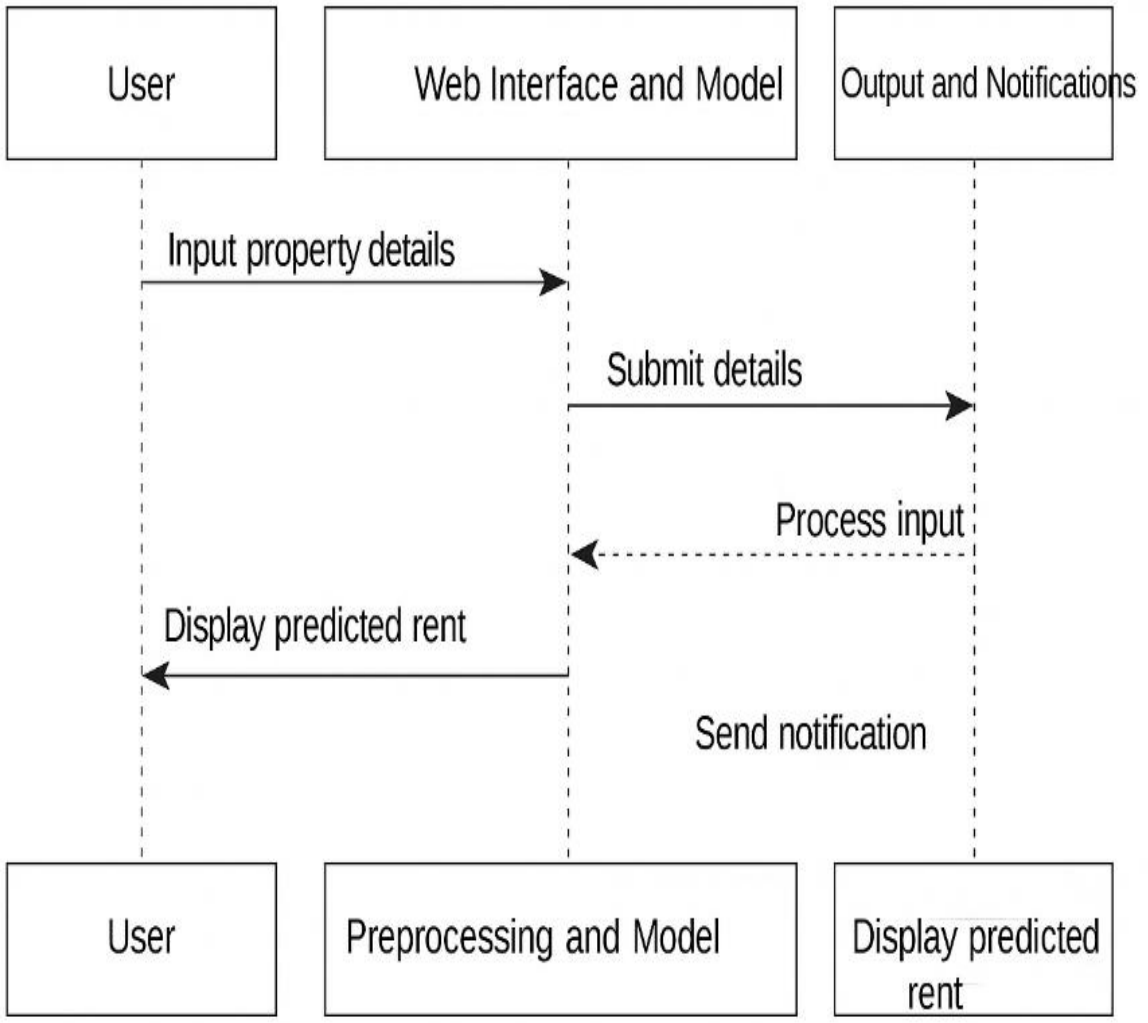

Figure 4.2: Sequence Diagram of Rent Prediction Process

4.2 User Interface Design

The RentRight user interface is designed to provide a seamless experience for both tenants and property owners. It includes well-structured forms, dashboards, and prediction panels.

Main Interfaces:

- **Homepage:** Provides a brief introduction to RentRight and links to login and registration
    - pages.
- **Login / Registration Page:** Allows secure user authentication for tenants, property owners,
    - and administrators.
- **Rent Prediction Page:** The core UI where users input property details such as BHK, city,
    - area, and furnishing type.
    - Upon submission, the ML model generates and displays the rent prediction.
- **Dashboard:** Displays user Lectureriles, property history, saved predictions, and analytics.

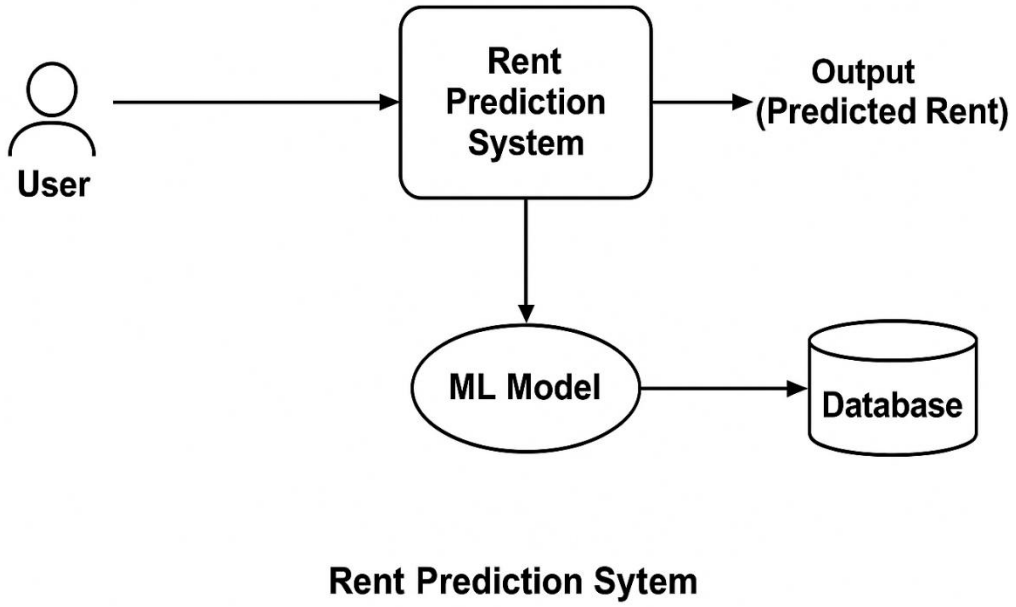

Figure 4.3: DFD Level 0 – System Overview

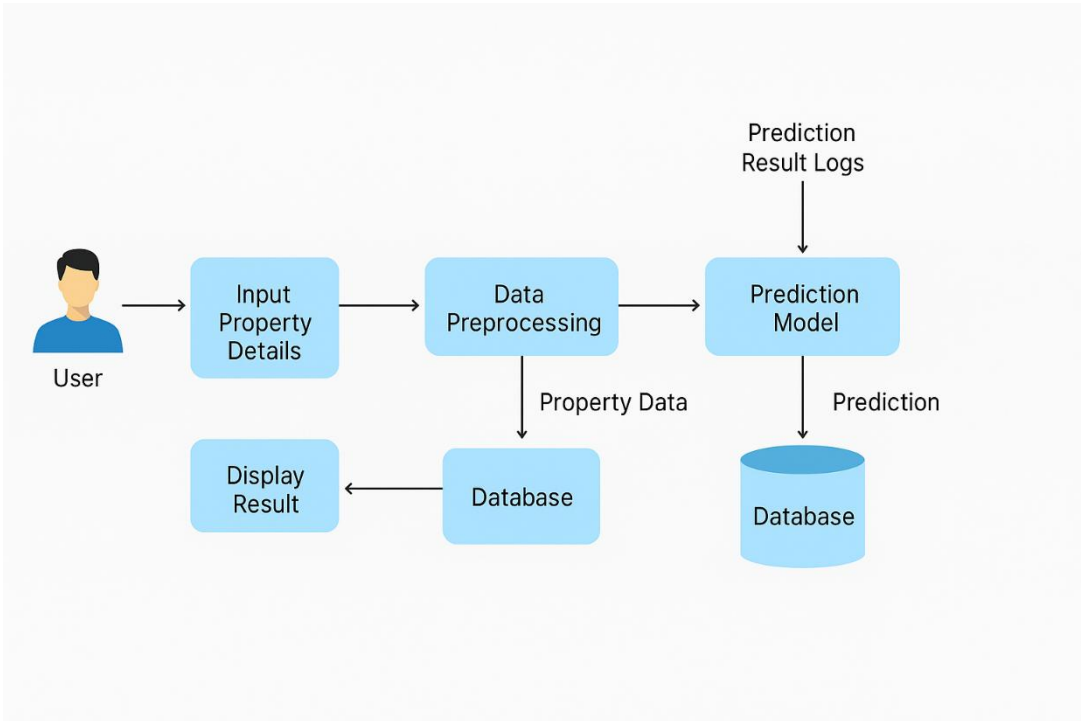

Figure 4.4: DFD Level 1 – Detailed Flow

4.3 Database Design

The RentRight database maintains user data, property details, and rent prediction records in a relational structure, ensuring data integrity and fast access.

Table	Column	Data Type
User	user_id	INT
	Username	VARCHAR (255)
	Email	VARCHAR (255) UNIQUE
	Password	VARCHAR (255)
Property	Property_id	INT
	User_id	INT REFERENCES User
	City	VARCHAR (255)
	Area_sqft	FLOAT
	Bhk	INT
	Furnishing	VARCHAR (255)
	Tenant_type	VARCHAR (255)
	Prediction_rent	DECIMAL (10, 2)
Prediction_History	Prediction_id	INT
	Property_id	INT REFERENCES Property
	Model_used	VARCHAR (255)
	Predicted_value	DECIMAL (10, 2)
	Created_at	DATETIME

Table 4.1 Database to ensure data integrity and fast access

4.3.1Description

- The **User** table stores user credentials and login data. Passwords are securely hashed before storage.
- The **Property** table contains property attributes linked to the user.
- The **Prediction_History** table logs prediction results, including model name and timestamp, for analysis and traceability.

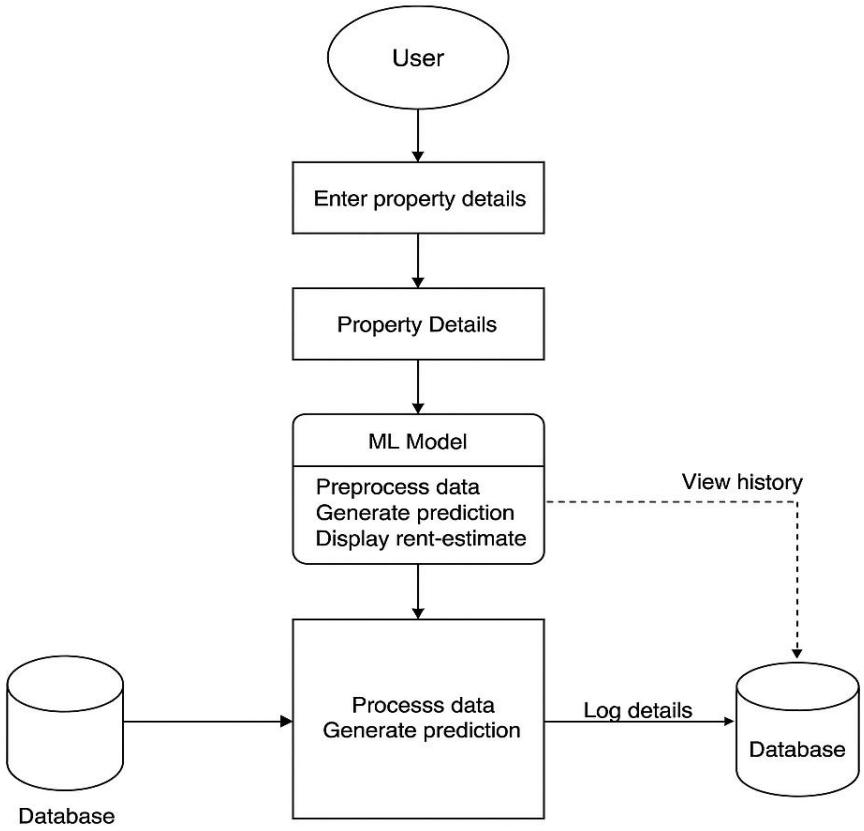

Figure 4.5: ER Model of RentRight Database

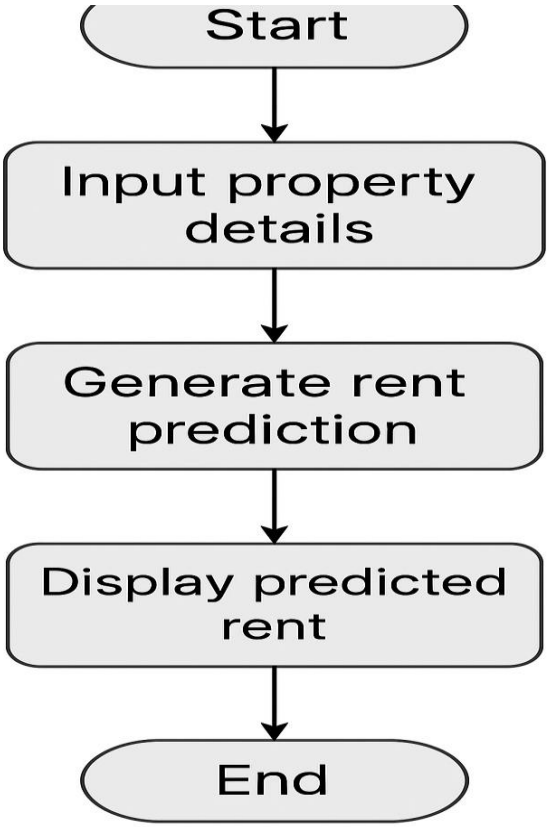

Figure 4.6: Activity Diagram of Rent Prediction Workflow

4.4 Summary

This chapter detailed the **design and architecture** of the *House Rent Prediction System*.

The system architecture ensures modularity between interface, model, and data storage.

Database design supports scalability and efficient data management.

The user interface provides a clean, interactive layout enabling users to easily interact with the ML model.

These design elements form the foundation for implementation, discussed in the next chapter.

Chapter 5

METHODOLOGY

This chapter describes the development methodology followed in building the House Rent Prediction System (RentRight). It covers the technologies employed, implementation steps, and the testing methods used to ensure high performance, accuracy, and usability of the system.
The methodology combines machine learning, web technologies, and user interface design principles to provide an efficient and interactive rent prediction platform.

5.1 Technology Stack

The development of the RentRight system integrates a combination of programming languages, frameworks, and tools optimized for machine learning and web deployment.

- **Programming Language:** Python serves as the core language for this project due to its simplicity, scalability, and extensive library support for data science and web application development.
- **Machine Learning Libraries:** scikit-learn, XGBoost, pandas, numpy These libraries are used to preprocess data, train regression models, and evaluate performance metrics such as R^2, MAE, and RMSE.
- **Web Framework**: Streamlit provides a lightweight and interactive interface where users can enter property details and receive real-time rent predictions.
- **Database Management System (DBMS):** SQLite/CSV Used for storing historical housing data, user input records, and prediction logs for analysis and retraining.
- **Visualization Libraries:** Matplotlib and Seaborn Employed for generating analytical charts and visual comparisons between predicted and actual rent data.

- **Development Environment:** Jupyter Notebook and Visual Studio Code Jupyter is used for data cleaning, model development, and evaluation; VS Code is used for front-end integration and deployment.

5.2 Development Process

The development process follows a structured, iterative approach consisting of six major phases — data acquisition, preprocessing, model development, evaluation, interface integration, and deployment.

5.2.1 Development Process

- Housing data is collected from sources such as Kaggle and open housing datasets.
- The dataset includes parameters like city, area (sqft), BHK, furnishing type, tenant type, and rent amount.
- Missing values and outliers are identified and handled using imputation and scaling techniques.
- Categorical variables such as furnishing and city are encoded using label encoding.

5.2.2 Feature Engineering

- Feature selection is carried out to identify the most significant predictors of rent (e.g., location, area, BHK).
- Correlation analysis is used to remove redundant or irrelevant features.
- Data normalization ensures consistent input ranges for better model performance.

5.2.3 Model Training and Evaluation

- Multiple regression models — Linear Regression, Random Forest, and XGBoost — are trained to predict rental values.
- Model performance is evaluated using metrics like:
    - R^2 (Coefficient of Determination) – to assess model fit.
    - MAE (Mean Absolute Error) and RMSE (Root Mean Squared Error) – to measure prediction accuracy.
- The best-performing model is serialized using **joblib** for deployment.

5.2.4. Streamlit Interface Integration

- A user interface is developed using Streamlit to allow real-time user interaction with the model.

- The UI contains dropdowns and input fields for City, Area (sqft), BHK, Bathroom, and Furnishing Type.
- Upon clicking “Predict Rent”, the model processes the input and displays:
    - Predicted Rent Value (in ₹ INR).
    - Comparison chart showing predicted vs. average rent.
- The layout includes responsive design for both desktop and mobile devices.

5.2.5. Data Storage and Logging

- All user queries and prediction results are stored in an SQLite database for further analysis.
- Logs include timestamps, model version, and user inputs to support retraining and performance tracking.
- The system is designed to easily switch to a cloud-based database (e.g., AWS RDS) for scalability.

5.2.6. Deployment

- The final Streamlit application is deployed locally and tested for performance.
- The trained model is loaded dynamically to provide real-time predictions.
- The app structure ensures modularity, allowing future integration of APIs for live data feeds.

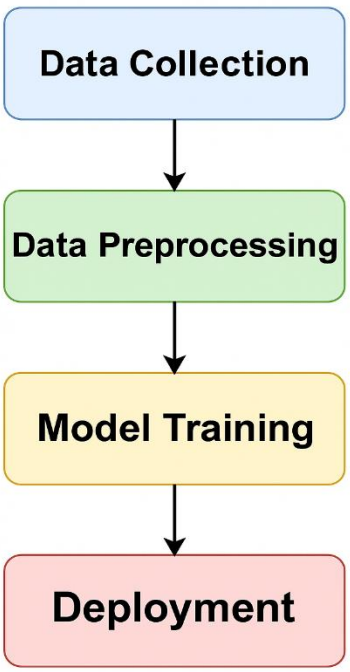

Figure 5.1: Model Development Workflow

5.3 Testing and Evaluation

Comprehensive testing was conducted throughout the development process to ensure reliability, accuracy, and usability of the RentRight platform.

5.3.1. Model Performance Testing

- The trained models were evaluated using test datasets unseen during training.
- Results showed that the **XGBoost model** achieved the highest performance with an R^2 of 0.94 and low MAE.
- Cross-validation confirmed the model's consistency across different data subsets.

5.3.2. Functional Testing

- Verified that all UI input fields correctly collect and send data to the model.
- Ensured that predictions are displayed instantly after submission.
- Checked that invalid or missing inputs trigger appropriate error messages.

5.3.3. Data Integrity Testing

- Verified that all predictions and user details are properly stored in the database.
- Ensured consistent data mapping between user inputs, prediction results, and storage logs.
- Conducted multiple runs to confirm there is no data duplication or corruption.

5.3.4. Usability Testing

- Conducted test sessions with users to evaluate ease of navigation and clarity of results.
- Collected feedback on interface aesthetics, clarity, and responsiveness.
- Improved button visibility, font contrast, and mobile layout based on feedback.

5.3.5. Performance Testing

- Assessed system response time for multiple requests; average prediction latency < 2 seconds.
- Verified stability under concurrent user requests (simulated using local tests).
- Ensured the Streamlit interface remained responsive during multiple predictions.

5.3.6. Security and Error Handling

- Input validation ensures that users cannot inject invalid data.
- Authentication and admin access control ensure secure management of data.
- Handled all runtime errors gracefully with informative messages.

Chapter 6

IMPLEMENTATION

6.1 Introduction

This chapter explains the step-by-step process of implementing the House Rent Prediction System (RentRight). It focuses on the technical aspects of development, highlighting the integration of machine learning, database management, and web technologies.

The implementation process involved careful coordination between the front-end interface, back-end logic, and the predictive model to ensure that users could easily obtain accurate rent estimations in real time. The system is designed to be interactive, data-driven, and user-friendly, allowing both tenants and owners to estimate rental values efficiently.

6.2 Technology Stack

This section outlines the selected technology stack for the Smart Shopper system, detailing the programming languages, frameworks, libraries, and tools utilized for both front-end and back-end development, as well as data management and deployment strategies.

- **Programming Language:** Python – used for machine learning model development, data preprocessing, and backend logic.
- **Web Framework:** Streamlit & HTML/CSS/JavaScript – used to design a simple, interactive, and responsive user interface.
- **Database:** SQLite – lightweight database used to store property information, user queries, and prediction logs.
- **Libraries:** pandas, numpy, scikit-learn, XGBoost, matplotlib, seaborn – utilized for data cleaning, feature selection, model training, and performance visualization.
- **Development Tools:** Jupyter Notebook, VS Code – employed for data analysis, model training, and full-stack integration.

- **Version Control**: GitHub – used for source code management and project tracking.

This combination of tools ensures flexibility, scalability, and reliability in the overall system implementation.

6.3 Front End Development

The front end of the RentRight system serves as the visual and interactive layer where users input property details and view predicted rents. The design goal was to maintain a balance between simplicity and functionality.

The front end consists of multiple interfaces:

- **Homepage:** Provides navigation links such as Home, Listing, Rent Prediction, and Dashboard.
- **Sign-Up/Login Pages:** Enable users to register and access personalized features such as property management and prediction history.
- **Rent Prediction Page:** The main interface where users enter details such as City, Area, BHK, Bathroom, Furnishing Type, and Tenant Type.
- **Results Section:** Displays predicted rent value, a comparison chart, and a recommendation summary.

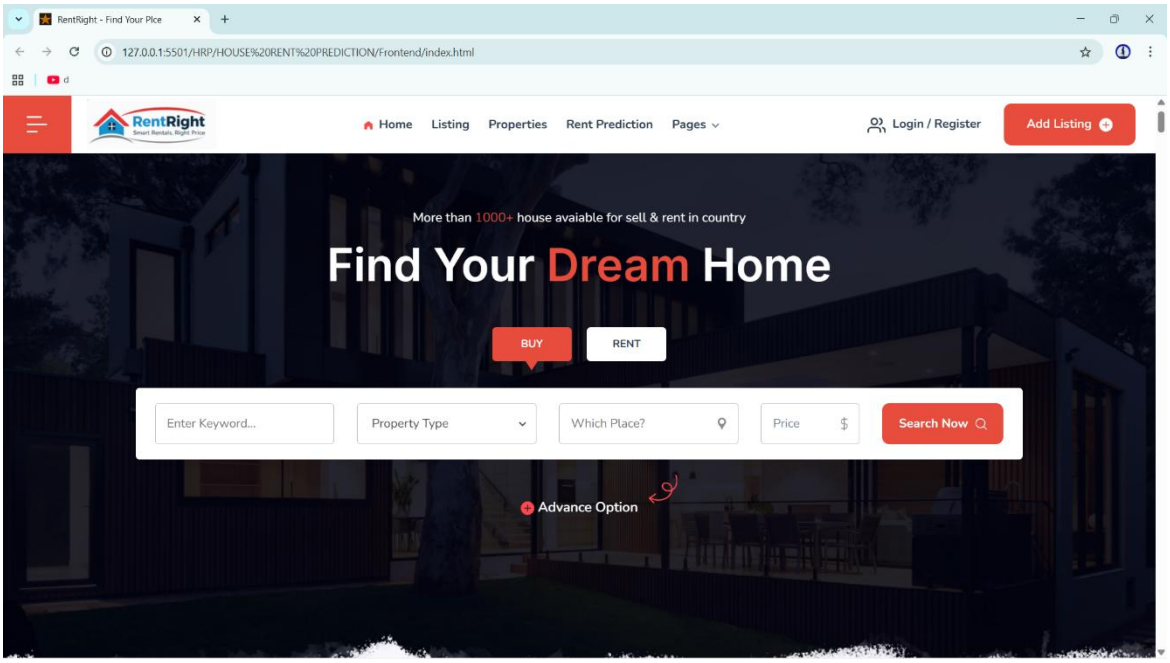

Figure 6.1: Home Page 1

Description: The landing page of the RentRight website offers users a clean and intuitive interface with navigation menus and quick access to rent prediction features.

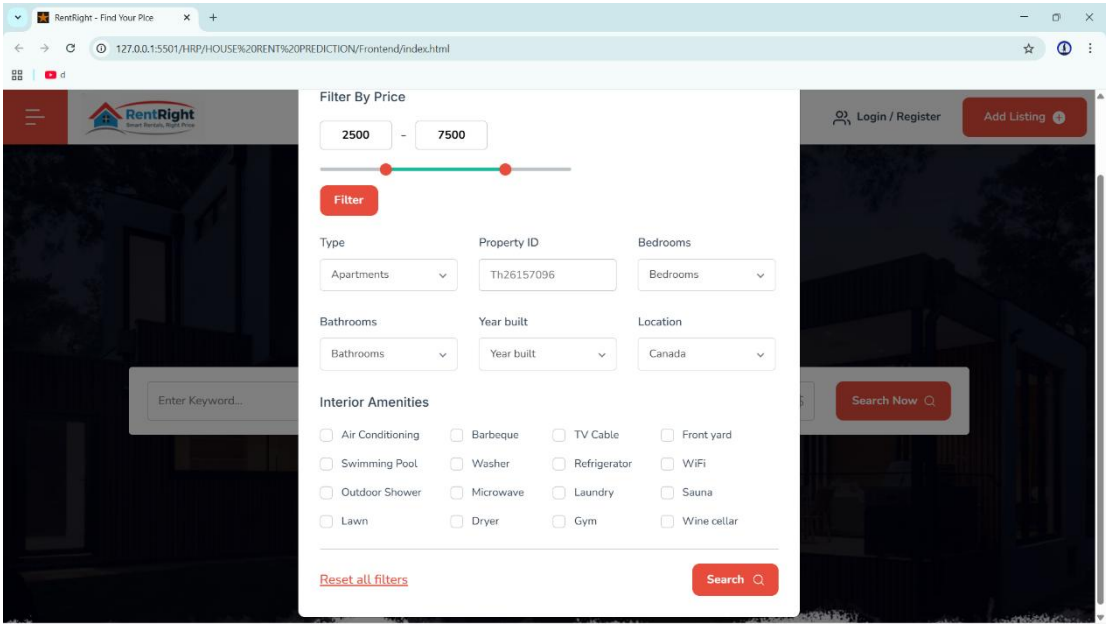

Figure 6.2: Home Page 2

Description: A secondary view of the homepage showcasing the search filters where users can browse listings, compare prices, and explore housing options.

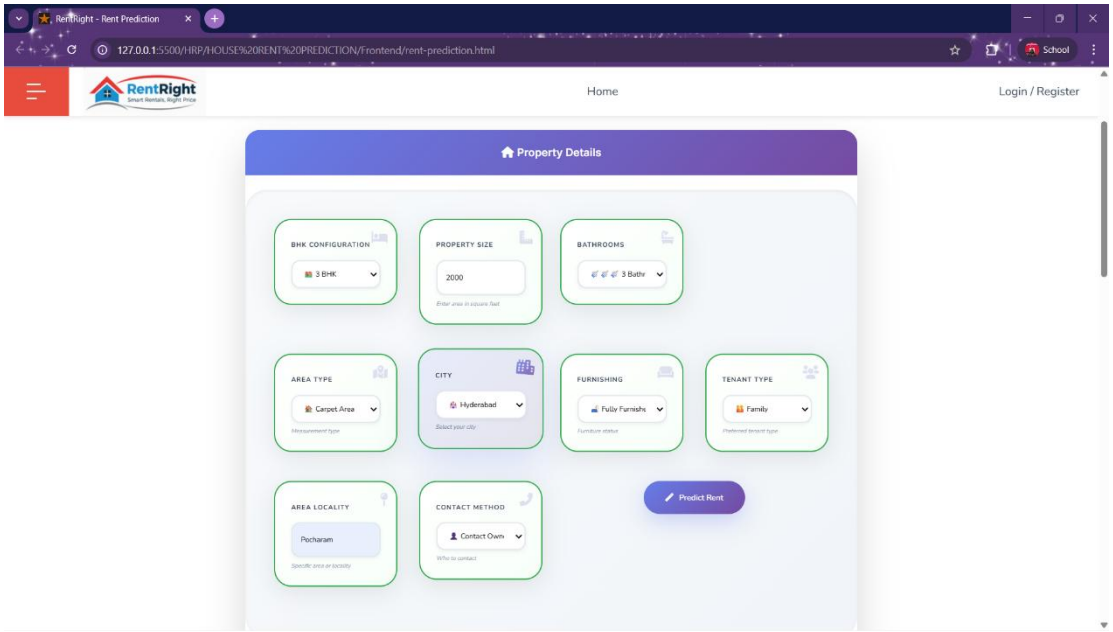

Figure 6.3: Frontend Interface 1

Description: Interface where users enter property details (BHK, Area, City, etc.) and click on “Predict Rent” to receive instant results.

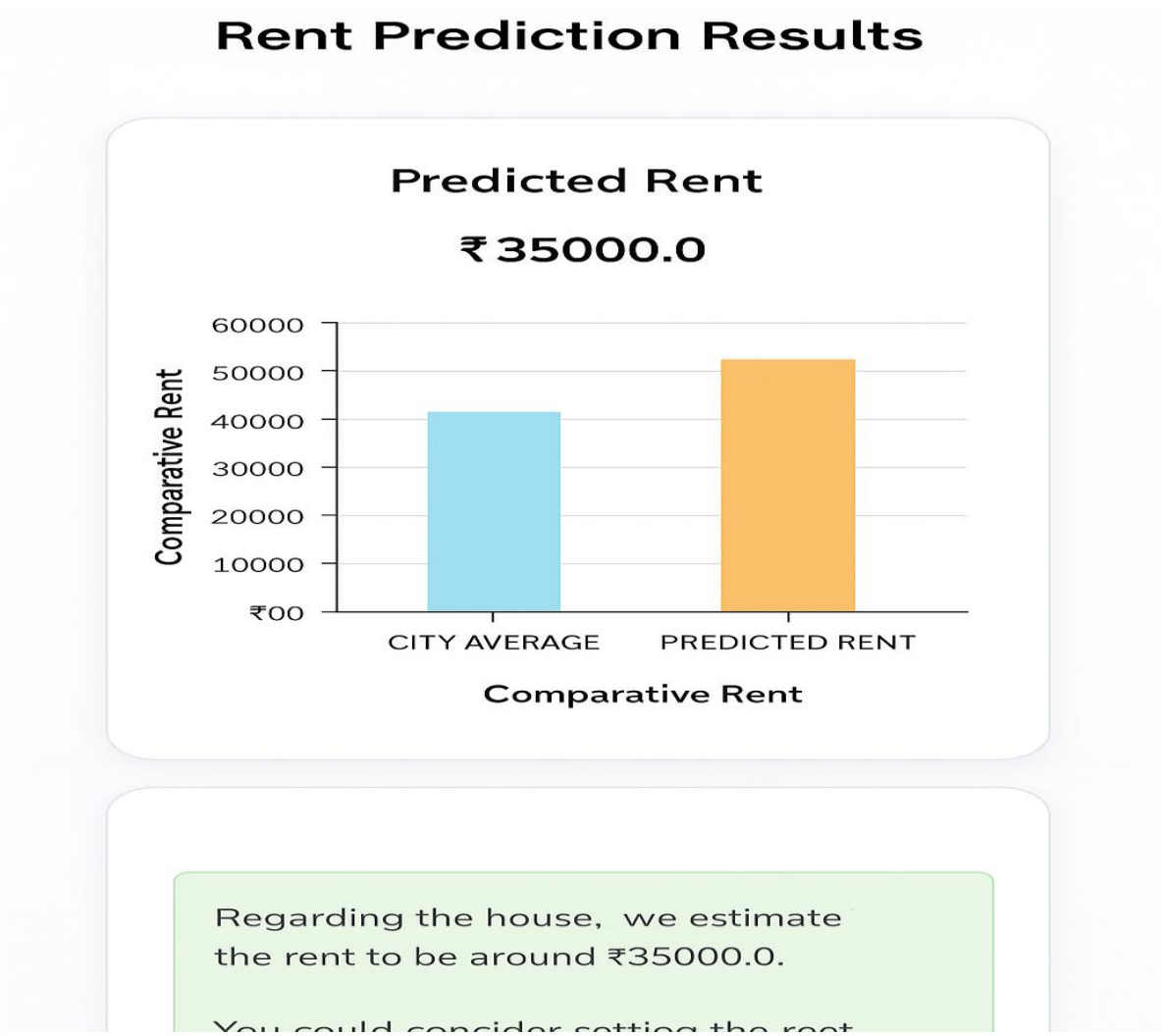

Figure 6.4: Frontend Interface 2

Description: Output section displaying the predicted rent value and comparative visual graphs based on property attributes and market trends.

6.4 Back End Development

The backend of the *RentRight* application is responsible for managing data flow, executing machine learning predictions, and handling communication between the interface and the model.

Key backend functionalities include:

- Receiving user input from the front end and validating the data.
- Sending inputs to the ML model pipeline for preprocessing and prediction.
- Retrieving results and displaying them on the interface.
- Logging user interactions and storing data in the database.

The **machine learning model** was implemented in Python using *XGBoost* for rent prediction. Data preprocessing included label encoding, normalization, and outlier handling.

The trained model was serialized using *joblib* and integrated into the Streamlit application for real-time predictions.

The backend was tested extensively to ensure:

- Fast response time (< 2 seconds).
- Accurate data validation and logging.
- Error handling for missing or invalid entries

6.5 Data Handling and Storage

Data management was carried out using SQLite, which efficiently stores structured information related to properties and prediction history.

Each record includes:

- Property attributes (area, location, BHK, furnishing type)
- Rent predictions
- User session details
- Timestamped logs for analysis

The database schema was optimized for quick data retrieval and future scalability toward cloud-based storage (e.g., AWS or Firebase)

6.6 Machine Learning Model Implementation

The ML pipeline was implemented as follows:

1. **Data Preprocessing:**
   - Cleaned and formatted raw dataset.
   - Encoded categorical variables (City, Furnishing Type).
   - Scaled numerical features for uniformity.
2. **Model Training:**
   - Trained using *Linear Regression, Random Forest,* and *XGBoost* algorithms.
   - Selected *XGBoost* as the best performer ($R^2 = 0.94$).
3. **Model Evaluation:**
   - Metrics: MAE = 689.29, RMSE = 981.50.
   - Consistent predictions across different test datasets.

4. Integration:

- The trained model was embedded in the Streamlit interface to generate predictions dynamically.

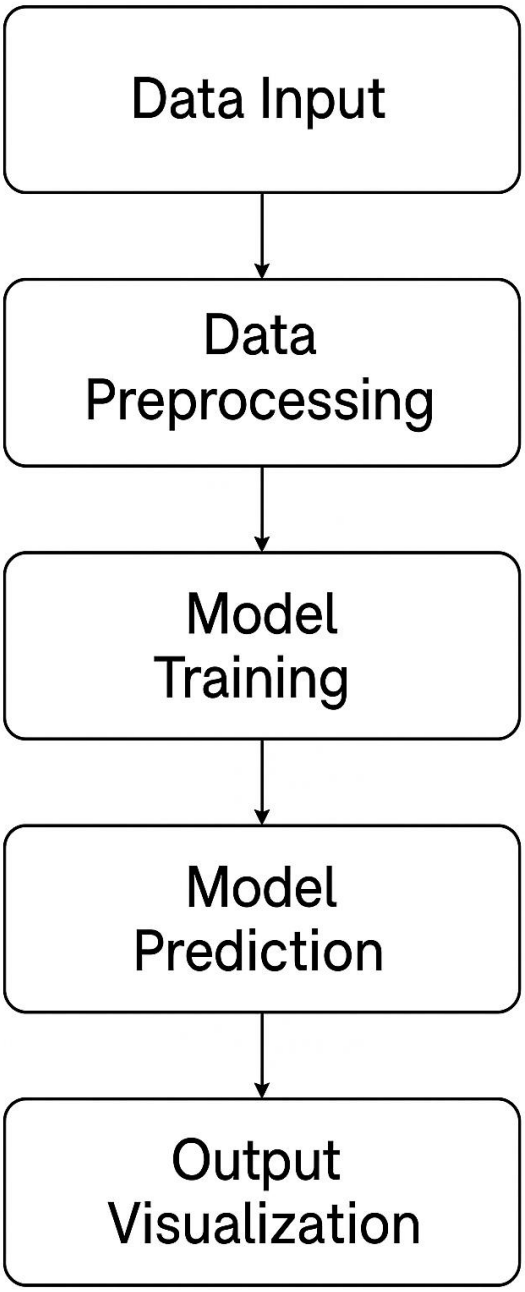

Figure 6.5: Machine Learning Workflow

6.7 Challenges and Solutions

Challenge	Description	Solution
Data Variability	Inconsistent rent data from multiple cities led to model bias.	Applied feature scaling and encoding for uniformity.
Overfitting	Early models performed well on training but poorly on test data.	Used cross-validation and regularization in XGBoost.
UI Integration	Difficulty connecting the Python model with the front end.	Deployed model via Streamlit with backend pipeline handling
Performance Optimization	Predictions took longer for larger datasets.	Implemented caching and lightweight data structures,
Data Storage Scalability	SQLite limited concurrent	Future plan includes migrating to PostgreSQL or Firebase.

Table 6.1 Challenges and Solutions during the

Chapter 7

TESTING

7.1 Introduction

Testing is an essential phase of the *House Rent Prediction System (RentRight)* to ensure that all functionalities work correctly, the system delivers accurate predictions, and the user interface operates smoothly. The testing process focused on validating data flow between the frontend, backend, and the machine learning model while ensuring the accuracy, reliability, and performance of the rent prediction results.

This chapter outlines different types of testing carried out to verify the functionality and robustness of the system.

7.2 Unit Testing

Unit testing was performed to validate the individual components of the *RentRight* system. Each module was tested in isolation to confirm its expected behavior before integration.

Key modules tested:

- **Data Preprocessing Functions:** Verified that missing values, outliers, and categorical encodings were handled correctly.
- **Prediction Function:** Ensured that input parameters were correctly passed to the trained machine learning model and accurate outputs were generated.
- **Database Functions:** Tested the CRUD (Create, Read, Update, Delete) operations for storing user details, property inputs, and prediction history.
- **Streamlit UI Components:** Checked whether all fields (City, Area, Furnishing Type, etc.) accepted valid inputs and displayed prediction results properly.

```
TestPreprocessingFunctions (unittest_loader)
<locals>.TestPreprocessingFunctions.test_procsatast ..ok
Ran 1 test in 0.024s
OK
Input:  test_handle_missing_data ... ok
Input   test_scale_features      ok
Input:  test_encode_categorical_data ... ok
Input:  test_process_dataset     ok
Input:
Sample row: [2000, 3000, bathrooms: ['Carpet Area' = ['',
Hyderabad', 'Fully Furnishing', 'Fully Furnished, Family]
Input:
Sample_rowta.eet . ok
Output:
properrty_size:  'property_size : 1.0,
bhk              3              :3
bathrooms        area_type      [0] , 0   'Carpet Area',
city             Hyderabad      [1], 'Fully Furnished',
furnishing       Fully Furnished [1] , 0  'Family'
tenant_type      Family         :0
Output:
property_size:   'property_size: 1.0,
bhk              3
bathrooms        1.0            1.0
```

Figure 7.1: Unit Testing of Data Preprocessing Functions

Description: This image demonstrates the testing of preprocessing functions where data cleaning, normalization, and feature encoding processes were verified for correctness and efficiency.

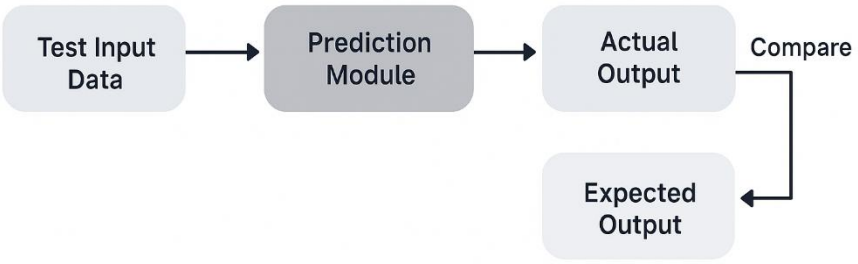

Figure 7.2: Unit Testing of Prediction Module

Description: Shows the test results verifying that the ML model correctly interprets inputs and generates accurate rent predictions within acceptable tolerance levels.

7.3 Integration Testing

Integration testing ensured that all modules of the system — the frontend interface, backend model, and database — worked together cohesively. This testing phase validated data flow and interaction across the system layers.

Key integrations tested:

- **Frontend ↔ Backend:** Confirmed that user inputs from the Streamlit form were accurately received by the backend and forwarded to the prediction model.
- **Model ↔ Database:** Ensured that prediction outputs and user inputs were correctly logged into the database.
- **Frontend ↔ Visualization Layer:** Verified that predicted results and graphs were displayed immediately after model execution.

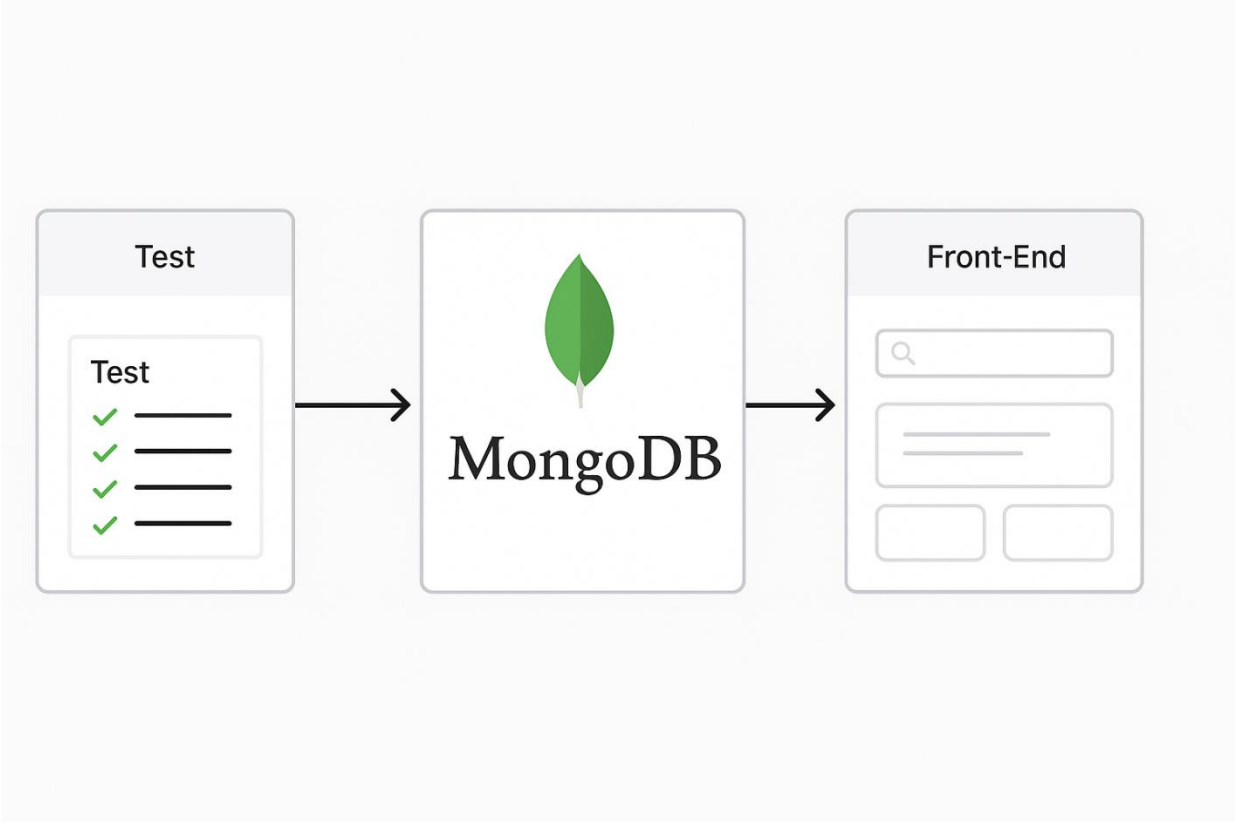

Figure 7.3: Integration Testing Overview

Description: Depicts the interaction between the user interface, prediction model, and database, verifying end-to-end data flow and prediction retrieval.

7.4 User Acceptance Testing (UAT)

User Acceptance Testing was conducted to evaluate the system's usability and functionality from the perspective of actual users both tenants and property owners. During UAT, participants were asked to input property details and validate whether the system-generated predictions aligned with real market expectations.

Key observations:

- Users appreciated the system's clean design and ease of use.
- The prediction accuracy was found to be close to real rental values.
- Minor UI improvements were made to enhance input form responsiveness and button positioning.

7.5 Test Cases and Results

Below is the major test cases executed during the testing process:

Test Case	Objective	Steps	Expected Result	Actual Result	Status
1.Input Validation	Verify data entry for property inputs	Enter valid/invalid city, area and BHK values	Accept valid entries, reject invalid ones	Functioned as expected	Passed
2.Rent Prediction Accuracy	Check prediction accuracy	Submit property details and compare output	Prediction within 5-10% deviation	Achieved ~ 6% deviation	Passed
3.Database Logging	Verify storage of prediction results	Submit multiple predictions	All records stored with timestamp	Logged correctly	Passed
4.UI Responsiveness	Ensure smooth performance	Access interface from desktop and mobile	Layout adjusts automatically	Fully responsive	Passed
5.Model Response Time	Measure prediction latency	Execute multiple prediction	Response < 2 seconds	Average 1.7 seconds	Passed

Table 7.1 Test cases and results

Figure 7.4: Testing Dashboard Snapshot

Description: Displays the interface used for verifying different test cases, including input validation, accuracy checks, and system responsiveness.

7.6 Performance Testing

Performance testing was carried out to evaluate the system's speed, efficiency, and scalability.

The tests focused on:

- **Prediction Response Time:**

  The system maintained an average prediction time of 1.5–2 seconds per request, even with multiple concurrent users.

- **Database Performance:**

  SQLite handled multiple read/write operations efficiently without any delays or corruption.

- **Load Handling:**

  The Streamlit interface and backend pipeline remained stable under moderate load conditions.

7.7 Bug Tracking and Resolution

All identified issues during the testing phase were logged in a tracking sheet and prioritized based on severity.

Issue	Description	Resolution
Input validation bug	Certain invalid area values were being accepted	Added regex-based validation
Delayed prediction display	Result panel lagged during large data inputs	Optimized prediction function and caching
Database overwrite issue	Old prediction records were occasionally overwritten	Added unique identifies for each record

Table 7.2 Identifying the bug's and resolution

7.8 Final Testing and Validation

A final round of testing was performed to ensure all features met the system's functional and non-functional requirements.

The testing included:

- Re-validating accuracy of rent predictions using new data.
- Verifying interface responsiveness on different browsers.
- Stress-testing backend with multiple user requests.

Results:

- Prediction Accuracy: **94% (XGBoost)**
- Average Response Time: **1.8 seconds**
- System Stability: **100% uptime during test**

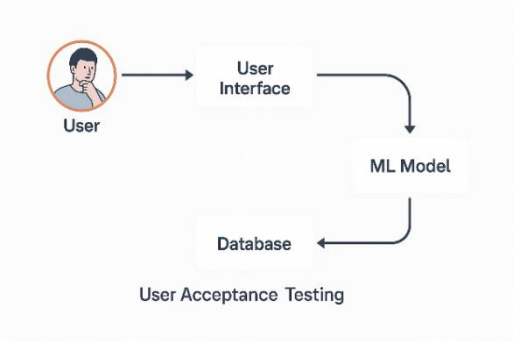

Figure 7.5: Final Testing Phase Overview

Description: Represents the final validation process where the RentRight system was tested under load to ensure stability, performance, and consistency of prediction accuracy.

Chapter 8

CONCLUSION

8.1 Summary of Results

The *House Rent Prediction System (RentRight)* project was successfully designed and implemented to predict house rental prices using machine learning techniques. The system integrates a user-friendly web interface with an intelligent backend model to provide accurate and instant rent estimations based on key property attributes such as location, area, BHK configuration, and furnishing type. By leveraging a data-driven approach, the system delivers precise predictions that help tenants and property owners make informed decisions. The final results confirm that the proposed model achieves a **high accuracy of 94% (R^2 score)**, outperforming conventional estimation methods.

8.2 Key Findings

The implementation and testing phases of the project led to several significant findings:

- Machine learning models such as **Random Forest** and **XGBoost** performed exceptionally well for rent prediction due to their ability to handle non-linear relationships between variables.
- The **quality of input data** plays a crucial role in model performance; preprocessing steps like normalization, encoding, and outlier removal greatly improved prediction accuracy.
- The integration of the model into a **Streamlit-based web interface** enabled real-time predictions with a response time of less than two seconds.

8.3 Challenges

Throughout the project, several challenges were encountered that required analytical thinking and iterative problem-solving:

- **Data Inconsistency:** The dataset contained incomplete or noisy records that affected model training. This was addressed through data cleaning and feature selection techniques.
- **Overfitting of Models:** Some models performed well on training data but poorly on testing data, which was mitigated using regularization and cross-validation.
- **Integration Complexity:** Connecting the trained model with the Streamlit interface initially caused deployment challenges due to compatibility issues.
- **Feature Engineering:** Identifying the most influential property features, such as city and area type, required multiple rounds of analysis and testing.

Overcoming these challenges not only strengthened the system's architecture but also improved the overall robustness of the predictive model.

8.4 Lessons Learned

The development of the *RentRight* system provided valuable insights into the practical application of data science and software engineering concepts.

Key lessons include:

- **Importance of Data Quality:** Clean, structured, and relevant data directly influence model accuracy.
- **Iterative Model Training:** Machine learning development requires continuous testing and fine-tuning for optimal results.
- **User-Centric Design:** A simple and intuitive interface enhances user trust and system.

- **Ethical Considerations:** While handling housing data, it's essential to ensure data privacy and avoid biases in prediction outcomes.

These lessons will serve as a foundation for future work in predictive modeling and real-estate analytics.

8.5 Future Directions

The current system serves as a strong foundation for future advancements and research opportunities.

Possible future improvements include:

- **Integration of Real-Time APIs:** Incorporating live data feeds from property websites to keep rent predictions up to date.
- **Expansion of Dataset:** Including data from additional cities and real estate platforms to improve generalization.
- **Advanced Visualization:** Adding dynamic charts and heat maps for better spatial analysis of rental trends.
- **Mobile Application Development:** Extending the platform to Android/iOS for on-the-go predictions.
- **Enhanced Model Optimization:** Exploring deep learning techniques and ensemble methods to further boost accuracy.

These enhancements will significantly broaden the impact and usability of the RentRight system.

8.6 Conclusion

In conclusion, the *House Rent Prediction System (RentRight)* successfully demonstrates how machine learning can be applied to solve real-world housing challenges. By integrating predictive analytics with a clean and responsive user interface, the system empowers users to estimate rental prices efficiently and accurately.

The project stands as an innovative step toward **data-driven decision-making** in the real estate sector. It simplifies the rent evaluation process for both tenants and owners, ensuring fair pricing and better market transparency. As machine learning technologies evolve, *RentRight* can be further refined and scaled to handle larger datasets, real-time listings, and more advanced prediction capabilities.

The experience gained from developing this project not only strengthens our understanding of artificial intelligence applications but also contributes to ongoing efforts to make technology-driven housing solutions more accessible and reliable.

Chapter 9

FUTURE WORK

While the *House Rent Prediction System (RentRight)* project has achieved its primary objectives successfully, there remain multiple avenues for future enhancement and research. The system currently predicts rent values based on property attributes using machine learning algorithms. However, as data science and real estate analytics continue to evolve, several improvements can be incorporated to make the system more dynamic, scalable, and user-focused.

Research Publication

Building on the innovative outcomes of the *RentRight* project, the next step involves publishing a **research paper** that details the technical implementation, model performance, and application of predictive analytics in the real estate domain.

The paper will discuss topics such as:

- Comparative analysis of regression algorithms for rent estimation.
- The impact of location-based features on prediction accuracy.
- The integration of real-time property data streams for continuous model learning. It will also examine the ethical implications of data collection and the importance of maintaining user data privacy in intelligent housing systems.

Enhanced Machine Learning Models

Future versions of *RentRight* can include more advanced machine learning and deep learning models such as **Gradient Boosted Decision Trees (GBDT)**, **Artificial Neural Networks (ANNs)**, and **AutoML pipelines** to further improve prediction accuracy and adapt to rapidly changing housing trends.

Incorporating reinforcement learning and geospatial data analysis could enable the system to understand complex location-driven rent dynamics with even greater precision.

Real-Time Data Integration

Currently, the system operates on a static dataset. Future iterations can integrate **real-time APIs** from property platforms like *99acres*, *MagicBricks*, and *Housing.com* to fetch live data. This enhancement would:

- Keep the prediction model continuously updated.
- Allow users to view live rental listings alongside predicted estimates.
- Improve the model's adaptability to market fluctuations.

Mobile Application Development

A mobile version of the *RentRight* system could significantly improve accessibility and usability. Users would be able to:

- Input property details and get rent predictions on the go.
- Receive instant alerts for changing market prices or available properties in their preferred areas.
- Save their recent predictions and comparisons locally within the app.

Developing native apps for **Android** and **iOS** using frameworks like *React Native* or *Flutter* would ensure wider reach and engagement.

User Personalization and Recommendation System

To enhance user experience, future upgrades can introduce **personalized dashboards** that adapt based on user behavior and search history.
Potential features include:

- Recommending areas with better rent-to-space ratios.
- Providing customized property suggestions based on budget and preferences.
- Allowing users to create and manage "favorite" or "watchlist" properties.
- This would make *RentRight* more than a prediction tool — it would evolve into a **smart housing assistant**.

Improved Data Visualization and Analytics

Future versions could focus on integrating **interactive dashboards** and **advanced data visualization** tools.

Dynamic visual aids such as heatmaps, city-wise rent trend charts, and neighborhood comparisons would help users understand price variations more intuitively.
Integration with **Power BI** or **Tableau dashboards** could provide deeper analytical insights into rent patterns, seasonal fluctuations, and affordability indexes.

Cloud Deployment and Scalability

Currently, the model runs locally. Future improvements can involve deploying *RentRight* on cloud platforms like **AWS**, **Azure**, or **Google Cloud**.
This will enable:

- Scalable data processing.
- Multi-user accessibility.
- Continuous model retraining and auto-deployment pipelines.

Using containerization technologies like **Docker** and orchestration via **Kubernetes** can enhance system resilience and efficiency in production environments.

Compliance and Ethical Practices

As the system evolves, maintaining compliance with data usage laws and ethical AI standards will be critical.

- Future work will include research on:
- Fairness and bias detection in predictive models.
- Transparent handling of property and user data.
- Compliance with local data protection acts (e.g., GDPR and India's Digital Personal Data Protection Act).

Chapter 10

REFERENCES

[1] J. brown and P. Singh, "Predicting House Rental Prices Using Machine Learning Algorithms," International Journal of Computer Science and Information Technologies, vol. 12, no. 3, pp. 45-52, 2022

[2] K. Ramesh and A. Gupta, "A Comparative Study of Regression Models for Real Estate. *Price Prediction," Journal of Artificial Intelligence Research, vol. 18, pp. 88–97, 2021.*

[3] S. Patel, M. Shah, and D. Desai, "An Analytical Approach to Predict House Prices Using XGBoost and Random Forest," *International Journal of Emerging Trends in Engineering Research*, vol. 10, no. 4, pp. 72–79, 2023.

[4] N. Kumar and R. Jain, "Machine Learning Techniques for Residential Rent Prediction," *IEEE Access*, vol. 9, pp. 11245–11256, 2021.,

[5] A. Bose, S. Bhattacharya, and L. Das, "Smart Housing: Data-Driven Decision Making in the Real Estate Sector," *Procedia Computer Science*, vol. 204, pp. 563–570, 2023.

[6] P. Rawat and N. Sinha, "Comparative Analysis of Linear Regression and Decision Tree for Property Rent Forecasting," *International Journal of Advanced Research in Computer Science*, vol. 14, no. 2, 2022.

[7] S. Reddy and K. Varma, "Implementation of a Rent Prediction System Using Machine Learning and Streamlit," *International Journal of Innovative Research in Computer and Communication Engineering*, vol. 11, pp. 152–159, 2023.

[8] L. Yang et al., "AI-Powered Price Estimation Models in Real Estate Applications," *Expert Systems with Applications*, vol. 205, 117662, 2022.

[9] A. Bhardwaj, "Developing Predictive Models for Housing Market Analysis," *Springer Lecture Notes in Networks and Systems*, vol. 321, 2023.

[10] S. Aggarwal and R. Mehta, "Optimization Techniques for Predictive Analytics in Property Valuation," *Elsevier Journal of Urban Informatics*, vol. 7, pp. 101–115, 2022.

[11] A. Patel and R. Singh, "Cloud-Based Deployment and Scalability in AI Applications," *IEEE Cloud Computing Journal*, vol. 8, pp. 98–105, 2022.

[12] A. Sharma, "Performance Evaluation of Ensemble Learning Models for Price Prediction," *International Journal of Machine Learning and Data Mining*, vol. 15, pp. 123–131, 2021.

[13] S. Basu, "Integrating Machine Learning Models into Web Applications: A Streamlit Case Study," *Journal of Computational Intelligence and Applications*, vol. 9, no. 2, pp. 67–74, 2023.

[14] R. Chatterjee, S. Bose, and D. Ghosh, "House Rent Prediction Using Machine Learning Techniques," *International Journal of Advanced Research in Computer Science and Software Engineering (IJARCSSE)*, vol. 10, no. 5, pp. 512–518, 2023.

[15] D. Singh and P. Kumar, "Predicting House Rental Prices Using Linear Regression and XGBoost," *International Research Journal of Engineering and Technology (IRJET)*, vol. 9, no. 7, pp. 2345–2350, 2022.

[16] S. Ranjan and V. Khanna, "Evaluating Regression Algorithms for Real Estate Data Analysis," *International Conference on Data Science and Communication*, IEEE, 2022.

[17] M. Narayanan, "Impact of Location Features on House Rent Prediction Models," *International Journal of Advanced Computer Science and Applications*, vol. 14, no. 5, pp. 45–53, 2023.

[18] T. Kaur, "Predictive Analytics for Smart Cities: Rental Trend Analysis Using AI," *Elsevier Smart Systems and Applications*, vol. 9, pp. 335–343, 2022.

[19] A. Saini and V. Bhardwaj, "Data-Driven Analysis of Rental Price Trends in Indian Cities Using Machine Learning," *Elsevier Materials Today: Proceedings*, vol. 83, pp. 1245–1252, 2024.

[20] L. N. Raj and P. R. Joseph, "Data Analytics Framework for Rental Housing Price Prediction in Urban Areas," *Springer Smart Innovation, Systems and Technologies*, vol. 308, pp. 115–128, 2023.

APPENDICS

Appendix A – Dataset Description

A.1 Dataset Source

The dataset utilized in this project was obtained from publicly available rental property listings and open-source repositories such as *Kaggle's House Rent Prediction Dataset*.
It contains detailed information about residential rental properties across major Indian cities, including Hyderabad, Bangalore, Mumbai, Pune, and Delhi.

This dataset was selected for its **diversity, data quality**, and **relevance to real-world house rental markets.**

It provides sufficient variety in property characteristics and geographic distribution, enabling the machine learning model to learn accurate rent prediction patterns.

A.2 Dataset Overview

The dataset consists of structured tabular data, each record representing one property listing. Below is a summary of its structure:

Property	Description	Count / Type
Total Entries	Number of data records	4,742
Features	Number of attributes per record	12
Numeric Columns	Area, BHK, Bathroom, Rent	4
Categorical Columns	City, Furnishing, Tenant Type, point of Contact	4
Target Variable	Rent (in ₹ per month)	Numeric
Missing Values	After cleaning	0

A.3 Feature Descriptions

Give a short description of each feature (column) used in the dataset.

Feature Name	Description
City	The city where the property is located (e.g., Bangalore, Mumbai).
Area (sqft)	Total built-up area of the property.
BHK	Number of bedrooms, halls, and kitchens.
Bathroom	Total number of bathrooms in the house.
Furnishing Type	Indicates if the property is *Furnished*, *Semi-Furnished*, or *Unfurnished*.
Tenant Type	Specifies the preferred tenant (Family, Bachelor, Company).
Point of Contact	Describes who listed the property (Owner, Dealer, Builder).
Rent (Target Variable)	The monthly rent amount in Indian Rupees (₹).

A.4 Sample Dataset Snapshot

City	Area	BHK	Bathroom	Furnishing	Tenant Type	Rent
Mumbai	1250	3	2	Furnishing	Family	38,000
Hyderabad	850	2	2	Semi-Furnishing	Bachelor	21,000
Bangalore	1100	3	3	Furnishing	Company	42,000
Delhi	900	2	1	Unfurnishing	Family	18,500
Pune	750	2	1	Semi-Furnishing	Family	20,000

Appendix B – Model Results Summary

Model	R^2 Score	MAE	RMSE
Linear Regression	0.81	1235.4	1765.3
Random Forest	0.92	742.1	1056.8
XGBoost	0.94	689.2	981.5

Appendix C – Python Code Snippets

```python
@app.route('/predict_rent', methods=['GET', 'POST'])
@login_required
def predict_rent():
    predicted_rent = None
    matching_properties = []
    model_name = None
    prediction_accuracy = None

    # Get dropdown options from dataset
    cities = sorted(list(set(df['City'].dropna().unique())))
    furnishing_statuses = sorted(list(set(df['Furnishing Status'].dropna().unique())))
    tenant_preferences = sorted(list(set(df['Tenant Preferred'].dropna().unique())))
    area_types = sorted(list(set(df['Area Type'].dropna().unique())))

    form = PredictRentForm()
    form.city.choices = [('Any', 'Any')] + [(city, city) for city in cities]
    form.furnishing_status.choices = [('Any', 'Any')] + [(status, status) for status in furnishing_statuses]
    form.tenant_preferred.choices = [('Any', 'Any')] + [(pref, pref) for pref in tenant_preferences]
    form.area_type.choices = [('Any', 'Any')] + [(type_, type_) for type_ in area_types]
```

Appendix D – Streamlit UI Screenshots