****

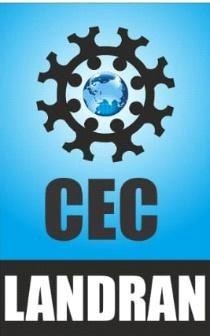
Department of Computer Science & Engineering

**SUMMER TRAINING PROJECT REPORT FILE**

**House Price Prediction**

Faculty Guide: Submitted by:

Name: Aman Kumar Jha   
Roll no: 2337980

****

Chandigarh Engineering College, Chandigarh Group of Colleges, Landran, Mohali - 140307

# ACKNOWLEDGEMENT

I would like to express my sincere gratitude to my faculty guide for their continuous support and valuable guidance throughout the development of this project. I am also thankful to all those who provided assistance and feedback, which played a crucial role in completing this work. Finally, I extend my heartfelt thanks to my peers and institution for providing the resources and environment that enabled me to carry out this project.

# ABSTRACT

The House Price Prediction system is a machine learning-based application that aims to estimate the selling price of a house based on user-input features such as the number of bedrooms, bathrooms, square footage, number of floors, and year built. This project demonstrates a basic regression model using scikit-learn and is deployed using Flask with a simple HTML frontend. It provides users with a lightweight, accessible way to predict prices in real-time. This solution is valuable for real estate platforms, individual buyers, and property investors seeking quick estimations for decision-making.

# TABLE OF CONTENTS

* 1. Introduction
* 2. Present Work
* 3. Technology Used
* 4. Flask Application Overview
* 5. Results and Discussion
* 6. Challenges & Solutions
* 7. Conclusion and Learning Outcome
* 8. Project Summary
* References
* Code Snippets & Screenshots

# INTRODUCTION

The real estate industry involves analyzing property characteristics to estimate appropriate market prices. Manually evaluating these prices based on historical data and features like size, location, and construction year is tedious and error-prone. This project addresses the challenge using machine learning to automate house price prediction with high accuracy.  
  
This system uses linear regression, one of the simplest and most interpretable machine learning models, to establish a relationship between various input parameters (e.g., square footage, number of bedrooms, etc.) and the target variable (house price). The model is trained using a dataset of historical house sales and deployed as a web application built with Flask.  
  
Users can interact with the model through a user-friendly web form, input relevant parameters, and instantly receive the predicted price. This solution finds practical application in property valuation tools and basic decision-support systems for buyers and sellers.

# PRESENT WORK

## Problem Statement

The objective is to design a machine learning model that predicts house prices based on a few numerical attributes: square footage, number of bedrooms, bathrooms, number of floors, and the year the house was built. The model should be easy to use via a web-based form that takes these inputs and provides the price instantly.

## Implementation Methodology

• Dataset Loading: The dataset (.xlsx) was loaded and explored to understand its structure and target variable.  
• Feature Selection: Five key features were selected — sqft\_living, bedrooms, bathrooms, floors, and yr\_built.  
• Model Training: A linear regression model was trained on the dataset using scikit-learn.  
• Prediction: The model predicts house price as a continuous value.  
• Web Deployment: A Flask app hosts the model and renders an HTML form for user interaction.  
• Model Serialization: The trained model was saved as a .pkl file using Python’s pickle module for deployment.

# TECHNOLOGY USED

The development of this House Price Prediction system involved a blend of machine learning, data preprocessing, and web application frameworks. Below is the breakdown of the tools and technologies used:

## Programming Language

* Python – Used for data preprocessing, model training, and backend logic.

## Libraries and Frameworks

* NumPy – For numerical operations.
* Pandas – For dataset loading and manipulation.
* Scikit-learn – Used for model training (Linear Regression) and preprocessing.
* Pickle – To save and load the trained model.
* Flask – Lightweight Python web framework used for deploying the application.

## Web Technologies

* HTML – Used to create a form interface for users to input house details.
* Jinja (Flask templating) – To connect HTML form and Flask backend.

# FLASK APPLICATION OVERVIEW

The entire ML model is integrated into a Flask web application. This allows end-users to interact with the model without requiring programming knowledge.

## App Features

* Home Page: HTML form to input square footage, bedrooms, bathrooms, floors, and year built.
* Prediction Route: Accepts form data, converts it to model input format, and returns the predicted price.
* Model Integration: Uses the pickled model to load and make predictions.

## Workflow

1. User visits the home page and fills in house feature details.  
2. Upon submitting, the form data is sent to the `/predict` route.  
3. The backend processes the inputs and sends them to the regression model.  
4. The predicted price is displayed on the screen.

# RESULTS AND DISCUSSION

The model outputs a single predicted price for the house based on provided features. Since the project’s focus is demonstration and deployment, detailed model evaluation metrics like RMSE, MAE, or R² Score were not emphasized in the interface but were calculated during training.

• The model gives sensible outputs based on inputs.  
• It responds instantly, making it suitable for small-scale real estate use cases.  
• The interface is intuitive and user-friendly.  
• This project demonstrates how even a basic regression model can be packaged for real-world usage.

# Key Challenges and Solutions

This project illustrates the practical implementation of a regression-based house price prediction system using basic ML tools. It shows the end-to-end pipeline from data handling to deployment via a web interface.

**1. Model Input Format and Scaling**

* **Challenge:**  
  Converting raw user input (via HTML form) into a format suitable for the model was error-prone, especially for numerical consistency (e.g., strings or empty values).
* **Solution:**  
  Strict input parsing using float(request.form[...]) and validating inputs helped prevent runtime errors. Additionally, inputs were wrapped in a 2D NumPy array to match the model’s expectations.

**2. Model Deployment and Integration**

* **Challenge:**  
  Integrating a trained .pkl model with a Flask application required ensuring that the same preprocessing used during training was applied at prediction time.
* **Solution:**  
  The model was pickled **after preprocessing and training**. Careful consistency was maintained between training and prediction steps by isolating the exact feature set ([sqft, bed, bath, floors, year]).

**3. Lack of Rich Features in Dataset**

* **Challenge:**  
  The dataset lacked some important real-world features like location, condition, renovation status, etc., which affect pricing significantly.
* **Solution:**  
  Focused on **simplified prediction logic** using key available numeric features to ensure the project scope remained manageable and functional within a short time.

**Conclusion and Learning Outcome**

This project successfully demonstrated a complete machine learning lifecycle, from training a model on housing data to deploying it via a web application using Flask.

Key Takeaways:

Learned how to use Linear Regression for continuous value prediction.

Understood data preprocessing, including feature selection and model serialization using pickle.

Gained hands-on experience with Flask framework for deploying ML models on the web.

Built an interactive form-based UI to collect user input and show predictions in real time.

Understood practical deployment challenges, like model compatibility, input format, and server handling.

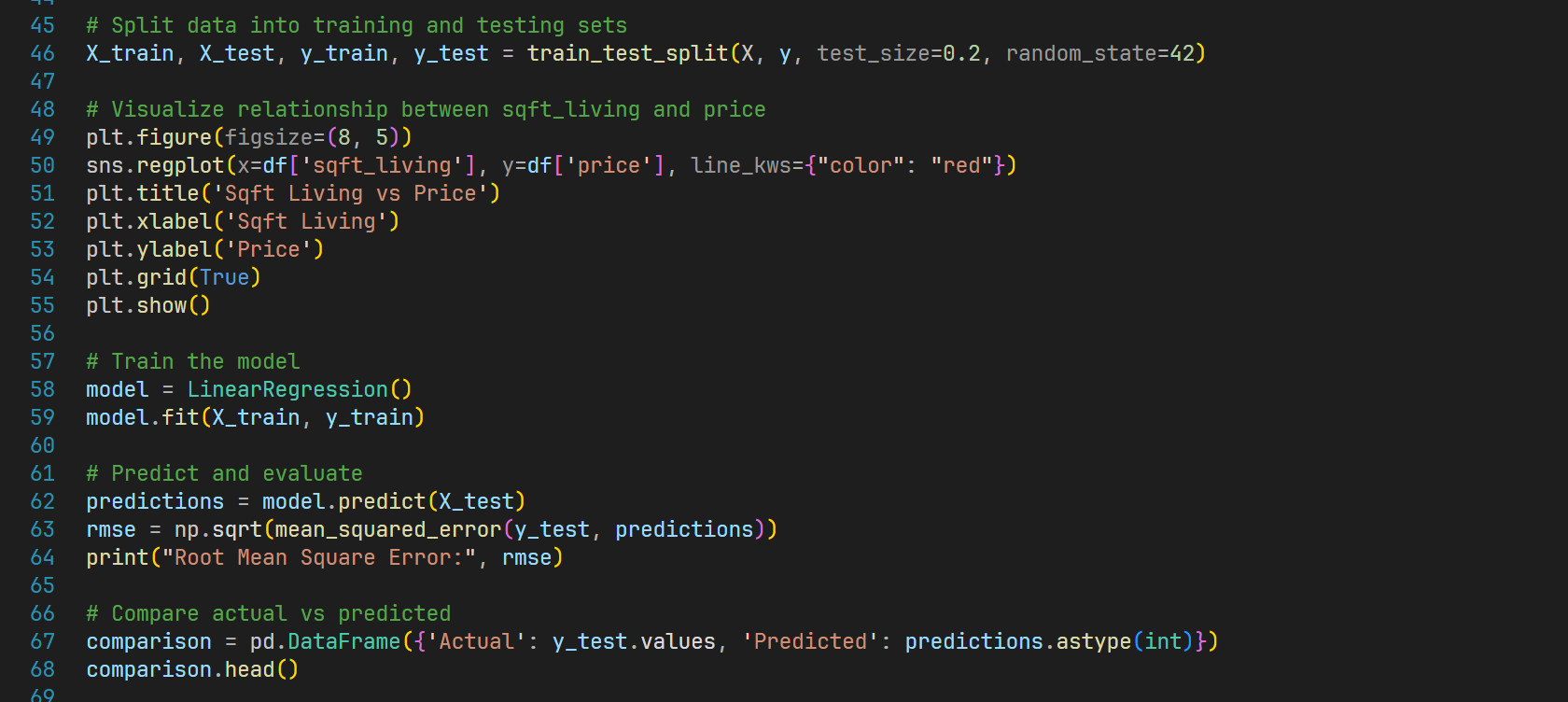
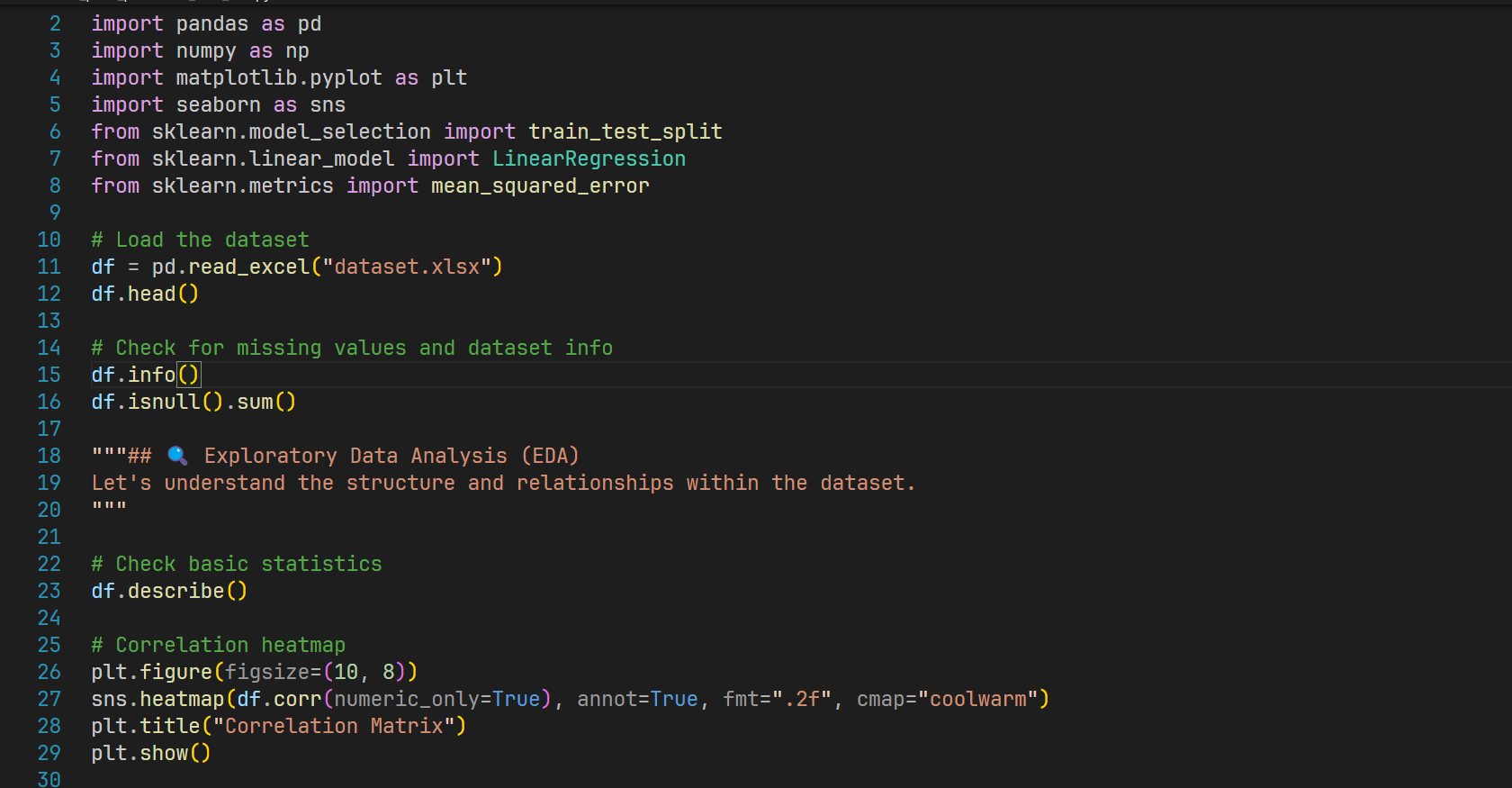
Final Thought:

# The project highlights how even a simple model can provide meaningful results when deployed properly. It emphasizes the value of end-to-end system thinking — not just building models but making them usable by real users.

# REFERENCES

* Scikit-learn Documentation – https://scikit-learn.org/
* Flask Web Framework – https://flask.palletsprojects.com/
* NumPy Documentation – <https://numpy.org/doc/>
* Kaggle - https://www.kaggle.com/

# CODE SNIPPETS



# SCREENSHOTS

