**Real Estate Price Prediction - Documentation**

**1. Project Overview**

This project is a **Real Estate Price Prediction** system that uses a **Machine Learning (ML) model** to predict house prices based on various input features. It consists of:

* **Frontend** (HTML, CSS, JS) for user interaction
* **Backend** (Flask API) to process user input and return predictions
* **Machine Learning Model** trained on real estate data

**2. Directory Structure**

Real Estate Price Prediction/

**├── Client/ # Frontend files**

**│ ├── app.html # UI file**

**│ ├── app.css # Stylesheet**

**│ ├── app.js # JavaScript logic**

**│ └── th.jpeg # Image asset**

**│**

**├── Model/ # Machine Learning model and data**

**│ ├── bengaluru\_house\_prices.csv # Training dataset**

**│ ├── Real\_Estate\_Prediction.ipynb # Jupyter Notebook for model training**

**│ ├── banglore\_house\_price\_prediction\_model.pickle # Trained model**

**│ ├── columns.json # Feature metadata**

**│**

**├── Server/ # Backend API (Flask)**

**│ ├── server.py # Flask app**

**│ ├── util.py # Utility functions**

**│ ├── artifacts/ # Stored model and metadata**

**│ └── \_\_pycache\_\_/ # Compiled Python files**

**│**

**└── .idea/ # PyCharm settings (optional)**

**3. Installation & Setup**

**Prerequisites**

* Python 3.x
* Flask
* Pandas, NumPy, Scikit-learn
* JavaScript-enabled browser

**Steps to Run**

**Backend (Flask API)**

1. Navigate to the **Server/** directory:

**cd Real Estate Price Prediction/Server**

1. Install dependencies:

**pip install -r requirements.txt** # If a requirements.txt exists

1. Run the Flask server:

**python server.py**

1. The server will start at **http://127.0.0.1:5000**

**Frontend (Client)**

1. Open Client/app.html in a browser.
2. Enter input values and get predictions.

**4. Model Training**

The model is developed in the Jupyter Notebook Real\_Estate\_Prediction.ipynb, following these steps:

1. **Data Loading**: The dataset (bengaluru\_house\_prices.csv) is loaded using Pandas.
2. **Data Cleaning & Preprocessing**:
   * Handle missing values.
   * Convert categorical data (location) into numerical values using one-hot encoding.
   * Remove outliers based on statistical analysis.
3. **Feature Engineering**:
   * Select relevant features such as total square footage, number of bedrooms, and location.
   * Normalize/scale numerical features.
4. **Model Selection & Training**:
   * Train a **Linear Regression** model using Scikit-Learn.
   * Test multiple models such as Decision Tree and Random Forest for comparison.
   * Optimize hyperparameters using **GridSearchCV.**
5. **Evaluation & Performance Metrics**:
   * Evaluate the model using **R-squared (R²)** and **Mean Absolute Error (MAE)**.
   * Validate with test data to ensure generalization.
6. **Model Export**:
   * Save the trained model using **Pickle (banglore\_house\_price\_prediction\_model.pickle)**.
   * Save feature names to columns.json for API reference.

**5. API Endpoints**

| **Endpoint** | **Method** | **Description** |
| --- | --- | --- |
| /predict | POST | Predicts house price based on input |
| /get\_location\_names | GET | Returns list of locations |

**6. Future Enhancements**

* Improve model accuracy
* Deploy to cloud (AWS/GCP/Heroku)
* Add more features like crime rate, school ratings, etc.