VITyarthi Project

HOUSE PRICE PREDICTION APPLICATION

Submitted by Indrajith Sen
25BET10015

Indrajith Sen
25BET10015

VITyarthi - Build your own project

Problem Statement

Creating a House Price Prediction Application

Problem Objective

The objective of this project is to develop a Python-based house price prediction system that applies machine learning techniques to estimate property prices based on key factors such as bedrooms, bathrooms, area, and location. The project aims to train an accurate predictive model, provide a user-friendly interface for entering house details, and store all generated predictions in a database for future reference. Through this implementation, the project seeks to demonstrate practical application of AI and ML concepts in solving real-world problems and supporting informed decision-making in the housing market.

Functional Requirement

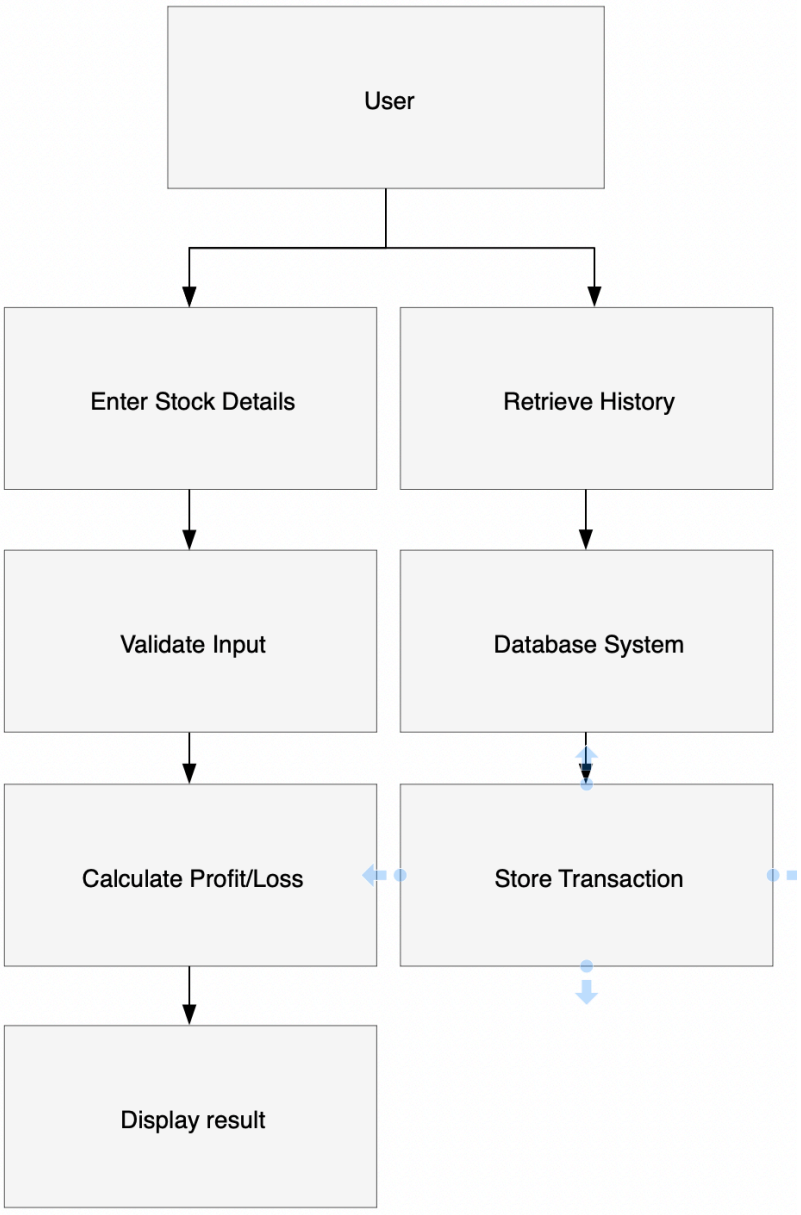
This system is required to provide a complete workflow that enables users to interact with the house price prediction application in a structured and efficient manner. The system must allow users to enter essential property details, including the number of bedrooms, number of bathrooms, built-up area in square feet, and the type of location, through a guided input process with appropriate validation to prevent incorrect or incomplete entries. Once the input is received, the application must process the data using a pre-trained machine learning model and generate an accurate predicted house price, which will be clearly displayed to the user along with a confirmation message. The system is also required to automatically store each prediction, together with the provided input values, in a SQLite database to maintain a persistent record of all transactions. Furthermore, the user is able to access a history module that retrieves and displays previously saved predictions in a readable format, ensuring transparency and traceability. The application operates through a menu-driven interface that allows users to seamlessly navigate between prediction, history viewing, and exit options, ensuring an intuitive and logical workflow throughout the system.

Non Functional Requirement

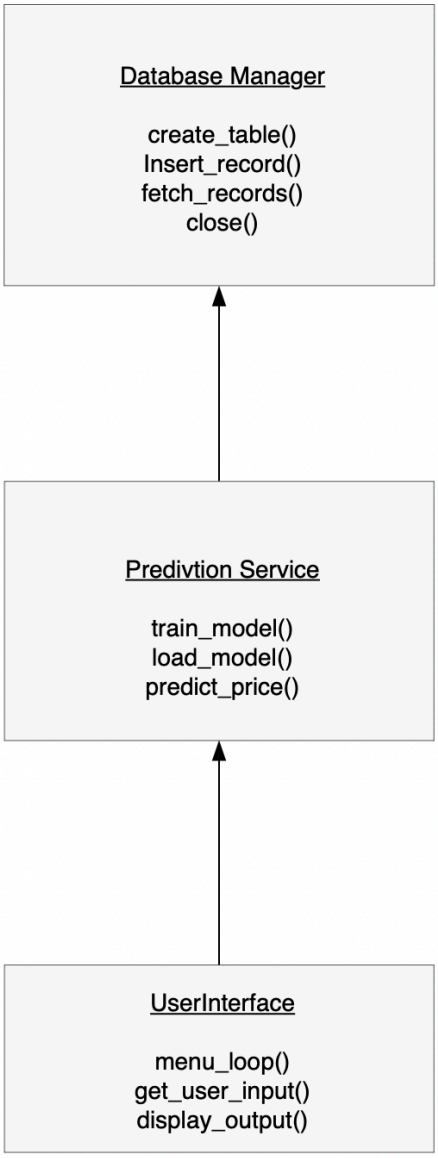
The system ensures high reliability by consistently performing predictions without unexpected failures and maintaining accurate storage of data in the database. The application provides good usability by offering a simple, menu-driven interface that is easy for users to navigate without requiring technical expertise. The performance of the system should remain efficient, ensuring that predictions and database operations are executed within a minimal response time. The software also maintains portability by running smoothly on any standard Python-supported environment without the need for specialized hardware. In terms of maintainability, the codebase has been structured clearly with modular components, allowing future updates such as model retraining, feature expansion, or interface improvements to be implemented with minimal effort.

Case Diagrams

Use Case Diagram



Class Diagram



ER diagram

Design Decisions & Rationale

The project uses Python as the primary programming language because it provides strong support for machine learning libraries and simplifies data processing and model development. A Random Forest Regression algorithm was selected for the prediction model due to its ability to handle non-linear relationships, reduce overfitting, and deliver more accurate results with small datasets compared to basic linear models. The system incorporates a SQLite database to store prediction history, as it is lightweight, file-based, and requires no external server setup, making it suitable for a small-scale, standalone application. StandardScaler was used for feature scaling to ensure that all input variables contribute equally to the model and improve prediction performance. A menu-driven console interface was used to maintain simplicity and accessibility, allowing users to interact with the system without needing graphical UI components. The codebase was designed in a modular structure, separating model training, prediction logic, and database operations to enhance readability, future scalability, and ease of maintenance.

Implementation Detail

The implementation of the House Price Prediction System was carried out using Python, integrating machine learning, data processing, and database storage within a single executable script. The system begins by setting up a SQLite database using the sqlite3 module, where a table named predictions is created to store user inputs and the corresponding predicted house prices. The dataset used for training is defined within the program and converted into a DataFrame using Pandas, after which the categorical location field is encoded into numerical values to make it suitable for machine learning processing. The features and target variable are then separated, and

the input data is standardized using StandardScaler to ensure uniform scaling across all attributes.

The model is trained using a Random Forest Regressor from the scikit-learn library, chosen for its robustness and ability to handle complex relationships in the data. The trained model and scaler are saved externally using the joblib library, allowing the system to load them later for real-time predictions without retraining. During execution, the application follows a menu-driven workflow where users can either enter new house details for prediction or view stored results. When the user enters input values, the system processes and scales the data, performs prediction using the pre-trained model, displays the estimated house price, and saves the result in the database. The program also includes a function to retrieve and display the complete prediction history from the database. Finally, the application runs in a continuous loop until the user selects the exit option, ensuring smooth interaction and efficient execution of all core functionalities.

Screenshots/Results

```
...   model trained successfully!

    ===== HOUSE PRICE PREDICTION SYSTEM =====
    1. Predict House Price
    2. View Prediction History
    3. Exit

    Enter choice: 1

    --- Enter House Details ---
    Bedrooms: 5
    Bathrooms: 4
    Area (sq ft): 2500
    Location (city/suburb/rural): rural
    /usr/local/lib/python3.12/dist-packages/sklearn/utils/validation.py:2739: UserWarning: X does not have va
      warnings.warn(

    Predicted House Price: ₹2615750.00

     Prediction saved to database!


    ===== HOUSE PRICE PREDICTION SYSTEM =====
    1. Predict House Price
    2. View Prediction History
    3. Exit

    Enter choice: 2

    ------ SAVED PREDICTIONS ------

    ID: 1 | Bedrooms: 5 | Bathrooms: 4 | Area: 2500.0 sq.ft | Location: rural | Price: ₹2615750.00 lakhs
    ID: 2 | Bedrooms: 5 | Bathrooms: 4 | Area: 2500.0 sq.ft | Location: rural | Price: ₹2615750.00 lakhs


    ===== HOUSE PRICE PREDICTION SYSTEM =====
    1. Predict House Price
    2. View Prediction History
    3. Exit

    Enter choice: 3

     Program closed. Thankyou!
```

Testing Approach

The testing approach validates both the functional correctness and reliability of the application. The system was first tested using manual input-based testing, where multiple combinations of bedroom count, bathrooms, area size, and location were entered to verify that the prediction function generates accurate and reasonable price outputs without errors. Boundary value testing was performed by providing minimum and maximum input ranges to ensure the program correctly handles extreme or unusual values, such as very small areas or high bedroom counts. Input validation was tested by entering invalid data types, including negative numbers and non-numeric values, to confirm that the application displays appropriate error messages and prevents crashes. The database functionality was verified by checking whether each prediction was successfully inserted into the SQLite table and accurately retrieved through the history-viewing feature. Additionally, the model loading process was tested to ensure that the pre-trained model and scaler files are accessed correctly without retraining during runtime. The overall testing confirmed that the application performs reliably under normal and edge-case scenarios, maintains data integrity, and handles user interaction smoothly throughout execution.

Challenges Faced

During the development of the House Price Prediction System, several challenges were encountered that required careful handling and problem-solving. One of the primary challenges was working with a limited sample dataset, which made it difficult to train the model with high accuracy and required thoughtful selection of features and preprocessing techniques to avoid overfitting. Encoding the categorical location data into numerical form also posed an initial challenge, as incorrect mapping could negatively affect model predictions. Integrating the trained machine learning model with a SQLite database required additional effort to ensure that predictions were stored and retrieved correctly without causing connection or commit errors. Handling user input validation was another challenge, especially preventing invalid entries such as negative values or non-numeric inputs from disrupting program execution. Additionally, ensuring that the model and scaler were correctly loaded from external files rather than retrained during each run required careful structuring of the code. Despite these challenges, systematic debugging, iterative testing, and incremental feature improvements helped achieve a stable and functional application.

Learnings & Key Takeaways

Through the development of the House Price Prediction System, several important learnings and insights were gained. The project provided a practical understanding of how machine learning models are trained, validated, and deployed using real-world data. It reinforced the importance of data preprocessing, including feature selection,

scaling, and categorical encoding, and demonstrated how these steps directly impact model accuracy and performance. Working with a Random Forest Regressor enabled a deeper understanding of ensemble-based algorithms and their advantages in handling non-linear data without extensive parameter tuning. The integration of SQLite highlighted the value of persistent storage for maintaining prediction records and improved the ability to design applications that combine machine learning with traditional software components. The project emphasizes the need for input validation, error handling, and modular coding to create reliable and user-friendly applications. Overall, the experience strengthened skills in Python programming, model deployment, database management, and problem-solving, while demonstrating how theoretical concepts can be applied to build a fully functional AI-based system.

Future Enhancements

In the future, the House Price Prediction System can be enhanced by incorporating a larger and more diverse real-estate dataset to improve the accuracy and generalization of the model. The application may also be upgraded to include additional predictive features such as locality rating, property age, nearby facilities, and market trends to provide more realistic price estimations. Integrating advanced machine learning techniques, including Gradient Boosting or Neural Networks, can further optimize performance and reduce prediction error. A graphical user interface or web-based platform could be developed to improve accessibility and user experience, allowing users to interact with the system without relying on console inputs. The database can be expanded to support user authentication, data filtering, and export options, as well as analytics dashboards for viewing trends over time. Additionally, automating periodic model retraining with newly collected data would ensure that predictions stay relevant to changing market conditions. Finally, adding features such as API integration with real-estate listing platforms or GPS-based property lookup could transform the system into a more comprehensive and real-world-ready application.

References

1. Github
2. Wikipedia
3. https://www.kaggle.com/competitions/house-prices-advanced-regression-techniques
4. https://housingprice.streamlit.app