



TRIBHUVAN UNIVERSITY

Prime College

Nayabazar, Kathmandu, Nepal

A PROJECT REPORT

ON

“CarCompanion”-A Secondhand Car Price Prediction System

SUBMITTED BY

Anahat Basnet (5-2-410-121-2018)

Ishan Rai (5-2-410-132-2018)

Swastika Banskota (5-2-410-159-2018)

A project report submitted for the partial fulfillment of the requirements for the degree of
Bachelor of Science in Computer Science and Information Technology (B.Sc.CSIT) 7th
Semester of Tribhuvan University, Nepal

May, 2023

“CarCompanion”- A Secondhand Car Price Prediction System

[CSC 412]

A project report submitted for the partial fulfilment of the requirements for the degree of
Bachelor of Science in Computer Science and Information Technology awarded by
Tribhuvan University.

SUBMITTED BY

Anahat Basnet (5-2-410-121-2018)

Ishan Rai (5-2-410-132-2018)

Swastika Banskota (5-2-410-159-2018)

SUBMITTED TO

Prime College

Department of Computer Science

Affiliated to Tribhuvan University

Khusibun, Nayabazar, Kathmandu



May, 2023

Date: 14th May, 2023

SUPERVISOR’S RECOMMENDATION

The project work report titled “CarCompanion- A Secondhand Car Price Prediction System” submitted by **Anahat Basnet, Ishan Rai**, and **Swastika Banskota** of Prime College located in Khusibun, Nayabazar, Kathmandu, is prepared under my supervision as per the required procedure and format set by the Institute of Science and Technology (IOST), Tribhuvan University in partial fulfilment of the requirements for the degree of Bachelors of Science in Computer Science and Information Technology (B.Sc.CSIT). Therefore, I recommend the project work report for future evaluation.

.....

Ms. Rolisha Sthapit

Supervisor

Department of Computer Science & Information Technology

Prime College

Date: 14th May, 2023

CERTIFICATE OF APPROVAL

This is to certify that the report is evaluated and recommended to the Department of Computer Science and Information Technology for acceptance of the report entitled “**CarCompanion- A Secondhand Car Price Prediction System**” submitted by **Anahat Basnet, Ishan Rai, and Swastika Banskota** in partial fulfilment for the degree of Bachelor of Science in Computer Science and Information Technology (B.Sc.CSIT), Institute of Science and Technology (IOST), Tribhuvan University.

Mr. Narayan Prasad Sharma
Principal

Ms. Rolisha Sthapit
Program Coordinator

Ms. Rolisha Sthapit
Supervisor

Mr. Jagdish Bhatta
External Examiner

ACKNOWLEDGEMENT

Foremost, we owe our deepest gratitude to Prime College for allowing us an opportunity to work on this project as part of our syllabus with necessary resources and assistance. We are heartily indebted to our supervisor, **Ms. Rolisha Sthapit** for her constant guidance throughout this project. Her insights and support helped us manage the emerging barriers through the development of this project from the beginning till the end. Furthermore, we extend our appreciation to other supervisors and panel members for their constructive suggestions and their views regarding the project that provided us room for the improvement.

Similarly, the development of this project was possible with the support, encouragement, and co-operation of our friends and every teaching staff of B.Sc.CSIT. Last but not the least, we would like to express heartfelt gratitude to the people who are directly and indirectly part of this project. Any kind of suggestion for criticism might be preferred and acknowledged.

With respect,

Anahat Basnet (20466/075)

Ishan Rai (20478/075)

Swastika Banskota (20506/075)

ABSTRACT

The global rise of the automobile industry and the advancement in the mode of transport have increased the usage of cars and have made cars one of the common modes of transport. Everyone is not capable enough to buy a new car instead they opt to buy pre-owned cars resulting in escalating the second-hand car businesses. But the randomness in the prices of the pre-owned cars has been a measure issue. To address this issue, CarCompanion, a web-based application is developed to predict the prices of used cars. Machine Learning algorithms such as Random Forest Regression, Bagging Regression, and Linear Regression algorithms are used to predict the price of second-hand cars, taking into account various factors that comprehend the value of a second-hand car such as the model of the car, its age, mileage, engine, type of fuel, type of owner, kilometers drove, etc. The prediction model is trained on a large dataset of previously sold cars, enabling it to learn and recognize patterns that affect the price of a used car. This application's prediction will benefit both the buyers and sellers of second-hand cars.

KEYWORDS: *Secondhand cars, Price Prediction, Random Forest Regression, Bagging Regression, Linear Regression*

TABLE OF CONTENTS

TITLE PAGE	ii
SUPERVISOR’S RECOMMENDATION	iii
CERTIFICATE OF APPROVAL	iv
ACKNOWLEDGEMENT.....	v
ABSTRACT.....	vi
TABLE OF CONTENTS	vii
LIST OF ABBREVIATIONS	ix
LIST OF TABLES	xii
CHAPTER 1 INTRODUCTION.....	1
1.1. Introduction	1
1.2. Problem Statement	2
1.3. Objectives.....	2
1.4. Scope and Limitation	3
1.5. Development Methodology.....	3
1.6. Report Organization	6
CHAPTER 2 BACKGROUND STUDY AND LITERATURE REVIEW	7
2.1. Background Study	7
2.2. Literature Review	8
CHAPTER 3 SYSTEM ANALYSIS	10
3.1. System Analysis	10
3.1.1. Requirement Analysis.....	10
3.1.2. Feasibility Analysis.....	13

3.1.3. Analysis	16
CHAPTER 4 SYSTEM DESIGN	22
4.1. Design.....	22
4.1.1. Refinement of Class Diagram.....	31
4.1.2. Component Diagram.....	36
4.1.3. Deployment Diagram.....	37
4.2. Algorithm Details	39
CHAPTER 5 IMPLEMENTATION AND TESTING	42
5.1. Implementation.....	42
5.1.1. Tools Used.....	42
5.1.2. Implementation Details of Modules	44
5.2. Testing.....	50
5.2.1. Test Cases for Unit Testing.....	50
5.2.2. Test Cases for System Testing.....	51
5.2.3. Real-Time Test Scenario.....	53
5.3. Result Analysis.....	56
CHAPTER 6 CONCLUSION AND FUTURE RECOMMENDATIONS.....	62
6.1. Conclusion.....	62
6.2. Future Recommendations.....	62
REFERENCES.....	64
APPENDICES	

LIST OF ABBREVIATIONS

CNN	Convolutional Neural Network
CSS	Cascading Style Sheet
EDA	Exploratory Data Analysis
GIT	Global Information Tracker
HTML	Hypertext Markup Language
MAE	Mean Absolute Error
ML	Machine Learning
MSE	Mean Squared Error
RDBMS	Relational Database Management System
RFR	Random Forest Regression
RMSE	Root Mean Square Error
RNN	Recurrent Neural Network
UI	User Interface
UML	Unified Modeling Language
URL	Universal Resource Locator
WWW	World Wide Web
XML	Extensible Markup Language

LIST OF FIGURES

Figure 1. 1 Incremental Development Method	5
Figure 1. 2 Development Methodology	6
Figure 3. 1 Use Case Diagram For Authentication	11
Figure 3. 2 Use Case Diagram For Users	12
Figure 3. 3 Gantt Chart	15
Figure 3. 4 Class Diagram Of Carcompanion	16
Figure 3. 5 Sequence Diagram Of Carcompanion	18
Figure 3. 6 State Diagram Of Carcompanion	19
Figure 3. 7 Activity Diagram Of Carcompanion	20
Figure 4. 1 System Architecture Of Carcompanion	22
Figure 4. 2 High-Level System Design Of Carcompanion	23
Figure 4. 3 Overview Of The Dataset	24
Figure 4. 4 Information On The Data	25
Figure 4. 5 Missing Data Visualization	26
Figure 4. 6 Numerical Features Visualization With Distribution Plot	27
Figure 4. 7 Replacing Null Values With Mean	28
Figure 4. 8 Removing The Units From Attribute Values I.E. Price	29
Figure 4. 9 Dropping New_Price Attribute	29
Figure 4. 10 Dropping Name Attribute	29
Figure 4. 11 Transforming Categorical Variables	30
Figure 4. 12 Transforming Data With Object Datatype Into Float Datatype	30
Figure 4. 13 Cross Validation Performed In Linear Regression	31
Figure 4. 14 Refined Class Diagram Of Carcompanion	32
Figure 4. 15 Refined Sequence Diagram Of Carcompanion	34
Figure 4. 16 Refined Activity Diagram Of Carcompanion	35
Figure 4. 17 Component Diagram Of Carcompanion	37
Figure 4. 18 Deployment Diagram Of Carcompanion	38
Figure 4. 19 Random Forest Regression	39

Figure 4. 20 Bagging Regression.....	40
Figure 5. 1 Implementation Of Registration Module.....	44
Figure 5. 2 Implementation Of Login Module.....	45
Figure 5. 3 Implementation Of Car Details Module	45
Figure 5. 4 Implementation Of View Car Details Module	46
Figure 5. 5 Generating Bootstrap Dataset From The Random Samples.....	46
Figure 5. 6 Generating Decision Trees From The Bootstrap Dataset.....	47
Figure 5. 7 Predictions From Each Decision Trees And Their Average	47
Figure 5. 8 Generating The Random Samples From The Original Dataset.....	47
Figure 5. 9 Feeding Sample Datasets Into The Model.....	48
Figure 5. 10 Predictions From Each Models And Their Average	48
Figure 5. 11 Initialization Of Instance Variables.....	48
Figure 5. 12 Training Linear Regression	49
Figure 5. 13 Calculation Of Predicted Target Values.....	49
Figure 5. 14 Computation Of Gradients	49
Figure 5. 15 Prediction Of Target Values	49
Figure 5. 16 Model Testing For Accurate Price Prediction	53
Figure 5. 17 Model Saving And Deployment.....	54
Figure 5. 18 Prediction Form Test	55
Figure 5. 19 Web App Testing.....	56
Figure 5. 20 R2 Score Obtained From Linear Regression.....	57
Figure 5. 21 R2 Score Obtained From Bagging Regression.....	57
Figure 5. 22 R2 Score Obtained From Random Forest Regression.....	57
Figure 5. 23 The Mse, Rmse, And Mae Of Linear Regression	58
Figure 5. 24 The Mse, Rmse, And Mae Of Bagging Regression	59
Figure 5. 25 The Mse, Rmse, And Mae Of Random Forest Regression	59
Figure 5. 26 Result Analysis.....	60

LIST OF TABLES

Table 3. 1 Hardware Requirements	14
Table 3. 2 Software Requirements.....	14
Table 3. 3 Schedule Table.....	15
Table 5. 1 Test Cases For Unit Testing (Test Case For Login And Registration).....	50
Table 5. 2 Test Cases For System Testing	51
Table 5. 3 Results Comparison	60

CHAPTER 1

INTRODUCTION

1.1. Introduction

The global used cars market size was valued at \$1.4 trillion in 2021, and is projected to reach \$2.6 trillion by 2031, growing at a CAGR of 6.5% from 2022 to 2031 [1]. Every single day, thousands of pre-owned cars are sold around the world. On that note, the secondhand car price prediction system is of high interest and is in urgent need. Especially in developing countries, because of the affordability of used cars, maximum people tend to purchase pre-owned cars.

In today's scenario of Nepal, if anyone wants to buy a secondhand car, he/she either approach an agent to find the market prices or research on different websites or physically visits the store. Still, no one can ensure that the money the buyer invest is worthy. Similarly, used car prices are not constant in the market. Determining if the listed price of used cars is accurate or not is challenging, as multiple factors drive the price of secondhand cars. Buyers, sellers as well as business professionals such as car dealers, lenders, and insurance companies need an intelligent system that will allow them to predict the correct price efficiently.

Thus, the CarCompanion is a Web App designed to accurately predict the price of pre-owned cars based on various features directing towards informed purchases. The Machine Learning model- Random Forest Regression trained on the past data precisely forecasts the cost of a used car on the basis of the features fed by the users to the system like model of the car, year, mileage, engine, owner type, transmission type, fuel type, and power. Once the user is registered, CarCompanion allows the user to feed the inputs to the system and predicts the car price. Furthermore, the system also keeps the record of every predictions for future insights.

1.2. Problem Statement

In this 21st century, everyone wants their own car. However, due to the rising cost of new cars, not everyone can afford to buy a brand new one. That is why buying used cars is increasing worldwide. However, buying a secondhand car from a dealer can be both frustrating and unsatisfying. Because not all dealers offer a worthy price. Some dealers may use wrong sales strategies to close the deal.

Similarly, the existing system involves a process where the seller decides the price randomly, without knowing in advance the current value of the car or the price at which he should sell it. Similarly, the buyer has minimal understanding of the used car and its current value.

Thus, the transaction between buyers and sellers is never exact because there is no basis on which used cars can be bought and sold at an exact price. However, there are many scams and market irregularities in dealing with secondhand cars.

The purpose of the "CarCompanion" application is to accurately predict the price of a used car based on the available facts. This research is important because it helps both car buyers and sellers determine what a fair price is for a used car.

1.3. Objectives

- To develop a user-friendly web-application that takes the cars' features and predicts the prices of the cars using Random Forest Regression.
- To provide valuable insights for individuals who are buying or selling secondhand cars.

1.4. Scope and Limitation

Scopes:

- The system provides predictions about used automobile pricing based on a variety of variables, including model, year, kilometers driven, mileage, engine, owner type, transmission type, fuel type, power, and seats.
- The system is beneficial for car dealerships and auction houses to estimate the value of used cars they have in their inventory. Furthermore, insurance companies can use this system to determine the value of a used car for insurance purposes.
- In addition, a used car price prediction system can be integrated with other systems such as online marketplaces, e-commerce sites, and mobile applications to provide users with real-time price estimates for used cars.

Limitations:

- Inadequate features of the cars have been the major limitation which hinders the efficiency of predicted price.
- The predictions do not consider external factors such as fuel prices, consumer confidence, and economic conditions that can affect the car market.
- The project only considers cars that are in running condition and not damaged or inoperable vehicles.
- The limited number of available cars and the estimation of new models or recently introduced cars can be challenging.

1.5. Development Methodology

In this project incremental method has been used as the developmental method. The project is built and delivered in small and incremental portions rather than all at once. Each increment builds upon the previous one and adds new functionality, gradually leading to the completion of the final product. Incremental method follows the following processes:

i. Increment 1:

- **Data Collection:** The problem to be solved is defined and sources are identified. The data is collected from Kaggle; a professional online community for data scientists and machine learning practitioners. The data is in 'csv (comma separated values)' format. This data include the model, year, kilometers driven, fuel type, mileage and many other additional features of a car that might impact the price.
- **EDA:** EDA involves analyzing the data to gain insights into the relationships between the different variables and to identify any trends or patterns.
- **Data Cleaning:** Next, cleaning of the data is done. Data cleaning involves removing any missing values or incorrect data and transforming the data into a format that can be used for analysis.

ii. Increment 2:

- **Feature Engineering and Selection:** Once the data is cleaned, the relevant data that helps in predicting the price of a secondhand cars are selected and engineered. It is one of the important steps as only the model trained on relevant data can give accurate output.

iii. Increment 3:

- **Model Training:** The next step is to train the model on the cleaned and transformed data. The parameters of the model to optimize its accuracy are adjusted. Random Forest Regression, Bagging Regression, and Linear Regression, are trained on the data.
- **Model Selection and Evaluation:** Finally, the appropriate model for the project is selected .i.e. Random Forest Regression is selected. This is done by evaluating different models and choosing the one that provides the best accuracy.

iv. Increment 4:

- **Model Validation:** It is important to validate the accuracy of the model by

testing it on a separate dataset. Thus model validation is performed with a purpose to identify any overfitting or underfitting that may have occurred during the training process.

- **Model Optimization:** The model is optimized by hyper parameter tuning of the selected algorithm and feature selection to improve the accuracy of the model.

v. **Increment 5:**

- **Model Deployment:** Therefore, the model is ready for deployment. This is done by integrating the model into a web-based platform that is be used to make predictions on new data.
- **Monitoring:** Lastly, the performance of the deployed model is to be monitored over time and necessary adjustments are to be made.

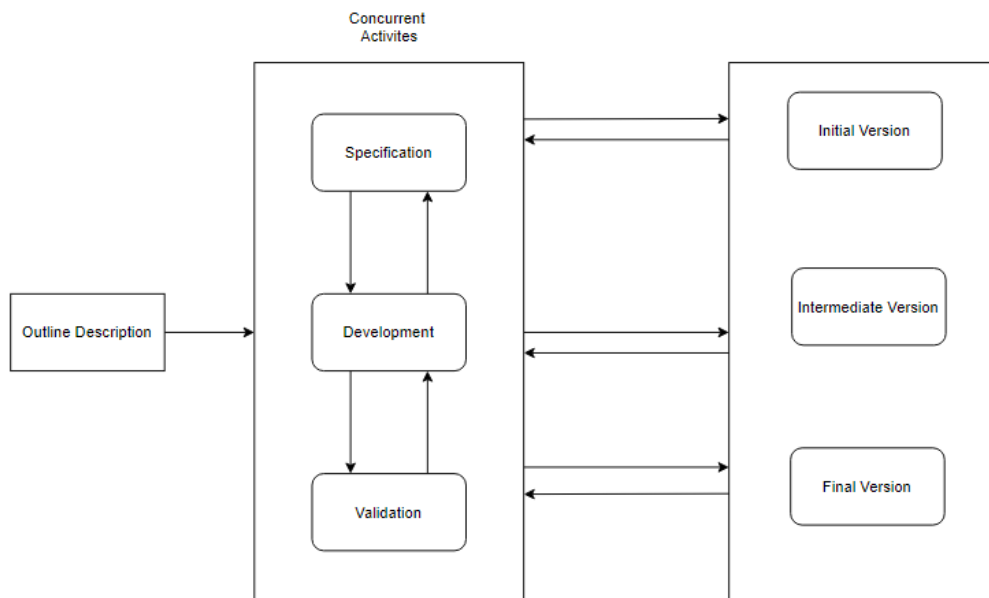


Figure 1. 1 Incremental Development Method

Source: [CS 410/510 - Software Engineering class notes \(ccsu.edu\)](#)

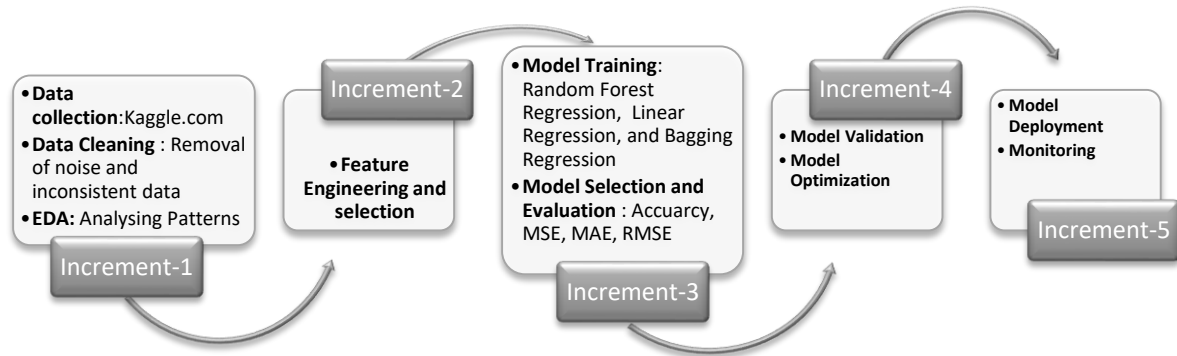


Figure 1. 2 Development Methodology

1.6. Report Organization

The report on “CarCompanion” comprises in six chapters.

Chapter1 provides the introduction regarding the project along with its development methodology, scopes and limitations.

Chapter2 describes background study of the present system available related to the car trading and briefing on the predictive analysis of the second hand car price prediction system.

Chapter3 provides an overview of all the functional and nonfunctional requirements binding the operation of the system along with analysis of the feasibility study of the system.

Chapter 4 portrays the diagram representing UML which describes the core design of the system alongside outline of the algorithms used to develop the system.

Chapter 5 contains a detailed description of the implementation phase and model phases of different modules. Moreover, this section also defines the testing processes providing an analytical result.

Chapter 6 provides summary of the system that has been developed and future possibility for the system to make it more sustainable, reliable, and efficient.

CHAPTER 2

BACKGROUND STUDY AND LITERATURE REVIEW

2.1. Background Study

According to the regional analysis of the market, the Asia-Pacific used cars market share is anticipated to grow at a CAGR of 7.6%, by generating a revenue of \$546.7 billion during the forecast period. The depreciation of new vehicles in terms of resale value after a period of time is a key driving factor that is expected to augment the demand for used cars and increase the growth of the used cars market size. The COVID-19 pandemic has enforced the companies in the used cars industry to adopt new methods of operation such as the use of online channels for the sales of used cars and it has positively impacted the used cars market. Carvana's sales, which sells only used cars online, increased 37% in the year 2020 as compared to 2019, according to CNN. These factors have positively impacted the global used cars market [2].

However, there is not any standard way for determining whether the price of used cars is accurate or not. The used car market is large and diverse, with a wide range of vehicles available at different prices. Factors such as age, mileage, model etc. significantly impact the price of a used car. However, predicting the exact price of a used car is challenging due to the complex and dynamic nature of the market. To address this challenge, various studies have been conducted to understand the impact of different factors on the price of a used car. These studies have found that factors such as the make and model of the car, the year of manufacture, the mileage, and the condition of the car are crucial in determining the price of a used car.

Based on these findings, we will be using a combination of these factors to develop a machine learning model that can accurately predict the price of a used car. The model will be trained using a large dataset of used cars, and the accuracy of the model will be evaluated using various metrics. The results of this project will provide valuable insights into the factors that impact the price of a used car and will help buyers, sellers, and dealers in determining used car value.

2.2. Literature Review

Foremost, the investigation of the application of supervised machine learning techniques to predict the price of used cars in Mauritius have been conducted. The predictions are based on historical data collected from daily newspapers. Different techniques like multiple linear regression analysis, k-nearest neighbors, naïve bayes and decision trees have been used to make the predictions [3].

Similarly, several studies and related works have been done previously to predict used car prices around the world using different methodologies and approaches, with varying results of accuracy from 50% to 90%. Using the random forest technique, the best accuracy to date is 83.63% on the Kaggle dataset. According to the thesis submitted by Abdulla AISHared at Rochester Institute of Technology, Dubai, all possible outcomes of a prediction were gathered and randomized with Bagging Regression. A better model was formed by combining many weak models. Bagging was used with the goal of reducing variance. In the thesis, the Bagging Regression of decision trees was trained with the intercept property where R-square of training data was almost 88%. [4].

According to some other research where linear and random forest regressions were used to predict the price of the pre-owned cars, the correlation study was done between the price and continuous variables. The importance of this study was to identify any high correlations between any variables, so that they can be used for further study. The correlation of price was high with variables Power, Kilometer, and Age with correlations 0.575, 0.440, and 0.338 respectively. Thus, these variables were considered for further study. The train data R Squared value was 0.780728326697251 with Linear Regression. Similarly for Random Forest Regression, the train data R squared value is 0.9201854511589997. Hence, in this research, the dataset was predicted over two regression methods—linear regression and random forest. In these two methods, random forest method fits more to the test data comparing to the linear regression. This can be extended by replacing the missing values with some imputed values. [5]

Similarly, Pal et al. used random forest to estimate the price of used cars, the model was chosen after careful exploration data analysis to determine the effect of each feature on the price, a random forest with 500 decision trees was created to train the data, from the test results, it was found that the accuracy of the training was 95.82% and the test accuracy was 83.63%, the model was able to accurately estimate the price of the cars by choosing the most suitable features [6].

At present, cardekho.com, carwale.com and zigwheels.com help user to know the prices of used car and is also used for retailing pre-owned cars. Basically, these applications serve as a website as well as mobile application that facilitates its users to buy right vehicles for them. They do this through a user-friendly website and mobile app that carry rich automotive content such as expert reviews, details of all specs and prices, comparisons as well as videos and pictures of different models that available in India.

CarDekho is a classified portal owned by Girnar Software and has tie-ups with many auto manufacturers, more than 4000 car dealers, and numerous financial institutions to facilitate the purchase of vehicles [7].

Similarly, CarWale is one of the India's leading source of new car pricing and other cars related information. Part of CarTrade Tech, CarWale's mission is to bring delight in car buying, offering a bouquet of reliable tools and services to help car consumers decide on buying the right car, at the right price and from the right partner [8].

Furthermore, Zigwheels is the best car & bike research app in India to buy, sell and compare cars and bikes, be it new or used. It gives you in-depth information, price alerts and offers about new cars, used cars, new motorcycles and scooters. You can compare any two cars or any two bikes to decide which car or bike to buy [9].

However, in context of Nepal, we still lack an online system specifically designed for secondhand car price prediction.

CHAPTER 3

SYSTEM ANALYSIS

3.1. System Analysis

System analysis is the process of collecting and evaluating data, defining problems and breaking down the system into components. It describes what the system must do. It is also considered a problem-solving technique that divides a system into parts and analyzes how well those parts work and interact with each other to achieve what is desired result. This is done by examining the system or its components to determine its goals.

3.1.1. Requirement Analysis

Requirement analysis is a technique to identify the expectations and goals of a new system. Requirement analysis is one of the most important processes for determining if a system or software project will be successful or not. The requirements for this system are mentioned below:

i. Functional Requirements

Functional requirements are those specifications or tasks that must be carried out by the software in order for it to satisfy user needs. These are the specifications that the system must meet in order to satisfy the end user's basic needs.

Functional requirements of a system consists a Use Case diagram. Basically, a use case diagram is a particular kind of UML (Unified Modeling Language) diagram that is used to represent how a system or application behaves from the perspectives of its users. Using CarCompanion's functional criteria as a guide, these are the functional requirements for a system to function as expected:

a. Authentication

The application enables new users to register for the service using their email. Once registered, the user can access the system by logging in with the proper credentials to their account. Once the system task is finished, the user can logout.

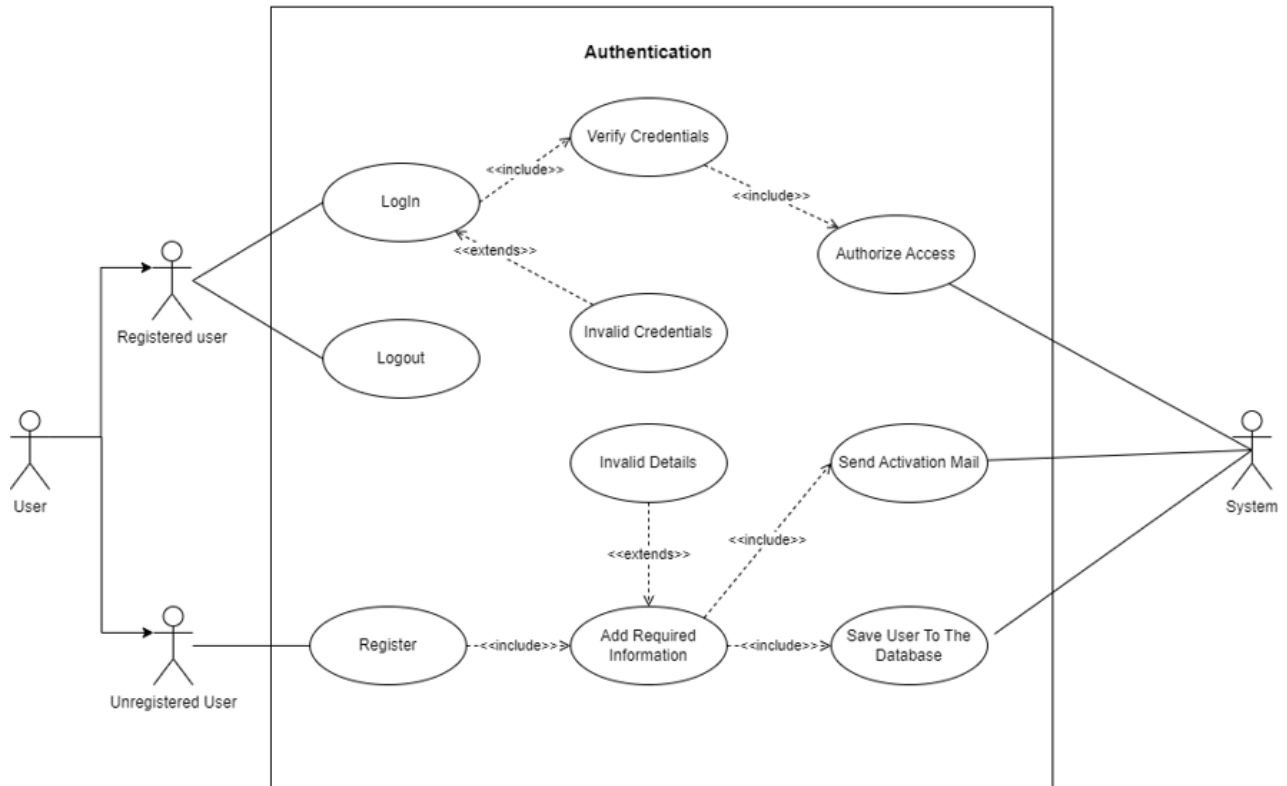


Figure 3. 1 Use Case Diagram for Authentication

In the above use case diagram for Authentication, foremost the user registers to the system providing the necessary details of the users including username, full name, email, and password. The system validates the credentials of the user and if all the details are valid as per the validation criteria set, the user is asked to login to the system, and with valid login details, the access to the system is granted. However, if the user's registration details are not valid, thus the user is not registered until correct credentials are provided. Similarly, if the login details are not valid, the user is not granted access to the system prompting an "Invalid Credentials" message.

b. Users

The application allows the users to feed the features of car into the system and predict the price. The application makes the necessary validation in the user's input. It then predicts car prices on the basis of the user's input. Once the price is predicted, the application maintains the record of each predictions for further insights and analysis for the users.

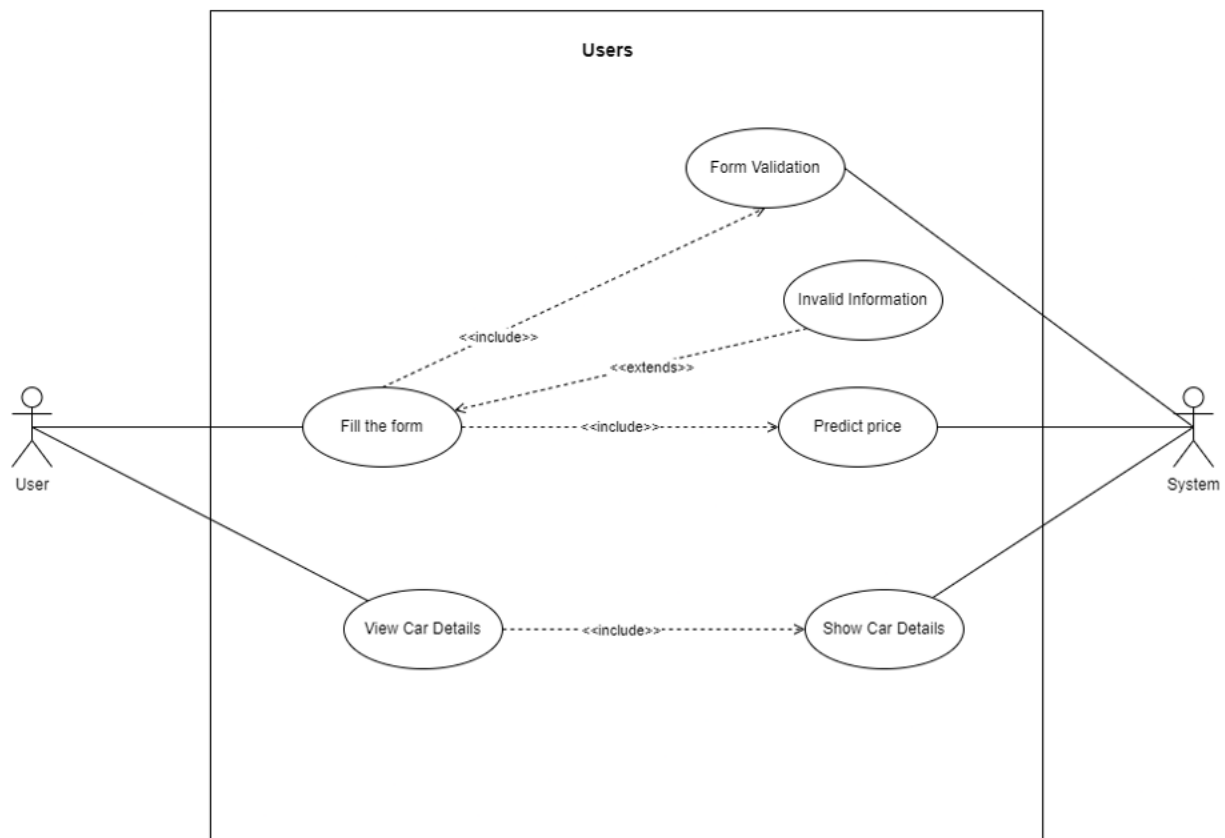


Figure 3. 2 Use Case Diagram for Users

In the above use case diagram for Users, foremost the user fills the form providing the inputs for instance the model of the car, how much kilometers the car has driven, and so on. If all the information provided by the user are correct, the system predicts the price of a car. However, the system prompts validation message of invalid information, if the user inputs are not correct. Thus, a user can also view the details of the cars whose price has been predicted.

ii. Non-Functional Requirements

Any needs that describe how the system performs a certain function are referred to as non-functional requirements. Non-functional need will explain how a system should act. They give specifics regarding the criteria used to compare a system's performance to various behaviors.

- a. **Performance:** The system provides accurate price estimates for used cars in a timely manner, regardless of the size of the dataset.
- b. **Reliability:** The system is available 24/7 and have a minimal downtime to ensure that users can access it whenever they need it.
- c. **Usability:** The system is user-friendly and easy to use, with a simple and intuitive interface.
- d. **Security:** The system is secure and protect user data and information, with proper access controls and encryption mechanisms.
- e. **Maintainability:** The system is easy to maintain and update, with clear documentation and support for future modifications.
- f. **Scalability:** The system is able to handle a large number of users and adapt to changes in demand over time.
- g. **Compatibility:** The system is compatible with different devices and operating systems to ensure maximum usability for users.
- h. **Accuracy:** The system provides accurate price predictions based on the data available and the model used, with a low margin of error.

3.1.2. Feasibility Analysis

The proposed project's viability is determined by the feasibility study. A feasibility study makes it possible to evaluate the project's success and reduce its risks. Before beginning the project, following feasibility studies were carried out which are as follows:

i. Technical Feasibility

All the tools and software product that are required for this project are easily available in the internet. The technical resources required for our application is tabulated below:

Table 3. 1 Hardware Requirements

Hardware Requirements	
Device	Laptop
Connection	Active and stable internet connection
Processor	i3,i5 or i7
RAM	Minimum 4GB and multi-core

Table 3. 2 Software Requirements

Software Requirements	
Operating System	Windows
Programming Language	Python
DBMS	Db.sqlite3
Web Development Tools	HTML, CSS
Machine Learning Libraries	scikit-learn, pandas, numpy, sklearn
Data Visualization Tools	matplotlib, seaborn
Version Control System	Git
Web Browsers	Google, Firefox, Edge
Framework	Django
Project Management Tool	Trello

ii. Operational Feasibility

Because the system's user interface is simple to use and accessible, this application is operationally viable.

iii. Economic Feasibility

Most computers generally have the necessary hardware and software resources

required for the project. Because the planned project does not necessitate the purchase of additional equipment and meets all necessary requirements without adding additional costs, the system is economically viable. Moreover, to lower development costs, open-source software and libraries can also be used.

iv. **Schedule Feasibility**

The project's total timeframe is shown in the Gantt chart below. It provides a chronological breakdown of the project's tasks along with the length of time required for each activity. The report preparation phase was run concurrently with the above phases from planning to test. Therefore, the project has been developed as per the following time schedule, thus our application is scheduled feasible:

Table 3. 3 Schedule Table

Task	Start Date	Days to Complete
Project Planning	11/7/2022	10
Project Analysis	11/17/2022	8
Project Design	11/25/2022	13
Implementation	12/8/2022	38
Testing	1/15/2023	23
Documentation	11/7/2022	92

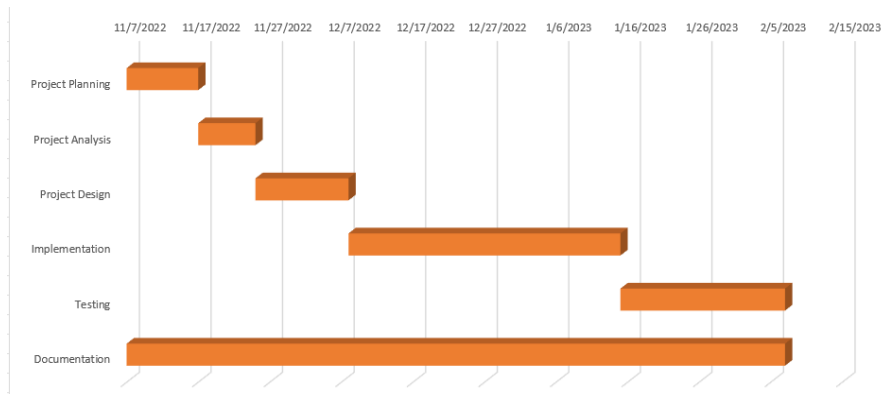


Figure 3. 3 Gantt Chart

3.1.3. Analysis

3.1.3.1. Object modelling using Class Diagram

Class diagram in the UML is a type of static structure diagram that describes the structure of a system, it shows system's classes along with their attributes, operations and relationships among objects.

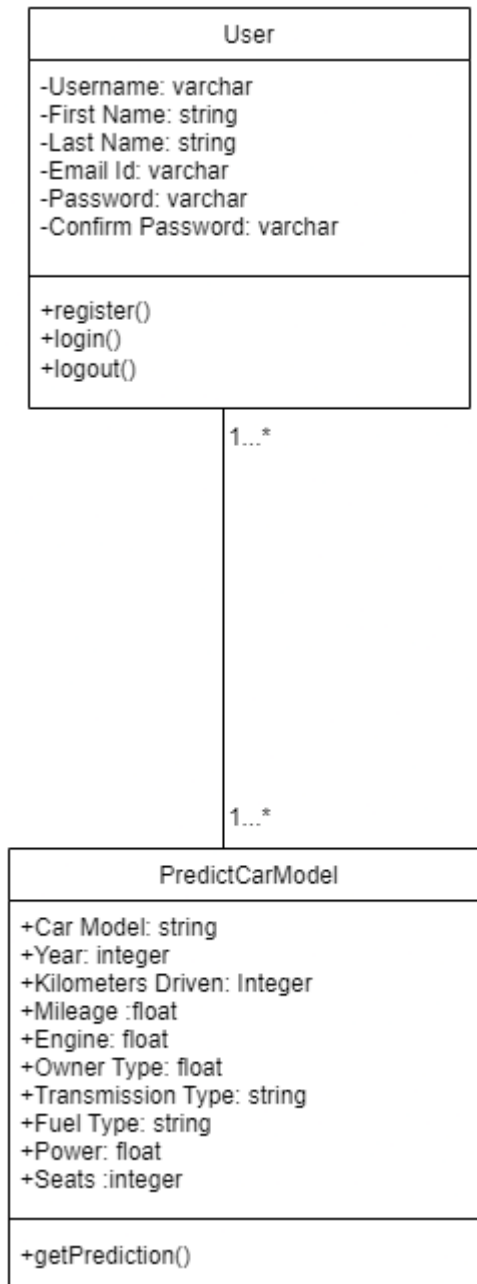


Figure 3. 4 Class Diagram of CarCompanion

In the above class diagram of CarCompanion, we have two classes as User and PredictCarModel. The User class represents the user of a system with the attributes as User Name, First Name, Last Name, Email Id, Password, and Confirm Password. These attributes are private to each users and are used to authenticate the users. On the other hand, the class PredictCarModel has the public attributes as Car Model, Year, Kilometers Driven, Mileage, Engine, Owner Type, Transmission Type, Fuel Type, Power, and Seats on the basis of which the price of a secondhand car is predicted.

The User class has the following methods as:

- a. **register():** this method takes the users' user name, first name, last name, email, and password and verifies the credentials to register.
- b. **login():** this method takes the users' username and password and verifies the credentials to login.
- c. **logout():** logs out the current user from the system.

Moreover, the PredictCarModel has the following method:

- a. **getPrediction():** This method takes the car objects as input and predicts the car price.

3.1.3.2. Dynamic modelling using Sequence Diagram

Sequence diagram in the UML is a type diagram that illustrates the sequence of messages between objects in an interaction. It consists of a group of objects represented by lifelines and message they exchange during the interaction denoted by arrow symbols.

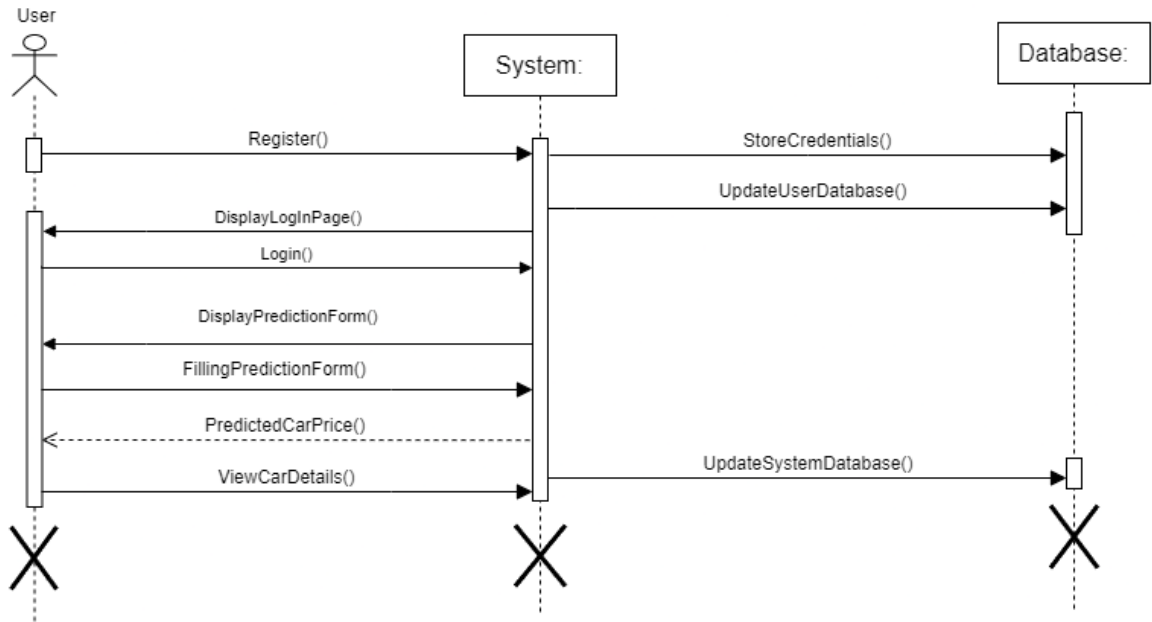


Figure 3. 5 Sequence Diagram of CarCompanion

In the above sequence diagram, we have three different components with their respective lifelines and they are User, System, and Database. Foremost the user registers to the system using the users' credentials. Those credentials are stored by the database. Once the database is updated, the user is requested to log in. Then the user logs in to the system and the system displays the prediction form. The users fill the prediction form and hence the price of a secondhand car is predicted. The database is then updated, hence the user can view the details of the car.

3.1.3.3. Dynamic Modelling using State Diagram

In UML, a state diagram, often referred to as a state machine diagram or state chart diagram, is a representation of the states an object can attain as well as the transitions between those states.

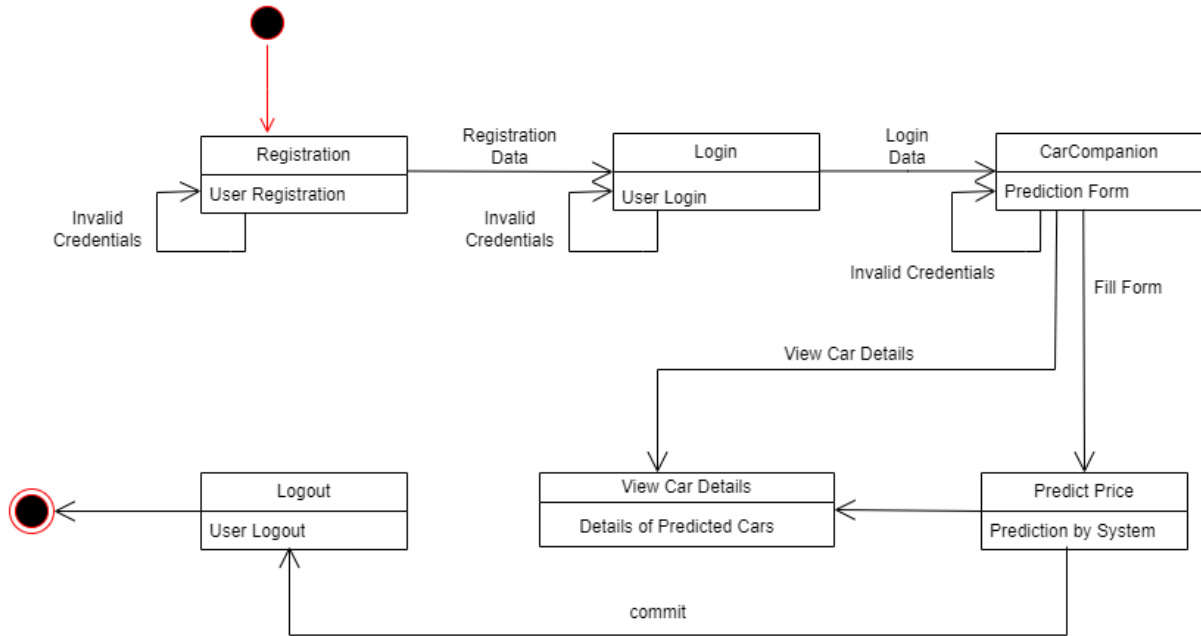


Figure 3. 6 State Diagram of CarCompanion

The above figure represents the state diagram of CarCompanion. The diagram consists of different states named as: Registration, Login, System (CarCompanion), Predict Price, View Car Details and Logout which are connected with each other with different path known as transition and performing different activities like data registration, filling form, committing etc. Each transition consists different message which are transmitted to another state for further process.

3.1.3.4. Process Modeling using Activity Diagram

Activity diagram in the UML is a type of diagram that visually presents a series of actions or flow of control in a system. It represents the workflow of activities with support for choice, iteration and concurrency.

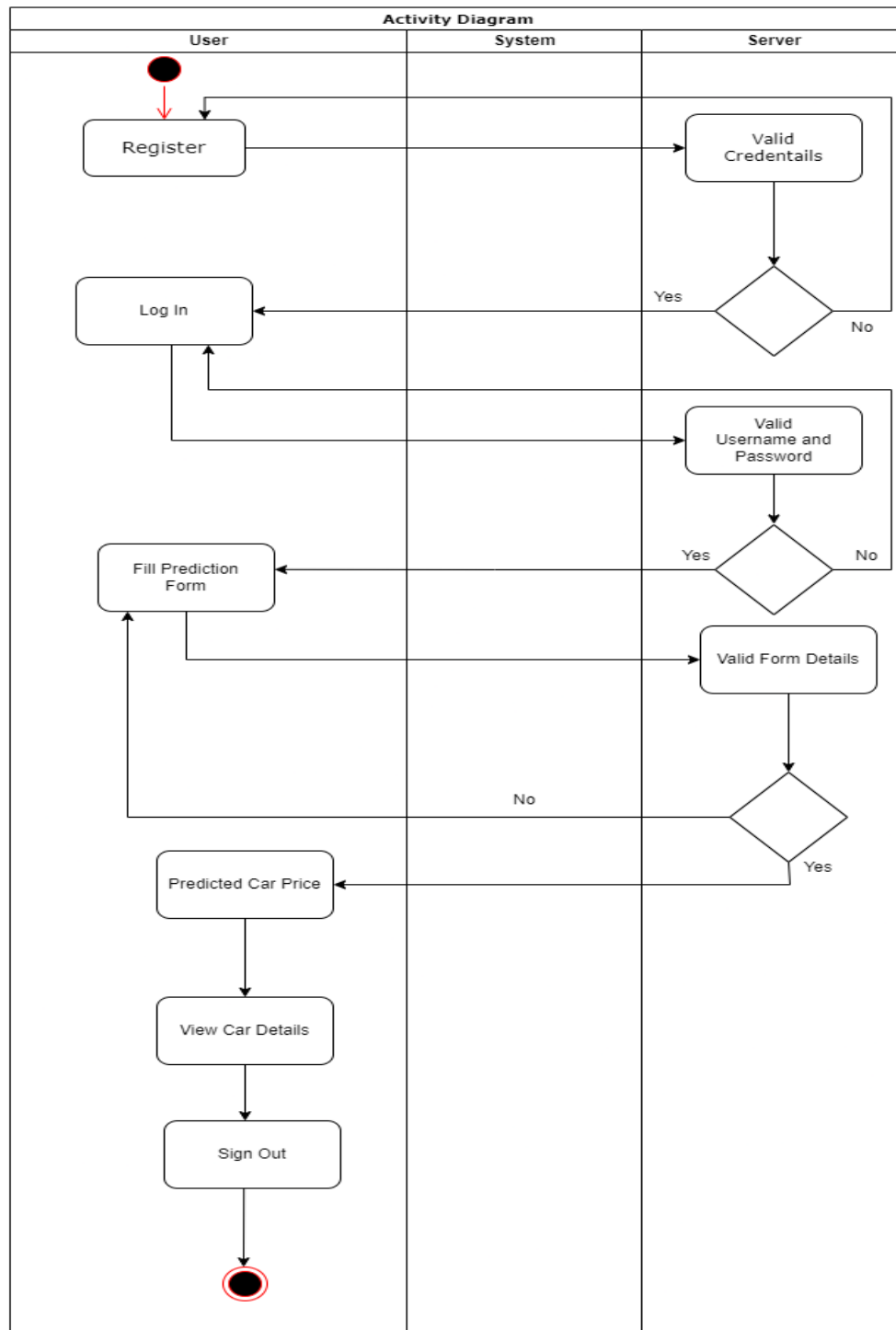


Figure 3. 7 Activity Diagram of CarCompanion

In above activity diagram, first the users register to the system providing the required credentials. The authentication is performed by the server. If the users' registration credentials

are not valid, the user is asked to provide valid credentials. However, if the users' registration credentials are valid, then the users are permitted to log in. Users then log in to the system. The validation of Username and Password takes place where the users only with the correct Username and Password are permitted to get logged in to the system. Once the users are logged in to the system, they fill the prediction form providing the features of the car as required by the system. Each details of the form go through the validation by server. Only the form that has been filled correctly predicts the price of a car. Therefore, the user views the details of the predicted cars and hence signs out from the system.

CHAPTER 4

SYSTEM DESIGN

4.1. Design

System Design is the process representing the elements of the system like architecture, modules and components with their interfaces that are included in the system. The main aim of system design is to provide information about the system and its elements.

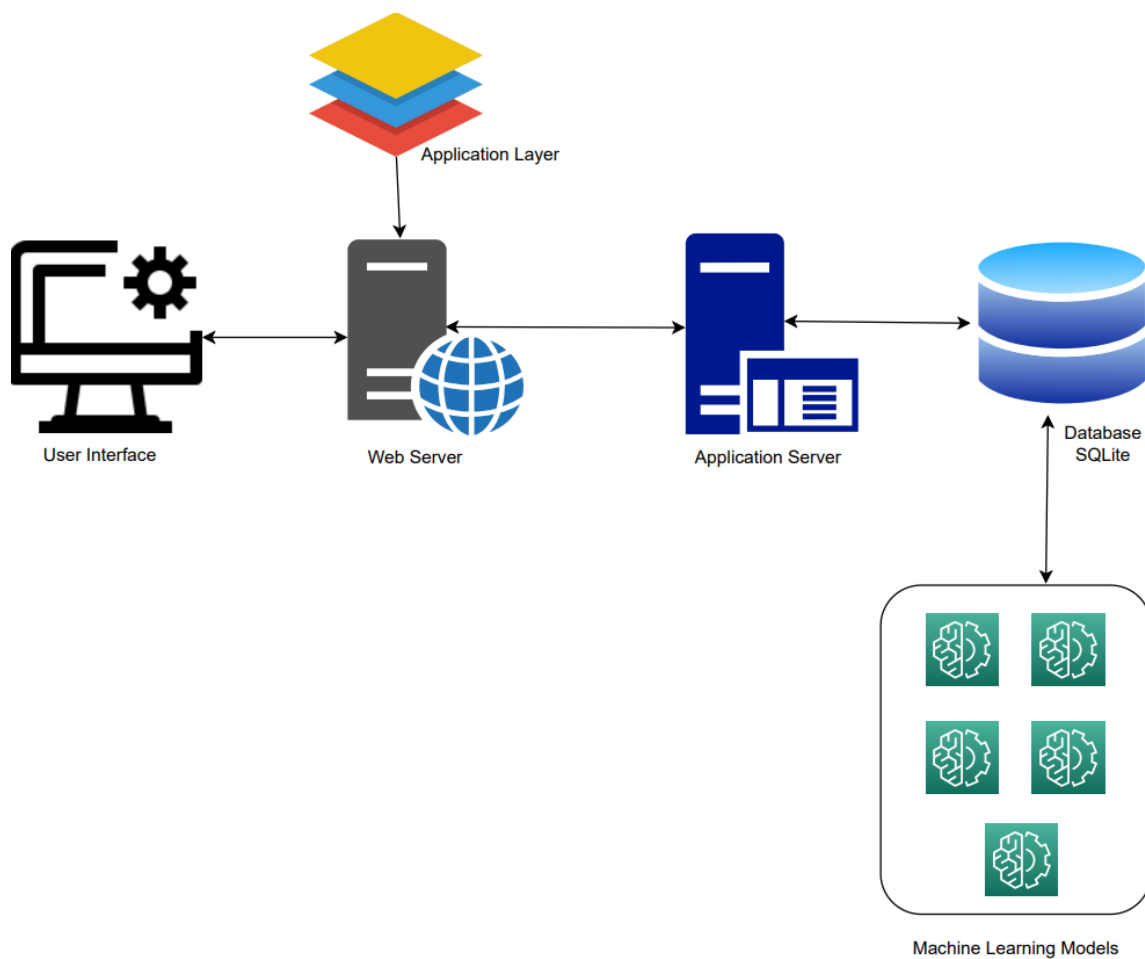


Figure 4. 1 System Architecture of CarCompanion

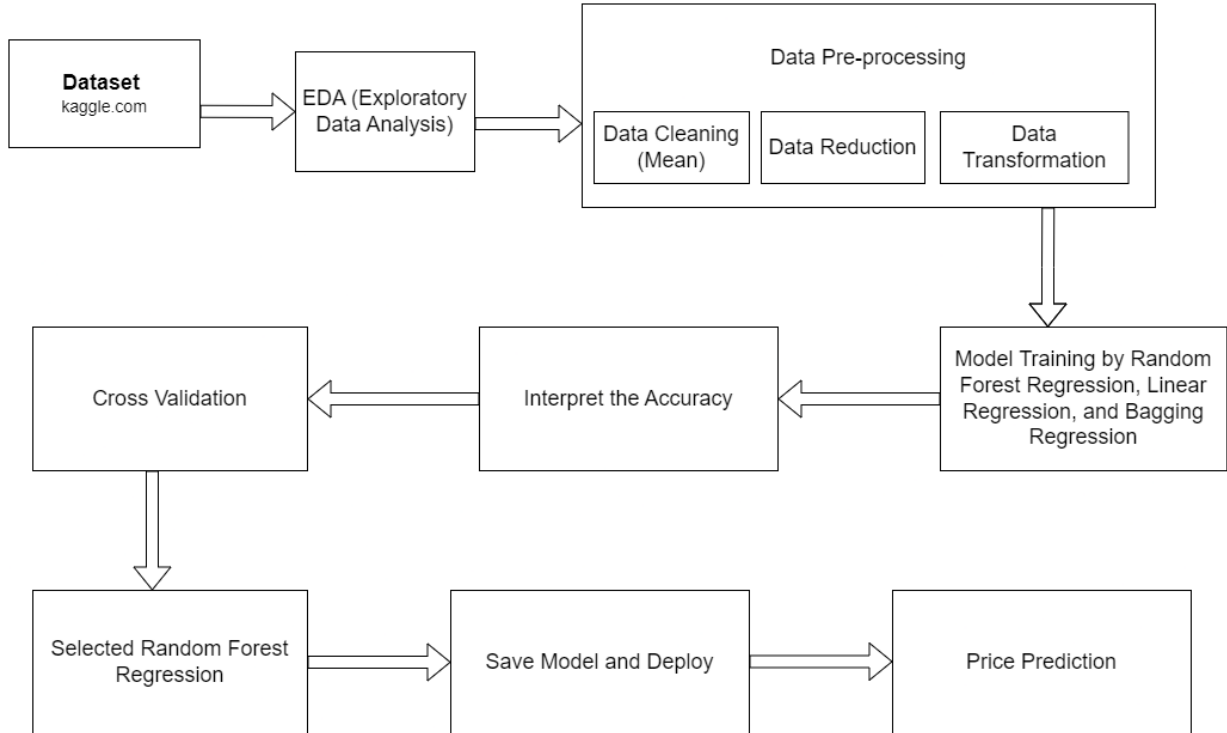


Figure 4. 2 High-Level System Design of CarCompanion

The high level system design of CarCompanion is shown in the above figure. The following steps were carried out throughout the project development:

1. Data Collection

Foremost, the dataset, train-data.csv of Used Car Price Prediction is collected from kaggle.com, a professional online community for data scientists and machine learning practitioners.

data

[154]

Python

...

	Name	Year	Kilometers_Driven	Fuel_Type	Transmission	Owner_Type	Mileage	Engine	Power	Seats	Price	New_Price
0	Maruti Wagon R LXI CNG	2010	72000	CNG	Manual	First	26.6 km/kg	998 CC	58.16 bhp	5.0	\$2,113.53	NaN
1	Hyundai Creta 1.6 CRDi SX Option	2015	41000	Diesel	Manual	First	19.67 kmpl	1582 CC	126.2 bhp	5.0	\$15,096.62	NaN
2	Honda Jazz V	2011	46000	Petrol	Manual	First	18.2 kmpl	1199 CC	88.7 bhp	5.0	\$5,434.78	\$10,398.55
3	Maruti Ertiga VDI	2012	87000	Diesel	Manual	First	20.77 kmpl	1248 CC	88.76 bhp	7.0	\$7,246.38	NaN
4	Audi A4 New 2.0 TDI Multitronic	2013	40670	Diesel	Automatic	Second	15.2 kmpl	1968 CC	140.8 bhp	5.0	\$21,425.12	NaN
...
6014	Maruti Swift VDI	2014	27365	Diesel	Manual	First	28.4 kmpl	1248 CC	74 bhp	5.0	\$5,736.72	\$9,516.91
6015	Hyundai Xcent 1.1 CRDi S	2015	100000	Diesel	Manual	First	24.4 kmpl	1120 CC	71 bhp	5.0	\$4,830.92	NaN
6016	Mahindra Xylo D4 BSIV	2012	55000	Diesel	Manual	Second	14.0 kmpl	2498 CC	112 bhp	8.0	\$3,502.42	NaN
6017	Maruti Wagon R VXI	2013	46000	Petrol	Manual	First	18.9 kmpl	998 CC	67.1 bhp	5.0	\$3,200.48	NaN
6018	Chevrolet Beat Diesel	2011	47000	Diesel	Manual	First	25.44 kmpl	936 CC	57.6 bhp	5.0	\$3,019.32	NaN

6019 rows x 12 columns

Figure 4. 3 Overview of the Dataset

From the above figure, it is seen that there are total 6019 rows and 12 columns in the dataset. The columns named as 'Name', 'Year', 'Kilometers_Diven', 'Fuel_Type', 'Transmission', 'Owner_Type', 'Mileage', 'Engine', 'Power', 'Seats' are the independent variables, however the column named 'Price' is the dependent or target variable. The target variable is dependent upon each of these independent variables. Any changes in these independent variables also changes the target variable.

2. EDA

Then the insights of the data were gathered performing EDA. Descriptive statistics and visualization were used to understand the overall structure and characteristics of the data.

```
data.info()

[156]

... <class 'pandas.core.frame.DataFrame'>
RangeIndex: 6019 entries, 0 to 6018
Data columns (total 12 columns):
#   Column                Non-Null Count  Dtype
---  -
0   Name                   6019 non-null   object
1   Year                   6019 non-null   int64
2   Kilometers_Driven      6019 non-null   int64
3   Fuel_Type              6019 non-null   object
4   Transmission           6019 non-null   object
5   Owner_Type             6019 non-null   object
6   Mileage                6017 non-null   object
7   Engine                 5983 non-null   object
8   Power                  5983 non-null   object
9   Seats                  5977 non-null   float64
10  Price                  6019 non-null   object
11  New_Price              824 non-null    object
dtypes: float64(1), int64(2), object(9)
memory usage: 564.4+ KB
```

Figure 4. 4 Information on the Data

From above figure, it can be analyzed that the columns 'Name', 'Year', and 'Seats' have numeric values, other columns have the DataType as objects. Similarly, different patterns and trends were looked to identify unusual observations if any. Moreover, it can also be analyzed that there are maximum null values in the column 'New_Price' i.e. only 824 values out of 6019 are non-null. Likewise, other three attributes named as 'Mileage', 'Engine', 'Power', and 'Seats' also have null values. These missing data has been visualized below:

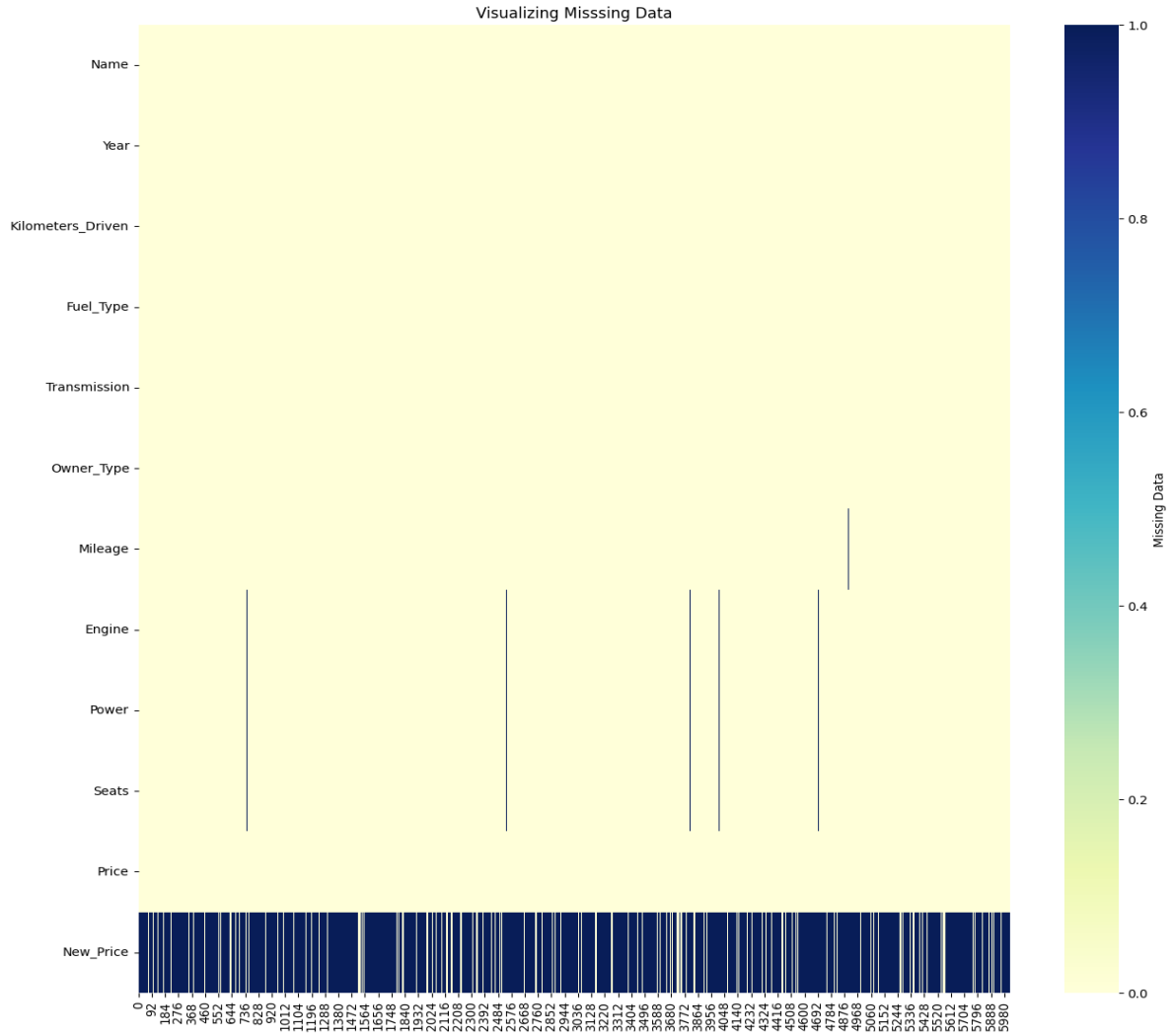


Figure 4. 5 Missing Data Visualization

The figure above visualizes the missing data in the dataset. Even the figure depicts that the attributes 'Mileage', 'Engine', 'Power', 'Seats', and 'New_Price' have null values. It shows that there are maximum null values in New_Price column of the data. However, Engine, Power, Seats, and Mileage columns have minimum null values. These null values are tackled further in the preprocessing phase.

Moreover, before the null values are tackled, we need to understand the numerical features of the data. Null values can be tackled with different ways. It can be tackled by calculating mean, median, mode, and many more. However, before any ways are

applied, it is important that there is information related to its numerical features. To gain that information, the visualization is done as shown below:

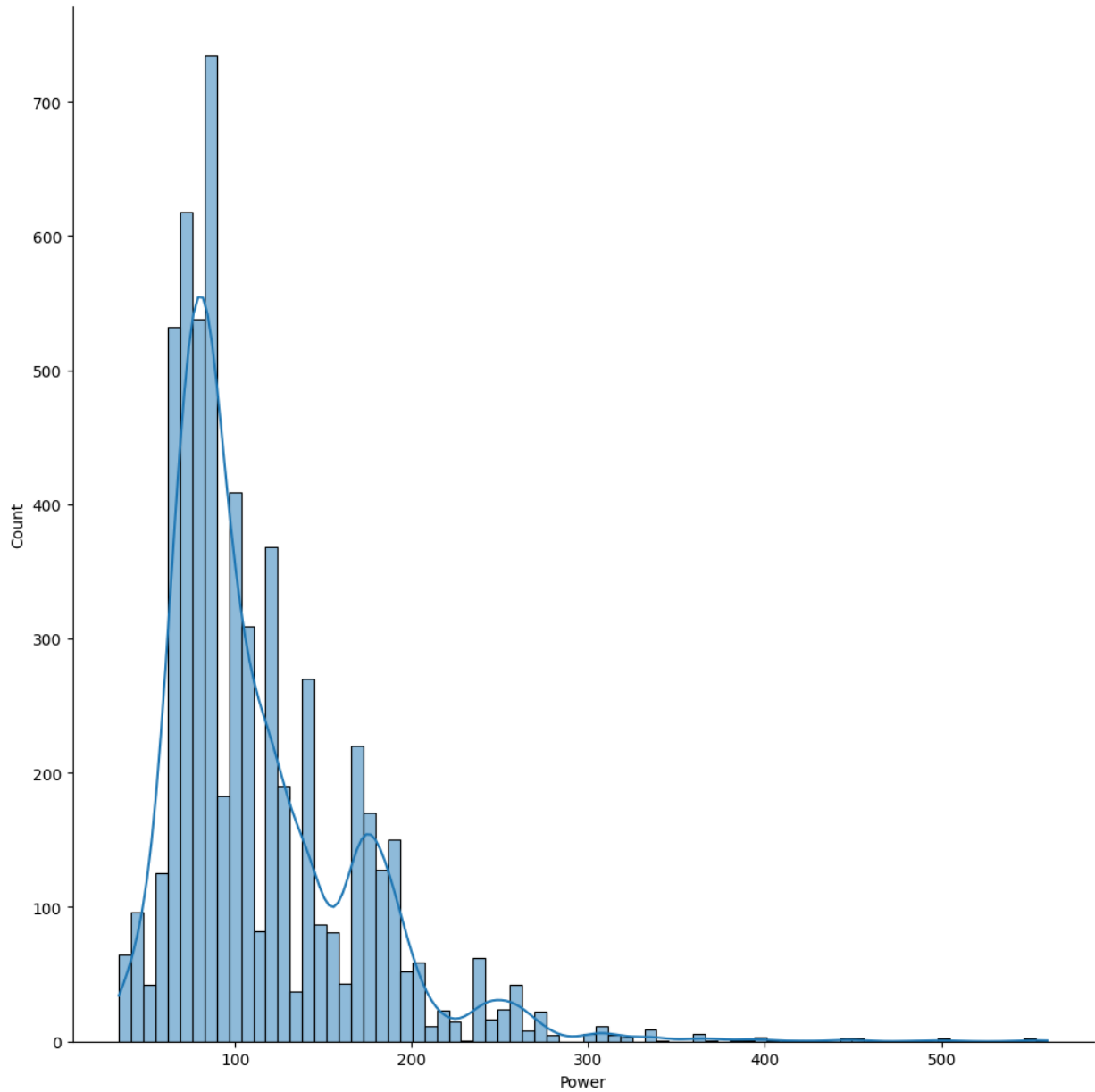


Figure 4. 6 Numerical Features Visualization with Distribution Plot

The above figure visualizes the numerical features of 'Power' column of the dataset using Distribution Plot. It shows that the data is skewed towards right. Thus, from this it can be understood that mean can be used to tackle the null values.

3. Data Pre-Processing

Furthermore, data was preprocessed. Data cleaning, data reduction, and data transformation were performed.

Data Cleaning: The data were cleaned handling missing values. The missing values in 'Engine', 'Mileage', 'Power', and 'Seats' columns were handled calculating mean of the given values and replacing the null values with the mean value.

```
data["Mileage"].fillna(data["Mileage"].astype("float64").mean(), inplace = True)
data["Engine"].fillna(data["Engine"].astype("float64").mean(), inplace = True)
data["Power"].fillna(data["Engine"].astype("float64").mean(), inplace = True)
data["Seats"].fillna(data["Engine"].astype("float64").mean(), inplace = True)
```

[173]

Figure 4. 7 Replacing Null Values with Mean

Here, as shown in the above figure, the mean (average) of the values were calculated in attributes 'Mileage', 'Engine', 'Power', and 'Seats', and then we filled the null positions with that calculated mean value. 'mean()' method calculates the mean and 'fillna' fills the null values with the mean value.

Similarly, the units of the values removed as the machine learning model only takes the numeric values. Any values with the string attached will not be trained. Thus, the units are removed.


```

import re

def remove_currency_symbol(s):
    # Use regular expression to remove any currency symbol
    return re.sub(r'[$,]', '', s)
data['Price'] = data['Price'].apply(remove_currency_symbol)
[193]

```

Figure 4. 8 Removing the Units from Attribute Values i.e. Price

In the above figure, the '\$' unit from the 'Price' attribute is removed using regular expression. And, that updated Price data is applied to the original data.

Data Reduction: Those columns in the dataset that had no significance in predicting the price of a secondhand cars and those with maximum null values were removed with drop() method as shown in the figures below:

Dropping New_Price

As the New_Price column has maximum null values, it is better that we drop this column.

```

data.drop(['New_Price'],axis = 1, inplace = True)
[164]

```

Figure 4. 9 Dropping New_Price Attribute

Dropping Name column

```

data.drop("Name", axis = 1, inplace = True)
[187]

```

Figure 4. 10 Dropping Name Attribute

Thus, the column New_Price of the dataset was removed. Similarly, later Name column was also removed after extracting the Models from the data.

Data Transformation: Those columns that represented the data in the form of string were transformed into the numeric form for training and prediction.

```
[215] data_transformed = oe.transform(data[['Owner_Type', 'Transmission', 'Fuel_Type']])
```

Figure 4. 11 Transforming Categorical Variables

```
[225] data["Price"] = data["Price"].astype(float)
```

Figure 4. 12 Transforming Data with Object DataType into Float DataType

4. Model Training and Accuracy Interpretation

Moreover, Random Forest Regression, Bagging Regression, and Linear Regression were trained on the data whose accuracy were interpreted.

5. Cross Validation

Then, cross validation was also performed to evaluate the accuracy. Cross Validation is the process of dividing the dataset into multiple folds. Among these folds, one of the folds is used for validation set i.e. the test set and the model is trained on the remaining folds. The process is repeated multiple times each time using different fold as a validation set. Lastly, the result is averaged to get the robust estimate of the model.

```
[257] scores = cross_val_score(linearRegression, X_train_scaled, Y_train, cv=5)
print("Cross Validation Score:", scores)

... Cross Validation Score: [0.66668896 0.63373418 0.31838931 0.58369439 0.69184092]

[258] print("%.2f Accuracy" %(scores.mean()))
print("And Standard Deviation of %.2f" %(scores.std()))

... 0.58 Accuracy
And Standard Deviation of 0.14
```

Figure 4. 13 Cross Validation Performed in Linear Regression

6. Model Selection, Saving, and Deployment

Finally, the model with best accuracy was selected and was then saved and deployed, thus predicting the price of a secondhand car.

4.1.1. Refinement of Class Diagram

The refined class diagram is the more detailed form of class diagram with full specification of methods and attributes of a class.

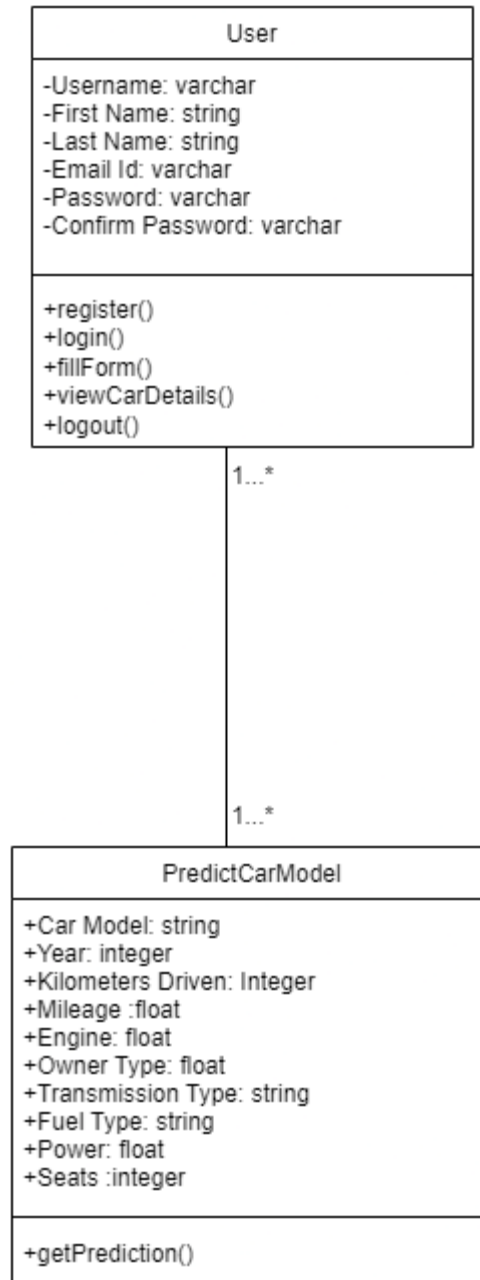


Figure 4. 14 Refined Class Diagram of CarCompanion

The refined class diagram is similar to class diagram with addition of methods to give it more detailed view. In the above refined class diagram of CarCompanion, we have two classes as User and PredictCarModel. The User class represents the user of a system with the attributes

as User Name, First Name, Last Name, Email Id, Password, and Confirm Password. These attributes are private to each users and are used to authenticate the users. On the other hand, the class PredictCarModel has the public attributes as Car Model, Year, Kilometers Driven, Mileage, Engine, Owner Type, Transmission Type, Fuel Type, Power, and Seats on the basis of which the price of a secondhand car is predicted.

The User class has the following methods as:

- a. **register():** this method takes the users' user name, first name, last name, email, and password and verifies the credentials to register.
- b. **login():** this method takes the users' username and password and verifies the credentials to login.
- c. **fillForm():** this method takes the features of the car and verifies those features for prediction.
- d. **viewCarDetails():** this method displays the details of cars whose price has been predicted.
- e. **logout():** logs out the current user from the system.

Moreover, the PredictCarModel has the following method:

- a. **getPrediction():** This method takes the car objects as input and predicts the car price.

4.1.2. Refinement of Sequence Diagram

A refined sequence diagram gives a thorough perspective of the interactions between the components in a system. It displays the order in which these interactions take place as well as the sequence of messages sent and received by various system components or objects.

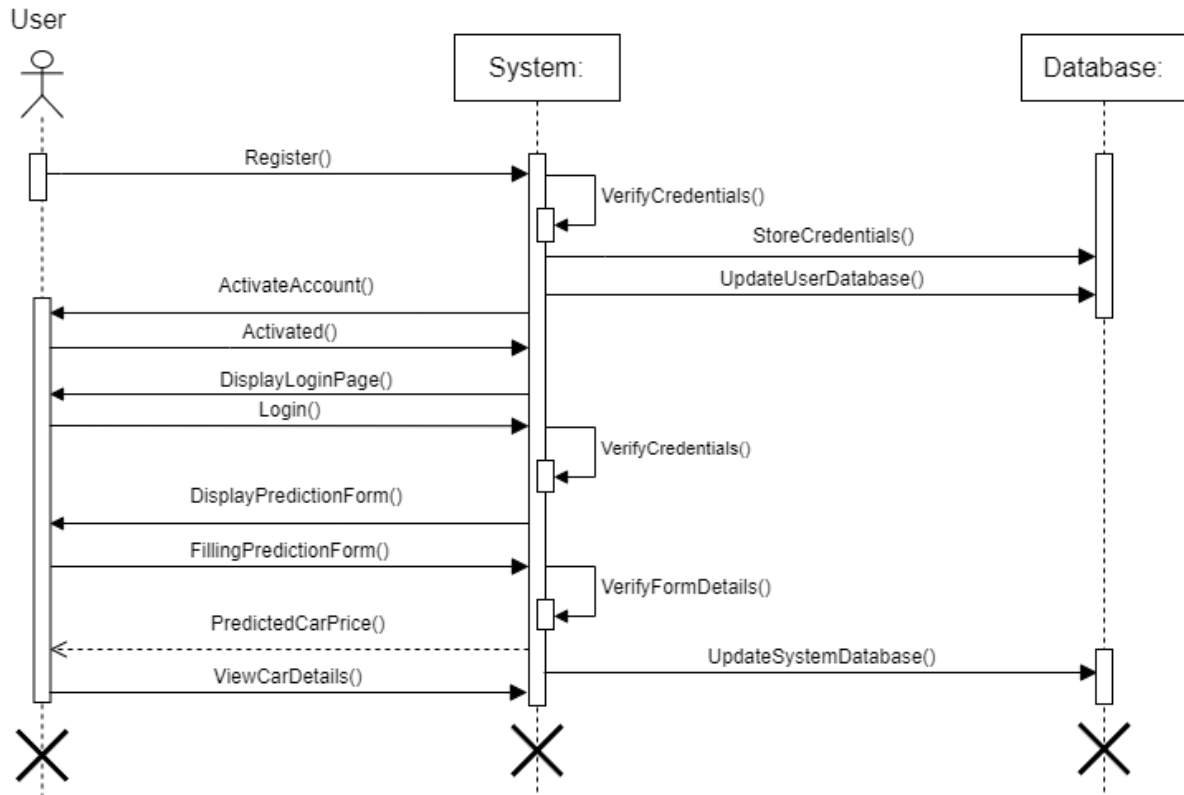


Figure 4. 15 Refined Sequence Diagram of CarCompanion

In the above sequence diagram we have three different components with their respective lifelines and they are User, System, and Database. Foremost the user registers to the system using the users' credentials. The system verifies the users' credentials to registration. If the credentials provided by the users are valid then those credentials are stored by the database and the database is updated. Once the database is updated, the user is sent a link to activate the account. The users activate their account with the activation link and then they are requested to log in to the system. The users log in to the system. Then system then verifies the Username and Password of the users and requests to fill the prediction form. Then the users fill the prediction form whose details are again verified by the system, thus predicting the price of a secondhand car. The database is then updated, hence the user can view the details of the car.

4.1.3. Refinement of Activity Diagram

A refined activity diagram provides a more thorough illustration of the activities and steps involved in the process or workflow. By adding extra information and processes to the basic activity diagram, it expands upon it and enables a more in-depth understanding of the activities.

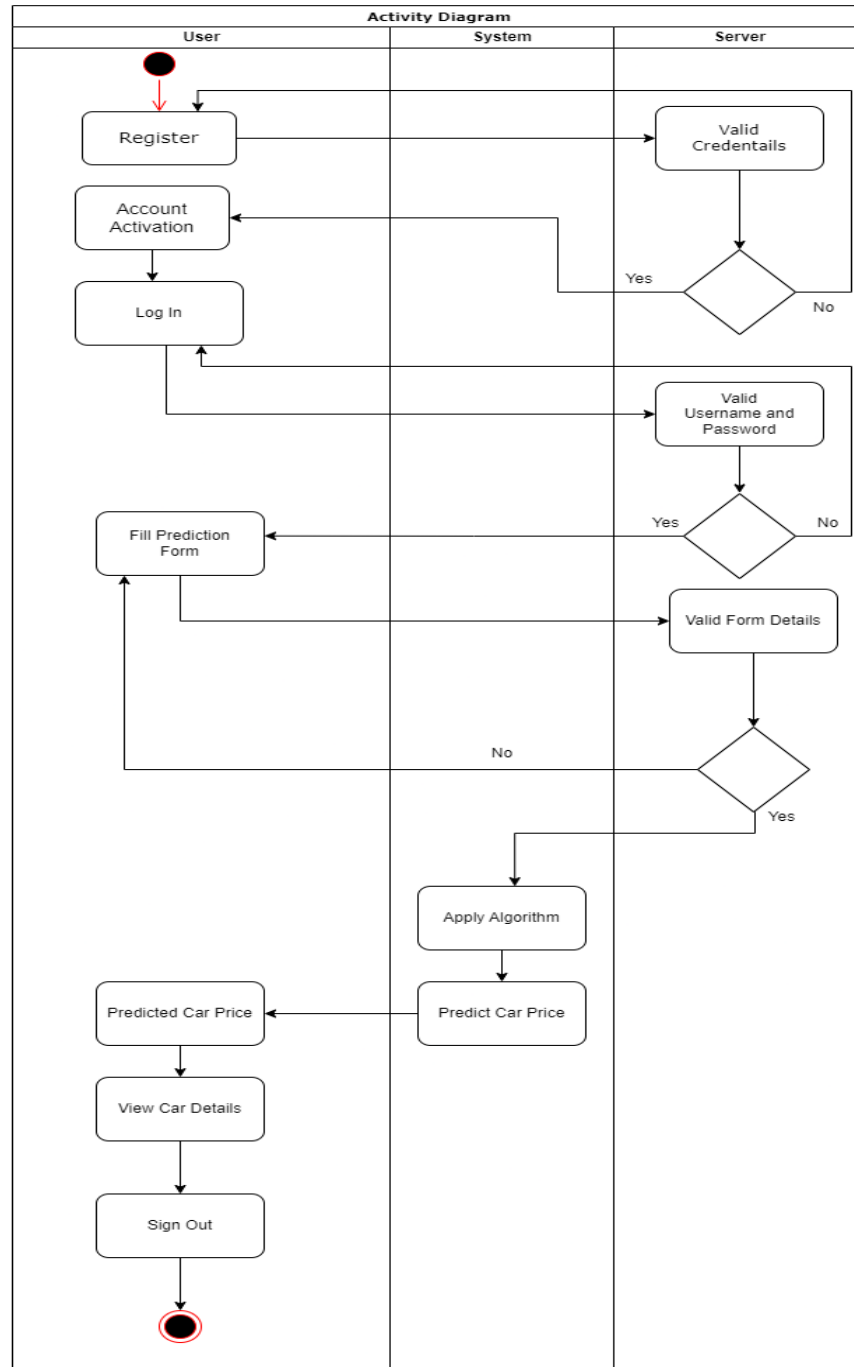


Figure 4. 16 Refined Activity Diagram of CarCompanion

In above activity diagram, first the users register to the system providing the required credentials. The authentication is performed by the server. If the users' registration credentials are not valid, the user is asked to provide valid credentials. However, if the users' registration credentials are valid, then the users are sent an activation link in their email through which the users activate their account. Then the users log in to the system. The validation of Username and Password takes place where the users only with the correct Username and Password are permitted to get logged in to the system. If the Username and Password are not valid, they are asked to login with valid credentials. Once the users are logged in to the system, they fill the prediction form providing the features of the car as required by the system. Each details of the form go through the validation by server. If the form is not valid, the users fill the form again with correct details. Once the information of the form is valid, Machine Learning Algorithm is applied which predicts the price of a car which is displayed to the user. Therefore, the user views the details of the predicted cars and hence signs out from the system.

4.1.2. Component Diagram

A component diagram, also referred to as a UML component diagram, describes the organization and wiring of the physical components within a system. Component diagram consists of components, interface, and ports. The main purpose of the component diagrams is to show the relationship between the varying components in the system.

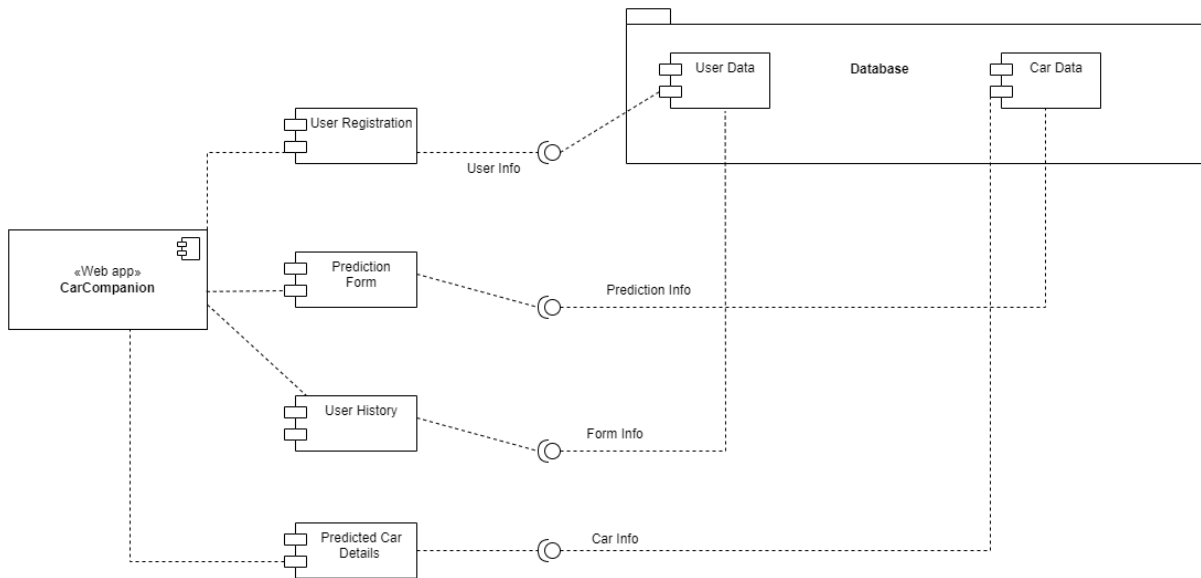


Figure 4. 17 Component Diagram of CarCompanion

The figure above is the component diagram of CarCompanion that shows how the components are connected and dependent upon one another. The diagram consists of four components named as: User Registration, Prediction Form, User History and Predict Car Details. The User Registration and User History components have interface with User Data module through User Info and Form Info respectively. Similarly, the Prediction Form and Predicted Car Details have interface with Car Data Module through Prediction Info and Car Info respectively.

4.1.3. Deployment Diagram

A deployment diagram is a UML diagram type that shows the execution architecture of a system that includes nodes such as hardware or software execution environments and the middleware connecting them.

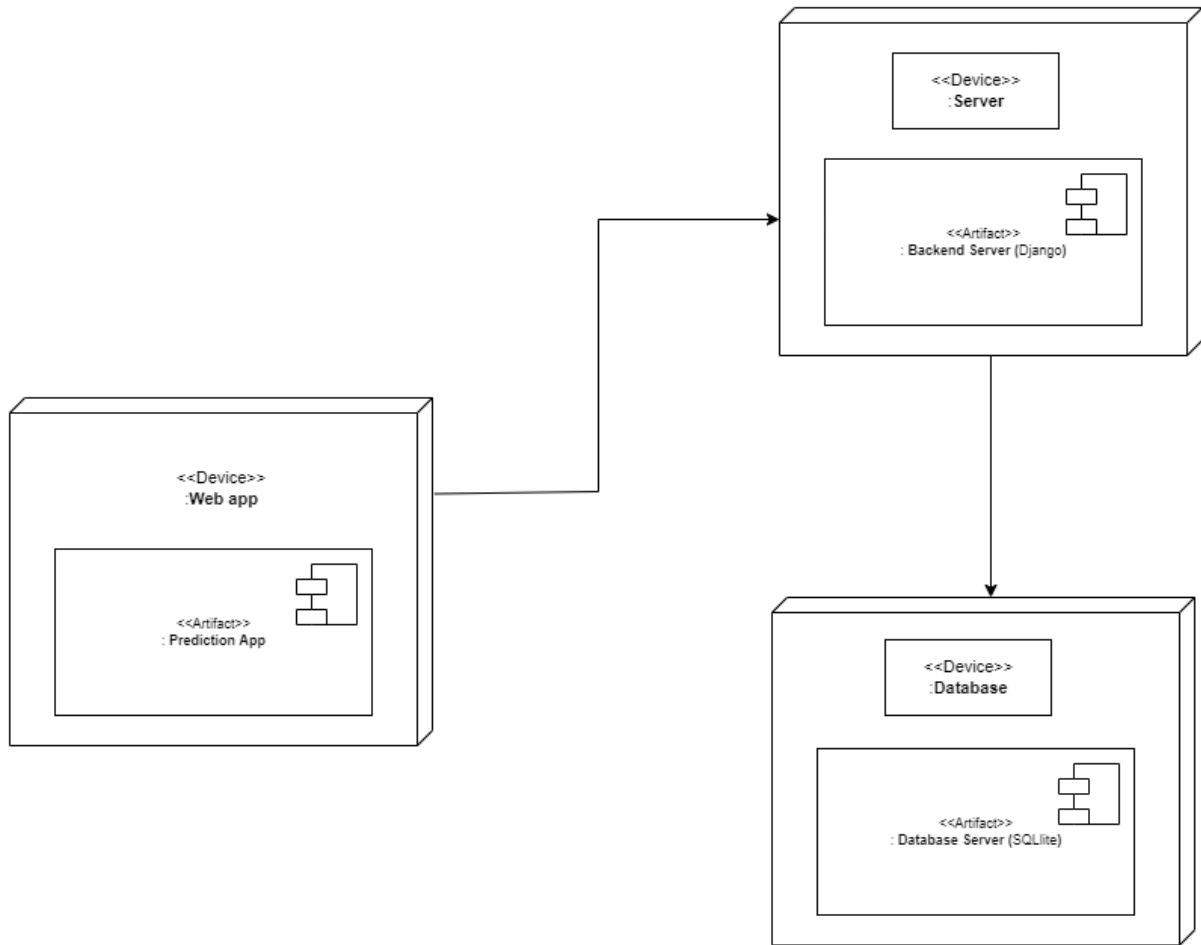


Figure 4. 18 Deployment Diagram of CarCompanion

In above Deployment diagram, the diagram consists of three nodes named as: Web app, Server and Database where every nodes consists of components named as: Prediction app, Backend Server, and Database Server respectively. The server node hosts the CarCompanion application and the user interacts with the application with client node. The database node stores the data. The web app node is connected to the server node and server node is connected to the database node thus showing the relationship between hardware and software components of our system.

4.2. Algorithm Details

CarCompanion app uses Machine Learning Regression Algorithms to predict the price. Although Random Forest Regression is used for the prediction, however, other two regressions like Bagging Regression and Linear Regression are also trained on the data.

a. Random Forest Regression Algorithm

Random Forest Regression is a part of Supervised Learning Algorithm that uses Ensemble Learning technique for regression. The Ensemble Learning technique combines predictions from various machine learning algorithms and provides prediction that is more accurate than those from a single ML model. The regression is performed with the following steps:

Step 1: Foremost, from the training dataset, randomly pick k data points.

Step 2: From those data sets, randomly build the decision tree.

Step 3: Choose the N number of Decision Trees to build.

Step 4: Repeat Step 1 and Step 2.

Step 5: Finally, for a new data point say x , each one of the N Decision Trees predicts the value of y as y_1, y_2, \dots, y_n , thus the average of these predictions gives the final Random Forest Prediction [10].

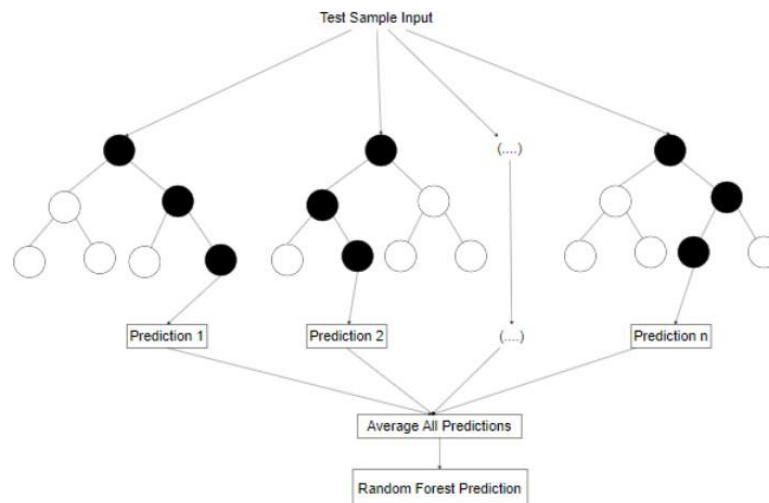


Figure 4. 19 Random Forest Regression

b. Bagging Regression

Bagging regression uses Ensemble Learning technique for regression. It selects the random subsets from the original dataset and then aggregates their individual predictions (either by voting or average) to generate the final accurate prediction. It is also called Bootstrap Aggregating.

Bagging Regression is performed with the following steps:

Step 1: Given the standard training dataset m , select a random sample from the standard training dataset with replacement and create new 'n' number of datasets say $d1, d2, \dots, dn$. ($m > n$)

Step 2: Feed $d1, d2, \dots, dn$ into the models to train the models (also called classifiers).

Step 3: Finally, the predicted outcomes from the models are aggregated to generate final accurate prediction.

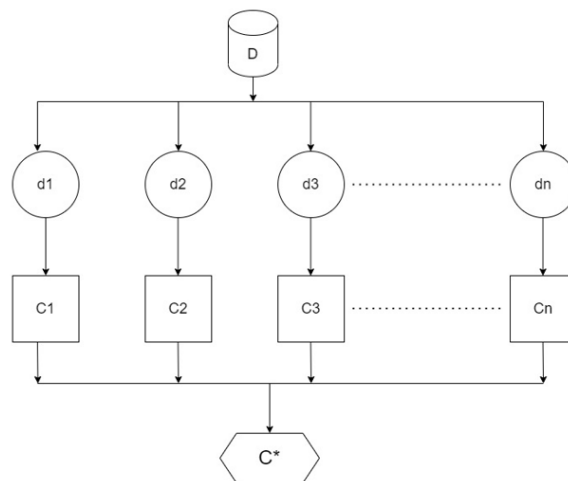


Figure 4. 20 Bagging Regression

c. Linear Regression

Linear Regression represents the relationship between dependent variable and one or more independent variables. It is the simplest way to predict output using a linear function of input features.

$$\hat{y} = w[0] \times x[0] + w[1] \times x[1] + \dots + w[n] \times x[n] + b$$

In the equation above, the linear model is based on the n number of features. Considering only a single feature w [0] will be slope and b will represent intercept. Linear regression looks for optimizing w and b such that it minimizes the cost function. The cost function can be written as:

$$\sum_{i=1}^M (y_i - \hat{y}_i)^2 = \sum_{i=1}^M \left(y_i - \sum_{j=0}^p w_j \times x_{ij} \right)^2$$

In the equation above let us assume the data-set has M instances and p features. Once we use linear regression on a data-set divided into training and test set, calculating the scores on training and test set can give us a rough idea about whether the model is suffering from over-fitting or under-fitting. If we have very few features on a dataset and the score is poor for both training and test set then it's a problem of under-fitting. On the other hand if we have large number of features and test score is relatively poor than the training score then it's the problem of over-generalization or over-fitting. Ridge and Lasso regression are some of the simple techniques to reduce model complexity and prevent over-fitting which may result from simple linear regression [11].

CHAPTER 5

IMPLEMENTATION AND TESTING

5.1. Implementation

The incremental development model is a method for developing software products that involves designing, implementing, and testing the final product one by one. When the requirements are known before the product is developed and can be implemented in stages, the incremental development technique is used.

For our project, the incremental approach includes gradually updating the model with updated data over time to increase its accuracy. This can be accomplished either by retraining the model with fresh training data or by adjusting the parameters of the current model. In order to increase the model's prediction accuracy, additional data regarding the used car market will be incorporated, including updates to market conditions, vehicle specifications, and consumer preferences. Over time, the incremental approach can produce predictions that are increasingly accurate by continuously modifying the model to the characteristics of the used car market.

5.1.1. Tools Used

i. Front End Tools

a. HTML

In this project, HTML is used as a front-end tool to create online pages and web application. HTML is used to organize materials and specify web page's layout. The structure and content of a web page are defined using a number of components and tags in HTML. These components include headings, sentences, lists, tables, forms, photos, and links. Thus, in this project, HTML is used to design the structure and content of the web page.

b. CSS

In the project, the display and layout of HTML and XML documents are described using the style sheet language known as CSS. It is simpler to construct and hence the project uses CSS to maintain consistent and aesthetically pleasing website. With CSS,

rules have been defined that determine how HTML components should be presented, controlling the look and feel of a webpage. Properties like font size, color, background color, margins, padding, border styles, and placement are determined with CSS.

ii. Back End Tools

a. SQLite

In this project, a popular open-source RDBMS for small applications, embedded systems, and mobile devices called SQLite is used to maintain the database. SQLite is used as it is self-contained, server less, and a zero-configuration database engine. The data is kept in a single file with SQLite that can be readily moved or backed up. Thus, our project uses SQLite to store the data.

b. Django

In this project, a high-level, open-source web framework called Django is used as a back-end tool to create web applications in Python. Django is used as it is simple to construct complex web applications rapidly and effectively. Moreover, for handling common web development tasks like URL routing, database access, authentication, and template rendering, Django offers a collection of tools and packages.

c. Machine Learning Pipeline with Python

In the back-end, this project builds Machine Learning Pipeline with Python. Machine Learning Pipeline was accomplished with python because Python is an object-oriented programming language that supports various programming paradigms such as functional programming, procedural programming, and structured programming. Moreover, Python is quite powerful and flexible with wide range of frameworks and libraries. Data preprocessing, training ML algorithms, etc. have been possible with libraries.

iii. Diagram Tool

Diagrams.net: All the UML diagrams of this project is drawn with Diagrams.net diagram tool. Diagrams.net provided a cross-platform, free and open source graph

drawing program. Moreover, the system design and the flowchart of the project is drawn with Diagrams.net.

iv. Document Editing Tool

The documentation of the project was created entirely with Microsoft Word-a word processor developed by Microsoft to draft, edit, format, and print various types of documents, including resumes, reports, letters, and more. Microsoft Word allowed to produce and maintain the document with user-friendly interface and abundance of functionality.

5.1.2. Implementation Details of Modules

This project can be decomposed into following modules:

i. Registration Module

In this system, the registration module allows users to create an account by providing their personal information such as name, email address, and password. This module ensures that the user's personal information is securely stored in the system. Here if the user submits required information then the user's input is taken and stored in different variables and the names email and passwords are validated and if the information provided by the users already exists in the database or is invalid than the errors are displayed accordingly. If the credentials are correct and the user has provided a valid email, then this module sends the activation link to the users Gmail account. The user account gets saved in the database.

```
myuser = User.objects.create_user(username,email,pass1)
myuser.first_name = fname
myuser.last_name = lname
myuser.is_active = False

myuser.save()
```

Figure 5. 1 Implementation of Registration Module

ii. Login Module

The login module allows registered users to access the system by providing their login

credentials such as email address and password. This module only works when the user clicks the confirmation link. This module is used when the user is properly saved in the database and is authenticated using:

```
if request.method == 'POST':
    username = request.POST['username']
    pass1 = request.POST['pass1']
    user = authenticate(username=username, password=pass1)
    if user is not None:
        login(request, user)
        fname = user.first_name
        return render(request, "authentication/index.html", {"fname": fname})
    else:
        print("SASDS")
        error_message = f"Invalid Credentials"
        return render(request, "authentication/signin.html", {"error_message": error_message})
return render(request, "authentication/signin.html")
```

Figure 5. 2 Implementation of Login Module

Also, certain errors are shown to the users if it doesn't meet any of the requirements or the user is not saved in the database. Thus, ensuring that only authorized users can access the system.

iii. Car Information Input Module

This is the main module used for displaying the actual predictions of the pre-owned cars. To make the system predict the prices of valid cars for the valid credentials, certain restrictions regarding the input credentials are made in order to generate authentic prediction such as MIN_MILEAGE =0, MAX_ENGINE = 6500, and so on. With this the limits have been made for the users input credentials and the credentials are validated accordingly. After getting the data from the users this system generates three different prices for three models loaded and trained using machine learning.

```
predictions = getPrediction(int(name), int(year), kilometer, mileage, engine, owner, transmission, fuel, power, seats)
data["predicted_price1"] = predictions[0]
data["predicted_price2"] = predictions[1]
data["predicted_price3"] = predictions[2]
PredictCarModel.objects.create(**data)
latest_entry = PredictCarModel.objects.latest('id')
predicted_price1 = round(latest_entry.predicted_price1 * 130, 2) if request.POST.get('model') else None
predicted_price2 = round(latest_entry.predicted_price2 * 130, 2) if request.POST.get('model') else None
predicted_price3 = round(latest_entry.predicted_price3 * 130, 2) if request.POST.get('model') else None
```

Figure 5. 3 Implementation of Car Details Module

iv. View Car Details Module

The details of those cars whose price has been predicted by the user can be viewed in this module. The insights can be gained based on different models of the cars. Here we fetch and render all the data from the database using objects. All function helping in rendering all the cars in alphabetical order.

```
if request.method == 'GET' and 'search' in request.GET:
    search = request.GET.get('search')
    predict = PredictCarModel.objects.filter(model__icontains=search).order_by('model')
else:
    predict = PredictCarModel.objects.all().order_by('model')
```

Figure 5. 4 Implementation of View Car Details Module

5.1.3. Implementation Details of Algorithms

a. Implementation of Random Forest Regression

Step 1: Foremost, the sample data points are randomly picked and bootstrap dataset is created. It is done in the following way:

```
def fit(self, X, y):
    for i in range(self.n_estimators):
        # Create a bootstrap sample of the data
        sample_indices = np.random.choice(X.shape[0], size=X.shape[0], replace=True)
        X_sample = X[sample_indices]
        y_sample = y[sample_indices]
```

Figure 5. 5 Generating Bootstrap Dataset from the Random Samples

Step 2: From the generated Bootstrap Dataset, Decision Trees are generated. It is done in following way:

```
# Fit a decision tree to the sample
tree = DecisionTreeRegressor2(max_depth=self.max_depth)
tree.fit(X_sample, y_sample)

# Add the tree to the list of models
self.models.append(tree)
```

Figure 5. 6 Generating Decision Trees from the Bootstrap Dataset

Step 3: Now, the predictions with each Decision Trees are made and the results are averaged.

```
def predict(self, X):
    # Make predictions with each tree and average the results
    predictions = np.zeros((X.shape[0], len(self.models)))
    for i, model in enumerate(self.models):
        predictions[:, i] = model.predict(X)

    return np.mean(predictions, axis=1)
```

Figure 5. 7 Predictions from Each Decision Trees and Their Average

b. Implementation of Bagging Regression

Step 1: Foremost, the random samples from the original dataset is generated which is also called Bootstrap samples.

```
def fit(self, X, y):
    for estimator in self.estimators:
        idx = np.random.choice(X.shape[0], size=X.shape[0], replace=True)
        X_sampled = X[idx]
        y_sampled = y[idx]
```

Figure 5. 8 Generating the Random Samples from the Original Dataset

Step 2: Secondly, the Bootstrap samples are fed into the models to train the models.

```
estimator.fit(X_sampled, y_sampled)
```

Figure 5. 9 Feeding Sample Datasets into the Model

Step 3: Finally, the predicted outcomes from the models are aggregated to generate final accurate prediction.

```
def predict(self, X):
    predictions = np.zeros((X.shape[0], self.n_estimators))
    for i, estimator in enumerate(self.estimators):
        predictions[:, i] = estimator.predict(X)
    return np.mean(predictions, axis=1)
```

Figure 5. 10 Predictions from Each Models and Their Average

c. Implementation of Linear Regression

Step 1: Foremost, four instance variables: learning_rate, num_iterations, weights, and bias are initialized.

```
class LinearRegression2:
    def __init__(self, learning_rate=0.01, num_iterations=1000):
        self.learning_rate = learning_rate
        self.num_iterations = num_iterations
        self.weights = None
        self.bias = None
```

Figure 5. 11 Initialization of Instance Variables

Step 2: Then the Linear Regression Model is trained and weights and biases are further initialized in the process.

```
def fit(self, X, y):
    n_samples, n_features = X.shape

    # initialize weights and bias
    self.weights = np.zeros(n_features)
    self.bias = 0
```

Figure 5. 12 Training Linear Regression

Step 3: A loop is started where the predicted target values 'y_predicted' are calculated performing matrix multiplication between input variable X and weights adding bias.

```
for i in range(self.num_iterations):
    y_predicted = np.dot(X, self.weights) + self.bias
```

Figure 5. 13 Calculation of Predicted Target Values

Step 4: Then the gradient of weights and bias is computed and using the computed gradients and learning rate, the weights and bias are updated.

```
# compute gradients
dw = (1 / n_samples) * np.dot(X.T, (y_predicted - y))
db = (1 / n_samples) * np.sum(y_predicted - y)

# update weights and bias
self.weights -= self.learning_rate * dw
self.bias -= self.learning_rate * db
```

Figure 5. 14 Computation of Gradients

Step 5: Finally the 'predict()' method takes the input features 'X' and returns the predicted target values.

```
def predict(self, X):
    y_predicted = np.dot(X, self.weights) + self.bias
    return y_predicted
```

Figure 5. 15 Prediction of Target Values

5.2. Testing

Testing is the process of determining whether the generated software or application functions properly, or whether there is a match between the outcomes that were obtained and those that were anticipated. Testing is done as the software is being developed.

5.2.1. Test Cases for Unit Testing

Unit testing is a method of software testing that involves examining separately each unit or component of a software program, usually at the code level. Unit testing is used to verify that each software unit or component operates as intended and adheres to its design, requirements, and specifications.

Table 5. 1 Test Cases for Unit Testing (Test case for Login and Registration)

Test Case ID	Test Description	Input Data	Expected Result	Actual Result	Remarks
TC-REG-01	Register new user with valid email and other details	Username: Ishan2 First Name: Ishan Last Name: Rai Email: raiishan63@gmail.com Password: ***** Confirm Password: *****	User receives the verification link in email and login to the screen.	Verification link received.	Pass
TC-log-01	Login using valid email and other details	Username: Ishan2 Password: *****	Login successfully to the Prediction Form Page	Login successfully to the Prediction Form Page	Pass

TC-LOG-02	Login using invalid email and other details	Username: Abiral34 Password: *****	Login Unsuccessful to home screen.	Error message displayed in screen.	Pass
TC-REG-02	Register new user with invalid email and other details	Username: Ishan77 First Name: Ishan Last Name: Rai Email: gogoman63@gmail.com Password: ***** Confirm Password: *****	Verification link not received in email.	Verification link not received in email.	Pass

5.2.2. Test Cases for System Testing

System testing is a sort of software testing that examines a fully integrated software system's behavior in comparison to the requirements that were given to it. Instead of testing separate parts or units of the software, it tests the system as a whole.

Table 5. 2 Test cases for System Testing

Test Case ID	Test Description	Input Data	Expected Result	Actual Result	Remarks
TC-VAI-01	Test the system with valid Input data.	Car Model: Maruti Wagon Year: 9 Kilometer:14000 Mileage:19.2 Engine:998 Owner Type: Second Transmission Type:	The system should Predict the price of car.	The price is predicted.	Pass

		Manual Fuel Type: Petrol Seat: 5 Power:58.16			
TC- VAL- 02	Test the system with valid but different Input data.	Car Model: Hyundai Creta Year:5 Kilometer:10000 Mileage: 12 Engine:1498 Owner Type: Second Transmission Type: Manual Fuel Type: Petrol Seat:5 Power:128.16	The system should Predict the price of car.	The price is predicted.	Pass
TC- INV- 02	Test the system with word value or negative value in mileage.	Car Model: Hyundai Creta Year:5 Kilometer: 25000 Mileage: -12 Engine:1498 Owner Type: Second Transmission Type: Manual Fuel Type: Petrol Seat:6 Power:128.16	Error message should be displayed in Mileage.	The price is not predicted and error message is shown.	Pass

TC-INV-02	Test the system with Invalid data like giving credentials of Car model wrong or half.	Car model: Wagon Year:9 Kilometer:10500 Mileage: 19.2 Engine:998 Owner Type: Second Transmission Type: Manual Fuel Type: Petrol Seat:5 Power:58.16	Error message should be displayed in car model.	The price is not predicted and error message is shown.	Pass
------------------	---	---	---	--	------

5.2.3. Real-Time Test Scenario

a. Test Scenario 1: Model Testing

Test Objective: To Test Input Data

Test Data: Features of Secondhand Car

Expected Output: Accurate Price Prediction

Actual Output:

```

conv= standardScaler.transform(np.array([37,10,41000,23.08,1461,0,0,1,63.1,5]).reshape(1,-1))
[140] ✓ 0.0s Python
... c:\Users\Swastika\anaconda3\envs\testproject\lib\site-packages\sklearn\base.py:439: UserWarning: X does not
warnings.warn(

predicted_price = float(rf.predict(conv))
round(predicted_price, 2)
[141] ✓ 0.0s Python
... 4094.69

```

Figure 5. 16 Model Testing for Accurate Price Prediction

b. Test Scenario 2: Model Saving and Deployment

Test Objective: To Test Proper Model Saving and Deployment

Test Data: Model

Expected Output: Saved and Deployed Model

Actual Output:



Figure 5. 17 Model Saving and Deployment

c. Test Scenario 3: Prediction Form Test

Test Objective: To Test Correct Form Fill Up

Test Data: Secondhand Car Features

Expected Output: Correctly Filled Prediction Form

Actual Output:

The figure consists of two screenshots of a web application titled "CarCompanion". Both screenshots show a dark-themed interface with a car image in the background. The top screenshot displays a form for car model prediction with the following fields: "Car Model" (Maruti Wagon), "Year" (3), "Kilometers Driven" (41000), "Mileage" (26.59), and "Engine" (936). The bottom screenshot displays a form for owner type prediction with the following fields: "Owner Type" (First), "Transmission Type" (Manual), "Fuel Type" (CNG), "Power" (98.0), and "Seats" (5). Both forms have a "Hello !" message and "You're successfully logged in." text at the top.

Figure 5. 18 Prediction Form Test

d. Test Scenario 4: Web App Testing

Test Objective: To Predict Accurate Price of Secondhand Car

Test Data: Secondhand Car Features

Expected Output: Predict Price of Secondhand Car

Actual Output:

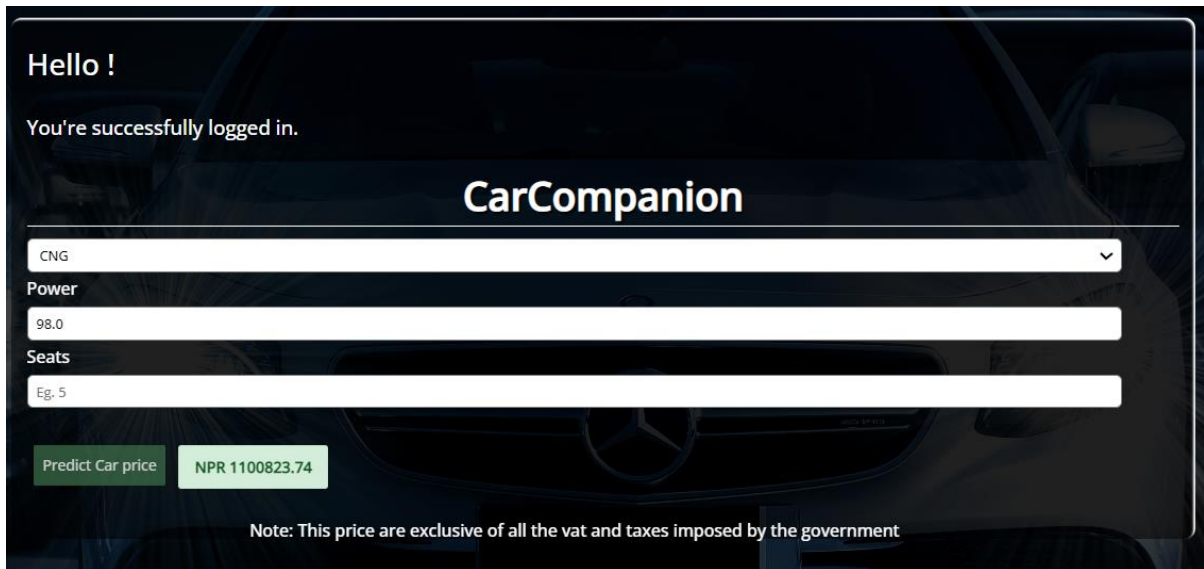


Figure 5. 19 Web App Testing

5.3. Result Analysis

Demonstration of this project shows that our system successfully predicts the price of the used cars. Both the functional and non-functional requirements were fulfilled while the testing was done. The algorithms used for this project are Random Forest Regression, Bagging Regression and Linear Regression. The accuracy of the Linear Regression Algorithm is 0.63, the Bagging Regression Algorithm is 0.90, and the Random Forest Regression Algorithm is also 0.90. Thus Random Forest Regression is used which gives the decent valuation of the pre-owned cars. The following metrics are used to compare the results among the Regression Algorithms:

Accuracy: Accuracy is an evaluation metric to measure the performance of a model. It is calculated as:

$$\text{Accuracy} = (\text{Number of Correct Predictions}) / (\text{Total Number of Predictions})$$

In this project, the accuracy is measured through **R2 (r squared score) Score**, which is also known as the coefficient of determination. It is a metric that is used to evaluate the performance of regression models of the machine learning. It is measured by the difference between the samples in the dataset and the predictions made by the model. The figures below depicts the

R2 Score for all three algorithms.

```
linearRegression = LinearRegression()
linearRegression.fit(X_train_scaled, Y_train)
y_pred = linearRegression.predict(X_test_scaled)
r2_score(Y_test, y_pred)
```

✓ 0.0s

0.6354654899259813

Figure 5. 20 R2 Score obtained from Linear Regression

```
baggingRegression = BaggingRegressor(n_estimators = 100, random_state = 42)
baggingRegression.fit(X_train_scaled, Y_train)
y_pred = baggingRegression.predict(X_test_scaled)
r2_score(Y_test, y_pred)
```

0.90548056078088

Figure 5. 21 R2 Score obtained from Bagging Regression

```
rf = RandomForestRegressor(n_estimators = 100, random_state = 42)
rf.fit(X_train_scaled, Y_train)
y_pred = rf.predict(X_test_scaled)
r2_score(Y_test, y_pred)
```

0.9031518162133888

Figure 5. 22 R2 Score obtained from Random Forest Regression

Similarly, in this project, to calculate the error, MSE, RMSE, and MAE metrics are used.

MSE: MSE calculates the average squared error, which is the difference in between the estimated and actual values. By taking the difference between the model's predictions and the actual data, squaring it, and averaging it across the entire dataset, MSE is obtained. It is given

by the equation:

$$MSE = \frac{1}{N} \sum_{i=1}^N (\hat{y}_i - y_i)^2$$

RMSE: RMSE is an evaluation metric that shows how far predictions fall from measured true values using Euclidean distance. For each data point, a residual i.e. a difference between the prediction and the truth is calculated. The norm of the residual for each data point is then added, along with the mean of the residuals, and the square root of that mean is taken. It is calculated as:

$$RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^N (\hat{y}_i - y_i)^2}$$

MAE: The MSE and MAE differ slightly from one another, but surprisingly, they offer nearly the same features. To calculate MAE, take the difference between the model's predictions and the actual data, multiply it by its absolute value, and then average the result across the entire dataset. Since the absolute value of the errors is taken in MAE, the MAE is never negative. It is given by following equation:

$$MAE = \frac{1}{n} \sum_{j=1}^n |y_i - \hat{y}_j|$$

The figures below depicts MSE, RMSE, and MAE for all three algorithms:

```
mse = mean_squared_error(Y_test, y_pred)
rmse = math.sqrt(mse)
mae = mean_absolute_error(Y_test, y_pred)
print("Mean Square Error", mse)
print("Root Mean Square Error", rmse)
print("Mean Absolute Error", mae)
```

✓ 0.0s

Mean Square Error 0.0018141766917098708
Root Mean Square Error 0.04259315310833269
Mean Absolute Error 0.025391302778479063

Figure 5. 23 The MSE, RMSE, and MAE of Linear Regression

```
mse = mean_squared_error(Y_test, y_pred)
rmse = math.sqrt(mse)
mae = mean_absolute_error(Y_test, y_pred)
print("Mean Square Error", mse)
print("Root Mean Square Error", rmse)
print("Mean Absolute Error", mae)
```

✓ 0.0s

Mean Square Error 0.0004703943215417309
Root Mean Square Error 0.02168857583018606
Mean Absolute Error 0.009257414082815314

Figure 5. 24 The MSE, RMSE, and MAE of Bagging Regression

```
mse = mean_squared_error(Y_test, y_pred)
rmse = math.sqrt(mse)
mae = mean_absolute_error(Y_test, y_pred)
print("Mean Square Error", mse)
print("Root Mean Square Error", rmse)
print("Mean Absolute Error", mae)
```

✓ 0.0s

Mean Square Error 0.00048198377054734296
Root Mean Square Error 0.02195412878133275
Mean Absolute Error 0.009275148097661884

Figure 5. 25 The MSE, RMSE, and MAE of Random Forest Regression

Now from the above table and analysis, the linear regression has 64% accuracy, followed by bagging regression and random forest regression with 90% accuracy. The linear regression has MSE, RMSE, and MAE as 0.0018141766917098708, 0.04259315310833269, and 0.025391302778479063 respectively. Similarly, the bagging regression has MSE, RMSE, and MAE as 0.0004703943215417309, 0.02168857583018606, and 0.009257414082815314 respectively. Finally, the random forest regression has MSE, RMSE, and MAE as 0.00048198377054734296, 0.02195412878133275, and 0.009275148097661884 respectively.

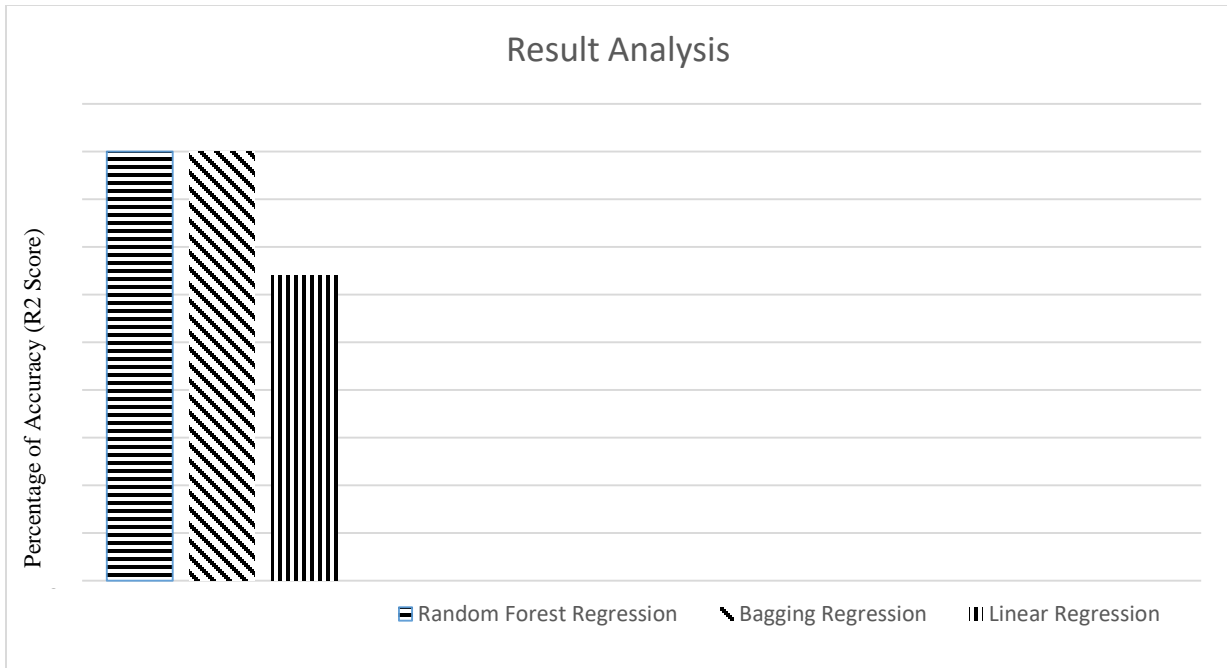


Figure 5. 26 Result Analysis

From the result analysis above, the Linear Regression has very less accuracy as compared to Bagging Regression and Random Forest Regression. The Random Forest Regression and Bagging Regression have same accuracy.

Table 5. 3 Results Comparison

S.N.	Algorithm	Accuracy (R2 Score)	MSE	RMSE	MAE
1	Linear Regression	0.63	0.00181	0.0426	0.0253
2	Bagging Regression	0.90	0.00047	0.0217	0.0092
3	Random Forest Regression	0.90	0.00048	0.0219	0.0092

From the above table and obtained measures, Linear Regression has very less accuracy and maximum errors as compared to Random Forest Regression and Bagging Regression, thus it is not considered for price prediction. Nevertheless, Random Forest Regression and Bagging Regression has similar accuracy and error metrics, however, Random Forest Regression model is used for price prediction because Random Forest Regression is less prone to overfitting as compared to Bagging Regression. Similarly, Random Forest handles categorical data more efficiently as compared to Bagging Regression and when there is large dataset Random Forest reduces the computational complexity as it considers the subset of features at each split whereas bagging regression considers all features.

CHAPTER 6

CONCLUSION AND FUTURE RECOMMENDATIONS

6.1. Conclusion

In conclusion, the CarCompanion presents a used car price prediction model using the Random Forest Regression Algorithm. The results show that the model effectively predicts the price of a car based on several characteristics such as mileage, age, model and others. The Random Forest algorithm has proven to be an effective tool for predicting car prices because it can handle both categorical and continuous variables and is robust to overfitting. Moreover, the model was trained on a large dataset of car sales data, which helped improve its accuracy and generalization.

Thus, CarCompanion has significant implications for the automotive industry as it can help dealers and buyers make more informed pricing and purchasing decisions. By accurately predicting the price of a car, the project can also help reduce information asymmetry and improve market efficiency.

6.2. Future Recommendations

Future research on car prediction models using machine learning algorithms could focus on several areas to improve their accuracy and applicability.

One area for improvement is the incorporation of additional features into the model. For example, location-based factors such as regional demand and supply, local economic conditions, and weather patterns could be included to better reflect the local market. The condition of the car, including factors such as mileage, wear and tear, and accident history, could also be incorporated to improve the model's accuracy in predicting car prices.

Another potential avenue for research is exploring alternative machine learning algorithms.

While the Random Forest algorithm has proved to be effective in this study, there may be other algorithms that could achieve higher accuracy or better performance on specific tasks. For example, deep learning models such as CNN and RNN may be more effective at processing image data or time-series data, respectively.

Furthermore, research could also investigate how these models could be applied in real-world scenarios, such as in online car marketplaces or dealership sales. This could involve testing the accuracy of the model on new and unseen data, or evaluating its performance in different market conditions.

Overall, the future of research in car prediction models using machine learning algorithms is promising, and there is scope for further research to improve the accuracy and applicability of these models in the car industry.

However, the system is limited by the quality and quantity of the data. The model used is trained on 216 car models, thus the system only predicts the price for those models of the cars. There are multiple factors like external factors such as fuel prices, consumer confidence, and economic conditions that can affect the car market including the price of cars. Thus, these factors are not considered in the system. The system does not accurately predict the prices of luxury or high-end cars that have unique and rare features.

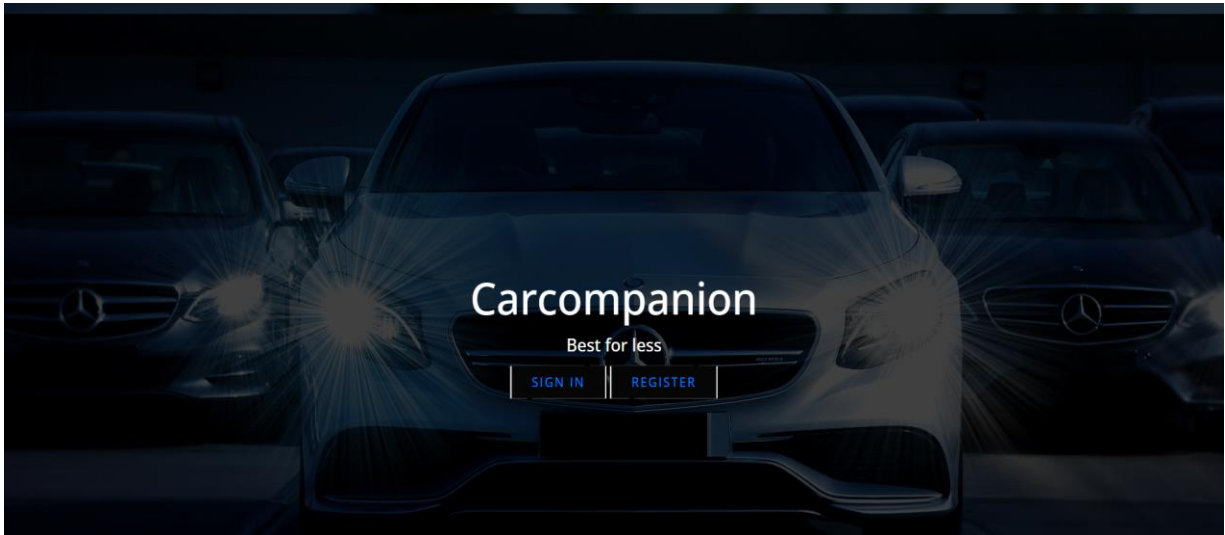
REFERENCES

- [1] Siddhant W, Sonia M, "Allied Market Research," Allied Market Research, March 2023. [Online]. Available: <https://www.alliedmarketresearch.com/used-cars-market-A06429>. [Accessed april 15 2023].
- [2] "Research Dive," Research Dive, Sep 2021. [Online]. Available: <https://www.researchdive.com/8497/used-cars-market>.
- [3] P Rane, D Pandya, D Kotak, "USED CAR PRICE PREDICTION," *International Research Journal of Engineering and Technology (IRJET)*, vol. 8, no. 4, p. 3, April , 2021.
- [4] A. AlShared, "Used Cars Price Prediction and Valuation using Data Mining Techniques," Rochester Institute of Technology, Dubai , 2021.
- [5] Lakshmi Sucharitha, B.Raghavendran, Ch. V.Venkataramana, B, Advances in Computational Intelligence and Informatics, R. R. S. Chillarige, Ed., Springer Singapore, 2020, pp. 253--263.
- [6] Chen; Jian; Fangfang, Li; Jing; Quing, Wang; Qingzhen , Han; Ming Yan, "Comparisons of different methods used for second-hand car price prediction.," *In 2nd International Conference on Applied Mathematics, Modelling, and Intelligent Computing (CAMMIC 2022)*, vol. 12259, no. 2022, pp. 1191-1201, 2022.
- [7] "CarDekho.com," Cardekho, 2000. [Online]. Available: https://www.cardekho.com/info/about_us. [Accessed 30 Novermber 2022].
- [8] "Carwale.com," Carwale, 2006. [Online]. Available: <https://www.carwale.com/about-us/>. [Accessed 16 4 2023].
- [9] "Zigwheels.com," Zigwheels, 8 2008. [Online]. Available: <https://www.zigwheels.com/aboutus>. [Accessed 16 4 2023].
- [10] chaya, "Medium," Level Up Coding, 9 6 2020. [Online]. Available: <https://levelup.gitconnected.com/random-forest-regression-209c0f354c84>. [Accessed 16 4 2023].
- [11] S. Bhattacharyya, "Towards Data Science," Towards Data Science, 26 Sep 2018.

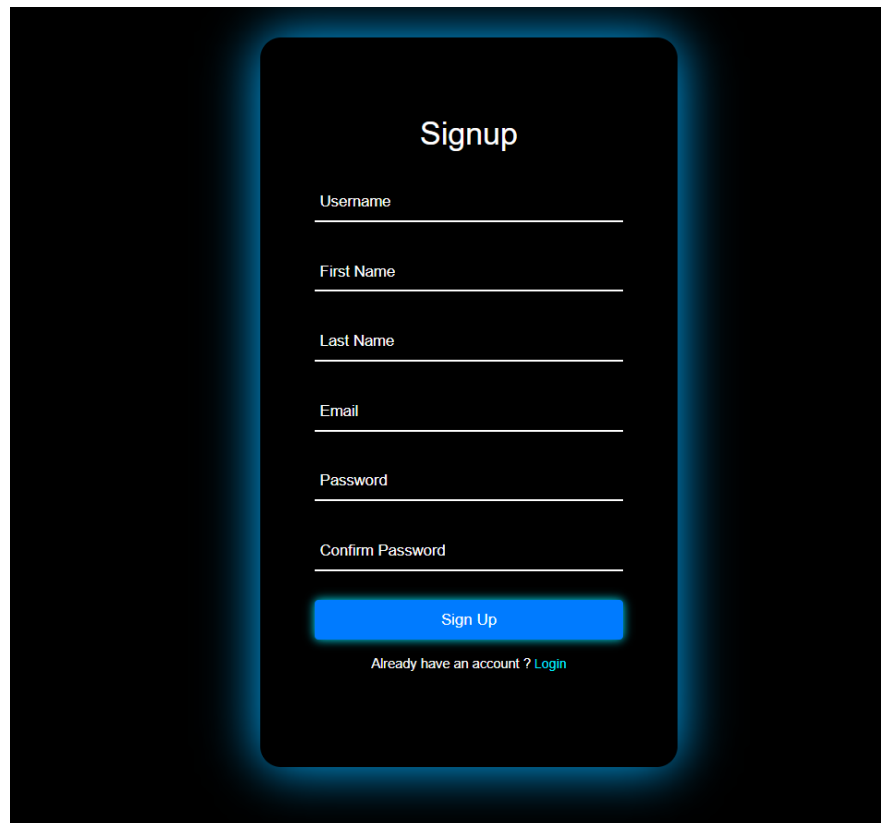
[Online]. Available: <https://towardsdatascience.com/ridge-and-lasso-regression-a-complete-guide-with-python-scikit-learn-e20e34bcbf0b>.

- [12] S. R. Manker, "Youtube," 5 Minute Engineering, 28 4 2019. [Online]. Available: <https://www.youtube.com/watch?v=kW4XSvuBtDw>. [Accessed 16 4 2023].

APPENDICES



Home Page

The image shows the signup page of the Carcompanion application. The page has a dark background with a glowing blue rectangular frame in the center. Inside the frame, the word 'Signup' is centered at the top in a white, sans-serif font. Below it are six input fields, each with a white label and a white underline: 'Username', 'First Name', 'Last Name', 'Email', 'Password', and 'Confirm Password'. At the bottom of the frame is a large, solid blue button with the text 'Sign Up' in white. Below the button, centered, is the text 'Already have an account ? Login' in a small, light blue font.

Registraion Page

Signup

Username

Ishan2

First Name

Ishan

Last Name

Rai

Email

raiishan63@gmail.com

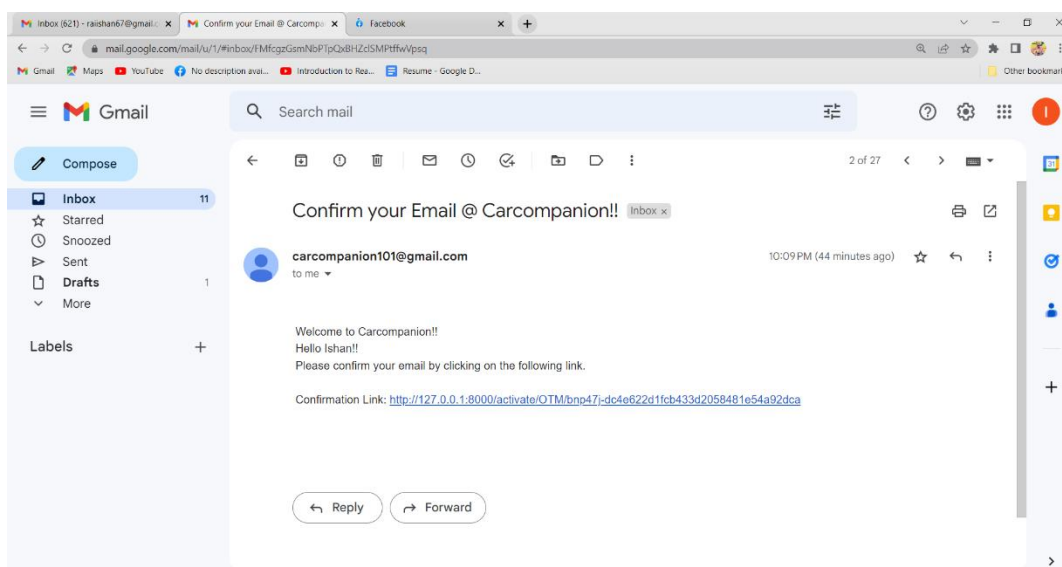
Password

Confirm Password

Sign Up

Already have an account ? [Login](#)

Successful Registration



Email for Successful Registration

Signup

Username
Ishan77

First Name
Ishan

Last Name
Rai

Email
gogoman67@gmail.com

Password

Confirm Password

Sign Up

Already have an account ? [Login](#)

Unregistered Account

Signup

Username

First Name

Last Name

Email

Password

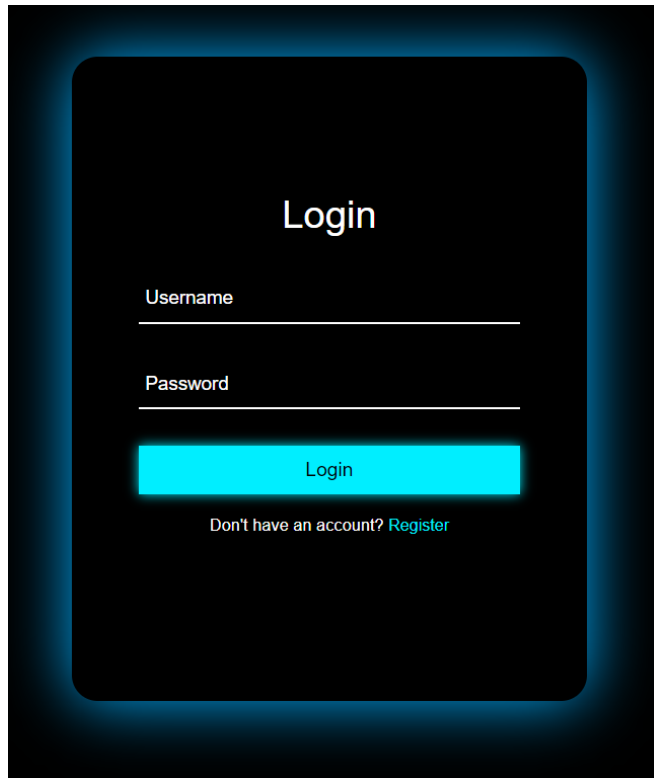
Confirm Password

Sign Up

Already have an account ? [Login](#)

Password must be above 8 characters!

Unsuccessful Registration



A login form on a dark background with a glowing blue border. The form contains a title, two input fields, a button, and a link.

Login

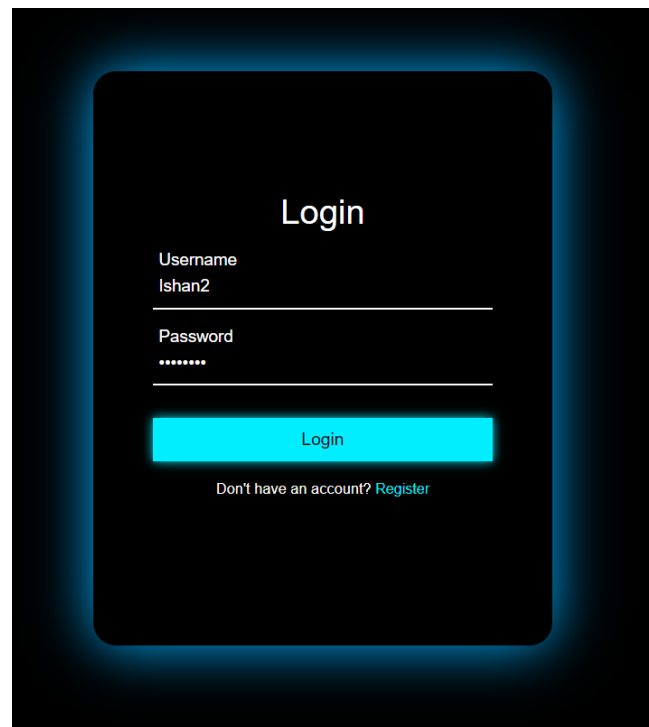
Username

Password

Login

Don't have an account? [Register](#)

Login Page



A login form on a dark background with a glowing blue border. The form contains a title, two input fields with text, a button, and a link.

Login

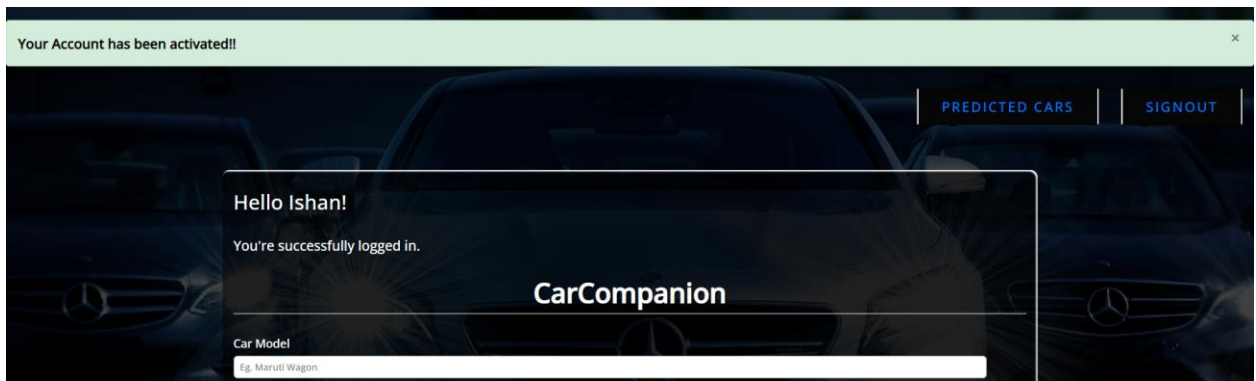
Username
Ishan2

Password

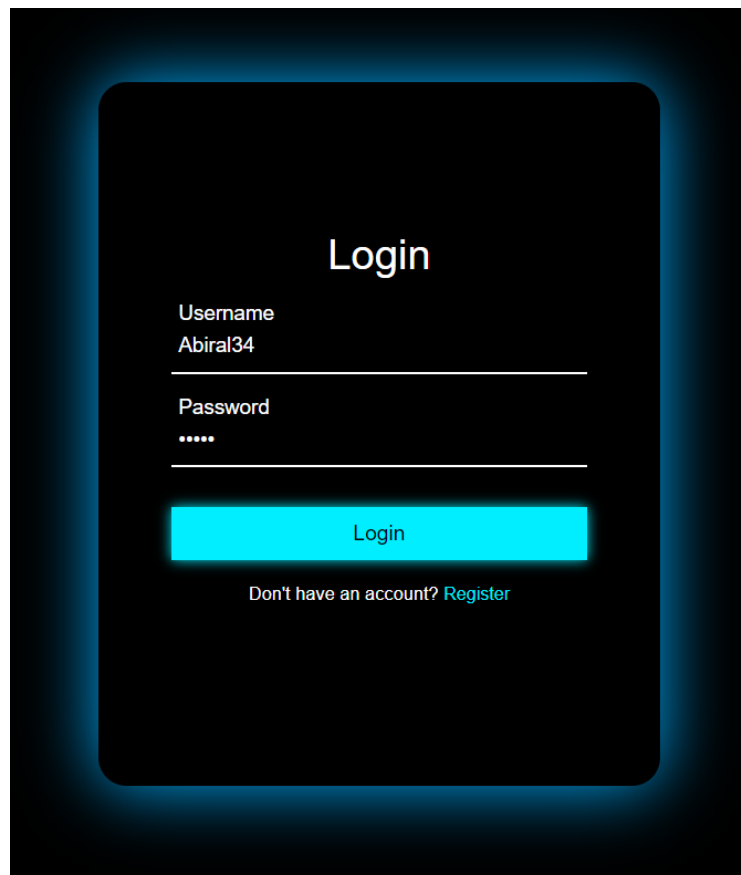
Login

Don't have an account? [Register](#)

Login Successful



Successfully Logged In To Home Page



Invalid Credentials

Login

Username

Password

Login

Don't have an account? [Register](#)

Invalid Credentials

Login Unsuccessful

[PREDICTED CARS](#) | [SIGNOUT](#)

Hello Ishan!

You're successfully logged in.

CarCompanion

Car Model
Maruti Wagon

Year
9

Kilometers Driven
14000

Mileage
19.2

Engine
998

Car Price Prediction Form

[PREDICTED CARS](#)[SIGNOUT](#)

Hello Ishan!

You're successfully logged in.

CarCompanion

Owner Type

Second

Transmission Type

Manual

Fuel Type

Petrol

Power

58.16

Seats

5

Valid Form Details

[PREDICTED CARS](#)[SIGNOUT](#)

Hello !

You're successfully logged in.

CarCompanion

Seats

5.0

Predict Car price

Random Forest Regression => NPR 523407.3	Bagging Regression => NPR 527049.9	Linear Regression => NPR 53085.5
R2 Score=> 0.90	R2 Score=> 0.90	R2 Score=> 0.63
MSE=> 0.00048	MSE=> 0.00047	MSE=> 0.00181
RMSE=> 0.0219	RMSE=> 0.0217	RMSE=> 0.0426
MAE=> 0.0092	MAE=> 0.0092	MAE=> 0.0253

Note: This price are exclusive of all the vat and taxes imposed by the government

Price Predicted

PREDICTED CARS | SIGNOUT

Hello !
You're successfully logged in.

CarCompanion

Car Model

Wagon

Year

9

Kilometers Driven

10500

Mileage

19.2

Engine

998

Activate Windows

Invalid Model

PREDICTED CARS | SIGNOUT

Hello !
You're successfully logged in.

CarCompanion

CNG

Power

Eg. 58.16

Seats

Eg. 5

Predict Car price

Wagon Model does not exists. Please enter valid model.

Note: This price are exclusive of all the vat and taxes imposed by the government

Activate Windows

Error Message

[PREDICTED CARS](#)
[SIGNOUT](#)

Hello !

You're successfully logged in.

CarCompanion

Fig. 26.69

Engine

789.0

Please fill out this field.

Owner Type

First

Transmission Type

Manual

Fuel Type


CNG

Power

Activate Windows

Go to Settings to activate Windows.

Form Validation

<div>  <div>Enter Model</div> </div> <h1>CAR PREDICTIONS</h1>		
<div>Audi A3</div> <div> <p>Year: 8</p> <p>Kilometers Driven: 56000.0</p> <p>Mileage: 12.0</p> <p>Engine: 6322.0</p> <p>Owner Type: First</p> <p>Transmission Type: Manual</p> <p>Fuel Type: CNG</p> <p>Power: 56.0</p> <p>Seat: 6</p> </div> <div> <p>Random Forest Regression: 3325251.80</p> <p>Bagging Regression: 3410820.40</p> <p>Linear Regression: 6618940.90</p> </div>	<div>Audi A4</div> <div> <p>Year: 8</p> <p>Kilometers Driven: 56000.0</p> <p>Mileage: 12.0</p> <p>Engine: 6322.0</p> <p>Owner Type: First</p> <p>Transmission Type: Manual</p> <p>Fuel Type: CNG</p> <p>Power: 56.0</p> <p>Seat: 8</p> </div> <div> <p>Random Forest Regression: 3322205.90</p> <p>Bagging Regression: 3377801.70</p> <p>Linear Regression: 6595810.00</p> </div>	<div>Audi A4</div> <div> <p>Year: 8</p> <p>Kilometers Driven: 56000.0</p> <p>Mileage: 12.0</p> <p>Engine: 6322.0</p> <p>Owner Type: First</p> <p>Transmission Type: Manual</p> <p>Fuel Type: CNG</p> <p>Power: 56.0</p> <p>Seat: 9</p> </div> <div> <p>Random Forest Regression: 3322205.90</p> <p>Bagging Regression: 3377801.70</p> <p>Linear Regression: 6595373.20</p> </div>

Activate Windows

Predicted Cars Details

SEARCH RESULTS

Audi A4

Year: 8
Kilometers Driven: 56000.0
Mileage: 12.0
Engine: 6322.0
Owner Type: First
Transmission Type: Manual
Fuel Type: CNG
Seat: 8

Random Forest Regression: 3322205.90
Bagging Regression: 3377801.70
Linear Regression: 6595810.00

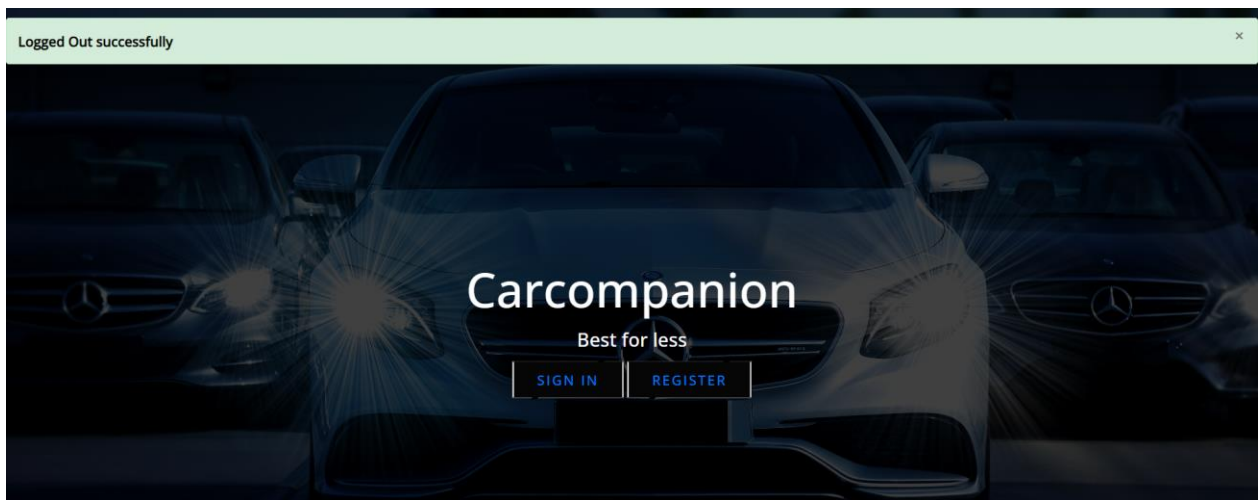
Audi A4

Year: 8
Kilometers Driven: 56000.0
Mileage: 12.0

Activate Windows
Go to Settings to activate Windows.

Search Cars with Model

Logged Out successfully



Carcompanion

Best for less

SIGN IN REGISTER

Logged Out of the System