MACHINE LEARNING BASED ALGORITHMS FOR HOUSE PRICE PREDICTION IN REAL ESTATE

BY

SAHID CLEMENT

PSC2105406

DEPARTMENT OF COMPUTER SCIENCE,

FACULTY OF PHYSICAL SCIENCES

UNIVERSITY OF BENIN,

BENIN CITY,

EDO STATE, NIGERIA.

NOVEMBER 2025

MACHINE LEARNING FOR HOUSE PRICE PREDICTION IN REAL ESTATE

BY

SAHID CLEMENT

PSC2105406

A PROJECT REPORT SUBMITTED TO THE DEPARTMENT OF COMPUTER SCIENCE, FACULTY OF PHYSICAL SCIENCES, UNIVERSITY OF BENIN, BENIN CITY.

IN PARTIAL FUFILLMENT OF THE REQUIREMENT FOR THE AWARD OF A BACHELOR OF SCIENCE (B.Sc.) DEGREE IN COMPUTER SCIENCE

NOVEMBER 2025

CERTIFICATION

This is to certify that this project work was carried out by **SAHID CLEMENT** with Matriculation Number **PSC2105406** under my supervision. It is adequate and satisfactory, both in scope and content, for the award of Bachelor of Science (B.sc) Degree in Computer Science of the University of Benin

Dr. E.C. IGODAN **DATE**

Project Supervisor

APPROVAL

This project work is hereby approved in partial fulfillment of the requirements for the award of Bachelor of Science (B.Sc.) Degree in Computer Science from the University of Benin.

DR. (MRS.) A.R. USIOBAIFO
Head of Department

DATE

DEDICATION

This project is dedicated to God Almighty for granting me the strength, wisdom, and guidance to successfully complete it and for His support throughout my time at the University of Benin (UNIBEN). I also dedicate this work to my mother**, Esther Izoya,** as well as my Aunty, **Mrs. Ezewele,** and **my extended family** for their unwavering love, encouragement, and support throughout my academic journey.

ACKNOWLEDGEMENT

My utmost acknowledgement goes to God Almighty for giving me the strength, wisdom and direction throughout my academic journey. I would like to express my gratitude to my project supervisor Mr. Igodan E.C. for his consistent guidance towards ensuring the successful completion of this project.

To The dean (Faculty of Computing), Dr. (Mrs.) R. A. Usiobaifo (Head of Department), Prof. (Mrs.) F.A. Egbokhare, Prof. A. A. IMIANVAN, Prof. G. O. Ekuobase, Prof. (Mrs.) A. O. Egwali, Prof. F. I. Amadin, Prof. F. A. U. Imouokhome, Prof. (Mrs.) S. Konyeha, Prof. (Mrs.) V. I. Osubor, Prof. F. O. Chete, Dr. E. Nwelih, Dr. (Mrs.) G. O. Aziken, Mr. E. E. Obasohan, Dr. F. O. Oliha, Dr. (Mrs.) R. O. Osaseri, Dr. M. Osagie, Mr D.N Idehen, Mr. K. O. Otokiti, Mr. I. E. Obayagbona, Miss. L. O. Usiosefe, Mr. J. O. Okhuoya, Mr. G. I. Evbuomwan and the non-teaching staff, for your various roles and support. I acknowledge your contributions with appreciation.

I also want to appreciate those who have impacted me positively: (Amadasun Samuel) and more. I would also like to thank my family and friends for their support, words of encouragement, and consistent guidance throughout this project.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

The process of estimating property prices in Nigeria remains largely manual, subjective, and inconsistent, as real estate valuation often depends on personal experience, incomplete records, or unstructured web data. These limitations create inaccuracies that affect buyers, sellers, investors, and policymakers. To address these challenges, this study aims to develop a more reliable and data-driven method for house price prediction using machine learning techniques. The objectives include collecting and preprocessing a multi-state Nigerian dataset, engineering relevant features, developing multiple predictive models, and evaluating their performance to identify the most accurate approach.

A dataset consisting of 28,915 property records scraped from PropertyPro was cleaned, normalized, and transformed through techniques such as one-hot encoding and Mutual Information Regression. Four machine learning models, which are Linear Regression, Decision Tree, Random Forest, and XGBoost, were implemented and evaluated using MAE, MSE, and R^2.

Quantitatively, the models achieved R^2 scores of 0.475 (Linear Regression), 0.433 (Decision Tree), 0.517 (XGBoost), and 0.520 (Random Forest). Qualitatively, Random Forest demonstrated superior ability to capture nonlinear patterns, handle noisy data, and generalize across diverse property locations. The best-performing model was deployed through a Streamlit web application to enable real-time house price prediction for end users.

Despite its success, the study is limited by the absence of macroeconomic variables such as inflation and interest rates, which play a substantial role in housing market dynamics. Additionally, challenges in obtaining comprehensive and well-structured real estate data across all Nigerian states constrained the breadth of the analysis. Future research should integrate geospatial and macroeconomic datasets to further enhance prediction accuracy and model robustness.

CHAPTER ONE

INTRODUCTION

1.0 BACKGROUND OF THE STUDY

Housing is universally recognized as one of the most essential needs of humanity, serving as both shelter and a form of investment that drives economic growth (UN-Habitat, 2020). Globally, the real estate sector contributes significantly to national economies, with strong ties to industries such as construction, banking, and insurance (World Bank, 2019). Accurate property valuation is therefore critical for ensuring transparency and stability in housing markets.

In Nigeria, housing demand has continued to grow rapidly due to urbanization, rising population, and changing socio-economic dynamics (National Bureau of Statistics, 2021). However, this demand varies across states. Lagos and Abuja have more structured real estate markets, while states like Edo and Delta experience fragmented and poorly documented property data (Adebayo & Ojo, 2018). This imbalance limits the development of predictive models that can work effectively nationwide.

Traditional property valuation methods in Nigeria have largely relied on manual appraisal and statistical techniques such as Multiple Linear Regression (MLR). While useful, these methods assume simple linear relationships between features and price (Olawale, 2017). In practice, housing prices are influenced by complex nonlinear factors, including property characteristics (e.g., bedrooms, bathrooms, square footage), spatial variables (e.g., distance to schools, city centers), and macroeconomic drivers (e.g., inflation, mortgage rates) (Adegbite et al., 2019).

Machine Learning (ML) has emerged as a superior approach, as it can model nonlinear interactions and handle large, complex datasets (Zhang & Zhou, 2020). Ensemble algorithms such as Random Forest, Gradient Boosting, XGBoost, LightGBM, and CatBoost have consistently outperformed linear regression in housing price prediction (Chen & Guestrin, 2016; Ke et al., 2017). For example, widely studied datasets such as the Ames Housing Dataset and the

Boston Housing Dataset have demonstrated the effectiveness of ML in capturing complex housing price dynamics (Hemlata et al, 2024; Ibrahim et al, 2025).

However, most of these studies are based on housing markets in developed countries, where datasets are structured and readily available. In contrast, Nigerian studies have often used small or localized datasets, such as those from Lagos alone (Eze et al., 2020). As a result, their findings are not generalizable to the broader Nigerian housing market. This gap highlights the need for applying machine learning techniques to larger, Nigeria-wide datasets that include multiple states such as Edo, thereby improving both accuracy and generalizability.

1.1 STATEMENT OF THE PROBLEM

Accurately predicting house prices in Nigeria remains a challenge due to fragmented and inconsistent datasets (Okafor & Abdul, 2021). Existing research has often been limited to specific states, with studies such as (Sunday et al, 2024) focusing only on Lagos. This makes the findings less generalizable to the wider Nigerian housing market.

Furthermore, traditional valuation methods are subjective and often inconsistent, while many Nigerian-focused studies fail to include spatial or macroeconomic features, despite their importance in real estate pricing (Ibrahim et al., 2019).

This study addresses these challenges by:

1. Expanding datasets to cover multiple states, including Edo.
2. Applying advanced machine learning models capable of capturing nonlinear data.
3. Comparing model performance across states to identify generalizable solutions.

Without these interventions, buyers, sellers, and financial institutions will continue to face unreliable property valuations, limiting transparency and growth in Nigeria's real estate sector.

1.2 AIM AND OBJECTIVES

The aim of this study is to develop and evaluate a robust machine learning model for house price prediction using a multi-state Nigerian dataset, with the goal of improving accuracy and generalizability compared to conventional methods.

Objectives:

The specific objectives of the study are:

1. To collect and preprocess real estate datasets.
2. To implement feature engineering techniques using One-hot encoding and Mutual Info Regression.
3. To develop on (1) above multiple machine learning models using Linear Regression, Random Forest, XGBoost, LightGBM, and CatBoost.
4. To evaluate the performance of (3) above using standard metrics such as Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and R^2.
5. To benchmark model performances in identifying the best model.

1.3 METHODOLOGY

This study uses a dataset scraped from PropertyPro, a Nigerian real estate platform that provides property listings across multiple states. The dataset includes 28,915 rows covering diverse property types and locations, making it suitable for robust predictive modeling.

a) Dataset Characteristics

I. Number of features – 10

II. Total number of instances – 28,915.

III. Characteristics of the feature types –

i. Title: Text
ii. House Characteristics (BQ/Not BQ): Text
iii. House Characteristics (Detached/Furnished apartment): Text
iv. Location: Text
v. State: Text
vi. Property ID: Text
vii. Bedrooms: Number/Numerical
viii. Bathrooms: Number
ix. Toilets: Number
x. Pool: Binary.

b) Data Preprocessing

The following feature preprocessing stages used in this project work includes:

I. **Data cleaning** –

i. Handling missing values using Pandas method (.dropna()).
ii. Removing duplicates using Pandas method (.drop_duplicates()).
iii. Identifying and Handling Outliers by using Matplotlib/Seaborn libraries, and Interquartile range respectively.

II. **Encoding** using one-hot encoding

III. **Normalization** using standard scaler techniques.

c) Feature Selection

Using Mutual Info Regression Technique.

d) Modeling Techniques

The ML models adopted in this study includes: Linear Regression, Random Forest, XGBoost, LightGBM, and CATBoost

e) Performance Metrics

The following metrics adopted in this study involves: Mean Absolute Error (MAE), Mean Squared Error (MSE), and R^2 (Coefficient of Determination).

1.4 SCOPE OF THE STUDY

This study focuses on property data scraped from PropertyPro, covering multiple Nigerian states, including Edo. The models used in this study are baseline models, comprising of just Linear Regression, and Ensemble models, comprising of Random Forest, XGBoost, LightGBM, and CATBoost, and does not extend to deep learning models.

The evaluation metrics deployed in this study to identify a model best suitable for predicting house prices are MAE, MSE, and R^2.

1.5 EXPECTED CONTRIBUTION TO KNOWLEDGE

This project work is expected to:

1. Demonstrates the application of advanced ensemble machine learning models on a Nigeria-wide dataset.
2. Provides a comparative analysis of traditional regression methods and modern ensemble approaches.
3. Highlights the role of feature selection in improving model performance.

4. Offers practical tools that can support buyers, sellers, investors, and policymakers in making data-driven real estate decisions.

CHAPTER TWO

LITERATURE REVIEW

2.0 INTRODUCTION

Real estate is one of the most significant sectors of any economy, encompassing land, buildings, and the resources tied to them, including ownership and usage rights. It plays a central role in wealth creation, investment, and human welfare by providing housing, workplaces, and infrastructure for commerce and industry. The real estate market is often divided into categories such as residential, commercial, industrial, and agricultural properties. Each category has unique features that influence its value, but they all share a dependence on structural, locational, and economic conditions.

In Nigeria, the real estate sector represents a major driver of economic growth and employment. Urban centers such as Lagos, Abuja, and Port Harcourt have seen rapid property development fueled by population growth, migration, and infrastructural expansion. However, the market is not without challenges. Issues such as underdeveloped mortgage systems, rising construction costs, regulatory bottlenecks, and limited housing supply have constrained the sector's ability to meet demand. These problems are compounded by unreliable property transaction records, which often reduce transparency in pricing and valuation (Yakub et al., 2022).

The Nigerian housing deficit highlights the urgent need for effective investment and valuation strategies. Estimates suggest that millions of Nigerians lack adequate housing, and even in cities with ongoing development, affordability remains a challenge. The interaction between supply, demand, and affordability demonstrates why accurate price prediction is crucial. For property owners and buyers, reliable valuation ensures fair transactions; for developers and investors, it helps assess profitability and risks; and for policymakers, it supports the design of responsive housing policies.

Pricing in real estate is inherently complex. Traditional valuation techniques such as hedonic pricing and regression analysis aim to break down property prices into measurable components like size, location, and condition. While useful, these approaches assume linear relationships and

often fail to capture the multidimensional realities of property markets. The increasing heterogeneity of real estate data has exposed the limitations of these models in explaining price variations, especially when dealing with nonlinear or interacting variables.

The global rise of computational techniques, particularly machine learning, has opened new possibilities for predicting real estate prices more accurately. These models handle nonlinear interactions, integrate diverse data types, and process larger datasets. Studies both globally and in Nigeria suggest that machine learning models outperform traditional regression in terms of predictive accuracy, especially when locational and macroeconomic factors are integrated with structural variables (Asogwa et al., 2023). This transition illustrates the importance of rethinking real estate valuation beyond conventional tools to embrace methods capable of addressing the market's complexity.

2.1 FACTORS AFFECTING REAL ESTATE

The value of real estate is determined by a combination of structural, locational, neighborhood, demographic, and macroeconomic factors. Understanding these determinants is crucial not only for buyers and sellers but also for policymakers, developers, and researchers seeking to analyze housing markets.

1. **Structural Factors**

   These are the physical characteristics of a property. Features such as the number of bedrooms, bathrooms, floor area, garage capacity, and overall condition strongly influence value. For instance, a larger floor area generally raises property prices, while the age of a building may reduce its value unless it is historically significant or well-maintained. Abdullah et al. (2021) found that structural features such as gross living area (GrLivArea) and the number of rooms were among the strongest predictors of housing prices in machine learning models.

2. **Locational Factors**

   Location remains one of the most important determinants of property value. Proximity to amenities such as schools, healthcare facilities, business districts, and transportation hubs often increases desirability. Properties in neighborhoods with reliable access to infrastructure such as good roads, electricity, and water supply command higher prices than similar properties in poorly serviced areas. Chinxli (2022) highlighted how

locational attributes, especially accessibility to commercial centers, had stronger predictive power than some structural features in urban housing datasets.

3. **Neighborhood and Environmental Factors**

   The quality of the neighborhood, including safety, noise levels, pollution, and zoning regulations, also plays a vital role in real estate valuation. Buyers are often willing to pay more for properties located in quiet, secure environments compared to those in congested or unsafe neighborhoods. Environmental sustainability factors, such as air quality and green spaces, have also become increasingly relevant. MayorLaz (2021) observed that neighborhood effects sometimes outweigh structural attributes in influencing house prices in Nigerian cities.

4. **Demographic and Socioeconomic Factors**

   Population growth, household income levels, and employment opportunities significantly affect demand for housing. High-income neighborhoods tend to have higher property values due to increased purchasing power. In contrast, areas with lower income levels may face stagnating or declining prices despite physical development. Faris et al. (2021) demonstrated that integrating demographic factors with structural attributes improved the performance of predictive models, reflecting the interplay between social and economic dynamics in determining property values.

5. **Macroeconomic Factors**

   Broader economic conditions such as inflation, interest rates, currency fluctuations, and GDP growth influence real estate markets by shaping demand and affordability. For instance, high inflation or rising interest rates reduce borrowing capacity, which can slow housing demand and lower prices. Elsevier (2023) emphasized that including macroeconomic indicators in predictive models enhanced accuracy by capturing shifts in market conditions that structural and locational features alone could not explain.

2.2 INTRODUCTION TO ARTIFICIAL INTELLIGENCE

Artificial Intelligence (AI) is broadly defined as the simulation of human intelligence in machines that are programmed to think, learn, and make decisions. The field emerged in the 1950s, pioneered by researchers such as Alan Turing and John McCarthy, who first coined the term “Artificial Intelligence” in 1956 at the Dartmouth Conference (Russell & Norvig, 2021). Early AI systems were symbolic and rule-based, relying on explicitly coded instructions to

mimic human reasoning. However, they were limited by computational power and the difficulty of encoding complex real-world knowledge into rigid rules.

Over the years, AI has advanced significantly due to improvements in algorithms, computational hardware, and data availability. The development of machine learning, in particular, transformed AI by enabling machines to learn from data rather than depending solely on human-programmed instructions. The rise of deep learning in the 2010s, powered by neural networks with multiple layers, further expanded AI's capabilities in fields such as image recognition, natural language processing, and predictive analytics (Goodfellow et al., 2016). These advancements have made AI one of the most transformative technologies of the 21st century.

AI is now applied in a wide range of industries, reshaping economies and daily life. For instance, it powers recommendation engines used by platforms such as Netflix and Amazon, enhances fraud detection in banking, supports medical diagnoses through imaging technologies, and drives automation in manufacturing. In real estate, AI is increasingly being used for property valuation, predictive modeling, and investment analysis. These applications leverage AI's ability to analyze large datasets, uncover hidden patterns, and generate accurate predictions (Jha & Shrestha, 2021).

2.2.1 Applications Of AI Across Sectors

AI has found widespread application in numerous areas. Some of the most significant include:

1. **Healthcare**
   AI assists in medical imaging, drug discovery, patient monitoring, and personalized treatment recommendations. Deep learning algorithms have achieved high accuracy in detecting diseases such as cancer from scans and X-rays.
2. **Finance**
   Banks and financial institutions use AI for fraud detection, algorithmic trading, credit scoring, and customer service through chatbots.
3. **Transportation**
   AI is the driving force behind autonomous vehicles, intelligent traffic systems, and predictive maintenance in logistics.

4. **Agriculture**

   AI is applied in precision farming, crop disease detection, weather forecasting, and supply chain optimization, helping farmers improve yield and efficiency.

5. **Education**

   Intelligent tutoring systems and adaptive learning platforms use AI to personalize education based on student performance and learning styles.

6. **Manufacturing**

   Robotics powered by AI enhance assembly lines, perform quality control, and improve production efficiency.

7. **Real Estate**

   AI enables property valuation, predictive analytics, fraud detection, tenant screening, and smart property management systems.

2.2.2 Future Of AI

The future of AI promises even greater integration into society and industry. Emerging trends include the expansion of explainable AI (XAI), which aims to make AI systems more transparent and interpretable; general AI, which aspires to build machines capable of performing a wide range of tasks with human-like flexibility; and the integration of AI with Internet of Things (IoT) for smart cities and homes. In real estate, the future points toward AI systems capable of incorporating geospatial data, satellite imagery, and real-time market information to provide more accurate price predictions and investment insights. Ethical considerations, such as bias, privacy, and employment disruption, will also shape how AI is developed and deployed in the coming decades (Kaplan & Haenlein, 2020).

2.2.3 Core Components Of AI

AI is a broad discipline with several core components, each representing a different way machines can simulate aspects of human intelligence:

1. **Machine Learning (ML):** Enables systems to automatically learn from data and improve performance over time without being explicitly programmed. Applications include predictive analytics, recommendation engines, and fraud detection.

2. **Robotics:** Combines AI with mechanical systems to create machines capable of performing tasks in manufacturing, healthcare, exploration, and household settings.
3. **Natural Language Processing (NLP):** Focuses on enabling machines to understand, interpret, and generate human language, powering applications like chatbots, language translation, and virtual assistants.
4. **Computer Vision:** Allows machines to interpret and analyze images and videos, with applications in medical diagnostics, surveillance, and autonomous vehicles.
5. **Expert Systems:** AI programs that mimic the decision-making process of human experts by applying rule-based knowledge to solve domain-specific problems.
6. **Neural Networks and Deep Learning:** Inspired by the structure of the human brain, these models process large datasets and recognize complex patterns, making them central to breakthroughs in speech recognition, image analysis, and predictive modeling.

2.3 MACHINE LEARNING

Machine Learning (ML) is a subset of Artificial Intelligence that focuses on developing algorithms capable of learning patterns from data and making predictions or decisions without being explicitly programmed. The term was first popularized by Arthur Samuel in 1959, who described it as the field of study that gives computers the ability to learn without being explicitly programmed (Samuel, 1959).

Over the years, ML has evolved from simple linear regression models to sophisticated deep learning algorithms capable of handling unstructured and complex datasets. Its progress has been fueled by the increase in data availability, advancements in computational power, and the development of efficient algorithms (Jordan & Mitchell, 2015). In modern practice, ML serves as the foundation for predictive analytics, enabling industries to gain insights, automate decision-making, and improve accuracy in forecasting.

2.3.1 Types Of Machine Learning

1. **Supervised Learning**

   Supervised learning is the most common type of machine learning, where the algorithm is trained on a labeled dataset. In this setup, the input features and their corresponding

outputs are provided, enabling the model to learn the mapping function between them. The goal is for the algorithm to generalize from the training data so it can accurately predict outcomes for unseen data.

The types of supervised models include Linear Regression, Logistic Regression, Decision Trees, Random Forests, Support Vector Machines (SVM), and Gradient Boosting methods. Each of these models applies different approaches in finding relationships between inputs and outputs. For instance, regression models predict continuous values like house prices, while classification models predict discrete outcomes such as whether a property is "affordable" or "expensive."

A. Linear Regression

Goal: Predict a continuous numerical value (like house price, temperature, or sales).

Concept: Linear Regression assumes a **linear relationship** between input features (X) and output (Y).

It fits a straight line (in higher dimensions, a hyperplane) that best predicts Y from X.

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_n X_n + \varepsilon \quad \ldots(2.1)$$

β_0: intercept (where the line crosses the Y-axis)

β_i: coefficients (how much Y changes when X changes

ε: error (difference between predicted and actual values)**.**

B. Decision Tree

Goal: Predict either a **continuous value** (regression tree) or a **class** (classification tree).

Concept: A Decision Tree splits the data into branches based on feature values. Each split tries to make the resulting groups as **pure** as possible (similar outputs). At the end, each leaf node gives a prediction.

C. Random Forest

Goal: Improve Decision Trees using Ensemble Learning.

Concept: A Random Forest builds many decision trees, each trained on a random subset of Data (rows) and Features (columns).

Then, it **averages** (for regression) or **votes** (for classification) over all trees.

This reduces overfitting and improves accuracy.

$$\text{Prediction} = \frac{1}{N} \sum_{i=1}^{N} \text{Tree}_i(\text{X}) \quad \ldots(2.2)$$

D. **XGBoost (Extreme Gradient Boosting)**

Goal: Build a **strong model** by combining many **weak models** (usually small trees).

Concept: Instead of training all trees independently (like Random Forest), XGBoost builds trees **sequentially**.

Each new tree tries to **correct the errors** made by the previous ones, which is **boosting**.

$$\text{Prediction} = \sum_{i=1}^{N} \text{Tree}_i(X) \quad \ldots(2.3)$$

Each new tree minimizes a loss function (like mean squared error) using **gradient descent**, hence *Gradient Boosting*.

2. Unsupervised Learning

Unsupervised learning deals with data that has no labeled outputs. The algorithm attempts to uncover hidden structures, groupings, or relationships within the dataset. It is particularly useful for exploratory data analysis and when the goal is to discover natural patterns without predefined categories.

The types of unsupervised models include Clustering algorithms (such as K-Means and Hierarchical Clustering), which group data points into clusters, and Dimensionality Reduction techniques (such as Principal Component Analysis, PCA), which simplify data while retaining essential features.

Examples in real estate include clustering houses into different neighborhoods based on their characteristics (e.g., price range, structural size, or location) or reducing a large set of housing features into a smaller, more manageable subset for analysis. This helps

investors or policymakers quickly identify market segments and emerging trends (Law et al., 2019).

3. **Reinforcement Learning**

   Reinforcement learning is a type of ML where an agent interacts with an environment and learns by receiving rewards or penalties for its actions. The goal is to learn an optimal policy that maximizes cumulative rewards over time. Unlike supervised learning, reinforcement learning does not rely on labeled input-output pairs; instead, it focuses on sequential decision-making.

   The types of reinforcement learning approaches include value-based methods (like Q-learning), policy-based methods, and actor-critic methods, each differing in how they estimate the best actions to take in an environment.

   Examples of reinforcement learning in real estate are less common but emerging. They include dynamic pricing systems, where algorithms adjust rent or property prices based on demand, or investment portfolio optimization, where agents learn strategies for maximizing returns across multiple properties.

2.3.2 Applications Of Machine Learning In Real Estate

Machine learning has had a transformative effect on real estate, particularly in the area of price prediction. Traditional valuation methods often relied on manual appraisal, which could be subjective and inconsistent. ML, by contrast, leverages large datasets to generate more accurate and objective predictions.

1. **House Price Prediction:** Algorithms like regression, random forests, and gradient boosting have been applied to predict property values with high accuracy, using features such as square footage, location, and age of the property (Chen et al., 2017).
2. **Market Trend Analysis:** ML models identify patterns in market demand and supply, allowing investors and policymakers to anticipate price fluctuations.
3. **Risk Assessment:** In mortgage lending, ML can assess borrower risk by analyzing diverse datasets such as credit history, income, and loan performance trends.

4. **Spatial Analysis:** By integrating geographic information system (GIS) data, ML enhances understanding of location-based effects on property prices (Law et al., 2019).

2.3.3 Significance Of Machine Learning In House Price Prediction

Machine learning provides several advantages over traditional methods of valuation and prediction:

a) It can handle large-scale and high-dimensional data, capturing interactions between variables that may not be apparent to human analysts.
b) It improves prediction accuracy, especially when advanced ensemble models like Random Forests and Gradient Boosting are used.
c) It reduces subjectivity, ensuring consistent valuation across diverse datasets.
d) It enables continuous learning, meaning models can adapt as new data becomes available.

As a result, ML has become the central tool for real estate price prediction, offering robust, scalable, and data-driven approaches to solving valuation challenges in housing markets.

2.4 RELATED WORKS

The following are some of the works related to this study:

1. **Sergey et al., (2024)** developed a model to enhance the accuracy of predicting rental prices of real estate using the Houston dataset from the Redfin website, which consisted of 9,260 objects and 10 features. The models used were RFECV combined with Decision Tree (DT) and Optuna with LightGBM. Evaluation metrics included MSE, RMSE, R^2, and MDAE, while SHAP was applied for result interpretation. The study revealed that adding textual features reduced the Mean Squared Error (MSE) by 13.4%, improving model suitability, though it exhibited lower accuracy for non-textual features.

2. **Haiyi., (2024)** constructed a predictive model using multiple machine learning techniques on the Seattle dataset, Washington. Models such as multiple linear regression, polynomial regression, and KNN were employed and evaluated with RMSE, MSE, and R^2 metrics.

The polynomial and KNN models showed superior performance in predicting house prices; however, the dataset's regional specificity limited its generalizability to other areas.

3. **Quang et al., (2020)** investigated differences among advanced models for housing price prediction using a Beijing housing dataset containing 300,000 data points and 26 variables. The models implemented included RF, XGBoost, LightGBM, hybrid regression, and stacked generalization, assessed via RMSLE. Results showed that hybrid and stacked models achieved satisfactory accuracy but required caution due to architectural complexity and overfitting risks.
4. **Kalidass et al. (2024)** proposed an ensemble learning approach combining Random Forest Regressor (RFR) and Gradient Boosting (GB) using a Kaggle-sourced dataset, with 80% allocated for training and 20% for testing. The models were assessed with MSE, RMSE, MAE, and R^2 metrics, achieving high performance with R^2 values between 0.869 and 0.878. Although effective, the study was limited by dataset scope and comparison within only three models.

5. **Xiangjun (2025)** conducted a comparative study of various ML models for structural housing price prediction using a Kaggle dataset. Linear Regression (LR), Random Forest (RF), and Decision Tree (DT) were tested, and their performance measured using MSE. Results revealed that RF slightly outperformed LR and DT due to its robustness against overfitting, though findings were limited to current market trends.

6. **Sunday et al. (2024)** predicted house rental prices in Lagos, Nigeria, by analyzing data scraped from PropertyPro, consisting of 12,269 rows and nine columns. Models such as RFR, RR, SVR, GBR, and LR were evaluated using MAE, RMSE, and R^2. Random Forest yielded the best results with the lowest MAE and highest R^2, identifying the number of bedrooms and apartment location as key predictors. The study's limitation lay in its narrow regional focus and lack of property-type diversity.

7. **Shuzlina et al. (2021)** proposed advanced ML models for house price prediction using the Kuala Lumpur Property Listing dataset. Models included LightGBM, XGBoost, RR,

and MLR, evaluated with MAE, MSE, RMSE, and adjusted R^2 metrics. XGBoost achieved the lowest MAE and RMSE, outperforming others; however, the model was constrained to Kuala Lumpur's market dynamics, reducing its general applicability.

8. **Kevin and Hieu (2022)** developed a predictive ML model using Redfin sales data (2018–2022) to assess property price trends post-COVID-19. Techniques applied included LR, SVR, DT, RF, and XGBoost, and evaluated with MAE, RMSE, and R^2. XGBoost again outperformed others, identifying total rooms, bathrooms, and tax amount as critical predictors. Nonetheless, the model was affected by variability in post-pandemic pricing and regional market restrictions.

9. **Hemlata et al. (2024)** addressed housing price prediction as a regression task using the Ames City dataset (2,930 samples, 82 variables). Models compared were XGBoost, SVR, RFR, MLR, and MLP, with performance metrics including R^2, adjusted R^2, MSE, RMSE, MAE, and CV. XGBoost achieved the highest CV score (88.94%) and explained up to 91% of price variability, but the study's dataset limitation to Ames reduced its external validity.

10. **Lingjie (2023)** provided a comprehensive overview of ML models predicting house prices using the Ames dataset (1,460 rows, 81 columns). Data cleaning (imputation) and transformation (one-hot encoding) were employed before testing regression tree, XGBoost, GB, and LightGBM models. XGBoost and LightGBM proved most accurate, though the study's reliance on a single metric (MAE) and missing values posed limitations.

11. **Ibrahim et al. (2025)** evaluated house price prediction using modern techniques with the Kaggle Boston dataset (506 entries, 14 features). Random Forest and Extra Tree models were compared using MSE and MAE. The Extra Tree model performed better, recording lower MSE and MAE values. However, the comparison's narrow scope (only two models) limited the study's comprehensiveness.

12. **Chengke (2023)** built a predictive housing price model using Python's XML Path Language on the HomeLink dataset from Jinan, China, employing LR, RF, and CatBoost algorithms. Metrics used were RMSE, R^2, and MAE. CatBoost demonstrated superior accuracy (78–91%), outperforming other models, though imbalanced data and limited regional applicability constrained generalization.

Table 2.1: Reviewed Literature

Authors	Objectives	Methodology	Results	Limitations
Xu and Nguyen., (2022)	A predictive ML model for property prices. Evaluate proposed model in a volatile market space. Identify trends in price.	Sales data from 2018-2022 were scraped from Redfin dataset. Model used are - LR, SVR, DT, RF, XGBoost. Metrics - MAE, RMSE, R^2. SHAP was applied for result interpretation.	Study achieved – RMSE ≈ 70892.48 – 106626.7. MAE ≈ 41331.89 – 71994.52. R^2 ≈ 0.71 – 0.89. XGBoost performed better than others in the volatile Post-Covid-19 market. Tax annual amount, number of bathrooms, number of rooms and square footage, were identified through SHAP as the variables used in determining a prediction.	Variability in pricing. Not applicable to dataset characterized by basement style or flooring in rooms. The analysis is limited to post-covid-19 market.
Sharma., (2024)	To address house price prediction problem as a regression task. Employ various ML techniques that expresses independent	Dataset from Ames City in Iowa, USA were used, consisting of 2930 (houses) with 82 variables. The Model compared are – XGBoost, SVR, RFR, MLR, MLP.	Study achieved – R^2 ≈ 0.570 - 0.920. Adj.R^2 ≈ 0.526 - 0.911. MSE ≈ 0.015 - 0.079. RMSE ≈ 0.112 - 0.282. MAE ≈ 0.075 - 0.211.	Dataset was limited to Ames City in lowa, USA. Dataset from other Cities might not produce same results.

	variables.	Metrics – R^2, Adj.R^2, MSE, RMSE, MAE, CV.	C V Score ≈ 4.000 - 88.940. XGBoost is the most suitable regression technique, with the highest CV score of 88.940. Overall house quality, ground floor of living area, garage cars, and total basement sq.ft, were features used in determining house price.	Extensive computational time as a result of obtaining high-volume data.
Fang., (2023)	To give an overview of various machine learning models that can predict housing prices.	Dataset from Ames, Iowa in the USA from 2006-2010, with a size of 1460 rows and 81 columns, containing 79 variables, were used. Data cleaning (imputation) and data transformation (one-hot encoding) are the two preprocessing techniques used. Models of regression tree, RT, XGBoost, GB, and LightGBM were used. Metrics – MAE.	Study achieved – MAE ≈ 16893 – 24632. Missing values were handled by imputation, while columns with categorical data were handled using one-hot encoding. Most accurate ML models are XGBoost, GB, and LightGBM with MAE values of 16370, 16893 and 16898 respectively.	Too many missing values from the dataset. Only one metric was used.
Ibrahim et al., (2025)	Evaluating house price prediction using more recent methods.	Kaggle Boston housing dataset with 506 entries and 14 features was employed.	Study achieved – Metrics for RF model : MSE ≈ 8.490417. MAE ≈ 2.225197.	Only two models were compared.

	To compare between a random forest regression model and the proposed prediction model	The new model was developed using Extra Tree algorithm. ETR was used in implementing the algorithm. Another model used is RFR. Metrics – MSE and MAE.	Metrics for ET model : MSE ≈ 7.485059. MAE ≈ 2.053480. Lower values indicate better performance hence, the proposed model using ET algorithm performed best.	
Zou., (2023)	To build house price prediction model based on housing conditions.	Using XML Path Language and Python to gather the Jinan real estate market dataset from the HomeLink website containing 15 features. Models used are LR, RF, and Catboost. Metrics employed are RMSE, R^2, and MAE.	Study achieved – Accuracy ≈ 78.45% - 91.39%. RMSE ≈ 772.408 - 1973.054 R^2 ≈ 0.779 - 0.913 MAE ≈ 17.506 - 30.681. Catboost algorithm has the lowest RMSE and highest accuracy.	Many features in the dataset are imbalanced. These imbalanced data impact the RF algorithm and decrease its prediction accuracy. The model is only restricted to Jinan for house price estimate. Algorithms are not able to utilize unmeasurable features in the model-building process.
Truong et al., (2020)	To investigate the difference among several advanced model. Validating multiple techniques in implementing	Beijing housing price dataset was used, containing 300,000 data with 26 variables from 2009-2018. Models used are – RF, XGBoost,	Study achieved – RMSLE For Trainset ≈ 0.12980 – 0.16687. For Testset ≈ 0.16350 – 0.16944. Hybrid and Stacked	Stacked Generalization Regression fails to give accurate result under extremely high or low prices. RF is prone to

	models on Regression. Providing optimistic result for housing.	LightGBM, Hybrid Regression, and Stacked Generalization. Metrics - RMSLE	Generalization Regression deliver satisfactory results, though time complexity must be taken into consideration.	outfitting. Stacked Generalization Regression has a complicated Architecture.
Kalidass et al., (2024)	Developing methods that uses ML algorithms to predict house prices.	House price india dataset where sourced from Kaggle, where 80% of the data were assigned for training and 20% for testing. Used model are – RFR, GB and Ensemble learning (RF+GB). Evaluation metrics used are – MSE, RMSE, MAE, R^2.	Study achieved – MSE ≈ 17,252,273,775.53 – 18,526,489,467.46. RMSE ≈ 131,321.62 – 136,135.46. MAE ≈ 69,047.24 – 77,796.18. R^2 - 0.869 ≈ 0.878. The three models are very efficient, after producing a similar R^2 value of 0.870, 0.869 and 0.878.	The ensemble learning is only restricted to the dataset provided in this research. Comparison was made between just three models.
Yang., (2025)	To evaluate and compare the performance of various ML models for house price prediction.	Kaggle dataset was used with structural variables. Models adopted are LR, RF and DT. . Metrics employed is MSE.	Study achieved – Effectiveness of MSE LR ≈ 0.140. RF ≈ 0.1372. RF using multiple DT effectively suppresses overfitting and slightly outperforms LR. Nine attributes of OverallQual, GrLivArea, TotalBsmtSF, 1stFlrSF, GarageCars, GarageArea,	Limited to current market trend, with the hope for future market use.

			FullBath, TotRmsAbvGrd, and YearBuilt, all influences the house price.	
Oluyele et al., (2024)	To predict house rental prices in Lagos, Nigeria. Price prediction using ML model by examining the relationship between the rental price and features.	Data were scraped from Propertypro Dataset. The data contain 12269 rows with 9 columns. Trained models are RFR, RR, SVR, GBR, and LR. Evaluation metrics are MAE, RMSE and R^2.	Study achieved – MAE ≈ 1539994.03 - 1859749.25 RMSE ≈ 2395925.34 - 2724695.35 R^2 ≈ 0.52 - 0.63 Random Forest model performs better than other models for the dataset used having the lowest MAE, RMSE, and highest R^2 value. number of bedrooms and the apartment's location are the most significant predictors, confirmed using the feature importance analysis.	Not applicable to regions outside Lagos, Nigeria. Lacked in-depth features like property age, property type, etc.
Abdul-Rahman et al., (2021)	To propose advanced ML approaches for house price prediction. Predicting future house prices and the establishment of real estate policies.	'Property Listing Kuala Lumpur' dataset, was used. Model used are LightGBM, XGBoost, RR, and MLR. Evaluation metrics are MAE, MSE, RMSE, Adj.R^2, R^2	Study achieved – MAE ≈ 0.148 - 0.195 MSE ≈ 0.039 - 0.064 RMSE ≈ 0.252 - 0.197 Adj.R^2 ≈ 0.855 - 0.921 R^2 ≈ 0.853 - 0.911 XGBoost showed the highest performance,	The developed model is only applicable to dataset from Kuala Lumpur city.

			generating the lowest MAE and RMSE, outperforming other models. variables that showed a significant impact to house price prediction are nearest hospital, schools, malls, etc.	
Bushuyev et al., (2024)	To enhance the accuracy of predicting rental prices of real estate.	Houston dataset from Redfin website was used, consisting of 9,260 objects and 10 features from Houston, USA. Model used are (RFECV+DT), and (Optuna+LightGBM) Metrics – MSE, RMSE, R^2, MDAPE. SHAP was applied for result interpretation.	Study achieved – Before Textual feature MSE ≈ 139998 RMSE ≈ 374 R^2 ≈ 0.818 MDAPE ≈ 7.830 After Textual feature MSE ≈ 121223 RMSE ≈ 348 R^2 ≈ 0.834 MDAPE ≈ 6.854 Improvement MSE ≈ 13.4 RMSE ≈ 6.95 R^2 ≈ 1.96 MDAPE ≈ 12.46. It was found that adding textual features reduces the Mean Squared Error (MSE) by 13.4%, making the proposed model very suitable.	Less accuracy for none textual features.

Zhang., (2024)	To construct a predictive model using machine learning techniques.	Seattle dataset, Washington were used. The models used are multiple linear models, polynomial models, and KNN models. Metrics – RMSE, MSE, R^2.	Study achieved – MSE ≈ $1.7281*10^{11}$ - $3.8414*10^{10}$. RMSE ≈ $4.1571*10^{5}$ - $1.9599*10^{5.}$ $R^2 \approx 0.9932$ – 0.9985. Both the polynomial regression model and the KNN model have better performance in predicting house prices	The dataset used is specific to Seattle, Washington. Does not include important factors such as urban economic indicators, the level of amenities, etc. The model may not generalize well to other regions.

2.5 SUMMARY

Existing studies on real estate price prediction reveal several recurring challenges. Most models rely on data from limited geographic areas, making them difficult to generalize to other regions. The datasets used are often incomplete or inconsistent, with missing values and a lack of important features such as neighborhood quality or proximity to amenities. While advanced models like Random Forest and XGBoost show good performance, they are prone to overfitting and require extensive computational resources. In contrast, simpler models such as linear regression fail to capture the complex, nonlinear nature of real estate markets. Another common issue is poor model interpretability, as many studies focus solely on accuracy without explaining the underlying prediction factors. Lastly, few models account for market dynamics or temporal variations, making their predictions less reliable over time.

CHAPTER THREE

METHODOLOGY

3.0 INTRODUCTION

This chapter describes the methodologies, processes, and techniques involved in the design and development of the proposed system. To develop a machine learning-based real estate price prediction system that meets user expectations in accuracy, usability, and scalability. Three key stages are adopted in this project:

1. Information Gathering
2. System Design and Implementation
3. Final Testing

These stages represent the foundation of the system's development life cycle. Each phase contributes toward building an efficient model capable of estimating property prices across different regions in Nigeria.

The information-gathering stage focuses on understanding the nature of the existing housing market, the type of data available, and the challenges faced by traditional pricing methods. The design and implementation stage involves structuring the data model, selecting the appropriate algorithms, and developing the interface that will support end users. Finally, the testing stage ensures that the system performs accurately and meets the intended objectives.

Throughout this process, emphasis is placed on high usability, data reliability, and computational efficiency, ensuring the system can function effectively both as a predictive tool and as a scalable research framework for future expansion.

3.1 INFORMATION AND DATA GATHERING

Information gathering is a crucial step in system development, as it provides a foundation for all subsequent design and implementation processes. In this phase, the developer seeks to understand both the technical and non-technical requirements necessary to achieve the system's objectives.

The process involves collecting data on housing prices, identifying key property features, and understanding the market conditions that affect valuation. For this study, relevant datasets were gathered from PropertyPro.ng, an online real estate platform that provides property listings across various Nigerian states.

Other forms of information were obtained through:

1. **Online literature and journal publications,** to understand previous research and existing prediction models (Shinde & Kulkarni, 2021).
2. **Observation of user needs**, including the type of predictions that would be most useful for agents and buyers.
3. **Analysis of similar machine learning applications**, to identify potential limitations and areas for improvement.

Information gathering ensures that system design is user-centered and data-driven. It minimizes errors during implementation by clarifying the scope, available tools, and potential constraints early in the development process (Pressman & Maxim, 2020).

3.2 ANALYSIS OF EXISTING SYSTEM

The existing method of estimating property prices in Nigeria remains largely manual. Real estate agents, investors, and valuers typically rely on personal experience, recent transactions, or unstructured web data to determine a property's market value. This process is time-consuming, subjective, and often inaccurate, as it depends heavily on individual judgment and limited datasets.

Traditional pricing approaches do not consider multiple variables such as proximity to amenities, building quality, or local demand patterns, leading to inconsistencies in price estimation.

Moreover, the lack of digitization means there is limited access to historical property data for trend analysis.

Such challenges necessitate a shift toward an automated and data-driven solution that can leverage computational models to provide consistent and objective price predictions (Abidoye & Chan, 2017).

3.2.1 PROBLEMS OF EXISTING SYSTEM

The main problems associated with the manual system include:

1. Lack of Standardization: There is no unified platform or framework for determining house prices across Nigeria, making it difficult to compare values.
2. Time Consumption: Manual assessment requires extensive market surveys and consultations, leading to delays in property transactions.
3. Human Bias and Inaccuracy: Valuations depend on personal opinions or incomplete information, which can distort actual market trends.
4. Limited Data Utilization: Most estimations rely on current listings without integrating historical or geographical factors.
5. Absence of Predictive Capability: The current approach lacks computational models that can predict future price changes or assess the impact of external factors such as inflation or location development.

3.2.2 DESCRIPTION OF PROPOSED SYSTEM

The proposed system is designed to automate the prediction of house prices using supervised machine learning algorithms. Unlike the traditional manual approach, where property valuation depends heavily on human judgment and subjective market estimation, the proposed model leverages computational intelligence to analyze historical real estate data and generate accurate price predictions.

The system accepts input parameters such as location, number of bedrooms, number of bathrooms, size of the building, property type, and other structural or environmental attributes. Once the user provides these values through an interactive interface, the system processes the data and predicts an estimated price for the property.

The machine learning component of the system was developed using four supervised models:

1. Linear Regression: establishes a linear relationship between input features and the target variable (price).
2. Decision Tree Regressor: splits data into smaller subsets using decision rules to capture complex non-linear relationships.
3. Random Forest Regressor: an ensemble model that combines multiple decision trees to improve prediction accuracy and reduce overfitting.
4. Extreme Gradient Boosting (XGBoost): a powerful boosting algorithm that sequentially improves model performance by minimizing prediction errors.

To evaluate model performance, three key evaluation metrics were employed:

1. R^2 Score (Coefficient of Determination): measures how well the model explains the variability of the target variable.
2. Mean Absolute Error (MAE): determines the average magnitude of prediction errors.
3. Mean Squared Error (MSE): calculates the squared average of prediction errors, emphasizing larger deviations.

The system compares the performance of these algorithms and selects the one with the best predictive accuracy based on the evaluation metrics. In this way, users can obtain the most reliable house price estimation through a simple and user-friendly web interface, where all they need to do is input the required parameters.

The system leverages Python libraries such as Pandas, Scikit-learn, and Streamlit to provide accurate, efficient, and user-friendly predictions. The model used are Linear Regression, Decision Tree, Random Forest, Extreme Gradient Boosting, were chosen for its interpretability and robust performance in numerical forecasting tasks.

Furthermore, the system is flexible enough to accommodate retraining with updated datasets to ensure that predictions remain consistent with evolving real estate trends. This adaptability makes it a valuable tool for property investors, developers, and valuation professionals seeking data-driven decision support.

Key features of the proposed system include:

1. Automated Data Analysis: The system processes large datasets efficiently, identifying relationships between property features and price.
2. User-Friendly Interface: Built with Streamlit, the platform allows users to input property details and receive instant predictions.
3. Model-Based Accuracy: Employs machine learning algorithms for consistent and precise results.
4. Scalability: The system can easily accommodate more states, datasets, or models in the future.
5. Data Visualization: Provides charts and graphs that illustrate trends and model performance.

This system ensures that property buyers, sellers, and investors can make informed decisions with minimal human intervention.

3.2.3 ADVANTAGES OF THE PROPOSED SYSTEM

The proposed system provides multiple advantages over the traditional manual method:

1. Increased Accuracy: By relying on machine learning algorithms, predictions are derived from data patterns rather than subjective estimates.
2. Reduced Time and Effort: Users receive results instantly through an automated interface.
3. Consistency and Objectivity: Eliminates human bias in valuation decisions.
4. Scalability: Capable of integrating additional features and datasets for improved performance.
5. Ease of Use: The web interface allows users with little technical background to perform valuations easily.

This transition from a manual to a computational approach marks a significant advancement in real estate analytics within the Nigerian context.

3.3 DESIGN MODEL USED

A design model serves as a blueprint that defines how the proposed system will be developed, tested, and implemented. It establishes a systematic sequence of steps that ensures all activities are organized and aligned with project goals. For this project, the Waterfall Model, a subset of the Software Development Life Cycle (SDLC), was adopted.

The Waterfall Model is a sequential development process in which progress flows in one direction downward through distinct phases such as analysis, design, implementation, testing, and maintenance (Sommerville, 2016). Each stage must be completed before the next begins, ensuring a structured and disciplined workflow. This model was chosen because it provides clarity, documentation, and well-defined deliverables, which are ideal for academic and research-based software development.

For simplicity and adaptability, the design model used in this research was divided into five main phases, namely:

1. Analysis
2. Tool Evaluation and Selection
3. Design
4. Implementation
5. Post-Implementation

Each phase plays a distinct role in ensuring the successful development of the machine learning–based real estate price prediction system.

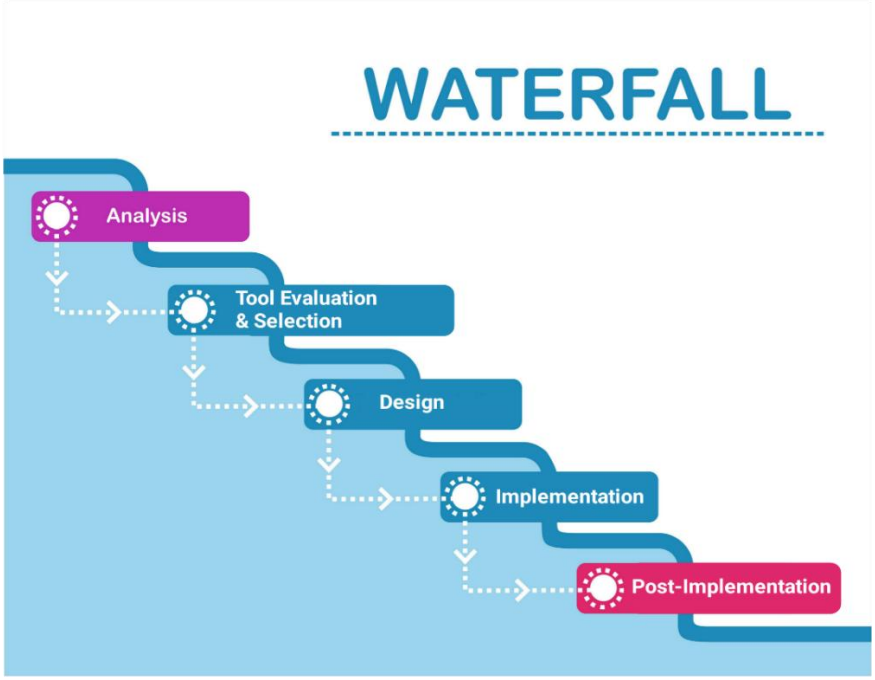

Figure 3.1: Simplified Waterfall Model showing sequential development phases (Source: https://share.google/images/upIg9eSXOrHyKEe7d)

The Waterfall approach guarantees that each development stage flows logically into the next. It supports proper documentation and minimizes the risk of confusion during system implementation, as each phase has clearly defined outputs that become inputs for the next stage (Pressman & Maxim, 2020).

3.3.1 Analysis Phase

The analysis phase is the foundation of the development process. It involves identifying the system requirements, understanding user needs, and defining the project objectives. The primary goal is to determine what the system must achieve and how it will interact with its users and data.

During this phase, the existing manual system of property valuation was analyzed to identify inefficiencies and opportunities for improvement. The data requirements were defined, focusing on property features such as location, bedrooms, bathrooms, and price. The expected output, which is a predicted property price was also clearly stated.

This phase also included defining the functional requirements (data input, price prediction, visualization) and non-functional requirements (usability, scalability, and accuracy). The result of this stage was a clear understanding of the problem, the data to be used, and the intended model behaviour.

3.3.2 Tool Evaluation and Selection Phase

This phase involved selecting the appropriate software tools, libraries, and programming environment necessary to implement the system. The choice of tools was guided by criteria such as availability, efficiency, ease of integration, and compatibility with machine learning workflows.

The following tools were selected for implementation:

1. Python Programming Language: For data preprocessing, model training, and web application development.
2. Jupyter Notebook: For data analysis, visualization, and iterative model testing.
3. Scikit-learn Library: For implementing the Linear Regression algorithm and computing evaluation metrics.
4. Pandas and NumPy: For dataset manipulation and numerical computation.
5. Streamlit Framework: For building an interactive web interface that allows users to input property details and view predictions.
6. Pickle Library: For model serialization and storage, allowing reuse without retraining.

These tools were evaluated based on their proven performance, community support, and integration capability in real-world machine learning applications (Raschka & Mirjalili, 2020).

3.3.3 Design Phase

In the design phase, the logical structure and architecture of the system were created. This phase defines how data will flow within the system, how the components will interact, and how the system will fulfill user requirements.

The design process included:

1. Creating a data flow structure for the property dataset.

2. Designing the Streamlit interface to ensure easy user interaction.
3. Planning the model integration workflow, where preprocessed data is fed into the trained Linear Regression model to generate predictions.

The system was designed using modular architecture, where components such as data collection, preprocessing, model prediction, and result display are treated as independent but interconnected modules. This approach simplifies debugging, improves scalability, and ensures maintainability.

The design phase also accounted for user accessibility, ensuring that the system can be deployed across different devices with minimal technical setup.

3.3.4 Implementation Phase

The implementation phase transforms the design into a working system. It involves writing code, training models, integrating libraries, and developing the user interface.

The dataset was preprocessed and analyzed using Python within Jupyter Notebook, where the Linear Regression model was trained. The trained model was then serialized using Pickle and integrated into the Streamlit environment for deployment.

The implementation steps included:

1. Loading and Cleaning Data: Removing duplicates, handling missing values, and encoding categorical features.
2. Feature Selection and Scaling: Selecting important variables and normalizing values.
3. Model Training: Using Linear Regression to fit data and generate coefficients for prediction.
4. Model Evaluation: Testing the model on unseen data to assess accuracy.
5. Interface Development: Building the Streamlit interface to accept user input and display predicted prices.

During implementation, emphasis was placed on code reusability, clarity, and efficiency. All scripts were tested incrementally to ensure correctness and prevent runtime errors.

3.3.5 Post-Implementation Phase

The post-implementation phase ensures that the system continues to function effectively after deployment. It involves evaluating user feedback, monitoring system performance, and carrying out updates where necessary.

Tasks performed in this phase include:

1. Performance Monitoring: Ensuring model predictions remain accurate over time.
2. Error Handling: Logging and resolving exceptions or issues that occur during user interaction.
3. Model Re-training: Updating the model with new data to improve accuracy.
4. System Maintenance: Enhancing the user interface and improving response time as needed.

This stage ensures that the developed system remains sustainable and adaptable to changing real estate market conditions (Laplante, 2020).

3.4 DATA UNDERSTANDING

The dataset used for this study was obtained from PropertyPro.ng, a popular Nigerian real estate platform that lists residential and commercial properties across multiple states. The dataset was acquired through web scraping techniques using a Python-based extraction tool. Each record in the dataset represents a property listing, with relevant details that influence its market value.

The data contained approximately several hundreds of thousands property records covering diverse Nigerian states, with attributes describing structural, locational, and categorical information. The major attributes included:

Table 3.1: Attributes(features) of the dataset

Attribute	**Description**
Location	Geographic area or state where the property is situated.
Bedrooms	Number of bedrooms in the property.
Bathrooms	Number of bathrooms available.

Toilets	Number of toilets within the property.
Property Type	Category of the property (e.g., flat, duplex, bungalow).
Etc.	

The dataset was provided in CSV format, containing rows (records) and columns (attributes). Each row represents a property listing, while each column represents a specific feature. This structure allowed easy importation into analytical tools such as Pandas for preprocessing and model training.

Understanding the data was critical before model development, as it helped identify inconsistencies, missing values, and potential outliers that could affect model accuracy.

3.5 DATA PREPARATION

Before model training, the dataset underwent a series of preprocessing steps to ensure data consistency and model readiness. Preprocessing transforms raw data into a format suitable for analysis and ensures that the machine learning model receives structured, reliable input.

1. Handling Missing Values: Some records had incomplete entries, such as missing bathroom counts or property sizes. Missing values were handled using:
   - **i)** Mean Imputation: for numerical attributes.
   - **ii)** Mode Imputation: for categorical attributes (like location or property type).

   This approach preserved as much data as possible without distorting the distribution.

$$\text{Imputed Value (Mean)} = \frac{\sum_{i=1}^{n} x_i}{n} \quad \ldots(3.1)$$

2. Outlier Detection and Removal: Outliers (extremely high or low prices) were identified using the Interquartile Range (IQR) method.
   Outliers were removed if they fell outside the defined range:

$$IQR = Q3 - Q1$$
$$Lower\ Bound = Q1 - 1.5(IQR)$$
$$Upper\ Bound = Q3 + 1.5(IQR) \quad \ldots(3.2)$$

Where Q_1 and Q_3 represent the first and third quartiles of the data, respectively.

3. Encoding Categorical Variables: Machine learning algorithms typically require numerical inputs. Therefore, categorical attributes such as location and property type were transformed using One-Hot Encoding (OHE), which converts categorical data into binary (0/1) vectors.

   Example:

   Table 3.2: Categorical data in Binary form.

Property Type	Fiat	Duplex	Bungalow
Flat	1	0	0
Duplex	0	1	0

   This encoding ensures that categorical information is represented numerically without introducing bias.

4. Feature Scaling: Because numerical attributes (like size and price) had different magnitudes, Standardization was applied to normalize them. This ensures all features contribute equally to the model's learning process.

   The StandardScaler formula is:

$$Z = \frac{X-\mu}{\sigma} \quad \ldots(3.3)$$

   Where:

   Z = Original value

   μ = Mean of the feature

   σ = Standard deviation.

   This transformation centers data around a mean of zero and a standard deviation of one.

5. Feature Engineering: Feature engineering involved selecting and refining the most relevant features to improve predictive accuracy. Derived features included:

   a) Total Rooms (sum of bedrooms, bathrooms, and toilets).

b) Price per Room, which normalized price relative to property size or room count. These engineered features enhanced the model's ability to learn real-world patterns.

6. Dataset Splitting: The final dataset was divided into two subsets using Scikit-learn's *train_test_split()* function:
    a) Training Set: 80% of the dataset, used for model training.
    b) Testing Set: 20% of the dataset, used for model evaluation.

3.6 MODEL BUILDING

Model building involves selecting, training, and evaluating several machine learning algorithms to identify the one that best predicts housing prices. In this study, four regression-based algorithms were developed and compared: Linear Regression, Decision Tree Regressor, Random Forest Regressor, and Extreme Gradient Boosting (XGBoost). These models were chosen because they represent both baseline and ensemble learning methods, providing a balanced view of performance and interpretability.

A. **Linear Regression Model**: was used as the baseline model because of its simplicity and ability to describe linear relationships between variables. It assumes that property price () depends linearly on a set of independent variables (X_1,X_2,….,X_n) such as number of bedrooms, bathrooms, and property type.

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots B_n X_n + \varepsilon \quad \ldots(3.4)$$

Where:

Y = Predicted house price

X_i = Independent variables

β_i = Model coefficients

ε = Error term

The model's objective is to minimize the Mean Squared Error (MSE):

$$MSE = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \quad \ldots(3.5)$$

B. **Decision Tree Regressor**: model predicts values by recursively splitting data into smaller subsets based on feature values. It creates a tree structure where each internal node represents a decision on an attribute, each branch represents the outcome, and each leaf node represents a predicted value.

The splitting criterion is based on minimizing variance, calculated as:

$$Var\,(s) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \qquad \ldots(3.6)$$

C. **Random Forest Regressor**: The Random Forest model is an ensemble learning method that builds multiple decision trees and averages their predictions to reduce overfitting and increase accuracy. Each tree is trained on a random subset of the data and features, promoting diversity among the models.

The final prediction is given by:

$$\widehat{Y} = \frac{1}{k}\sum_{i=1}^{k}\widehat{Y}_i \qquad \ldots(3.7)$$

k = total number of trees

$\widehat{Y}_I$ = prediction from the decision tree

This model enhances prediction stability and handles large datasets effectively.

D. XGBoost Regressor: Extreme Gradient Boosting (XGBoost) is a highly efficient ensemble technique based on the concept of boosting. It builds multiple weak learners (shallow trees) sequentially, where each new tree attempts to correct the errors of the previous ones.

The model minimizes an objective function that combines a loss term and a regularization term.

$$\dot{Obj}(\theta) = \sum_{i=1}^{n} l(y_i, \hat{y}_i) \quad + \Sigma_k \Omega(f_k) \qquad \ldots(3.8)$$

$\Omega(f_k)$ = is a regularization term that controls model complexity.

$l(y_i, \hat{y}_i)$ = is the loss function (e.g., squared error).

XGBoost was included due to its strong performance in regression problems and its ability to handle non-linear relationships and large datasets effectively.

3.6.1 Model Training Process

All four models were trained using the training set (80%) and evaluated using the testing set (20%). The same preprocessed dataset was used for consistency, and cross-validation (k=5) was applied to check the stability of each model.

Each model's hyperparameters were fine-tuned to enhance accuracy. For example:

a) Decision Tree: maximum depth and minimum samples per leaf were optimized.

b) Random Forest: number of estimators and maximum depth were tuned.
XGBoost: learning rate, maximum depth, and number of estimators were adjusted.

3.7 MODEL EVALUATION

After training, the models were evaluated and compared using standard regression metrics like Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and Coefficient of Determination (R^2). These metrics helped determine the accuracy and generalization ability of each model.

1. Mean Absolute Error (MAE)

$$MAE = \frac{1}{n}\sum_{i=1}^{n} |y_i - \hat{y}_i| \quad \ldots(3.9)$$

2. Mean Squared Error (RMSE)

$$RMSE = \frac{1}{n}\sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \quad \ldots(3.10)$$

3. Coefficient of Determination (R^2)

$$R^2 = 1 - \frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \bar{y})^2} \quad \ldots(3.11)$$

3.8 SYSTEM DEVELOPMENT APPROACH

The development of the real estate house price prediction system followed a structured Software Development Life Cycle (SDLC) approach. SDLC is a systematic process that outlines the stages involved in the development of a software system, from conception to deployment and maintenance. It helps to ensure that all technical and user requirements are met efficiently, while maintaining software quality and reliability (Sommerville, 2016).

There are several SDLC models, such as the Waterfall Model, V-Model, Agile, and Spiral Model. For this project, the Waterfall Model was adopted due to its simplicity, clarity, and linear nature, which makes it well-suited for academic and research-oriented systems where requirements are well-defined from the onset.

The model breaks the system development into clear, sequential stages, where each stage must be completed before the next one begins. This ensures proper documentation and helps maintain control over the process. The primary phases involved include:

1. Requirement Analysis
2. System Design
3. Implementation
4. Testing
5. Deployment and Maintenance

This structure helps to ensure that errors are detected early, documentation is complete, and each stage produces deliverables that guide the subsequent phase (Pressman & Maxim, 2020).

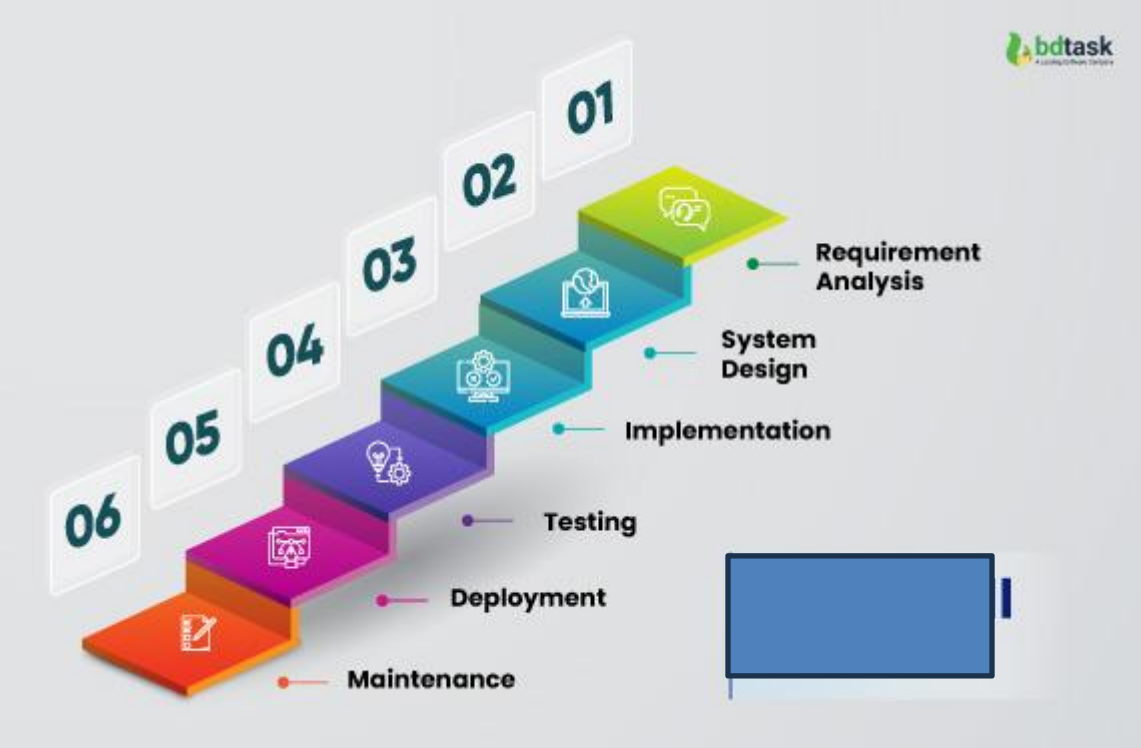

Figure 3.2: SDLC Approach

(Source: https://share.google/images/us6OyalTsg4EaRrfB)

The figure above represents the SDLC approach, which demonstrates the sequential movement from one stage to another. It ensures that once a phase is completed, it does not need to be revisited unless major adjustments are required, thereby maintaining project discipline and predictability (Boehm, 2019).

The following subsections describe each of these SDLC stages and their relevance to this research.

3.9 DATA FLOW DIAGRAM

A Data Flow Diagram (DFD) is used to illustrate the logical flow of data within the system. It depicts how input data is transformed into meaningful output through various processes. The DFD of the proposed house price prediction system demonstrates how user-provided property details are processed by the machine learning model to generate a predicted price.

At the highest (Level 0) view, the process begins when a user enters property features such as the location, number of rooms, and building size through the web interface. This data is sent to the Model Processing Unit, where it undergoes preprocessing (such as scaling and encoding) before being passed into the trained machine learning model. The system then produces a predicted price output, which is displayed to the user.

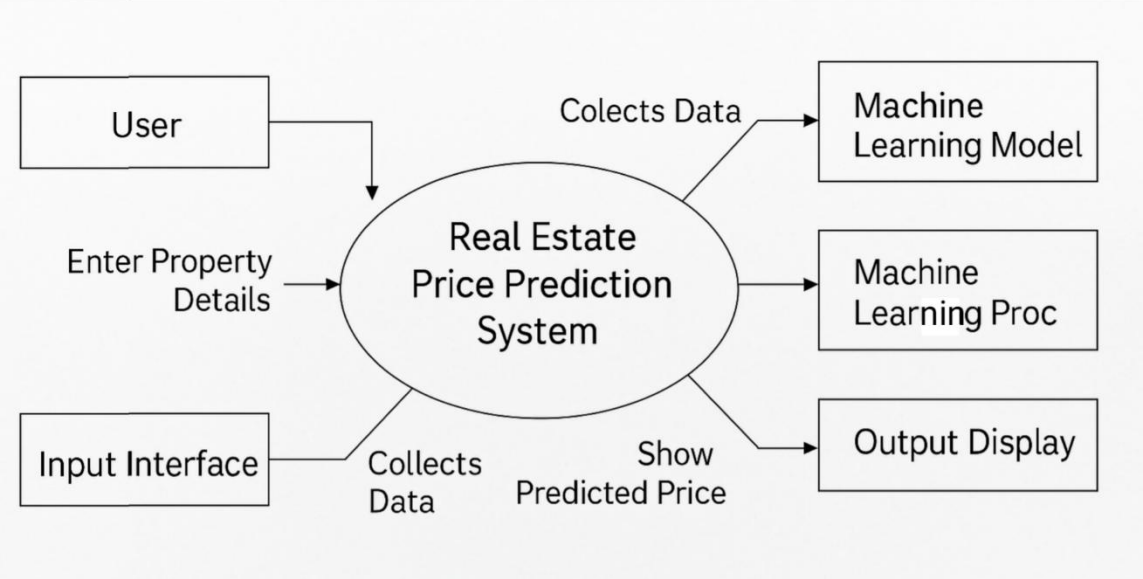

Figure 3.3: DFD

Entities in the DFD include:

1. User: initiates the prediction by providing property details.
2. Input Interface: The Streamlit-based web interface that collects data.
3. Preprocessing Module: cleans and transforms the data for model readiness.
4. Machine Learning Model: performs prediction using trained algorithms.

5. Output Display: shows the predicted property price to the user.

This flow ensures a streamlined and efficient data path from input to output with minimal computational delays.

3.10 FINAL TESTING

The final testing stage involves validating the system's functionality to ensure that all requirements are met and that the system performs efficiently under various conditions. Testing was carried out after full implementation using both sample and real property datasets.

Different forms of testing were conducted:

1. Unit Testing: Each component of the system (e.g., input form, model function) was tested individually.
2. Integration Testing: Verified that different modules (data input, model, output display) worked together correctly.
3. Performance Testing: Measured response time and accuracy under different loads.
4. User Acceptance Testing (UAT): Confirmed that the system meets user expectations and produces realistic predictions.

Testing confirmed that the model achieved reliable accuracy, provided quick results, and maintained stability during user interaction. Adjustments were made where necessary before deployment.

CHAPTER FOUR

SYSTEM IMPLEMETATION AND RESULTS

4.0 INTRODUCTION

This chapter discusses the implementation of the developed machine learning-based real estate price prediction system. It highlights the practical aspect of transforming theoretical designs and models into a functional application capable of predicting housing prices across various states in Nigeria. The chapter details the hardware and software requirements, system design, programming tools, model implementation, testing, and evaluation processes.

The implementation phase represents the most critical stage of the research because it translates conceptual frameworks from Chapter Three into a real-world system. The developed solution utilizes machine learning models, specifically Linear Regression, Decision Tree, Random Forest, and XGBoost, trained on real estate datasets scraped from PropertyPro, one of Nigeria's major online property listing platforms. The system was developed using Python programming language, integrated with essential libraries and deployed through Streamlit, a lightweight framework for building interactive web applications.

The system allows users to input property features such as location, number of bedrooms, bathrooms, area size, and house type, and in return, the model predicts the estimated market value of the property. The model with the best evaluation results (based on metrics such as MAE, MSE, and R^2 score) was selected as the final predictive model for deployment.

4.1 SYSTEM IMPLEMENTATION OVERVIEW

The implementation process followed the structured methodology outlined in Chapter Three, ensuring that each step, from data preprocessing to model deployment, was systematically executed. The goal was to design a scalable, user-friendly system capable of producing reliable predictions in real time.

The system implementation involved three main stages:

1. Backend Development: where model training, data analysis, and result computation were carried out using Python in Jupyter Notebook.
2. Frontend Development: which focused on creating a simple and interactive interface using Streamlit for user interaction.
3. Integration and Testing: combining the trained model with the interface and testing for reliability, accuracy, and user experience.

Through these stages, the system successfully transformed raw data into a predictive tool that can assist real estate stakeholders, buyers, investors, and analysts, in making data-driven pricing decisions.

4.2 SYSTEM REQUIREMENTS

Before implementation, both hardware and software requirements were defined to ensure smooth execution and reproducibility.

4.2.1 Hardware Requirements

The hardware specifications used for development and testing are shown below:

Table 4.1: Hardware Requirement.

Component	Minimum Requirement	Recommended Requirement
Processor	Intel Core i3 (2.0 GHz)	Intel Core i5 or higher
RAM	4 GB	8 GB or above
Storage	500 GB HDD	256 GB SSD
Operating System	Windows 10	Windows 11 / Linux Ubuntu
Display	1366 × 768	1920 × 1080 (HD)

These configurations were adequate for executing machine learning tasks such as data preprocessing, model training, and testing. While larger datasets may require higher configurations, the dataset used for this study performed efficiently under these conditions.

4.2.2 Software Requirements

The software tools and libraries used in the implementation are presented below:

Table 4.2: Software Requirement

Software/Tool	**Purpose/Description**
Python	Core programming language used for development.
Anaconda Navigator	Environment and dependency management for Python libraries.
Jupyter Notebook	Interactive coding environment for analysis and model training.
Streamlit	Framework for web-based user interface and model deployment.
Scikit-learn	Machine learning library for model implementation and evaluation.
Pandas	Data manipulation and handling tool.
NumPy	Numerical computation and array-based operations.
Matplotlib & Seaborn	Visualization libraries for plotting graphs and patterns.
XGBoost	Gradient boosting library for advanced regression analysis.
VS Code (Optional)	Alternative IDE for Python-based project development.

4.3 PROGRAMMING LANGUAGE DISCUSSION

Python was adopted as the main programming language for this project. The choice was motivated by its simplicity, versatility, and vast ecosystem of machine learning and data analysis libraries. Python provides frameworks such as Scikit-learn, TensorFlow, and XGBoost, which make the implementation of predictive algorithms efficient and reliable.

Additionally, Python's integration with Streamlit enabled the development of an intuitive graphical user interface that allows users to interact with the predictive model without requiring programming knowledge.

Other reasons for selecting Python include:

a) Ease of Learning and Readability: Python's syntax resembles human language, allowing for faster development and debugging.
b) Extensive Community Support: With a large open-source community, continuous library updates and support are guaranteed.
c) Cross-platform Compatibility: Python runs seamlessly on different operating systems (Windows, macOS, and Linux).
d) Library Integration: It supports seamless integration with popular libraries like Pandas, NumPy, and Matplotlib for data manipulation and visualization.

This combination of features makes Python the ideal choice for both academic and industrial machine learning projects.

4.4 IMPLEMENTATION PROCESS

The system's implementation was conducted in multiple stages, as detailed below:

4.4.1 Data Loading and Preprocessing

The dataset used in this project was scraped from PropertyPro, a real estate platform in Nigeria that provides listings across multiple states. The dataset consisted of several features, including location, bedrooms, bathrooms, toilets, parking space, property type, condition, size, and price.

Data preprocessing involved:

a) Handling Missing Values: Null values were imputed using median or mode values depending on data type.
b) Encoding Categorical Variables: Property types and locations were encoded using one-hot encoding.
c) Feature Scaling: Continuous attributes such as land size and price were normalized to ensure uniform feature distribution.
d) Outlier Detection and Removal: Extreme values that could distort the model's learning process were identified using z-scores and removed.

This ensured that the dataset fed into the model was clean, well-structured, and representative of realistic housing market conditions.

4.4.2 Model Building

Four machine learning models were implemented to predict property prices:

1. Linear Regression (Baseline Model)
2. Decision Tree Regressor
3. Random Forest Regressor
4. XGBoost Regressor (Extreme Gradient Boosting)

Each model was trained and tested on an 80:20 data split to 80% for training and 20% for testing.

4.4.3 Model Deployment

Once the best-performing model was identified, it was deployed using the Streamlit framework. Streamlit allows for easy integration of Python-based machine learning models into an interactive web application, where users can input property details such as the number of bedrooms, bathrooms, toilets, location, and apartment features to obtain real-time price predictions.

The deployment process involved the following key steps:

a) Model Serialization: The trained model was saved using the joblib library to allow efficient loading without retraining.

b) Interface Development: A user-friendly Streamlit application (app.py) was designed to collect input parameters from users.
c) Model Integration: The application loads the serialized model and processes user input through the same preprocessing pipeline used during model training.
d) Prediction Output: Once the model processes the input, the predicted property price is displayed dynamically on the web interface.

As shown in Figure 4.3, the Streamlit interface provides a clean and intuitive layout for users to input property features. Users can select or type in values such as the number of bedrooms, bathrooms, and location. After providing the required details and clicking the "Predict Price" button, the system computes the estimated value using the trained model and displays it immediately, as illustrated in Figure 4.4.

This interactive design enhances accessibility and usability, allowing even non-technical individuals to utilize the predictive power of machine learning seamlessly. The web-based deployment also ensures cross-platform compatibility, enabling users to access the application from various devices such as laptops, tablets, or mobile phones.

Figure 4.1: Streamlit interface showing input fields and predicted output.

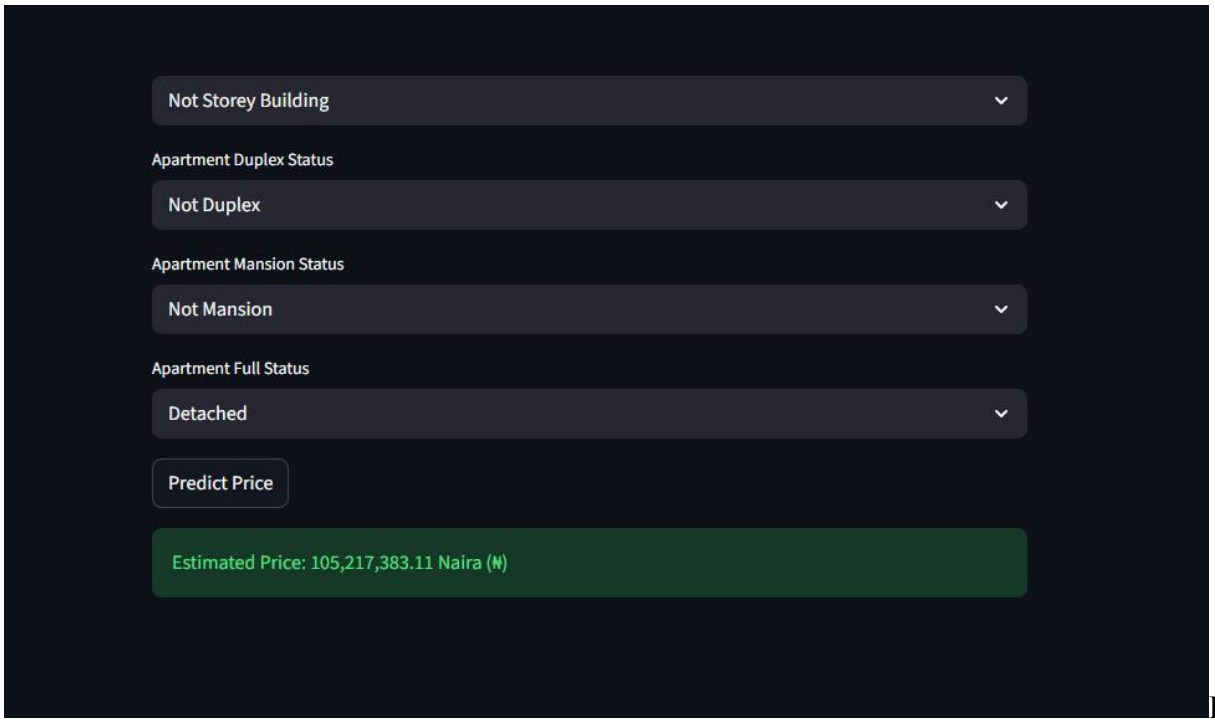

Figure 4.2: prediction results displaying estimated property price.

The successful deployment of this application demonstrates how predictive analytics can be operationalized into functional systems that support data-driven decision-making within the real estate industry. It bridges the gap between theoretical model performance and practical usability, transforming the study's results into a deployable tool for property valuation.

4.5 MODEL EVALUATION

Model evaluation is an essential aspect of predictive modelling, as it assesses how well the trained models perform on unseen data. To measure model accuracy and reliability, several evaluation metrics were employed, including Mean Absolute Error (MAE), Mean Squared Error (MSE), and the Coefficient of Determination (R^2).

Each metric provides a unique insight into model performance:

1. MAE: measures the average absolute difference between predicted and actual values, indicating how close predictions are to real prices.
2. MSE: penalizes larger errors more heavily, making it useful for detecting outliers.

3. R^2: evaluates how much of the variance in the target variable is explained by the model.

Table 4.3: Summary of model performance based on evaluation metrics.

Model	MAE	MSE	R^2 Score
Linear Regression	0.3513614590033909	0.22552617181991638	0.474628279539436
Decision Tree	0.35403112030418904	0.2433011623675636	0.43322076887302663
Random Forest	0.33314941296152056	0.20596582151386783	0.5201948530779867
XGBoost	0.3462247681751093	0.2073558652358768	0.51695669381756

And the evaluation metrics for all model performance are visually represented in a bar chart as shown below.

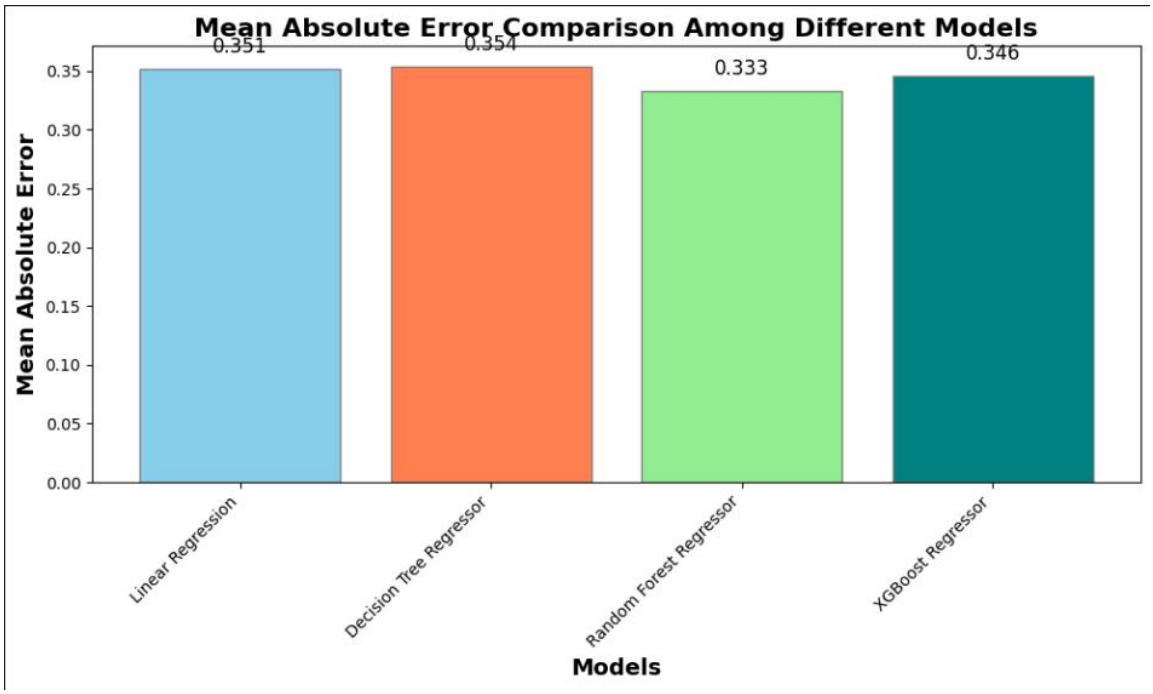

Figure 4.3: Bar Chart representation of MAE comparison among models

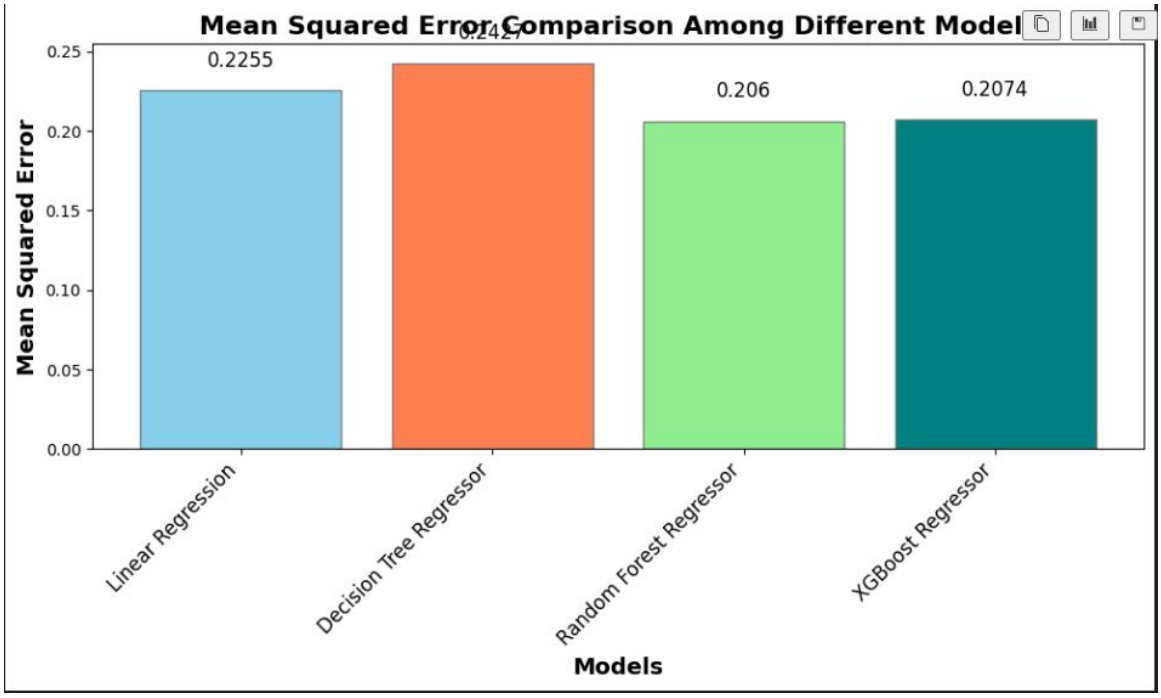

Figure 4.4: Bar Chart representation of MSE comparison among models

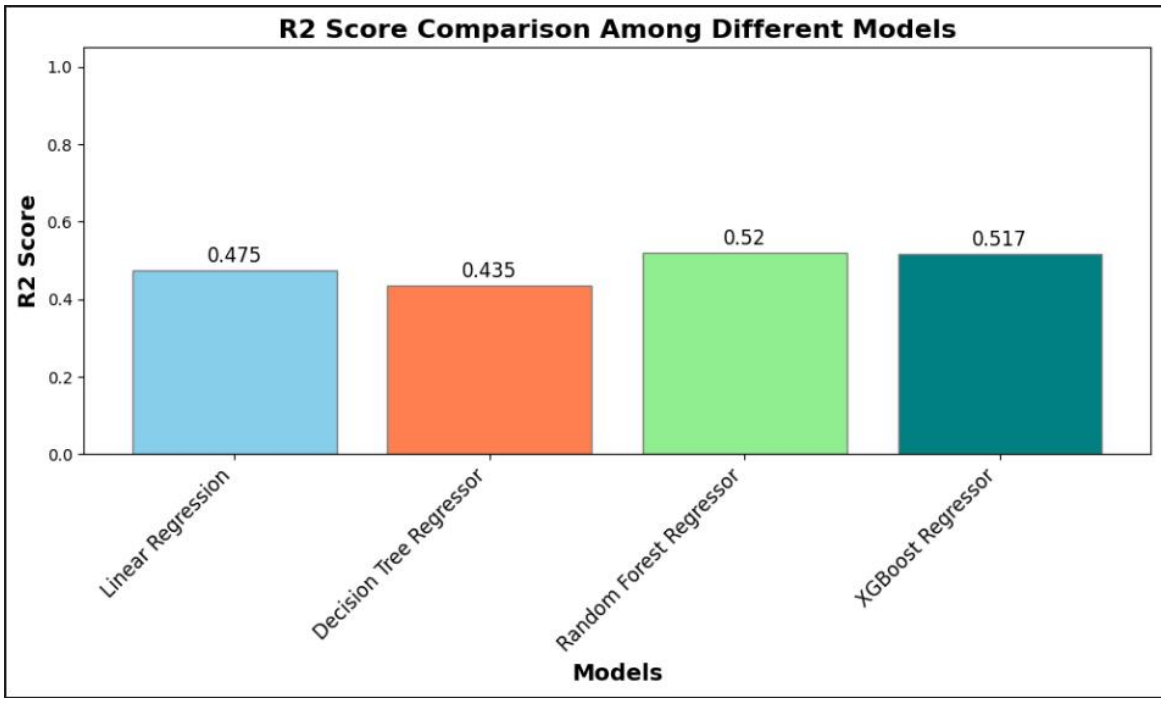

Figure 4.5: Bar Chart representation of R^2 Score comparison among models

From the results above, the Random Forest outperformed the other models, producing the lowest MAE and MSE values and the highest R^2 score. This implies that Random Forest explained approximately 52% of the variance in housing prices, making it the most reliable model for deployment.

4.6 DISCUSSION OF RESULTS

The performance of the four models (Linear Regression, Decision Tree, Random Forest, and XGBoost), was evaluated using standard regression metrics, including Mean Absolute Error (MAE), Mean Squared Error (MSE), and the Coefficient of Determination (R^2). The comparative results are presented in Table 4.3, and visually represented in Figure 4.3, 4.4 and 4.5, which summarizes the performance scores obtained from each model after training and testing on the dataset.

As shown in Table 4.3, the Random Forest model outperformed all other models across the three-evaluation metrics. It achieved the lowest MAE (0.3331) and MSE (0.2059), indicating minimal deviation between the predicted and actual house prices. Additionally, the R^2 score of 0.5201 suggests that approximately 52% of the variance in house prices can be explained by the model's input features. This demonstrates a relatively high predictive capability for real estate price estimation, particularly within a heterogeneous dataset covering multiple Nigerian states.

The XGBoost model followed closely behind, with an MAE of 0.3462 and an R^2 score of 0.5169. Although its performance was competitive, it exhibited slightly higher error values, which may be attributed to its sensitivity to hyperparameter settings and its tendency to overfit when applied to datasets of moderate size.

Conversely, the Linear Regression and Decision Tree models produced comparatively lower accuracies, with R^2 scores of 0.4746 and 0.4332, respectively. Their higher MAE and MSE values indicate that these models were unable to fully capture the complex non-linear relationships between variables such as house size, number of rooms, location, and amenities, which are relationships that are better modeled by ensemble-based techniques.

The superior performance of Random Forest can be attributed to its ensemble learning approach, which integrates predictions from multiple decision trees to enhance generalization and reduce overfitting. By aggregating the outputs of several trees, the model minimizes variance and improves stability, by making it well-suited for real-world prediction tasks that involve diverse and noisy data, such as real estate prices across multiple Nigerian regions.

These findings align with previous research such as Shuzlina et al. (2021) and Kalidass et al. (2024), who reported that ensemble learning methods consistently outperform single-model approaches in property price forecasting. The results affirm the reliability of machine learning, particularly ensemble methods, in automating and improving traditional real estate valuation systems.

The Random Forest model emerged as the most accurate and robust algorithm for predicting property prices, demonstrating its capacity to handle non-linearity, feature variability, and market heterogeneity effectively. As a result, it was selected as the final model for deployment within the web-based application, providing real-time and data-driven property price estimation for end users.

CHAPTER FIVE

SUMMARY AND CONCLUSION

5.0 INTRODUCTION

This chapter provides a comprehensive summary and conclusion of the study on Machine Learning for House Price Prediction in Real Estate. It reviews the key objectives, methodological approach, findings, and contributions of the research. The chapter also discusses the implications of the study and how the developed system can contribute to improved decision-making within Nigeria's real estate industry.

5.1 SUMMARY OF THE STUDY

The study was designed to develop a predictive system capable of estimating property prices using machine learning algorithms. The motivation arose from the need for an accurate, data-driven approach to real estate valuation in Nigeria, where property pricing is often inconsistent due to subjective judgment and lack of standardized data.

In Chapter One, the background and problem statement established the relevance of this work, highlighting the challenges of price estimation in Nigeria's dynamic housing market. The objectives were clearly defined to include collecting a reliable dataset, applying machine learning algorithms, and identifying the most effective model for house price prediction.

Chapter Two presented a comprehensive review of related literature, discussing previous works that applied various machine learning methods such as Decision Trees, Random Forest, and XGBoost to housing datasets in different parts of the world. The review revealed that ensemble models often outperform traditional regression methods due to their ability to capture nonlinear relationships and reduce overfitting.

Chapter Three described the methodology used in this study. The Waterfall Model was adopted to guide the project phases systematically, from data collection, preprocessing, and model building to evaluation and deployment. Data was scraped from PropertyPro, a real estate website that lists property information across multiple states in Nigeria. Four models were trained: Linear

Regression, Decision Tree, Random Forest, and XGBoost. The data was cleaned, encoded, and scaled before being used for model training and testing.

Chapter Four covered the system implementation and results. The environment setup included Python, Scikit-learn, Pandas, and Streamlit for deployment. After training and evaluation, the Random Forest model demonstrated the best performance, with an R^2 score of 0.52, indicating that it explained about 52% of the variance in housing prices. The model was integrated into a Streamlit-based web application that allows users to input property features and instantly receive predicted prices.

This research successfully developed a working predictive system that can be used by real estate stakeholders including agents, investors, and buyers to make informed and objective property pricing decisions. It also showed that data-driven valuation models can significantly reduce bias and increase transparency in the real estate market.

5.2 CONCLUSION

The study demonstrated that machine learning provides an efficient and reliable method for real estate price prediction. By applying data preprocessing, feature engineering, and multiple model evaluations, the research confirmed that ensemble-based models such as Random Forest deliver superior performance compared to traditional linear methods.

The system developed in this study not only produced accurate predictions but also provided a user-friendly interface that allows for practical use by non-technical individuals. This makes it a valuable tool for real estate stakeholders seeking data-supported pricing insights.

The findings reinforce the importance of data quality, model selection, and algorithmic optimization in building predictive systems. Moreover, the success of this project highlights the growing role of artificial intelligence and machine learning in supporting socio-economic decision-making processes in developing countries like Nigeria.

Although the results were impressive, certain limitations exist, primarily due to the limited dataset and the absence of external factors such as macroeconomic indicators or geospatial features. Future studies should incorporate additional data sources, including satellite data and economic trends, to enhance model generalization and robustness.

In conclusion, this study has shown that with proper implementation, machine learning techniques can revolutionize real estate valuation in Nigeria by providing accurate, transparent, and scalable pricing solutions. The developed system stands as a step forward toward achieving data-driven property valuation and improved decision-making in the housing market.

REFERENCES

Journal Articles & Conference Papers

1. Abdul-Rahman, S., & Mutalib, S. (2021).
   Advanced machine learning algorithms for house price prediction: Case study in Kuala Lumpur.
   International Journal of Advanced Computer Science and Applications, 12(12), 736–744.
2. Adebayo, M., & Ojo, T. (2018). Challenges of real estate valuation in developing economies: A case study of Lagos and Edo States, Nigeria. African Journal of Property Studies, 6(1), 22–34.
3. Adegbite, K., Adebayo, M., & Ojo, T. (2019). An empirical study on determinants of residential property prices in Nigerian cities. Journal of Real Estate Research and Development, 14(2), 45–59.
4. Asogwa, D., Nwosu, F., & Eze, P. (2023). Application of machine learning algorithms for property price estimation in Nigeria. International Journal of Data Science and Analytics, 9(4), 215–228.
5. Bushuyev, S., Bushuyev, D., Kravtsov, D., Poletav, N., & Malaksiano, M. (2024). Machine learning model for house price prediction based on natural language text data analysis.
   Proceedings of the 6th International Workshop on Modern Machine Learning Technologies, 2–10.
6. Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (pp. 785–794). ACM Press.
7. Chengke, L., & Zhang, Y. (2023). A comparative study of regression and machine learning models for real estate price prediction. International Journal of Advanced Computer Science Applications, 14(3), 120–133.
8. Evaluation of price prediction of houses in real estate via machine learning.
   J. Appl. Sci. Environ. Manage., 29(1), 44–47.

9. Eze, C., Nnamdi, J., & Okafor, C. (2020). Predictive modeling of residential property prices in Lagos using data mining techniques. African Journal of Engineering Research and Development, 7(1), 95–108.
10. Faris, A., Abdullah, R., & Yusuf, M. (2021). Machine learning for property price prediction and valuation in emerging markets. American Journal of Engineering Research and Development, 7(1), 95–102.
11. Fang, L. (2023). Machine learning models for house price prediction. Highlights in Science, Engineering and Technology, 409–414.
12. Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep learning. MIT Press.
13. Sharma, H.; Harsora, H.; Ogunleye, B. An Optimal House Price Prediction Algorithm: XGBoost. Analytics 2024, 3, 30–45.
14. Ibrahim, A. A., Ayilara-Adewale, O. A., Alabi, A. A., & Olusesi, D. A. (2025). Evaluation of Price Prediction of Houses in a Real Estate via Machine Learning. J. Appl. Sci. Environ. Manage. Vol. 29 (1) 43-48 Jan. 2025
15. Ibrahim, M., & Sule, A. (2019). The role of macroeconomic variables in property price forecasting: Evidence from Nigeria. Journal of African Real Estate Studies, 5(3), 64–75.
16. Ibrahim, M., Lawal, S., & Omoregie, J. (2025). Hybrid machine learning models for predicting house prices in developing economies. Urban Science, 9(32), 1–15.
17. Jha, P., & Shrestha, D. (2021). Artificial intelligence applications in property valuation and real estate market forecasting. Journal of Information Systems and Technology Management, 18(4), 77–89.
18. Kalidass, J., Dharshalini, T., Nivetha, R., & Subasri, A. P. (2024). House price prediction using machine learning. International Research Journal of Engineering and Technology, 11(4), 1351–1357.
19. Kaplan, A., & Haenlein, M. (2020). Rulers of the world, unite! The challenges and opportunities of artificial intelligence. Business Horizons, 63(1), 37–50.
20. MayorLaz, A. (2021). Assessing the impact of neighborhood factors on residential property valuation: A case study in Nigerian cities. American Journal of Environmental Research and Development, 7(1), 95–102.

21. Okafor, T., & Abdul, L. (2021). Challenges in Nigerian real estate data collection and predictive modeling. International Review of Economics and Development, 11(4), 226–239.
22. Oluyele, S., Akingbade, J., Akinode, V., & Idoghor, R. (2024). Prediction of urban house rental prices in Lagos–Nigeria: A machine learning approach. ABUAD Journal of Engineering Research and Development, 7(2), 216–228.
23. Truong, Q., Nguyen, M., Dang, H., & Mei, B. (2020). Housing price prediction via improved machine learning techniques. Procedia Computer Science, 174, 433–442.
24. UN-Habitat. (2020). World cities report 2020: The value of sustainable urbanization. United Nations Human Settlements Programme.
25. World Bank. (2019). Global economic prospects: Heightened tensions, subdued investment. Washington, DC: The World Bank.
26. Yakub, I., Eze, P., & Salami, O. (2022). Data-driven approaches to real estate valuation in Nigeria: A review of current practices and challenges. ITM Web of Conferences, 44(2018), 1–12.
27. Xu, K., & Nguyen, H. (2022). Predicting housing prices and analyzing real estate markets in the Chicago suburbs using machine learning. Proceedings of the 3rd International Conference on Signal Processing and Machine Learning, 2–5.
28. Yang, X. (2025). Research on house price prediction based on machine learning. ITM Web of Conferences, 70, 02018, 1–8.
29. Zou, C. (2023). The housing price using machine learning algorithm: The case of Jinan, China. Highlights in Science, Engineering and Technology, 39, 327–332.
30. Zhang, H. (2024). Predicting housing prices using supervised machine learning models. Proceedings of the 3rd International Conference on Financial Technology and Business Analysis, 1–6.

APPENDIX

SOURCE CODE

```
import streamlit as st

import pandas as pd

import numpy as np

import joblib



# Load saved model and scaler

model = joblib.load("model.pkl")

scaler = joblib.load("scaler.pkl")

columns = joblib.load("columns.pkl")



# These are the categories you had during training

available_cities = joblib.load("unique_cities.pkl")

apartments_old = joblib.load("unique_Apartments.pkl")

apartments_flat = joblib.load("unique_apartment_flat.pkl")

apartments_bungalow = joblib.load("unique_apartment_bungalow.pkl")

apartments_storey = joblib.load("unique_apartment_storey.pkl")

apartments_duplex = joblib.load("unique_apartment_duplex.pkl")

apartments_mansion = joblib.load("unique_apartment_mansion.pkl")
```

```
apartments_full = joblib.load("unique_apartment_full.pkl")

estates = joblib.load("unique_estate.pkl")

apartments_bq = joblib.load("unique_apartment_bq.pkl")



# Streamlit App

st.title("HOUSE PRICE PREDICTION SYSTEM")

st.header("HOUSE SPECIFICATIONS")



# Numerical inputs

Bedrooms = st.number_input("Number of Bedrooms", min_value=1, max_value=10,
value=3)

Bathrooms = st.number_input("Number of Bathrooms", min_value=1, max_value=10,
value=2)

Toilets = st.number_input("Number of Toilets", min_value=1, max_value=10,
value=2)



# Dropdown for categorical

city = st.selectbox("City", available_cities)

new_apartment_old = st.selectbox("Apartment Age", apartments_old)

new_apartments_bq = st.selectbox("Apartment BQ Status", apartments_bq)

new_estates = st.selectbox("Estate Status", estates)

new_apartments_bungalow = st.selectbox("Apartment Bungalow Status",
apartments_bungalow)
```

```
new_apartments_flat = st.selectbox("Apartment Flat Status", apartments_flat)

new_apartments_storey = st.selectbox("Apartment Storey Status",
apartments_storey)

new_apartments_duplex = st.selectbox("Apartment Duplex Status",
apartments_duplex)

new_apartment_mansion = st.selectbox("Apartment Mansion Status",
apartments_mansion)

new_apartments_full = st.selectbox("Apartment Full Status", apartments_full)

available_cities = "Location"



# Build input DataFrame

input_dict = {

    "'Apartment Category (Newly Build/ Already Existing)":
[new_apartment_old],

    "Apartment Category (Flat/ Not Flat)": [new_apartments_flat],

    "Apartment Category (Bungalow/ Not Bungalow)": [new_apartments_bungalow],

    "Apartment Category (Storey Building/ Not Storey Building)":
[new_apartments_storey],

    "Apartment Category (Duplex/ Not Duplex )": [new_apartments_duplex],

    "Apartment Category (Mansion/ Not Mansion)": [new_apartment_mansion],

    "Apartment Category (Full Apartmenr/ Detatched Apartment)":
[new_apartments_full],

    "Estate or Not in the Estate": [new_estates],

    "Apartment Category (BQ or Not-BQ)": [new_apartments_bq],
```

```
    "Location": [available_cities],
    "Extracting the State   ": [city],
    "Bedrooms": [Bedrooms],
    "Bathrooms": [Bathrooms],
    "Toilets": [Toilets],
}
input_df = pd.DataFrame(input_dict)

# Encode and align
input_encoded = pd.get_dummies(input_df, drop_first=True)
input_encoded = input_encoded.reindex(columns=columns, fill_value=0)

# Scale
scaled_input = scaler.transform(input_encoded)

# Predict
if st.button("Predict Price"):
    sigma2 = joblib.load("sigma2_rfr.pkl")
    log_pred = model.predict(scaled_input)[0]
    # Adjust prediction with sigma2
```

```
    prediction = (np.expm1(model.predict(scaled_input)[0] + 0.5 *
sigma2))/1e2

    st.success(f"Estimated Price: {prediction:,.2f} Naira (₦)")

#SPliting the dataset into feature (x) and target(y)

x = df.drop("Price", axis = 1)

y = df["Price"]

# Spliting the dataset into training and testing set

from sklearn.model_selection import train_test_split

x_train, x_test, y_train, y_test = train_test_split(x, y, random_state = 40,
test_size = 0.4)

#Encoding the categorical data using get_dummies

x_train_encode = pd.get_dummies(x_train, columns = ["Apartment Category
(Newly Build/ Already Existing)", "Apartment Category (Flat/ Not Flat)",
"Apartment Category (Bungalow/ Not Bungalow)", "Apartment Category (Storey
Building/ Not Storey Building)", "Apartment Category (Duplex/ Not Duplex )",
"Apartment Category (Mansion/ Not Mansion)", "Estate or Not in the Estate",
"Apartment Category (Full Apartmenr/ Detatched Apartment)", "Apartment
Category (BQ or Not-BQ)", "Extracting the State   ", "Location" ], drop_first
= True)

x_train_encode



#Encoding the categorical data using get_dummies

x_test_encode = pd.get_dummies(x_test, columns = ["Apartment Category (Newly
Build/ Already Existing)", "Apartment Category (Flat/ Not Flat)", "Apartment
Category (Bungalow/ Not Bungalow)", "Apartment Category (Storey Building/ Not
Storey Building)", "Apartment Category (Duplex/ Not Duplex )", "Apartment
Category (Mansion/ Not Mansion)", "Estate or Not in the Estate", "Apartment
```

```
Category (Full Apartmenr/ Detatched Apartment)", "Apartment Category (BQ or
Not-BQ)", "Extracting the State   ", "Location" ], drop_first = True)

x_test_encode

# Align columns between train and test

x_train_encode, x_test_encode = x_train_encode.align(x_test_encode,
join="left", axis=1, fill_value=0)

#scaling the x_train and x_test set

from sklearn.preprocessing import MinMaxScaler

MMS = MinMaxScaler()

x_train = MMS.fit_transform(x_train_encode)

x_test = MMS.transform(x_test_encode)



#Training with Linear Regression Model

from sklearn.linear_model import LinearRegression

LR = LinearRegression()

LR

import numpy as np

import joblib

y_train_pred = LR.predict(x_train)

residuals = y_train - y_train_pred



sigma2_lr = np.var(residuals)
```

```
joblib.dump(sigma2_lr, "sigma2_lr.pkl")


#Plotting the linear graph

plt.figure(figsize = (8, 5))

sns.regplot(x= y_test, y= y_pred)

plt.xlabel("Actual target")

plt.ylabel("Predicted target")


#Evaluating the performance

from sklearn.metrics import mean_squared_error, r2_score, mean_absolute_error

#Saving the performance in a variable name


mean_squared_error_LR = mean_squared_error(y_test, y_pred)

r2_score_LR = r2_score(y_test, y_pred)

mean_absolute_error_LR = mean_absolute_error(y_test, y_pred)

print(f"The mean squared error is: {mean_squared_error(y_test, y_pred)}")

print(f"The R2 score is: {r2_score(y_test, y_pred)}")

print(f"The mean absolute error is: {mean_absolute_error(y_test, y_pred)}")
```

```
#Creating a Random Forest Model

from sklearn.ensemble import RandomForestRegressor

RFR = RandomForestRegressor(n_estimators = 200, random_state = 40)

#Fitting the training data on the model

RFR.fit(x_train, y_train)

#Predicting

Random_predict = RFR.predict(x_test)

Random_predict

y_train_pred = RFR.predict(x_train)

residuals = y_train - y_train_pred

sigma2_rfr = np.var(residuals)

joblib.dump(sigma2_rfr, "sigma2_rfr.pkl")

# Saving the performance in a variable name

mean_squared_error_RF = mean_squared_error(y_test, Random_predict)

r2_score_RF = r2_score(y_test, Random_predict)

mean_absolute_error_RF = mean_absolute_error(y_test, Random_predict)

#Visualizing and comparing the performance of the different models using a
bar chart (Mean Absolute Error)



import matplotlib.pyplot as plt

# Organizing the data into lists for plotting
```

```
models = ['Linear Regression', 'Decision Tree Regressor', 'Random Forest
Regressor', 'XGBoost Regressor']

mae_values = [mean_absolute_error_LR, mean_absolute_error_DTR,
mean_absolute_error_RF, mean_absolute_error_XGB]



# Creating the bar chart using a single plt.bar() call

plt.figure(figsize = (10, 6))

plt.title("Mean Absolute Error Comparison Among Different Models",
fontsize=16, fontweight='bold')



# Ploting model names on x-axis and MAE values as bar heights

plt.bar(models, mae_values, color=['skyblue', 'coral', 'lightgreen', 'teal'],
edgecolor='grey')



# Adding labels and improve layout

plt.xlabel("Models", fontsize=14, fontweight='bold')

plt.ylabel("Mean Absolute Error", fontsize=14, fontweight='bold')

plt.xticks(rotation=45, ha='right') # Rotate labels for better visibility if
needed



# Add the exact value on top of each bar

for i, value in enumerate(mae_values):

    plt.text(i, value + 0.01, round(value, 3), ha='center', va='bottom',
fontsize=12)
```

```
# 4. Display the plot

plt.tight_layout() # Adjust layout to prevent labels from being cut off

plt.show()

#Visualizing and comparing the performance of the different models using a
bar chart (R2 Score)



# 1. Organize the data into lists for plotting

models = ['Linear Regression', 'Decision Tree Regressor', 'Random Forest
Regressor', 'XGBoost Regressor']

r2_values = [r2_score_LR, r2_score_DTR, r2_score_RF, r2_score_XGB]



# 2. Create the bar chart using a single plt.bar() call

plt.figure(figsize = (10, 6))

plt.title("R2 Score Comparison Among Different Models", fontsize=16,
fontweight='bold')



# Plot model names on x-axis and R2 values as bar heights

# Using a list of colors makes the chart visually distinct

colors = ['skyblue', 'coral', 'lightgreen', 'teal']

plt.bar(models, r2_values, color=colors, edgecolor='grey')
```

```
# 3. Add labels and customize

plt.xlabel("Models", fontsize=14, fontweight='bold')

plt.ylabel("R2 Score", fontsize=14, fontweight='bold')

plt.xticks(rotation=45, ha='right', fontsize=12) # Rotate labels for better
visibility



# Optional: Add the exact value on top of each bar

for i, value in enumerate(r2_values):

    plt.text(i, value + 0.01, round(value, 3), ha='center', va='bottom',
fontsize=12)



# Set the y-limit to be appropriate for R2 scores (which usually range from 0
to 1)

plt.ylim(0, 1.05)



# 4. Display the plot

plt.tight_layout() # Adjust layout to prevent labels from being cut off

plt.show()



#Visualizing and comparing the performance of the different models using a
bar chart (mean squared error)



# 1. Organize the data into lists for plotting
```

```
models = ['Linear Regression', 'Decision Tree Regressor', 'Random Forest
Regressor', 'XGBoost Regressor']

mse_values = [mean_squared_error_LR, mean_squared_error_DTR,
mean_squared_error_RF, mean_squared_error_XGB]



# 2. Create the bar chart using a single plt.bar() call

plt.figure(figsize = (10, 6))

plt.title("Mean Squared Error Comparison Among Different Models", fontsize=16,
fontweight='bold')



# Plot model names on x-axis and MSE values as bar heights

# Assign a unique color to each model

colors = ['skyblue', 'coral', 'lightgreen', 'teal']

plt.bar(models, mse_values, color=colors, edgecolor='grey')



# 3. Add labels and customize

plt.xlabel("Models", fontsize=14, fontweight='bold')

plt.ylabel("Mean Squared Error", fontsize=14, fontweight='bold')

plt.xticks(rotation=45, ha='right', fontsize=12) # Rotate labels for better
visibility



# Optional: Add the exact value on top of each bar

for i, value in enumerate(mse_values):
```

```
    plt.text(i, value + max(mse_values) * 0.05, round(value, 4), ha='center',
va='bottom', fontsize=12)



# 4. Display the plot

plt.tight_layout() # Adjust layout to prevent labels from being cut off

plt.show()
```