Dissertation on

"Automation of Data Analysis and Web-Scraping to Value Old/Used Items"

Submitted in partial fulfillment of the requirements for the award of degree of

Bachelor of Technology
in
Computer Science & Engineering

UE20CS461A – Capstone Project Phase - 2

Submitted by:

Naren Chandrashekhar	**PES2UG20CS216**
Shriya Y.S.	**PES2UG20CS333**
Siddhant Verma	**PES2UG20CS338**
Aarav Babu	**PES2UG20CS486**

Under the guidance of

Prof. Shanthala P.T.
Assistant Professor
PES University

June - Nov 2023

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
FACULTY OF ENGINEERING
PES UNIVERSITY

(Established under Karnataka Act No. 16 of 2013)

Electronic City, Hosur Road, Bengaluru – 560 100, Karnataka, India

PES UNIVERSITY

(Established under Karnataka Act No. 16 of 2013)
Electronic City, Hosur Road, Bengaluru – 560 100, Karnataka, India

FACULTY OF ENGINEERING

CERTIFICATE

This is to certify that the dissertation entitled.

“Automation of Data Analysis and Web-Scraping to Value Old/Used Items”

is a bonafide work carried out by

Naren Chandrashekhar	**PES2UG20CS216**
Shriya Y.S.	**PES2UG20CS333**
Siddhant Verma	**PES2UG20CS338**
Aarav Babu	**PES2UG20CS486**

In partial fulfilment for the completion of seventh semester Capstone Project Phase - 2 (UE20CS461A) in the Program of Study -Bachelor of Technology in Computer Science and Engineering under rules and regulations of PES University, Bengaluru during the period June 2023 – Nov. 2023. It is certified that all corrections / suggestions indicated for internal assessment have been incorporated in the report. The dissertation has been approved as it satisfies the 7th semester academic requirements in respect of project work.

Signature	Signature	Signature
Prof. Shanthala P.T.	Dr. Sandesh B J	Dr. B K Keshavan
Assistant Professor	Chairperson	Dean of Faculty

External Viva

Name of the Examiners **Signature with Date**

1. ________________________ ________________________

2. ________________________ ________________________

DECLARATION

We hereby declare that the Capstone Project Phase - 2 entitled **“Automation of Data Analysis and Web Scraping for Valuing Second-Hand Items”** has been carried out by us under the guidance of Prof. Shanthala P.T and submitted in partial fulfillment of the course requirements for the award of degree of **Bachelor of Technology** in **Computer Science and Engineering** of **PES University, Bengaluru** during the academic semester June – Nov. 2023. The matter embodied in this report has not been submitted to any other university or institution for the award of any degree.

PES2UG20CS216	**Naren Chandrashekhar**
PES2UG20CS333	**Shriya Y.S.**
PES2UG20CS338	**Siddhant Verma**
PES2UG20CS486	**Aarav Babu**

ACKNOWLEDGEMENT

I would like to express my gratitude to Prof.Prof. Shanthala, Department of Computer Science and Engineering, PES University, for her continuous guidance, assistance, and encouragement throughout the development of this UE20CS461A - Capstone Project Phase – 2.

I am grateful to the Capstone Project Coordinator, Dr. Sarasvathi V, Professor and Dr. Sudeepa Roy Dey, Associate Professor, for organizing, managing, and helping. with the entire process.

I take this opportunity to thank Dr. Sandesh B J, Chairperson, Department of Computer Science and Engineering, PES University, for all the knowledge and support I have received from the department. I would like to thank Dr. B.K. Keshavan, Dean of Faculty, PES University for his help.

I am deeply grateful to Dr. M. R. Doreswamy, Chancellor, PES University, Prof. Jawahar Doreswamy, Pro Chancellor – PES University, Dr. Suryaprasad J, Vice-Chancellor, PES University and Prof. Nagarjuna Sadineni, Pro-Vice Chancellor - PES University, for providing to me various opportunities and enlightenment every. step of the way. Finally, this project could not have been completed without the continual support and encouragement I have received from my family and friends.

ABSTRACT

In the digital age, where technology intertwines with everyday life, PriceScout emerges as a beacon of innovation in the realm of device valuations. This web application redefines the traditional approach to estimating the value of mobile phones, laptops, and vehicles. Leveraging a combination of web scraping and machine learning, PriceScout ensures precision, speed, and unparalleled convenience for users seeking quick valuations.

The heart of PriceScout lies in its dynamic approach to data collection. Through web scraping, the platform gathers real-time information from diverse online sources, staying abreast of market trends and influencing factors. The integration of machine learning algorithms adds a layer of sophistication, analyzing device specifications, market demand, and other parameters to generate accurate estimates. The objective of this project is to develop an application that can help users determine the optimal price for second-hand items they want to sell. The application requires the user to enter a description of the item they want to sell. The input data is then stored in a database for further processing. The web scraper then scans different online marketplaces for similar items and the ML module determines their current prices. The application uses this data to calculate the optimal price for the user's item.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

The second-hand product market has only been growing each year, although listing products for sale has become easier. The process of finding an appropriate price to list the product at, however, has not. Traditional price prediction models that exist use outdated datasets. These outdated datasets, on which ML models are trained, can produce inaccurate results that do not reflect the current value. Every product is unique, with different features, collectible value, varying conditions, and depreciation costs, all of which must be considered when determining the final price for the end user. When listing a product, it must be compared to all other products currently listed on the market. The current demand for the product, as well as seasonal trends and fluctuations also need to be considered, which is tough for existing price prediction models. Our goal is to eliminate the manual and time-consuming process of considering all these constraints and comparing products on multiple sites. There is a need for software that can web-scrape/web-crawl, to retrieve data in real time to produce an accurate output. After retrieving the data, a method to analyze the dataset and create an algorithm that implements MI to find the best possible price. Considers the condition of the product and correlates the various attributes the user wants to specify in the product. The program will be able to suggest to the -user the most relevant data entry and provide a learning method that can give the most accurate prediction.

1.1 Web Scraping

In the vast landscape of the internet, where information sprawls across countless websites, web scraping emerges as the digital explorer's compass. This transformative technique involves the automated extraction of data from websites, transcending the confines of human capability.

Web scraping acts as a bridge between the untamed expanses of online content and the structured databases we crave. By navigating through HTML structures and parsing relevant data, web scraping unveils hidden insights, empowers data-driven decisions, and catalyzes innovation. Whether harvesting real-time market trends, aggregating product information, or conducting sentiment analysis, web scraping transforms the chaotic web into a structured well of knowledge.

Yet, with great power comes responsibility. Ethical considerations are paramount in web scraping, respecting the boundaries of website terms of service and legal regulations. In the hands of ethical practitioners, web scraping is a powerful tool for unlocking the treasures concealed in the digital wilderness, driving advancements across industries, and shaping the future of data-driven exploration.

1.2 Scikit-Learn: Empowering Machine Learning with Simplicity

Scikit-Learn, a robust machine learning library, stands as a beacon in the vast landscape of data science. Renowned for its user-friendly design and efficient implementation, Scikit-Learn accelerates the journey from raw data to meaningful insights.

1.2.1 Random Forest: Nature's Wisdom in Algorithms

Random Forest, a gem within Scikit-Learn, draws inspiration from nature's diversity. It assembles a multitude of decision trees, each contributing its wisdom to a collective decision. This ensemble approach not only enhances predictive accuracy but also mitigates overfitting, creating a resilient forest capable of navigating complex datasets.

1.2.2 Linear Regression: The Elegance of Simplicity

In the realm of predictive modeling, Linear Regression stands as an elegant cornerstone. Its simplicity belies its power—it seeks to establish a linear relationship between variables, providing a clear line of best fit. While sophisticated algorithms may dominate the landscape, the beauty of Linear

Regression lies in its interpretability and efficiency, making it a timeless choice for predictive analytics.

Together, Scikit-Learn, Random Forest, and Linear Regression form a triad of tools that demystify the complexities of machine learning. With Scikit-Learn as the orchestrator, Random Forest harnesses collective intelligence, and Linear Regression unfolds the poetry of simplicity—empowering data scientists to unravel the intricacies of their datasets and pave the way for informed decision-making.

1.3 Python Web Framework: Flask

Flask, a micro web framework for Python, has emerged as a beacon of simplicity in the realm of web development. With a philosophy of simplicity and modularity, Flask empowers developers to build robust web applications with ease.

Key Features:

- Minimalistic Design: Flask embraces minimalism, providing only the essentials needed for web development. This simplicity makes it easy to learn and quick to get started.
- Extensibility: While Flask comes with the basics, its modular design allows developers to seamlessly integrate extensions based on project requirements. This flexibility ensures that developers can tailor their applications precisely.
- Jinja Templating: Flask incorporates Jinja templating, enabling dynamic content rendering. This powerful feature simplifies the creation of dynamic web pages by allowing the embedding of Python-like code directly into HTML templates.
- Built-in Development Server: Flask includes a built-in development server, making it convenient for developers to test and debug their applications during the development phase.

Chapter 2

Problem Statement

Traditional price prediction models rely on outdated datasets, resulting in inaccurate pricing information, especially during market volatility. This creates challenges for sellers who struggle to price their products effectively, leading to unsuccessful sales. To address this issue, we aim to develop a system that simplifies the process of pricing products for resale by accurately predicting prices based on real-time data. The project will comprise of two parts, including web scraping/crawling to collect a valid dataset and automating the process of cleaning, pre-processing, scaling, and correlation of data to create a model that can identify the cost of certain old items. The end goal is to provide a final performance metric for the dataset of the user's requirement, enabling them to make informed pricing decisions. This system benefits both sellers and buyers by ensuring fair pricing and increasing the likelihood of successful sales in a rapidly changing market.

Chapter 3

Literature review

3.1 Auto-Prep: Efficient and Automated Data Preprocessing Pipeline

3.1.1 Introduction

The paper discusses the development of an automated, data-driven, and interactive system for data preprocessing in the field of machine learning. The system is designed to identify potential flaws in the data and report results and recommendations to the user. The following components are meaningfully automated: data type detection, missing values imputation, qualitative data encoding features scaling, feature selection and extraction.

3.1.2 Implementation

The paper evaluates the proposed method on six regression and five classification datasets with diverse features. For evaluation, the proposed architecture is employed on ten different and diverse datasets for automatic data preprocessing before passing it to an ML algorithm. The results are then compared with the results generated by the same ML algorithm but implemented on manually preprocessed data.

3.1.3 Results

The results have shown that not only did this approach make the whole process uncomplicated and facile, but it was also able to improve the performance of the model significantly. The proposed method is competent in making rational decisions based on the given dataset.

The advantages of this approach include its automation, which reduces human error and saves time. It also improves model performance significantly compared to manually preprocessed data.

However, one limitation of this approach is that it may not be suitable for all types of datasets for machine learning tasks. Additionally, some users may prefer manual preprocessing for greater control over their data.

3.2 LSTM Online Training and Prediction: Non-Stationary Real Time Data Stream Forecasting

3.2.1 Introduction

The paper discusses an offline and novel online LSTM model for online cryptocurrency forecasting and algorithmic trading. The paper compares the results of this model to similar publications. The previously mentioned work is practical and reliable and can run in real-time. Such a model may be inciting to algorithmic traders who seek determinism in their trading strategies.

3.2.2 Key Pointers

On page 10, the paper mentions a small four-wheeled robotic land vehicle that demonstrates the practicality and benefits of offloading the continuous task of intrusion detection based on deep learning (LSTM). This approach achieves high accuracy much more consistently than with standard machine learning algorithms and is not limited to a single type of attack or the in-vehicle CAN bus as many previous works. Such attacks can include denial of service, command injection, and malware.

3.2.3 Result

However, the paper does not provide detailed descriptions of other existing approaches, results, advantages, or limitations beyond what is mentioned above.

3.3 New Evaluation Metric for Demand Response-Driven Real-Time Price Prediction Towards Sustainable Manufacturing

3.3.1 Introduction

The paper discusses a new approach called Knowledge-Preserving Decoder (KPD) for improving the accuracy of real-time prediction (RTP) of energy consumption in manufacturing processes. The paper presents a comparison of the proposed KPD approach with other commonly used prediction algorithms and shows that the KPD approach outperforms them in terms of prediction accuracy.

3.3.2 Approaches

The paper also discusses the suitability of integrating KPD into other commonly used prediction algorithms. The outcomes of this study can improve the RTP prediction quality, help manufacturers reduce energy costs through demand response, and contribute positively to sustainable manufacturing.

3.3.3 Results

The advantages of the proposed KPD approach include its ability to preserve knowledge from previous time steps, which improves its accuracy in predicting future energy consumption. The proposed method is also shown to be effective in various prediction algorithms.

3.3.4 Limitations

The limitations of the study include the fact that it was conducted on a single dataset, and therefore, its generalizability to other datasets may be limited. Additionally, the proposed method requires a large amount of training data to achieve optimal performance.

Overall, the paper provides a detailed description of existing approaches for RTP prediction and presents a new approach that outperforms them in terms of accuracy. The proposed KPD approach has several advantages over existing methods but also has some limitations that need to be addressed in future research.

3.4 Automation of Analytical Data Processing

3.4.1 Basic Approaches

The paper mentions several existing approaches, including:

- Using several different techniques for automated analysis of transactions
- Using statistical methods for analysis of both separate transactions and sequences of transactions
- Personalized medicine, which involves considering individual characteristics of the body when diagnosing diseases, predicting the health of a particular person, and creating personalized recommendations to prevent the development of diseases.

3.4.2 Limitations

However, it is important to note that these approaches are not exhaustive and there may be other existing approaches not mentioned in the given pages.

It does mention some benefits of using automated statistical analysis in general, such as reducing labor costs and time of reaction. Additionally, the personalized medicine approach has the potential to improve healthcare by tailoring treatments to individual patients.

3.5 Monitoring of Stocks using LSTM Model and Prediction of Stock Prices

3.5.1 Introduction

This paper discusses the use of interactive dashboards for monitoring stock market trends and real-time stock prices. The paper also explores the growing interest in forecasting stock prices and gaining profits through these predictions. Specifically, the paper proposes a predictive model that uses Long Short-Term Memory (LSTM) to predict future stock price values based on historical stock prices. The authors also discuss potential areas for improvement in their methodology, such as incorporating additional features like daily volume or fundamental ratios into the model. Overall, this paper aims to contribute to the field of stock market prediction by proposing a new approach that utilizes LSTM models and interactive dashboards for real-time monitoring and forecasting of stock prices.

3.5.2 About the Paper

This study has used technical analysis to predict the upcoming stock prices for the major oil and gas industry GAIL which is being enumerated in the National Stock Exchange (NSE) of India. To perform the above-mentioned task, an LSTM model has been created and compared the performance of the same with other existing machine learning models including linear regression, ridge regression and lasso regression by using the evaluation metric as R2 Score. The R2 Score is always between 0 and 1 for every model, the model whose value is closer to 1 performs better than those models whose R2 Score is closer to 0.

3.5.3 Limitations

However, this paper does have some limitations. National Stock Exchange (NSE) and Bombay Stock Exchange (BSE) are the two main stock exchanges known in India. The companies listed in India are available in both exchanges but the stock prices of the same companies at one point of time may or may not be equal in the two stock exchanges. The stock price of the same company differs in both i.e., they're not in sync in real time. Due to this they were forced to fetch the stock price data from only one of these exchanges. Another difficulty faced was in creating the real time dashboard. There are several APIs that provide historical stock price data for free, but we could not find any API that provides real time stock price for free. Web Scraping Python Script is shown in Fig 13. Only Premium APIs provide real time stock price data, and they charge around 10$ per month for the same. So, instead of relying on these Premium APIs, we worked on a Python script that scrapes the real time stock price data of any company from Google Finance website. This Python script utilizes the requests and bs4 (Beautiful Soup) library and will not be used for commercial purposes.

Chapter 4

Project Requirements Specification

4.1 Project Scope

The objective of this application is to reduce the time taken by users to price their second-hand products accurately. Users must search markets, compare prices and features to list a product and get a reasonable value for their product. This is a time-consuming process, so the end goal is to develop an application that can automatically evaluate a product at a reasonable price. The product being developed is an application that can be used by end users to help value their second-hand item by Scraping/Crawling multiple websites. Considering parameters such as product features and Images. Can be used to predict the price of new products by comparing features. There are some limitations to the product. The application is not foolproof. Web Scraping may be limited due to web-site policies complexity of the model might be high due to the combination of workflows.

4.2 Product Perspective

The Automation of Data Analysis and Web-Scraping to Value Old/Used Items product is developed in response to the growing need for a more efficient and accurate way to value old or used items. In the past, people relied on various methods to assess the value of their possessions, including personal knowledge and experience, consultation with experts, and manual research. However, with the rise of the internet and e-commerce, a wealth of data on products became available online, making it possible to automate the process of data analysis and web scraping to assess the value of old more accurately or used items. The origin of the product can be traced back to the development of web scraping and data analysis tools, which have become increasingly sophisticated over the years. As these technologies advanced, it became possible to automate many of the tasks involved in data collection, cleaning, and analysis, making it easier and more efficient to value old or used items.

The context of the product is primarily in the realm of personal finance, where individuals may need to assess the value of their possessions for insurance purposes, estate planning, or when selling or buying second-hand items. However, the product may also be used in other contexts, such as research or business analysis, where data analysis and web scraping are essential tools for gathering and analyzing information.

4.3 Product Features

Some of the major features and functions of this product include:

- Web Scraping: The product allows users to scrape data from various websites, including auction sites, online marketplaces, and classified ad sites. This feature allows users to gather data on a wide range of products and analyze them to determine their value.
- Data Cleaning: The product includes tools for cleaning and formatting data, which is essential for accurate analysis. This feature allows users to remove duplicate entries, correct errors, and standardize data for easier analysis.
- Data Analysis: The product includes tools for analyzing data and generating reports on various metrics such as price, age, condition, and more. This feature allows users to value quickly and accurately old or used items based on a variety of factors.
- Automation: The product is designed to automate many of the tasks involved in data analysis and web scraping, allowing users to save time and effort. This feature includes automated data collection, cleaning, and analysis, as well as automated reporting and notifications.
- Customization: The product allows users to customize their analysis and valuation criteria based on their specific needs and preferences. This feature includes the ability to create custom formulas, filters, and rules for data analysis.

4.4 User Classes and Characteristics

- Admin: The admin is responsible for managing the entire system. They can add new features, manage user accounts, monitor the system's performance, and troubleshoot any issues that arise.
- Data Scientist: The data scientist is responsible for developing the models and algorithms used by the system to automatically analyze data. They can use statistical analysis, machine learning, and other techniques to identify patterns, trends, and insights in large datasets.

- Data Analyzer: The data analyzer is responsible for automatically collecting and analyzing data on second-hand products, including their price, condition, and other relevant information. It can use data analysis tools to identify patterns and trends in the data and develop models for predicting prices.
- Web Scraper: The web scraper is responsible for collecting data from various websites and platforms that offer second-hand products for sale. They use web scraping tools to extract relevant information such as the product name, description, and price.
- User: The user is the end-user of the system, who can access the website or application to search for second-hand products and get a predicted price range for each item. They can also provide feedback and suggestions to improve the system's performance.

4.5 Operating Environment

The Automation of Data Analysis and Web-Scraping to Value Old/Used Items product can operate on a variety of hardware platforms, operating systems, and software components. Here are some of the key details:

- Hardware platform: The product is designed to operate on standard computing hardware, including desktops, laptops, and servers. There are no specific hardware requirements, but a higher-end computer may perform better when running more intensive web scraping or data analysis tasks.
- Operating system: The product is compatible with multiple operating systems, including Windows, Mac, and Linux. It can run on Windows 7, 8, 10, and Windows Server, macOS 10.12 and later, and Linux distributions such as Ubuntu, Debian, and Fedora.
- Software components: The product relies on several software components, including web browsers such as Google Chrome, Mozilla Firefox, or Microsoft Edge, which are used for web scraping. It also uses programming languages such as Python and R, which are used for data analysis, along with various libraries and packages, such as BeautifulSoup, Pandas, and Scikit-learn.

Additionally, the product may require specific software tools or packages to be installed, depending on the specific use case and requirements. For example, if the user needs to scrape data from a specific website, they may need to install additional libraries or extensions to enable the web scraping process.

Overall, the Automation of Data Analysis and Web-Scraping to Value Old/Used Items product is designed to be flexible and adaptable to a wide range of hardware and software configurations, making it accessible to users across various environments.

4.6 General Constraints, Assumptions and Dependencies

Dependencies and constraints for a second-hand product price prediction system using web scraping may involve various issues, including legal implications, usage limitations, and specific software/hardware requirements. Here are some potential examples:

- Legal Implications: Web scraping can potentially infringe upon intellectual property rights and other legal issues. The system needs to ensure that the data collected and used are legal and ethically obtained. The system must also comply with laws and regulations related to data privacy, data protection, and other relevant laws.
- Usage Limitations: The system may need to set limitations on the number of requests made to a website within a given period to avoid overloading the website with traffic or causing server crashes. The system may also need to limit the number of concurrent users or requests per user to prevent overloading the system.
- Specific Software/Hardware Requirements: The system may require specific software or hardware to run effectively, such as web scraping tools, data analysis software, or high-performance computing hardware. Users may need to have a specific operating system or web browser to use the system.
- Accuracy of Predictions: The system's accuracy in predicting prices may depend on the quality and quantity of data available for analysis. The system may need to consider factors such as the condition, age, and popularity of products, as well as the market demand and supply of similar items.

- Updating Data: The system will need to update its database frequently to ensure that the predicted prices are accurate and up to date. The system may need to set up automatic crawlers to retrieve the latest data or rely on user input to update the database.
- Data Storage and Security: The system will need to ensure that the collected data is stored safely and securely, protected from unauthorized access or data breaches. This may involve implementing encryption, user authentication, and other security measures to prevent data theft or misuse.

4.7 Assumptions for this project may include:

The system assumes that:

- The data collected from various sources is accurate and up to date.
- Sellers will provide accurate information about their products, including the condition and features of the items.
- The market demand and pricing trends will remain relatively stable over the period in which the system is used.
- The integration with existing second-hand marketplaces will be successful and seamless.
- There will be a sufficient user base to make the project financially viable.
- Users have a basic understanding of pricing and market trends.

4.8 Risks

- Technology Failure: The system may rely on multiple technologies, such as web scraping tools, data analysis software, and databases. Any failure or malfunction of these technologies can cause the system to break down or stop functioning correctly. For example, if the web scraping tool fails to extract data from a website, the system may not be able to update its database, leading to inaccurate predictions.
- Hardware Failure: The system may require high-performance computing hardware to handle large datasets and complex computations. If the hardware fails, the system may not be able to perform its tasks correctly or may slow down, causing delays in delivering the final project.
- Version Compatibility Problems: The system may rely on specific versions of software or libraries, and compatibility issues may arise when updating or upgrading these components.

This can cause the system to malfunction or stop working entirely, requiring additional time and resources to resolve the compatibility problems.

- Data Quality Issues: The accuracy of the price predictions relies heavily on the quality of the data used by the system. Poor quality data, such as incomplete or inaccurate data, can lead to incorrect predictions and unreliable results. Data quality issues can be difficult to detect and resolve and may require significant effort and resources to correct.
- Legal Issues: Web scraping can raise legal issues related to copyright infringement, data privacy, and intellectual property rights. If the system collects and uses data without permission or violates any legal requirements, it can face legal action that may cause delays in delivering the final project or even prevent its release.
- User Adoption: Even if the system is technically sound, its success ultimately depends on user adoption. If the system is difficult to use or the predictions are not accurate, users may not adopt it, rendering the final project useless or ineffective. The system needs to be designed with user needs and preferences in mind to ensure user adoption and engagement.

4.9 Functional Requirements

The user interacts with the user interface, and provides information of the item to be sold, which is the data collection stage. After which the below functions are performed.

- Data Collection: The system should be able to collect real-time data on product prices, market trends, and other relevant factors that affect pricing.
- Data Analysis: The system should be able to analyze the collected data to identify patterns and trends and make accurate predictions on pricing.
- Pricing Algorithm: The system should have a pricing algorithm that considers all relevant factors, such as product condition, features, and market demand.
- User Interface: The system should have a user-friendly interface that allows sellers to easily input product information and receive accurate pricing information in return.
- Error handling: If the data given by the user is incorrect or the description of the product is wrong, our web crawlers may provide a price point accordingly. The user must ensure to fill in the details and description of the product correctly.

4.10 External Interface Requirements

4.10.1 User Interfaces

- The system should respond quickly to user inputs, with minimal delay between actions and results. Inputs for users to describe the product and product features should be easy.
- The Screen Layout and Standard Functions should be intuitive, with commonly used functions and features. Navigation should be consistent and easy to follow.
- All the searching and processing done in the backend should be abstracted for the end-product. If integrated with other e-commerce websites, ads need to be automatically generated for the product user is trying to list.
- The system should have clear and informative error messages that help users understand and resolve any issues they encounter.

4.10.2 Hardware Requirements

Our goal is to deploy the end-product as a software that is free to use for all users.
Users will require a System that has a strong internet connection and has a GPU to clean and process the data. The software/program needs to be compatible with all operating systems like Windows, macOS and Linux. If the project is to be implemented as a Client-Server architecture, then the servers must be scalable and capable of handling traffic and users must be limited with no. of queries.

4.10.3 Software Requirements

Name: **PriceScout**

- Description:

  The project aims to create a software solution that simplifies the process of selling second-hand products by providing users with real-time pricing suggestions. The software will utilize web crawlers to collect data from various e-commerce platforms, such as Amazon, eBay, and OLX. This data will then be cleaned and preprocessed before being used to train machine learning models to predict the market price of a particular item. The software will be designed to handle non-stationary and dynamic data streams, and the models will be updated periodically using online learning techniques. The final product will have a user-friendly interface that allows users to input details about their product, and the software will output an accurate suggested selling price based on current market trends. The software will also include data visualization tools to help users understand how the suggested price was calculated.

- Version and Release Number:

  Initial version may be labeled as version 1.0, subsequently version numbers can change with respect to the changes done such as for minor changes or quality of life improvements only the third number will change with respect to the version the developers are working on for e.g. 1.1.3. If additional features are added to existing branches, then the second number in the version number changes for e.g. 1.2.3. If there's a major change with backend or front-end implementation, then the first number changes for e.g. 2.0.0.

- Databases:

  The database used for the product should be specified based on the development team's preferences and expertise. Common databases used for web applications include MySQL, PostgreSQL, and MongoDB.

- Operating Systems:

  The product should be designed to run on a variety of operating systems to ensure maximum compatibility and accessibility. This includes popular operating systems such as Windows, macOS, and Linux.

- Tools and Libraries:
  - Data Collection: Can be done by using web-crawling API or Custom Python-script parsers built by us.
  - Data Analysis: Data analysis and processing is done by ML Pipelines such as Auto-Prep, additional tools need to be used to detect outliers and eliminate them.
  - User Interface: The front-end of the software can be made as a web-app using tools such as React or Flask.
  - Security: Security for locally stored data must be encrypted and stored securely.
  - Source: The source code for the software product should be kept in a version control system such as Git. This allows for easy collaboration between developers and version management of the software components. The source code should be kept in a private repository to ensure the security of the product's intellectual property.

4.10.4 Communication Interfaces

The app will require a good internet connection. Retrieving and sending data will be done through HTTP requests.

4.11 Non-Functional Requirements

4.11.1 Performance Requirement

Once the user has given the description about the product and request for the price, our application shouldn't take more than a minute to give a price for which the article can be sold. It takes time as our model uses real time data and retrieving this data from multiple websites and finding an optimal price point can take a while.

Our product is reliable as it is using multiple platforms to gather data and give one price point.

With enough resources and compute power, our application can be robust enough to find prices of multiple products by multiple users.

4.11.2 Safety Requirements

Our web crawlers must send requests at a periodic well-spaced intervals of time as some websites may block our crawlers. In such a case, our application will be unable to gather enough data for finding the price of the product.

4.11.3 Security Requirements

The system should have robust security measures in place to protect user data and prevent unauthorized access or hacking.

4.12 Other Requirements

- Accuracy and Reliability: The system should be highly accurate and reliable, with a low margin of error in predicting prices.
- Scalability: The system should be able to handle a large volume of data and users and be easily scalable as the user base grows.
- Portability: The application can be accessed by either an application or a website on any device.
- Maintenance: The cost of running this application is high and will need to be timely checked.

4.13 Summary

All in all, our software calls for multiple intuitive interfaces that the user can interact with easy and prompt retrieval of relevant information. To store the data, we need to be able to export and import files from different sources to different formats quickly. To help better understand the data and information collected the same must be visually represented to the user. Our web-scraping and algorithm related to the retrieval of the costs has to be hidden from the user and only the predicted value must be represented.

Chapter 5

System Design

5.1 Detailed Design

5.1.1 Purpose

The purpose of the detailed design is to plan our system to meet the requirements specified at the start. In the detailed design we see what the input data is for each model, how the model implementation is carried out and how the output is interpreted. The main goal of the project is to design a tool that trains over second-hand product data and then predicts the final price.

5.1.2 Project Components

5.1.2.1 UI

- Web browser: The user interface should include a web browser component that allows the user to navigate to the target website and select the items they want to scrape.
- Input fields: The user interface should have input fields to specify the search keywords or URLs for the target items.
- Search and scrape buttons: The user interface should include buttons to initiate the search and web scraping process.
- Results display: The user interface should display the results of the web scraping process, such as the scraped data or item prices.
- Help and support: The user interface should provide help and support options, such as user guides, tutorials, or customer support channels, to assist users with any issues or questions.

5.1.2.2 Input-Processing

- Input validation: Validate the user input for the target website URL or search keywords to ensure that the input is valid and meets the project requirements.

- Data cleaning: Clean the input data to remove any irrelevant or redundant information and standardize the input data format for further processing.
- Data normalization: Normalize the input data to a standardized format to ensure consistency and compatibility with the web scraping algorithms and tools.
- Data filtering: Filter the input data based on specific criteria, such as data relevance, date range, or language, to ensure that the scraped data is relevant and useful.
- Data enrichment: Enhance the input data by adding additional metadata, such as tags, categories, or descriptions, to provide more context and insights for the scraped data.
- Data sampling: Sample the input data to test and validate the web scraping algorithms and tools before applying them to the full input dataset.
- URL normalization: Normalize URLs to ensure consistency and eliminate duplicates, as websites may have multiple URLs pointing to the same page or content.
- User agent spoofing: Spoof or modify the user agent string sent by the web scraper to simulate different web browsers or devices and avoid detection or blocking by the target website.
- Proxy management: Manage proxies to rotate IP addresses and avoid IP blocking or throttling by the target website, especially for large-scale or frequent web scraping tasks.
- Session management: Manage session cookies or tokens to maintain the user session and avoid frequent re-authentication or security checks by the target website.
- Captcha handling: Handle captchas or other security measures used by the target website to prevent automated scraping, by either solving them manually or using automated captcha solving services.
- Rate limiting: Limit the frequency and volume of web scraping requests to avoid overloading the target website and comply with ethical and legal guidelines.

5.1.2.3 Web Scraper

The web scraper is one of the crucial components for the system as the accuracy of the prediction can significantly improve if the web scraped data is highly correlated with the data the user has inputted. The real-time web scraping component will be responsible for scraping data from the selected websites and stream it to the data processing component. The framework should be able to handle dynamic websites and be able to traverse the websites with ease. The input to this component would be the data given by the user, using this data the web scraper needs to be able to query and look for similar products. The processing component of the web-scraper should be able to identify outliers and errors so that it does not feed that into the prediction algorithm. The web scraper should be designed to handle large volumes of data efficiently. The output from the web scraper must finally be able to clearly label the data that is being extracted, so that the storage of the scraped data is put in the correct columns.

5.1.2.4 Database Management System

MySQL is a very popular database used for different types of data storage and management. MySQL is a Relational Database Management System (RDBMS) that uses Structured Query Language (SQL) for managing data. It's ideal for storing structured data, such as data that fits well into tables with fixed columns and rows. MySQL is often used for transactional systems and applications that require a high level of data consistency and reliability. On the other hand, MongoDB is a NoSQL document-oriented database that uses JSON-like documents with dynamic schemas. It's ideal for storing unstructured or semi-structured data that can be represented in a hierarchical format, such as product catalogs, blog posts, and social media data.

MySQL will be used to store the data in a structured format after the data is selected and processed from the NoSQL database. This is important because MySQL is a popular and reliable RDBMS that is widely used in many industries. It can provide fast data retrieval, transaction management, and support for complex queries. By using MySQL, you can easily transform the data from the NoSQL database into a structured format that can be used with data analysis tools like Pandas and other SQL-based applications.

Overall, MySQL will provide a robust and flexible data management system that can handle structured data effectively. This will ensure that data can be managed effectively and provide accurate and relevant results to the user.

Chapter 6

High Level Design Document

6.1 Introduction

A high-level design document provides a roadmap for how a proposed solution or system will be implemented, by defining its overall structure, major components, and their interactions. This document provides a detailed overview of the proposed solution, outlining its key components, features, and functionality. It includes conceptual architecture models, system requirements, and specifications for each component or module. Overall, this document is a crucial tool for ensuring that a proposed system or solution is well-designed, meets requirements, and can be implemented successfully. It provides a clear roadmap for development, helps mitigate risks, and ensures that all stakeholders are aligned and informed.

This solution would automate both the preprocessing and algorithm selection steps, leveraging web scraping, data cleaning, automated data analysis, and reporting tools to provide a fast and accurate way to value old or used items. The solution could be particularly useful for individuals or organizations that need to manage large inventories of used items or are looking to sell or donate items, as well as for insurance companies, pawn shops, and other businesses that need to estimate the value of used items.

The first step in the process would be to gather data on the item being evaluated from relevant websites, including e-commerce sites, online marketplaces, and other sources. This can be done using web scraping tools or libraries that automatically extract data from these sites.

Once the data has been collected, an automated data preprocessing module can be used to clean and transform the data. This can involve tasks such as removing duplicates, correcting errors, and standardizing data formats. The automated preprocessing module can be trained on a large dataset of historical data to learn the best methods for cleaning and transforming data.

Once the data has been cleaned and transformed, an automated algorithm selection and tuning module can be used to select the best machine learning or statistical model for estimating the value of the item. The algorithm selection module can be trained on a large dataset of historical data to learn which algorithms perform best for different types of items and data.

With the data cleaned, transformed, and the algorithm selected, the next step would be to use the automated data analysis techniques, such as machine learning or statistical modeling, to estimate the item's value. The automated algorithm tuning module can be used to fine-tune the selected algorithm for the specific data being analyzed, considering factors such as market trends, seasonal fluctuations, and geographic location.

Finally, the results of the data analysis can be presented to the user in a clear and easily understandable format, such as a report or dashboard. This could include information on the estimated value of the item, the features that were used to estimate the value, and any other relevant information that can help the user make informed decisions.

6.2 Current System

There are several websites and applications that offer price prediction for used items based on machine learning algorithms, such as eBay, Amazon, and various price comparison websites. However, most of these systems rely on a combination of machine learning algorithms and human curation, with the algorithms being used to suggest prices and humans making the final decision on the prices.

As for fully automated systems like the one being proposed, it is a new and unique development that aims to provide an accessible and affordable solution to not only price prediction but the entire automated process of data analysis and model testing and training which can be used in any other project.

This makes the project highly feasible or valuable. This system offers quite a quick, unique, and innovative solution to the problem of automatically pricing used items without any human intervention.

6.2.1 Design Considerations

The project will utilize certain components and implement various methodologies to achieve the desired results.

6.2.1.1 Design Goals

- Data Collection: The data collection process in this project involves acquiring data from various sources based on the user's input. The user can input keywords, descriptions, or even images to specify the type of data they require. A web scraper will then be used to collect data from relevant websites, and the user can also specify specific sites from which they want data to be gathered. The collected data will then be stored in a suitable format for further processing.
- Data Cleaning: The data cleaning process involves preparing the collected dataset for the machine learning pipeline. This includes converting the dataset into a set of informative text that can be used to train the machine learning models. The keywords can either be chosen by the user or based on the method of cleaning applied to the dataset. This process may also involve removing irrelevant data, handling missing values, dealing with duplicates, and resolving inconsistencies to ensure the data is of high quality.
- Data Augmentation: Data augmentation is a crucial step in data preparation that involves generating additional training data by applying various transformations to the original dataset. This process can help regularize the dataset, reduce overfitting, and improve the model's performance by providing more diverse and representative training data. This step may involve techniques such as flipping, rotating, cropping, zooming, adding noise, or changing the color and contrast of the images. Understanding the required parameters for these transformations is also important to ensure the augmented data is realistic and representative of the original dataset.

6.2.1.2 Architecture Choices

- Feature selection: The feature selection process will be carried out using Genetic Algorithms, which is a metaheuristic optimization technique inspired by the process of natural selection. Specifically, the implementation of Genetic Algorithms will involve a random search approach to explore the search space of possible feature combinations in order to find the optimal subset of features for the given task.

- Feature Construction: The feature construction process will involve the use of decision tree-based methods and Genetic Algorithm (GA)-based methods to enhance the model's predictive power. The basic features will be considered as a starting point, and new features will be constructed from them using these techniques to capture more complex relationships and patterns in the data.
- Feature Extraction: The feature extraction process in this project will involve identifying relevant features in the data by altering the original features through mapping. Specifically, this will involve applying transformation functions to the original features to extract higher-level features that capture more informative and discriminative characteristics of the data. The goal is to reduce the dimensionality of the data while retaining the most relevant information for the given task.

6.3 Constraints, Assumptions and Dependencies

- Assumptions: The websites we are scraping have consistent and reliable data. The data extracted through web scraping is legal and ethical to use. The features we are extracting from the data are relevant to the problem you are trying to solve. The predictions generated by our application are accurate and reliable.
- Dependencies: The quality of the dataset and the accuracy of the predictions depend on the quality of the data extracted through web scraping. The performance of the application depends on the performance of the pipeline for feature extraction and prediction. The speed and reliability of the web scraping process depend on the stability and speed of the internet connection and the availability of the websites. The storage and processing capabilities of the machine running the application are sufficient to handle the size and complexity of the dataset and the pipeline.

.

6.4 High Level System Design

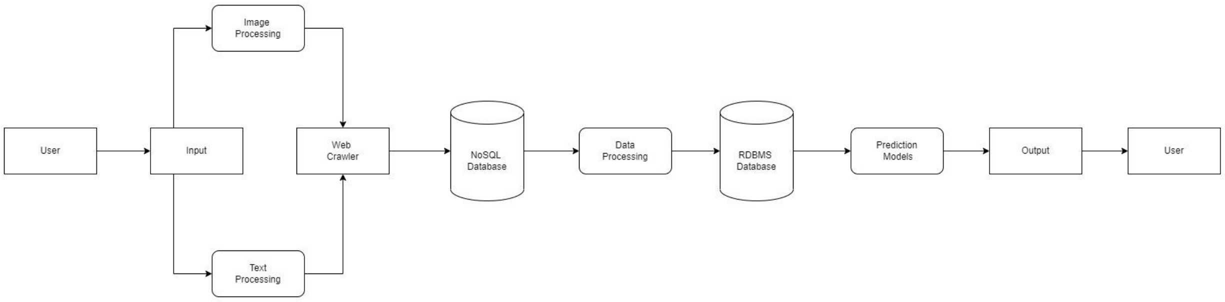


Fig 6.4.1 High Level System Design

Logical user group would be any user looking to sell a secondhand item. Application components are the User Interface, Price Prediction engine and web-crawler. The data components are RDBMS and NoSQL databases. Interfacing systems are API's and payment gateways being used in the system.

Some of the Technical specifications required to develop the system are:

- Custom Web Crawler to fit requirements of this system.
- DBMS for Data Processing and Storage
- Machine Learning Model to predict correct price using given parameters.
- User Interface that is easy to use for the end-user.

The system elements would be:

1. Conceptual or logical – Explained in section 10.1
2. Process - Explained in section 6.1
3. Physical – Explained in section 10.2
4. Module – The code organization will be done with the help of Jenkins software
5. Security – Text and Image inputs should be encrypted and kept only on the user's storage.

6.5 Design Description

The application consists of two main modules: the Web-Scraper module and the Machine Learning module. The Web-Scraper module is responsible for scraping data from online marketplaces, such as eBay or Amazon, and extracting relevant features for each item, such as product title, description, condition, and seller information. The Machine Learning module is automated which is further split into parts: the data preprocessing and the price prediction. Data preprocessing is an automated preprocessing pipeline that can clean and transform the scraped data into a suitable format for analysis. This pipeline handles missing data, outliers, and other common preprocessing tasks. The price prediction process involves an automated algorithm selection process that can identify the best machine learning algorithm for the given task. This may involve evaluating multiple algorithms and selecting the one with the best performance. Train the selected machine learning algorithm on the preprocessed data. Use the trained model to automatically value the old/used items based on their features.

Web-Scraper module interface:

- Input: URL of the online marketplace to scrape, search keywords to look for, maximum number of items to scrape
- Output: Extracted features for each item in a structured format (e.g., CSV file)
- Machine Learning module interface:
- Input: Extracted features for each item in a structured format (e.g., CSV file)
- Output: Trained machine learning model with optimized hyperparameters for predicting item prices

User interface:

- Input: Item to price (e.g., product title, description, condition)
- Output: Predicted price for the item, with a confidence interval and/or a range of prices based on similar items

6.6 Master Class Diagram

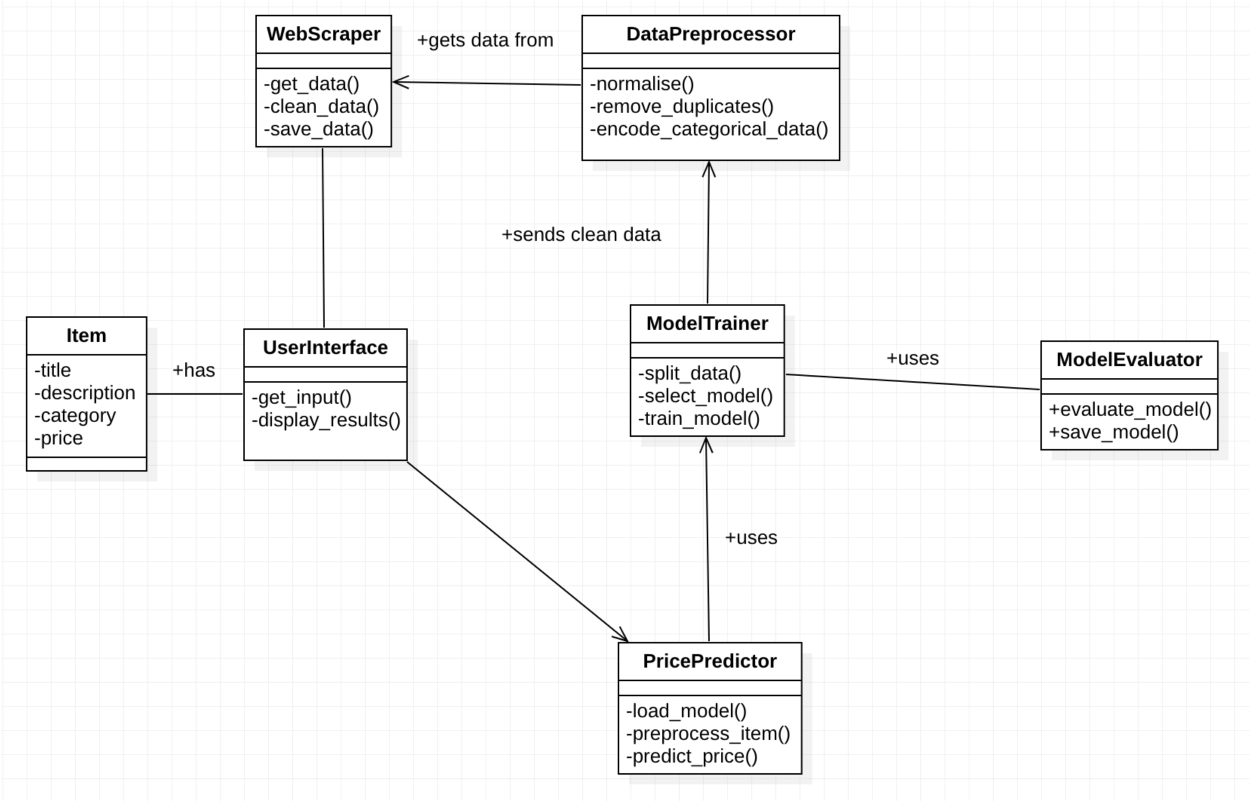


Fig 6.6.1 Master Class Diagram

6.7 Reusability Considerations

Reusability is an important consideration for any software project, as it can save time and resources in the long term. Here are some reusability considerations that could be planned for this project:

- Use of existing libraries and frameworks: The project can utilize existing libraries and frameworks for web scraping, data preprocessing, machine learning, and user interface design. For example, the BeautifulSoup library can be used for web scraping, scikit-learn for machine learning, and PyQt for user interface design.
- Modular design: The project can be designed in a modular way, with generic functionalities that can be used in other projects. The entire automated data analysis and machine learning system to predict the prices can handle a wide range of input data, making them reusable in other projects.

6.8 State Diagram

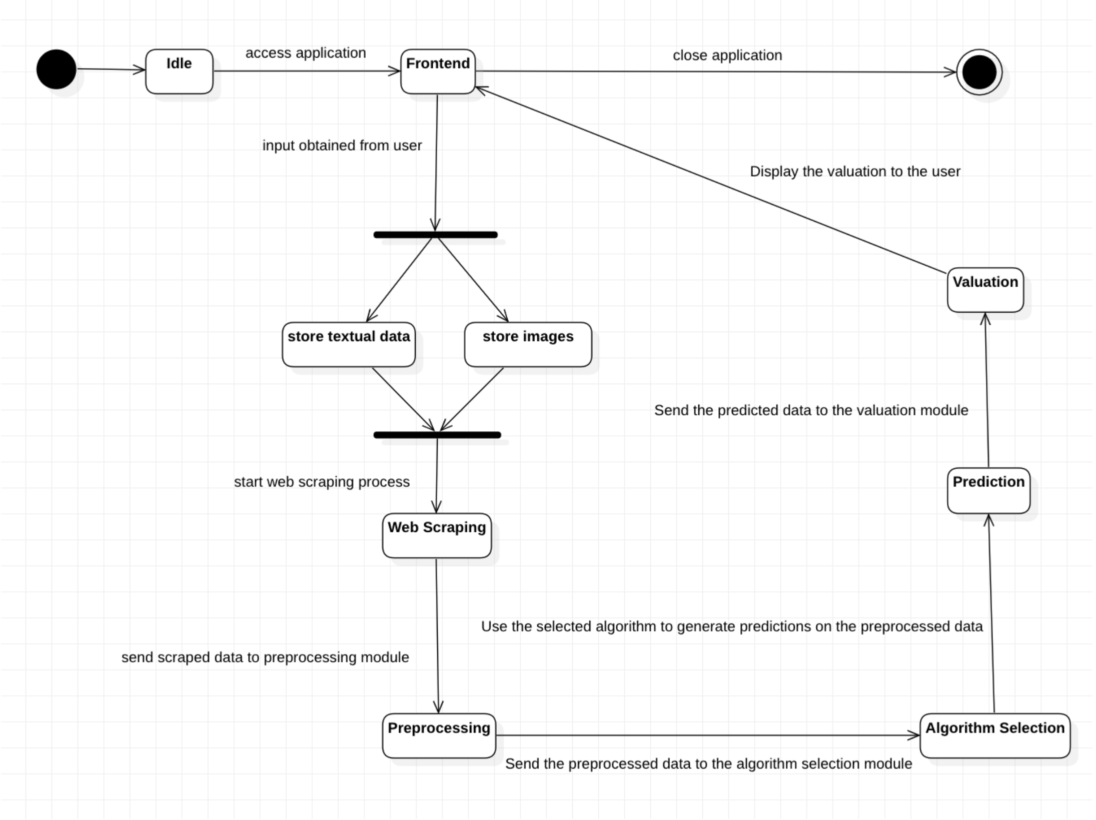
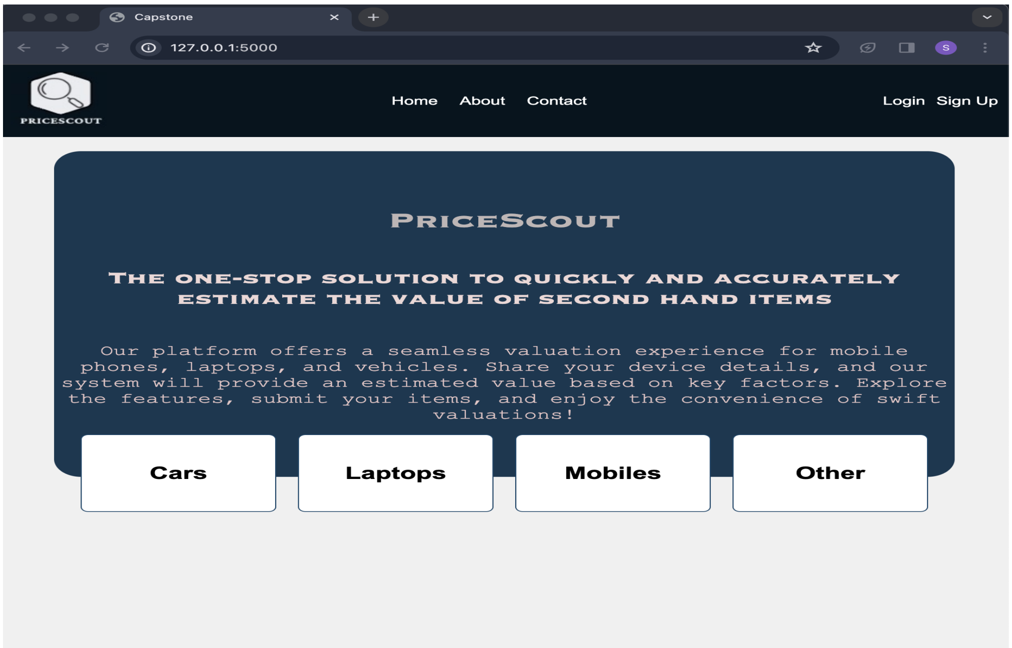

Idle
access application
Frontend
close application
input obtained from user
Display the valuation to the user
Valuation
store textual data
store images
Send the predicted data to the valuation module
start web scraping process
Prediction
Web Scraping
Use the selected algorithm to generate predictions on the preprocessed data
send scraped data to preprocessing module
Preprocessing
Algorithm Selection
Send the preprocessed data to the algorithm selection module

Fig 6.8.1 State Diagram

6.9 User Interface Diagrams

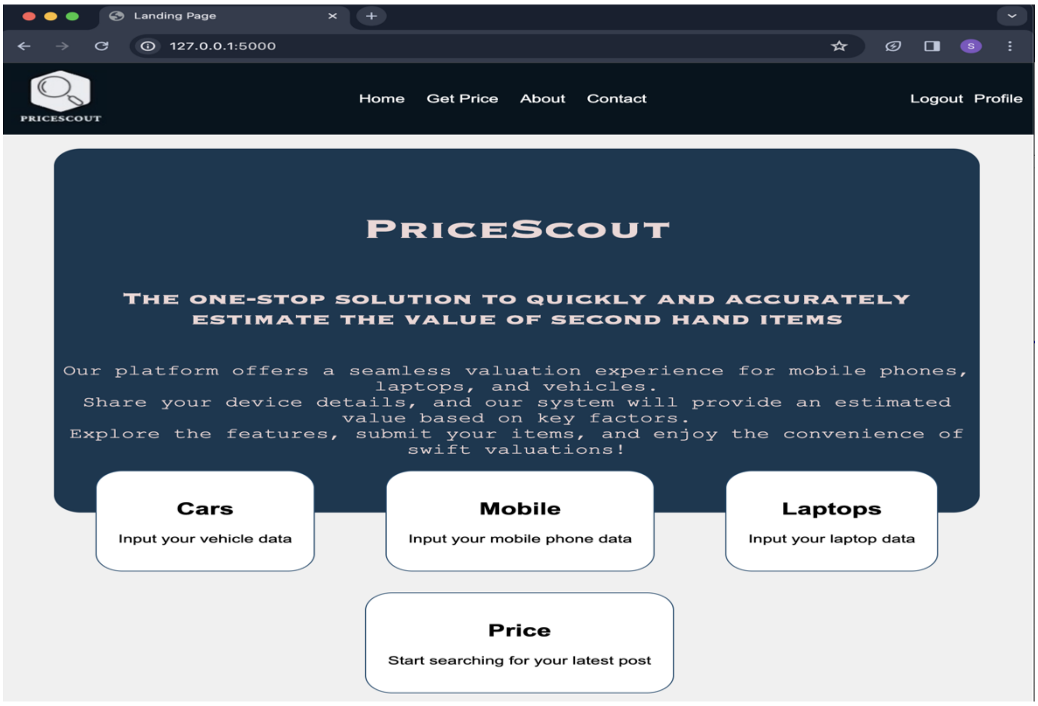


PriceScout

The one-stop solution to quickly and accurately estimate the value of second hand items

Our platform offers a seamless valuation experience for mobile phones, laptops, and vehicles. Share your device details, and our system will provide an estimated value based on key factors. Explore the features, submit your items, and enjoy the convenience of swift valuations!

(i)



PriceScout

The one-stop solution to quickly and accurately estimate the value of second hand items

Our platform offers a seamless valuation experience for mobile phones, laptops, and vehicles.
Share your device details, and our system will provide an estimated value based on key factors.
Explore the features, submit your items, and enjoy the convenience of swift valuations!

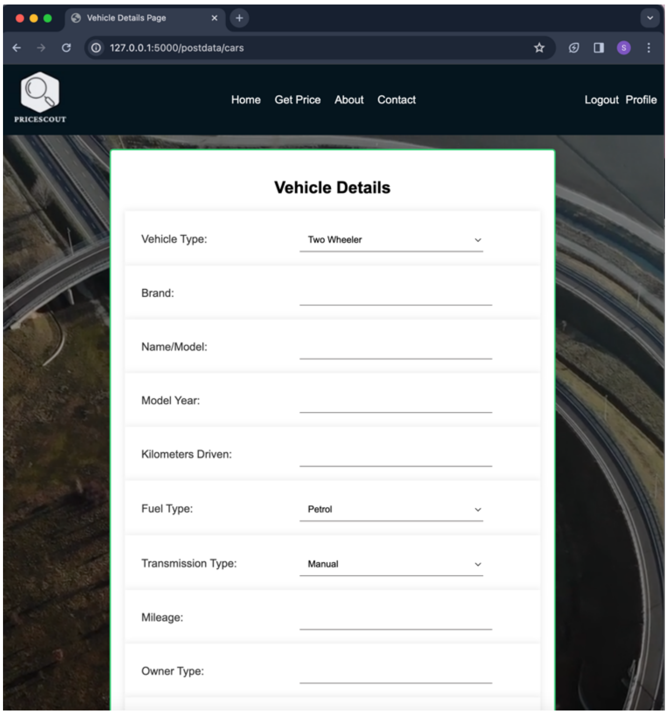
Input your vehicle data

Input your mobile phone data

Input your laptop data

Start searching for your latest post

(ii)



Vehicle Details

Vehicle Type:	Two Wheeler
Brand:	
Name/Model:	
Model Year:	
Kilometers Driven:	
Fuel Type:	Petrol
Transmission Type:	Manual
Mileage:	
Owner Type:	

(iii)

Fig 6.9.1 Sample User Interface (i) Home screen without login (ii) Home screen with Login (iii)Entry form for car data

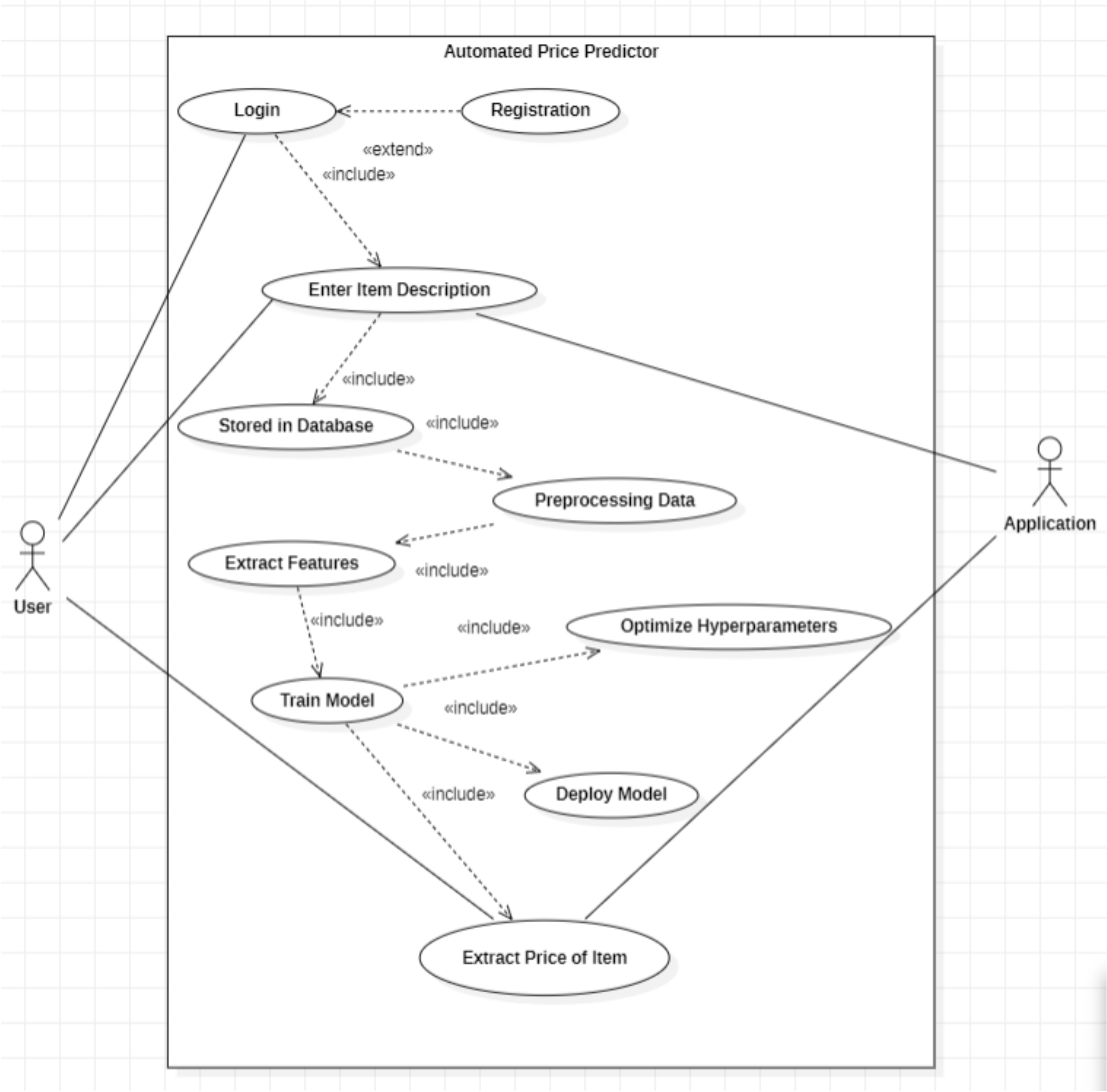


Fig 6.9.2 User Interface Diagram

6.10 External Interfaces

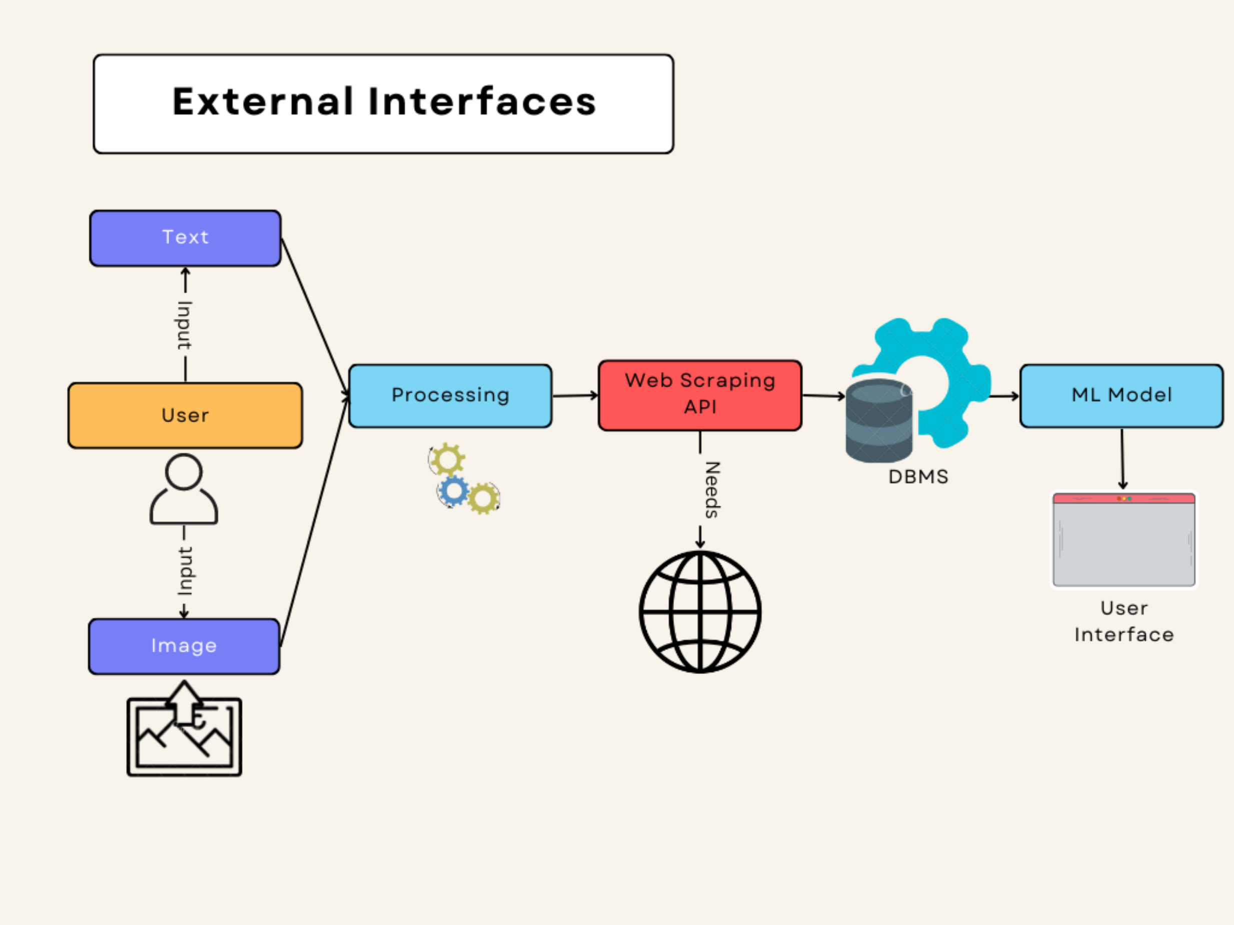


Fig 6.9.3 External Interface Diagram

External Interfaces for the project are as follows:

- Web scraping interfaces: The solution would need to interface with relevant websites, such as e-commerce sites and online marketplaces, to gather data on the items being evaluated. This would require web scraping interfaces or APIs (Application Programming Interfaces) to extract data from these external websites. This could include sending HTTP requests, handling cookies, handling responses, parsing data, and parsing HTML content.
- Database: The scraped data may be stored in a database for further analysis and processing. This could involve interfacing with a relational or NoSQL database management system to store, retrieve, update, and query data.

- File System: The scraped data may be stored in files, such as CSV or JSON, for further analysis or sharing. This could involve reading from and writing to files in the file system.
- APIs: The project may utilize APIs (Application Programming Interfaces) to interact with external services or platforms for data retrieval, authentication, or other functionalities. This could involve sending API requests, handling responses, and parsing data in a standardized format such as JSON or XML.
- Data cleaning and transformation interfaces: The automated data preprocessing module would require interfaces to clean and transform the data gathered from external sources. This could involve interfaces for removing duplicates, correcting errors, standardizing data formats, and other data cleaning tasks.
- Data analysis interfaces: The solution would require interfaces for performing automated data analysis techniques, such as machine learning or statistical modeling, to estimate the value of the items. These interfaces could include APIs or libraries for implementing the chosen algorithms and analyzing the cleaned and transformed data.
- Reporting and visualization interfaces: The results of the data analysis would need to be presented to the user in a clear and understandable format, such as reports or dashboards. This would require interfaces for generating and presenting reports or visualizations of the estimated item values and related information.
- User interfaces: The solution may require user interfaces, such as web or mobile interfaces, to interact with users and gather input or display results. These interfaces would need to be designed to be user-friendly and provide a seamless experience for users interacting with the system.
- External data sources: The solution would need interfaces to gather data from external sources, such as e-commerce sites, online marketplaces, or other relevant websites, which would require data retrieval mechanisms or APIs to access and extract data from these sources.

6.11 Packaging and Component Diagram

6.11.1 Packaging Diagram

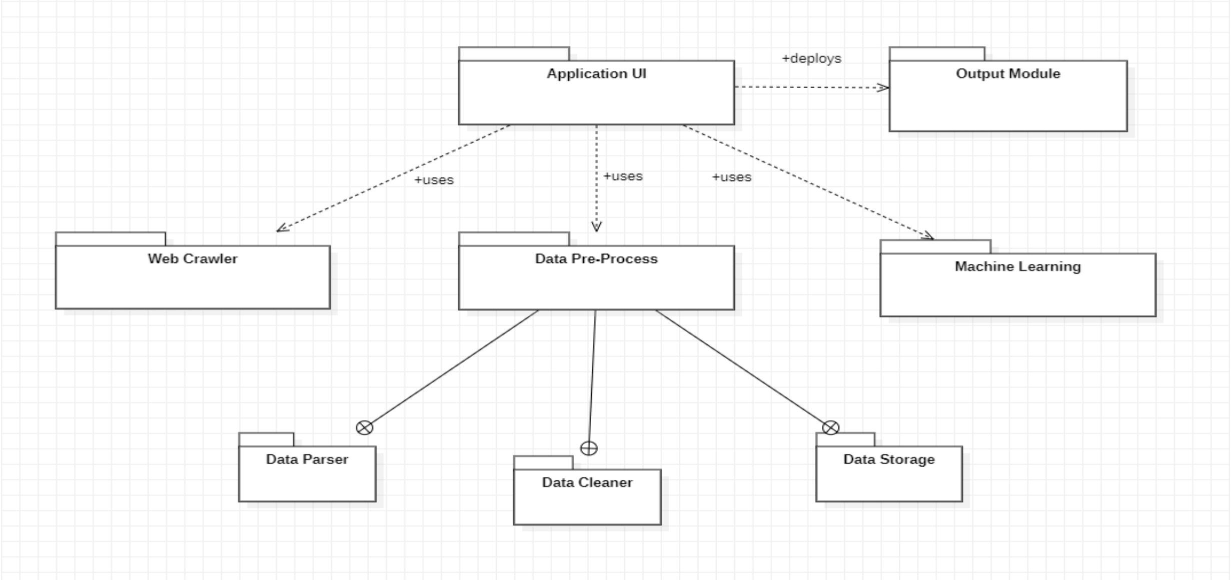


Fig 6.11.1 Packaging Diagram

By displaying the connections between packages, the provided package diagram illustrates the structure of a software system. A software system's related components, including classes, interfaces, and other packages, are grouped together in containers called packages. The primary packages that encompass all system functionalities are visible, as are the connections between the packages. The project is guaranteed to be modular and well-organized per the provided diagram.

6.11.2 Component Diagram

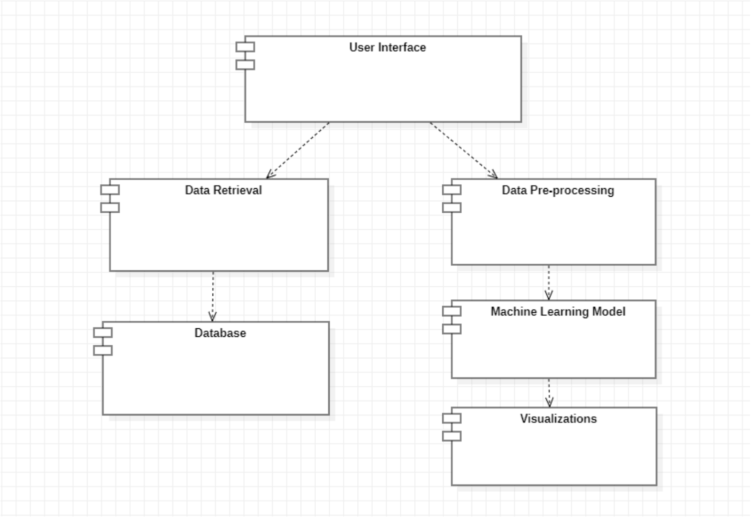


Fig 6.11.2 Component Diagram

Since the application is going to be completely run on the user's PC, there is no requirement of a deployment diagram. In the future when the model can be moved to a client-service architecture, a deployment diagram would be suitable to explain the architecture.

6.11.3 Help

The online platform offers a range of features to assist users in navigating through the application. The features include:

- A User Manual, which offers a detailed, step-by-step guide on how to use the application and its various features.
- Customizable settings that allow users to adjust the difficulty levels of the exercises according to their preferences.
- Support services, including an email address, are available to address any concerns or questions that users may have.
- Frequently Asked Questions (FAQ) section, which provides a set of commonly asked questions and their corresponding answers. This section is easily accessible to users to help clarify any doubts or discrepancies they may encounter while using the application.

6.12 Design Details

6.12.1 Novelty

This web scraping project lies in the extraction of pricing information for used items from multiple websites, utilizing a chosen programming language and web scraping library to automate the data collection process. The project involves navigating website structures, handling potential changes, and processing and analyzing the extracted data for insights, providing a unique approach to gathering market information on used items in a convenient and efficient manner.

6.12.2 Innovativeness

The innovativeness of this project lies in leveraging web scraping techniques to collect real-time pricing information for used items from various online sources, providing a comprehensive and up-to-date view of the market. By automating data collection and analysis, the project streamlines the process of gathering pricing data, potentially enabling quicker and more accurate insights for buyers, sellers, and market researchers. The project demonstrates an innovative approach to utilizing web scraping for market research and data-driven decision making in the context of used item pricing.

6.12.3 Interoperability

The interoperability of this project stems from its flexibility to work with different websites and adapt to changes in website structures. The choice of programming language and web scraping library allows for compatibility with various websites and platforms. The extracted data can be processed and analyzed using common data analysis tools, such as Excel, Python, or R, providing interoperability with existing data workflows. The project's design allows for seamless integration with other data sources and tools for further analysis or visualization, enhancing its interoperability and versatility for different use cases.

6.12.4 Performance

The performance of this project depends on several factors, including the efficiency of the web scraping code, the responsiveness and reliability of the websites being scraped, and the processing speed of data analysis tasks. Optimized web scraping code that minimizes redundant requests and handles exceptions can improve performance. The reliability and speed of the websites being scraped can impact the timeliness and accuracy of the data collected. Efficient data processing and analysis techniques can also enhance performance. Regular monitoring and updates to the code and data sources can help maintain optimal performance throughout the project's lifecycle.

6.12.5 Security

The security of this project involves ensuring that the web scraping activities are conducted in compliance with the ethical and legal guidelines of the websites being scraped. Respecting the terms of use, privacy policies, and robots.txt files of the websites is essential to maintain security. Care should be taken to avoid unauthorized access or data breaches, and sensitive or personal information should not be collected without proper consent. Regular updates and security measures, such as using HTTPS connections and securing login credentials, can also help protect the project from security risks. Adhering to best practices for web scraping and data handling, and being mindful of potential security concerns, are crucial for maintaining the security of the project.

6.12.6 Reliability

The reliability of this project depends on the stability and consistency of the websites being scraped, as well as the robustness of the web scraping code. Thoroughly testing and debugging the code, handling exceptions, and accounting for potential changes in website structures can improve reliability. Regular monitoring and updates to the project, along with proper error handling and data validation, can ensure the accuracy and integrity of the collected data. Adhering to best practices in web scraping and data processing, and ensuring the project is designed to handle potential challenges, are essential for maintaining the reliability of the project.

6.12.7 Maintainability

The maintainability of this project is facilitated by using clean and well-organized code, along with proper documentation and comments to aid in understanding and updating the code in the future. Modular and reusable code design allows for easy modifications and updates as needed. Following coding standards and best practices, along with version control, can streamline maintenance efforts. Regularly reviewing and updating the project's dependencies, libraries, and data sources can also ensure its maintainability. Proper documentation of data sources, data processing steps, and any changes made to the code or data can facilitate future maintenance and updates, making the project more maintainable in the long run.

6.12.8 Portability

The portability of this project is supported by using programming languages and web scraping libraries that are widely available and compatible with different operating systems and environments. Choosing portable file formats for storing the extracted data, such as CSV or JSON, ensures that the data can be easily transported and used in various tools and platforms. Avoiding platform-specific dependencies and using cross-platform libraries enhances portability. Additionally, documenting the setup and installation process, along with any required configurations, facilitates portability across different systems. Ensuring that the project can be easily replicated and executed in different environments promotes its portability and accessibility for different users and use cases.

6.12.9 Legacy to modernization

The legacy of modernization in this project lies in its adoption of contemporary web scraping techniques, programming languages, and libraries to gather and analyze pricing data for used items. By utilizing cutting-edge technologies and best practices, the project establishes a foundation for future enhancements and updates to keep up with evolving web technologies and data analysis methods. The project's modernization approach ensures its relevance and effectiveness in the long term, setting the stage for potential extensions, improvements, and further advancements in the field of web scraping and market research.

6.12.10 Reusability

The reusability of this project is facilitated by its modular and well-organized code design, which allows for easy extraction of pricing data from multiple websites with minimal modifications. The project's flexibility to adapt to changes in website structures and its compatibility with different platforms and data analysis tools enhances its reusability. The use of reusable code components and libraries can also facilitate future applications and extensions of the project for different use cases or markets. Proper documentation and comments within the code, along with comprehensive documentation of data sources and processing steps, enable the project to be easily understood and replicated, promoting its reusability for other similar projects or research endeavors.

6.12.11 Application compatibility

The application compatibility of this project is facilitated by its ability to integrate with different web applications and data analysis tools. The project's design allows for easy extraction of pricing data from various websites, making it compatible with different online marketplaces or e-commerce platforms. The extracted data can be processed and analyzed using popular data analysis tools such as Excel, Python, or R, making it compatible with different data workflows and analysis pipelines. The use of portable file formats for storing the extracted data, such as CSV or JSON, enables seamless integration with other data sources or applications. This compatibility makes the project versatile and adaptable for different applications and use cases, enhancing its overall usability and value.

6.12.12 Resource utilization, Etc.

The resource utilization of this project is optimized by employing efficient web scraping techniques, minimizing unnecessary requests to websites, and avoiding overloading their servers. The project utilizes appropriate libraries or tools to efficiently extract and process pricing data, minimizing resource consumption. Careful consideration is given to memory usage, CPU utilization, and network bandwidth to ensure efficient resource utilization. Proper error handling and exception management prevent resource waste and enhance the project's performance. Regular monitoring and optimization of the project's resource utilization help ensure efficient and effective operation while minimizing the impact on the websites being scraped or the system running the project.

Chapter 7

Low Level Design Document

7.1 Introduction

A Low-Level Design Document serves as a detailed guide for the implementation of a proposed solution or system. It meticulously outlines the structure, functionalities, and interactions of individual modules. This document is crucial for translating the high-level design into actionable development steps.

7.2 Overview

In this section, we present a high-level summary of the low-level design, offering a bird's-eye view of how the proposed solution will be realized. It highlights the key components, functionalities, and the intricate interplay between modules, setting the stage for a more in-depth exploration.

7.3 Purpose

In this section, we present a high-level summary of the low-level design, offering a bird's-eye view of how the proposed solution will be realized. It highlights the key components, functionalities, and the intricate interplay between modules, setting the stage for a more in-depth exploration.

7.4 Scope

Within this section, we define the boundaries and extent of the Low-Level Design Document. It explicitly outlines what the document will cover and what it won't. Understanding the scope ensures that the document remains focused, addressing critical aspects of the implementation process without unnecessary diversions.

7.5 Design Constraints, Assumptions, and Dependencies

Constraints made:

- Usability: Working for a solution that enables the user to connect to the application on different platforms
- Maintainability: The app working is stable and is always available for the users

Assumptions made:

- The websites we are scraping have consistent and reliable data.
- The data extracted through web scraping is legal and ethical to use.
- The features we are extracting from the data are relevant to the problem you are trying to solve.
- The predictions generated by our application are accurate and reliable.

Dependencies made:

- The quality of the dataset and the accuracy of the predictions depend on the data extracted through web scraping.
  - i.e. the newer the data is the more time the prediction of the model will take.
- The speed and reliability of the web scraping process depend on the stability and speed of the internet connection and the availability of the websites.
  - i.e. The difference of speeds between operating systems was significantly different being faster on Windows compared to Mac.

7.6 Design Description

7.6.1 Module 1: Data Collection

7.6.1.1 Class Description

Data required to be preprocessed and analyzed is collected in real time using web scraping from popular web sites where old/used items are sold in this module. The web scraper code is built using Selenium, a popular tool that supports browser automation.

7.6.1.2 Use Case Diagram

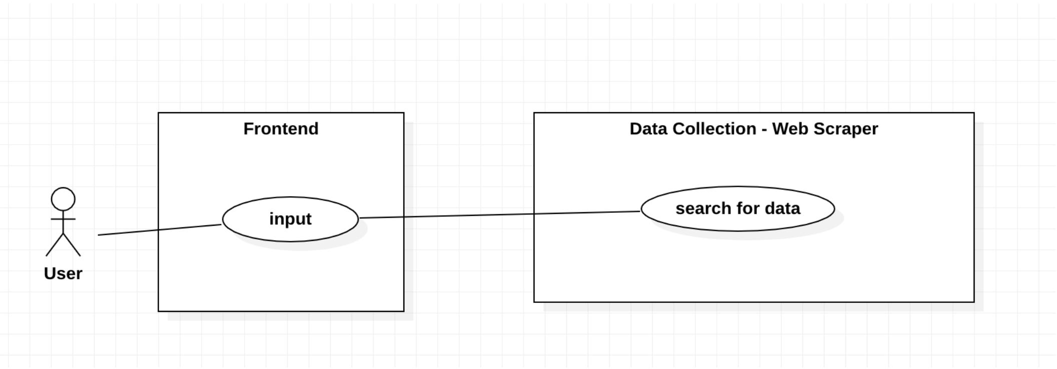


Fig 7.6.1.2 Use Case Diagram for Web-Scraper

7.6.1.3 Class Diagram

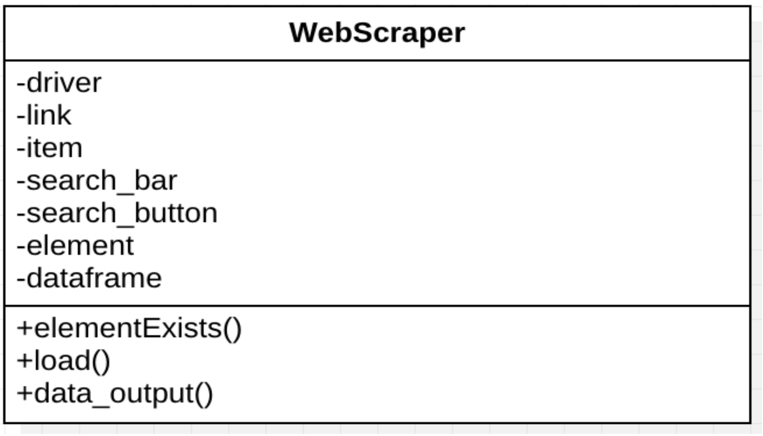


Fig 7.6.1.3 Web Scraper Class

7.6.1.3.1 Class 1: Web Scraper

7.6.1.3.2 Description:

This class is used to automate the process of looking through an entire web site for different sites that can be represented as different instances of the class. It looks through the site for relevant information using the html elements that have been extracted manually and produces a data frame with relevant data that is required for data analysis.

7.6.1.3.3 Data Members

Table 7.1 Data Members of Web Scraper Class

Data Type	Data Name	Access Modifiers	Initial Value	Description
WebDriver	driver	private	null	Represents the WebDriver for interacting with a web browser.
String	link	private	null	Stores a string representing a link or URL.
String	item	private	null	stores the name of the item to be searched for
String	search_bar	private	User input	stores a string representing a search bar element.
String	search_button	private	null	Stores a string representing a search button element.
String	element	private	null	Stores a string related to an HTML element.

Dataframe	df	private	null	Represents a Pandas DataFrame for storing relevant data.

7.6.2 Module 2: Data Preparation for Web Scraping

7.6.2.1 Class Diagram

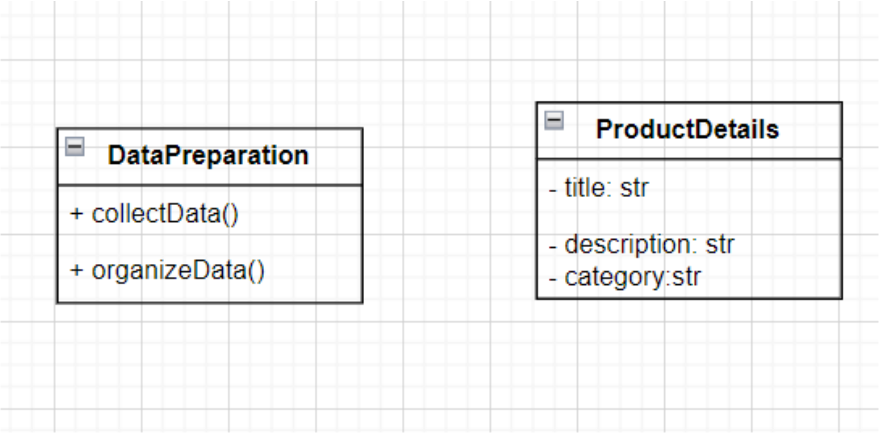


Fig 7.6.2.1 Class Diagram for Data Preparation

7.6.2.1.1 Class 1: Data Preparation

7.6.2.1.2 Class Description:

This module is responsible for preparing the data that will serve as input to the web scraper. It may involve collecting and structuring product descriptions and details from various sources, such as user input or external data feeds. It includes classes for managing and organizing the input data.

7.6.2.2 Data members

Table 7.2 Data Members of Data Preparation Class

Data Type	Data Name	Access Modifiers	Initial Value	Description
str	product	private	none	Name of the product

str	description	private	none	Description of the product

7.6.2.2.1 Class 2: ProductDetails

7.6.2.2.2 Class Description

This class represents a structured data container for holding essential information about a product. It is designed to store specific attributes of a product, including its title, description, category, brand, model, condition, and any additional relevant details. This class provides a consistent and organized structure for product data, making it easier to manage and work with product information within the system. It plays a crucial role in maintaining data integrity and facilitating the exchange of product details throughout the application.

7.6.2.1.6 Data members

Table 7.3 Data Members of Product Details Class

Data Type	Data Name	Access Modifiers	Initial Value	Description
str	name	private	null	Name of the product
str	desc	private	null	Description of the product/item
str	category	private	null	Which category of product(cars, mobiles, laptop, furniture)
str	brand	private	null	Which brand the product is off
str	model	private	null	Model of the product

str	condition	private	null	Current condition of the product. (damages, perfect condition)

7.6.3 Module 3: Preprocessing

7.6.3.1 Description

This module plays a critical role in the data pipeline by processing, preparing, and organizing data for the subsequent machine learning module. It includes various tasks such as data cleaning, transformation, and feature engineering. Once the data is ready, this module facilitates the transfer of the prepared dataset to the machine learning component.

7.6.3.2 Class Diagram

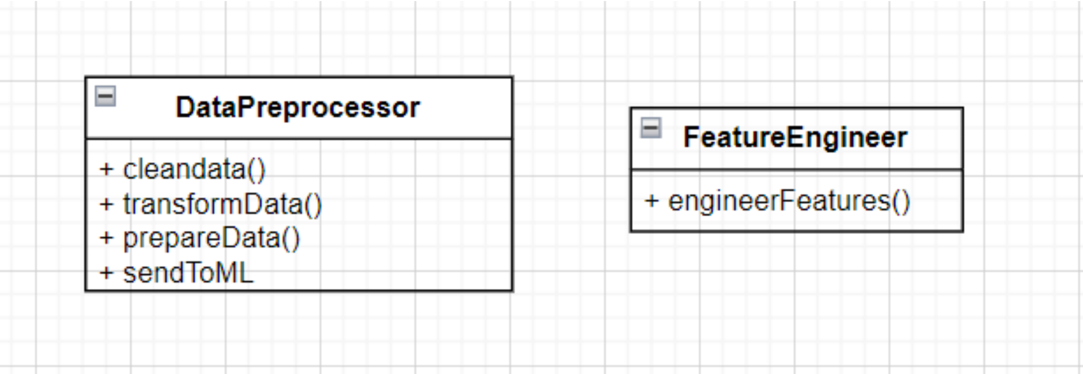


Fig 7.6.3.2 Class Diagram for Data Preprocessor

7.6.3.2.1 Class 1: Data Preprocessor

7.6.3.2.2 Class Description

This class is responsible for data preprocessing and preparation. It performs essential tasks like data cleaning, transformation, and dataset preparation. This class ensures that the data is in an optimal state for consumption by the machine learning model. It plays a pivotal role in maintaining data quality and consistency.

7.6.3.2.3 Data Members

Table 7.4 Data Members of Data Preprocessor Class

Data Type	Data Name	Access Modifiers	Initial Value	Description
dataset	rawData	private	none	Stores the raw dataset before processing
dataset	cleaneddata	private	none	Holds the cleaned dataset
dataset	transformedData	private	none	Stores the transformed dataset
dataset	preparedData	private	none	holds the final prepared dataset

7.6.3.2.4 Class 2: Feature Engineer Class

7.6.3.2.5 Class Description

This class enhances dataset features to improve machine learning model performance, revealing patterns and relationships in the data. It's a crucial step in data preparation for data-driven decision-making.

7.6.3.2.6 Data Members

Table 7.5 Data Members of Feature Engineer Class

Data Type	Data Name	Access Modifiers	Initial Value	Description
dataset	engineerFeatures	private	none	Holds the dataset after feature engineering.

7.6.4 Module 4: Machine Learning Model & Visualization

7.6.4.1 Description

The ML Model and Visualization Module is responsible for handling the machine learning model's prediction process and presenting the results in a visual format. It takes preprocessed data as input, performs predictions, and displays relevant information to users.

7.6.4.2 Class Diagram

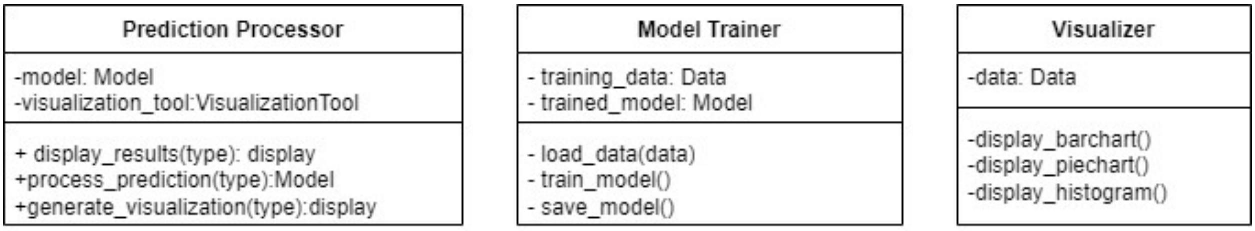


Fig 7.6.4.2 Class Diagram from ML Algorithm

7.6.4.2.1 Class 1: Prediction Processor

7.6.4.2.2 Class Description

The PredictionProcessor class is responsible for handling the processing of predictions generated by the trained machine learning model. It encapsulates the logic for taking input data, making predictions, generating visualizations based on those predictions, and displaying the processed results to the end-users. The class holds instances of the machine learning model (model) and a visualization tool (visualization_tool). It provides methods for processing predictions, generating visualizations, and displaying the results.

7.6.4.2.3 Data Members

Table 7.6 Data Members of Prediction Processor Class

Data Type	Data Name	Access Modifiers	Initial Value	Description
Model	**model**	**Private**	**None**	**Instance of a machine**

				learning model for making prediction
visualization_tool	**visualization_tool**	**Private**	**None**	**Instance of a visualization tool for generating visualizations**

7.6.4.2.4 Class 2: Model Trainer

7.6.4.2.5 Class Description

The ModelTrainer class manages the training process of the machine learning model. It is responsible for loading training data, training the model, and saving the trained model for future use. The class encapsulates the training data (training_data) and the trained machine learning model (trained_model). It provides methods to load data, train the model, and save the trained model.

7.6.4.2.6 Data Members

Table 7.7 Data Members of Model Trainer Class

Data Type	Data Name	Access Modifiers	Initial Value	Description
Data	**training_data**	**Private**	**None**	**Dataset used for training the machine learning model.**
Model	**trained_model**	**Private**	**None**	**The machine learning model that has been**

Data Type	Data Name	Access Modifiers	Initial Value	Description
				trained

7.6.5 Module 5: User Input and Authentication

7.6.5.1 Description

This module allows users to interact with the application through a web interface created using Streamlit. Users can log in with their credentials, sign up if they are new users, and once authenticated, they can access the main application features, including creating product posts.

7.6.5.2 Use Case Diagram

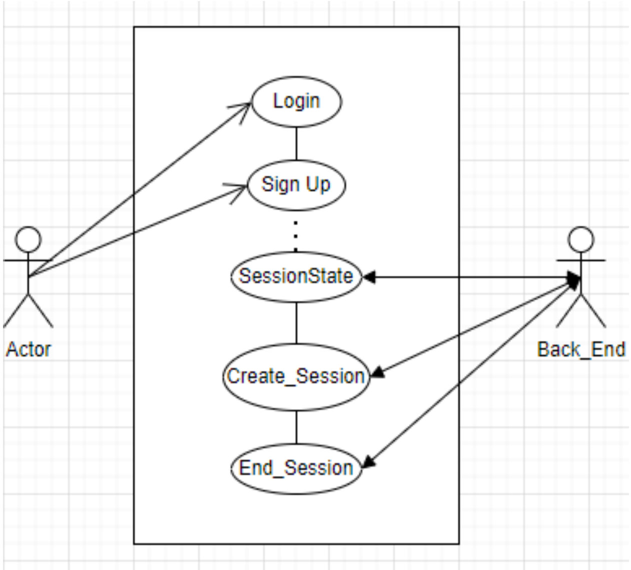


Fig 7.6.5.2 Use Case Diagram for User Input and Authentication

7.6.5.3 Class Diagram

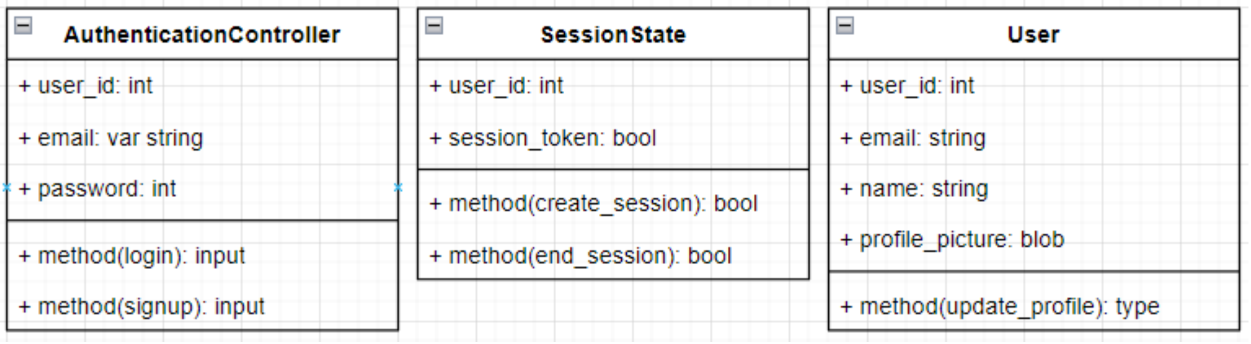


Fig 7.6.5.3 Class Diagram for Module User Input

7.6.5.3.1 Class 1: Authentication Controller

7.6.5.3.2 Class Description

This class is responsible for handling user authentication, including login and sign-up processes.

7.6.5.3.3 Data Members

Table 7.8 Data Members of Authentication Controller Class

Data Type	Data Name	Access Modifiers	Initial Value	Description
Data	**user_id**	**Private**	**None**	**Stores the unique identifier of the user**
Data	**email**	**Private**	**None**	**Stores the user's email address**
Data	**password**	**Private**	**None**	**Stores the user's password (hashed for security)**

7.6.5.3.4 Class 2: Session State

7.6.5.3.5 Class Description

This class is responsible for managing user sessions after successful login or sign-up.

7.6.5.3.6 Data Members

Table 7.9 Data Members of Session State Class

Data Type	Data Name	Access Modifiers	Initial Value	Description
Data	**user_id**	**Private**	**None**	**Stores the unique identifier of the user**
Data	**session_token**	**Private**	**None**	**Token generated for the user's session**

7.6.5.3.7 Class 3: User

7.6.5.3.8 Class Description

This class represents the user entity in the system, containing user-specific data.

7.6.5.3.9 Data Members:

Table 7.10 Data Members of User Class

Data Type	Data Name	Access Modifiers	Initial Value	Description
Data	**user_id**	**Private**	**None**	**Stores the unique identifier of the user**
Data	**email**	**Private**	**None**	**Stores the user's**

				email address
Data	**name**	**Public**	**None**	**Stores the users name**
Blob	**profile_picture**	**Public**	**None**	**Stores the image of the user**

7.6.6 Module 6: Handling Post Information

The Handling Post Information module manages the data related to the products being posted by users.

7.6.6.1 Description

This module is responsible for collecting, storing, and presenting information about the products users want to post on the platform. Users can provide details about the product, such as its type (e.g., mobile, laptop, car, or other items), specifications, images, and pricing. The module ensures that this information is properly stored and made available for other users to browse.

7.6.6.2 Use Case Diagram

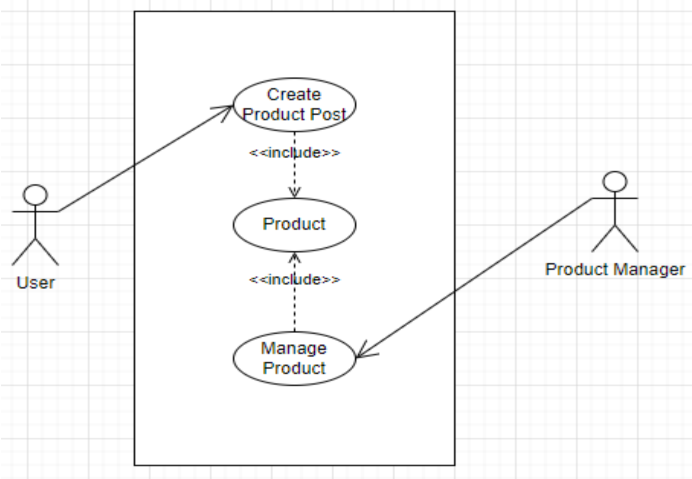


Fig 7.6.6.2 Use Case Diagram for Handling Post Information

7.6.6.3 Class Diagram

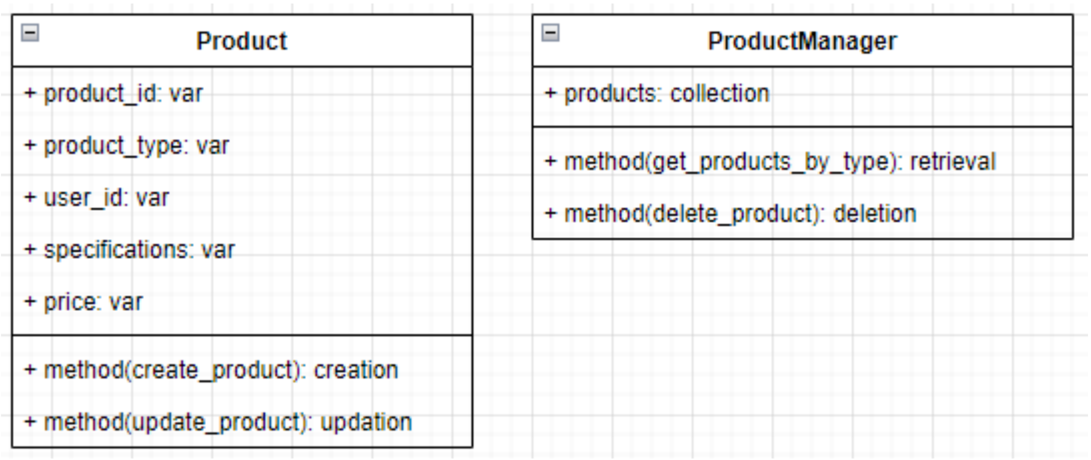

Product
+ product_id: var
+ product_type: var
+ user_id: var
+ specifications: var
+ price: var
+ method(create_product): creation
+ method(update_product): updation
ProductManager
+ products: collection
+ method(get_products_by_type): retrieval
+ method(delete_product): deletion

Fig 7.6.6.3 Class Diagram for Handling Post Information

7.6.6.3.1 Class 1: Product

7.6.6.3.2 Class Description

This class represents individual products that users post on the platform.

7.6.6.3.3 Data Members:

Table 7.11 Data Members of Product Class

Data Type	Data Name	Access Modifiers	Initial Value	Description
Data	**product_id**	**Private**	**None**	**Unique Identifier for the product**
Data	**product_type**	**Private**	**None**	**Indicates the type of product**
Data	**user_id**	**Private**	**None**	**Stores the user’s unique identifier**
Data	**specifications**	**Private**	**None**	**Contains details about the**

				product's specifications
Blob	**images**	**Private**	**None**	**Stores the image of the product**
Data	**price**	**Private**	**None**	**Indicates the price of the product**

7.6.6.3.4 Class 2: Product Manager

7.6.6.3.5 Class Description

This class is responsible for managing product posts and their lifecycle.

7.6.6.3.6 Data Members

Table 7.12 Data Members of Product Manager Class

Data Type	Data Name	Access Modifiers	Initial Value	Description
Collection	**products**	**Private**	**None**	**Stores all the products posted by the user**

7.6.7 Module 7: Output and Presentation

The Output and Presentation module handles the display of the user's session state and the product posts created by the user, as well as retrieving this data from the SQL database.

7.6.7.1 Description

This module focuses on presenting a user's session state, such as the current user, and showcasing the product posts created by that user. It retrieves data from the SQL database, where the previous modules have stored the user and product information, and formats it for presentation.

7.6.7.2 Class Diagram

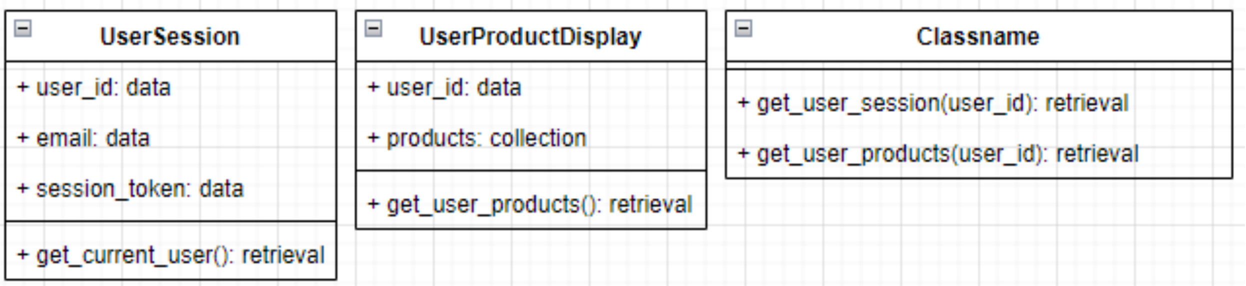

UserSession
+ user_id: data
+ email: data
+ session_token: data
+ get_current_user(): retrieval
UserProductDisplay
+ user_id: data
+ products: collection
+ get_user_products(): retrieval
Classname
+ get_user_session(user_id): retrieval
+ get_user_products(user_id): retrieval

Fig 7.6.7.2 Class Diagram for Output and Presentation

7.6.7.2.1 Class 1: User Session

This class represents the current user's session state.

7.6.7.2.2 Data Members

Table 7.13 Data Members of User Session Class

Data Type	Data Name	Access Modifiers	Initial Value	Description
Data	**user_id**	**Private**	**None**	**Stores the unique identifier of the current user**
Data	**email**	**Private**	**None**	**Stores the user’s email address**
Data	**session_token**	**Private**	**None**	**Stores the user’s session token**

7.6.7.2.3 Class 2: User Product Display

This class is responsible for displaying the product posts created by the current user.

7.6.7.2.4 Data Members

Table 7.14 Data Members of User Product Display Class

Data Type	Data Name	Access Modifiers	Initial Value	Description
Data	**user_id**	**Private**	**None**	**Stores the unique identifier of the user**
Collection	**products**	**Private**	**None**	**Stores all the products posted by the user.**

7.7 Sequence Diagram

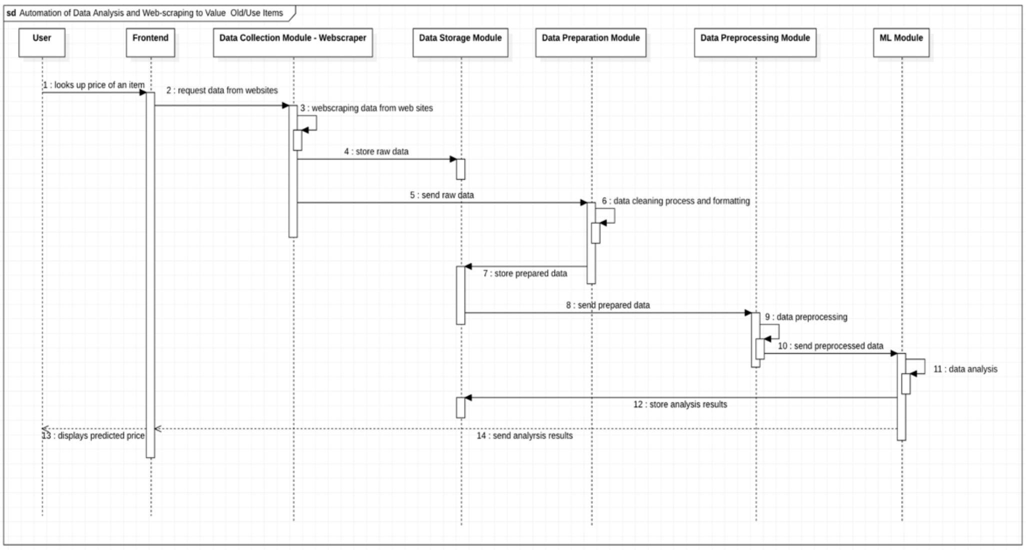

sd Automation of Data Analysis and Web-scraping to Value Old/Use Items
User
Frontend
Data Collection Module - Webscraper
Data Storage Module
Data Preparation Module
Data Preprocessing Module
ML Module
1 : looks up price of an item
2 : request data from websites
3 : webscraping data from web sites
4 : store raw data
5 : send raw data
6 : data cleaning process and formatting
7 : store prepared data
8 : send prepared data
9 : data preprocessing
10 : send preprocessed data
11 : data analysis
12 : store analysis results
13 : displays predicted price
14 : send analyrsis results

Fig 7.7 Sequence Diagram

7.8 Packaging and Deployment Diagrams

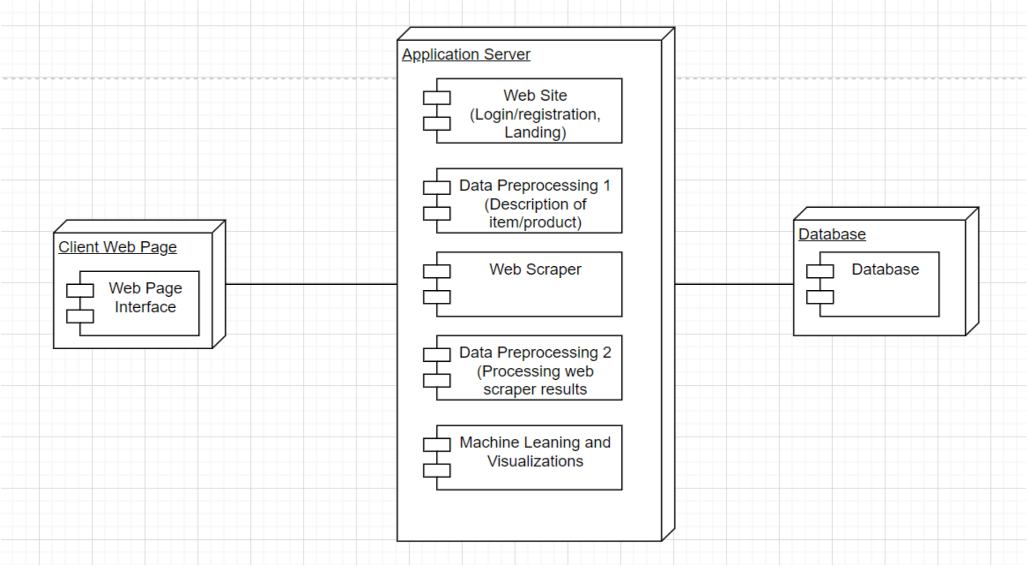

Application Server
Web Site (Login/registration, Landing)
Data Preprocessing 1 (Description of item/product)
Web Scraper
Data Preprocessing 2 (Processing web scraper results
Machine Leaning and Visualizations
Client Web Page
Web Page Interface
Database
Database

Fig 7.8 Packaging and Deployment Diagrams

Chapter 8

Proposed Methodology

8.1 PROPOSED SYSTEM

The proposed system aims to streamline and automate the valuation process for old or used items through an integrated approach, encompassing web scraping, automated data preprocessing, machine learning, and data analysis. This comprehensive solution is designed to provide rapid and accurate item valuation, benefiting individuals, organizations with large inventories, sellers, donators, insurance companies, pawn shops, and various businesses seeking precise estimates for used items.

Workflow Overview:

- Data Gathering:
  - Utilize web scraping tools to extract relevant information on the item from diverse sources, including e-commerce sites and online marketplaces.
  - Depending on the type of product the user selects we are changing our approach by selecting relevant features, and by optimizing our web scraping code to make it choose the closest items listed online that resemble the user's input.
- Automated Data Preprocessing:
  - Apply an automated preprocessing module to cleanse and transform the collected data.
  - Tasks include deduplication, error correction, and standardization of data formats.
  - Clean the data and remove any additional information that might affect the accuracy of the final price.
- Model Selection and Tuning:
  - Leverage an automated algorithm selection and tuning module to choose the most suitable machine learning for item valuation.
  - Since the number of rows returned from web-scraping is relatively low, we need to optimize the number of estimators used and ensure relevant dependent features are given more weight.

- Automated Data Analysis:
    - Employ machine learning to estimate the item's value.
    - Fine-tune the selected algorithm using an automated tuning module, considering factors such as market trends, seasonal variations, and geographic influences.
- Results Presentation:
    - Present the results of the data analysis in a clear and comprehensible format, such as a detailed report or user-friendly dashboard.
    - Include information on the estimated item value, the features influencing the valuation, and any other pertinent details to empower users in making informed decisions.
- Key Advantages:
    - Efficiency: Automation reduces manual effort, ensuring a swift and efficient valuation process.
    - Accuracy: Leveraging machine learning models enhances the precision of item valuations.
    - Adaptability: Modules are trained on extensive historical datasets, enabling adaptation to various item types, and changing market dynamics.
    - User-Friendly Reporting: Results are presented in a format that is easily understandable, aiding users in decision-making.
- Potential Users:
    - Individuals managing large inventories of used items.
    - Organizations involved in selling, donating, or managing used items.
    - Insurance companies require accurate valuations.
    - Pawn shops seeking efficient item appraisal processes.
    - Businesses looking to estimate the value of used items.

This proposed system integrates cutting-edge technologies to simplify the valuation process, offering a versatile and powerful tool for a diverse range of users and scenarios.

Chapter 9

Implementation and Pseudocode

9.1 Implementation Choices

9.1.1 Programming Language Selection

The programming language chosen for this project would be python due to the following reasons:

Options of using third party modules

Library support

User friendly data structures

Speed and Productivity

Open-source community support

9.1.2 Platform Selection

The Automation of Data Analysis and Web-Scraping to Value Old/Used Items product can operate on a variety of hardware platforms, operating systems, and software components. Here are some of the key details:

- Hardware platform: The product is designed to operate on standard computing hardware, including desktops, laptops, and servers. There are no specific hardware requirements, but a higher-end computer may perform better when running more intensive web scraping or data analysis tasks.
- Operating system: The product is compatible with multiple operating systems, including Windows, Mac, and Linux. It can run on Windows 7, 8, 10, and Windows Server, macOS 10.12 and later, and Linux distributions such as Ubuntu, Debian, and Fedora.
- Software components: The product relies on several software components, including web browsers such as Google Chrome, Mozilla Firefox, or Microsoft Edge, which are used for

web scraping. It also uses programming languages such as Python and R, which are used for data analysis, along with various libraries and packages, such as BeautifulSoup, Pandas, and Scikit-learn.

9.1.3 Libraries Required

- Selenium: Selenium is a free and open-source web crawling framework written in Python. It is a fast, high-level framework used to crawl websites and extract structured data from their pages.
- Scikit-learn: It can be used to import existing models that can be trained on our dataset, that will later be used to predict prices for each product.
- Numpy & Pandas: These python libraries can be used for our Data Processing and Analysis, have multiple functions that enable data preprocessing and analysis without hard coding.
- BeautifulSoup: Simple python library to enable web-scraping to visit a website and crawl to retrieve data.
- Flask: A back-end framework for python that enables developers to combine html templates to a database and connect all modules of a website application together.
- Celery: Library to enable multithreading for Flask to enable users to scrape multiple products at the same time.
- MySQL-Connector: Converting the user-input to a database schema to store the data which can later be converted to a csv file.
- Pseudocode of some relevant functions:

```
function getUserProductDetails():
        productDetails = {}  // Initialize an empty dictionary to store the product details

        displayMessage("Please provide the following product details:")

        productDetails["title"] = promptUserForInput("Product Title: ")
        productDetails["description"] = promptUserForInput("Product Description: ")
        productDetails["category"] = promptUserForInput("Product Category: ")

        // Collect additional details as needed
        productDetails["brand"] = promptUserForInput("Brand (if applicable): ")
        productDetails["model"] = promptUserForInput("Model (if applicable): ")
```

```
    productDetails["condition"] = promptUserForInput("Condition (e.g., new, used, refurbished): ")

    // You can continue to prompt for other details as required

    return productDetails  // Return the dictionary containing the product details
end function

function organizeProductData(rawProductData):
    // Initialize an empty dictionary to store organized data
    organizedData = {}

    // Extract and store product details
    organizedData["title"] = rawProductData["title"]
    organizedData["description"] = rawProductData["description"]
    organizedData["category"] = rawProductData["category"]

    // Extract and store additional details
    organizedData["brand"] = rawProductData["brand"]
    organizedData["model"] = rawProductData["model"]
    organizedData["condition"] = rawProductData["condition"]

    // You can continue to extract and organize other details as needed

    return organizedData
end function

    class ProductDetails:
        title: string
        description: string
        category: string
        brand: string
        model: string
        condition: string

        constructor(title, description, category, brand, model, condition):
            this.title = title
            this.description = description
            this.category = category
            this.brand = brand
            this.model = model
```

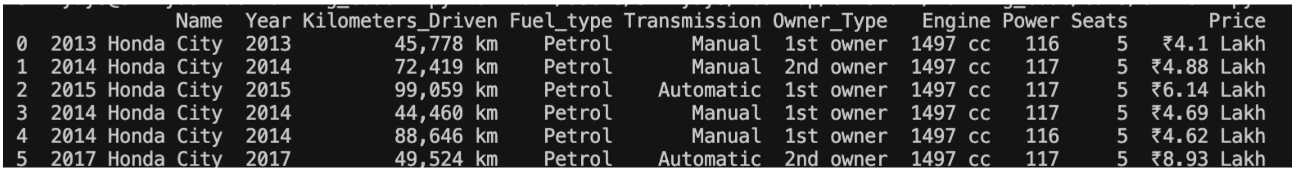
```
                this.condition = condition

            getAttribute(attribute):
                return this[attribute]

            setAttribute(attribute, value):
                this[attribute] = value
        end class

function login(email, password):
    # Query the database to retrieve user information by email
    user = retrieve_user_by_email(email)

    if user is not None and user.password == hash_password(password):
        # Create a user session for the authenticated user
        session_token = create_session(user.user_id)
        return "Login successful. Session token: " + session_token
    else:
        return "Login failed. Invalid credentials."

function hash_password(password):
    # Implement a secure password hashing algorithm (e.g., bcrypt)
    # Return the hashed password for comparison

function retrieve_user_by_email(email):
    # Query the database to retrieve user information by email
    # Return the user object if found, or None if not found

function create_session(user_id):
    # Generate a unique session token for the user
    # Store the session token in the database for future reference
    # Return the session token
```

Chapter 10

Results and discussion

10.1 Data Collection Results

The use of Selenium has been employed for web scraping data as per user’s requirements, in real time. It helps handling dynamic content and navigating through complex web pages by interacting with JavaScript.



```
             Name  Year Kilometers_Driven Fuel_type Transmission Owner_Type   Engine Power Seats       Price
0  2013 Honda City  2013         45,778 km    Petrol       Manual  1st owner  1497 cc   116     5   ₹4.1 Lakh
1  2014 Honda City  2014         72,419 km    Petrol       Manual  2nd owner  1497 cc   117     5  ₹4.88 Lakh
2  2015 Honda City  2015         99,059 km    Petrol    Automatic  1st owner  1497 cc   117     5  ₹6.14 Lakh
3  2014 Honda City  2014         44,460 km    Petrol       Manual  1st owner  1497 cc   117     5  ₹4.69 Lakh
4  2014 Honda City  2014         88,646 km    Petrol       Manual  1st owner  1497 cc   116     5  ₹4.62 Lakh
5  2017 Honda City  2017         49,524 km    Petrol    Automatic  2nd owner  1497 cc   117     5  ₹8.93 Lakh
```

Fig 10.1 Sample Web-scraped Data of Cars (Honda City)

10.2 Data Visualization

A correlation matrix is obtained from the web scraped data. The following can be inferred from one such correlation matrix (Fig: 10.2).

Table: 10.1 Correlation Matrix Inference

Negative Correlation	Positive Correlation
Price - Kilometers_Driven Car_Model - Kilometers_Driven (relative) Owner_type - Price	Owner_Type - Kilometers Driven Car_Model - Owner_Type (relative) Car_Model - Price (relative)

Negative correlation suggests that the two attributes are inversely proportional to each other and a positive correlation, directly proportionality. However, the relationship between any attribute and Car_model is relative as it depends on the encoding of different car models in a dataset which is automated.

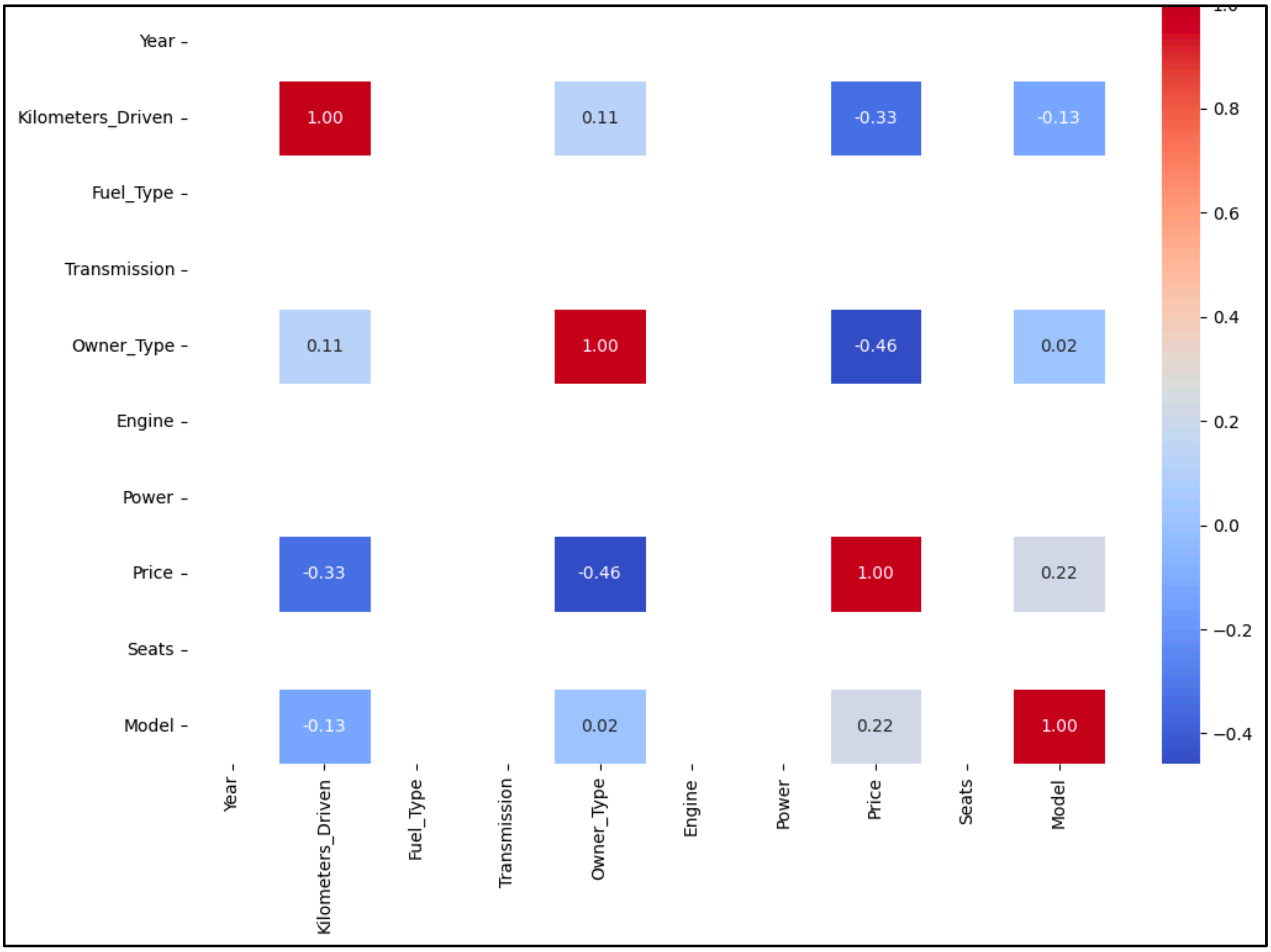


Fig 10.2 Correlation Matrix Heatmap for the Sample Web-scraped Data of Honda City Car

10.3 Model Training

Various regression models have been implemented to predict prices and here are the graphs comparing the Mean Square Error and the R^2 of different models respectively.

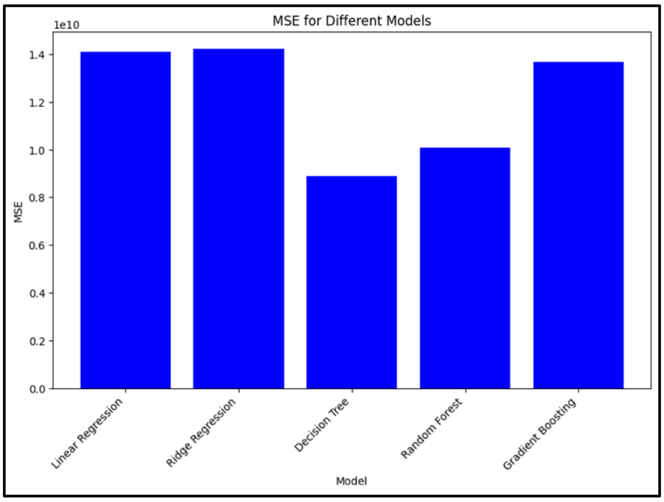


Fig: 10.3 Plot for comparison of MSE

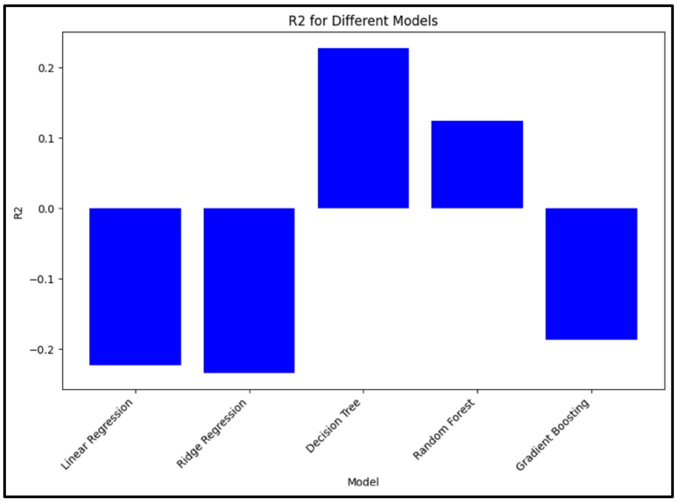


Fig: 10.4 Plot for comparison of R2

The datasets that are obtained through web scraping are highly filtered and hence have very limited rows. This gives us a need to pick a model that can handle the sparsity in data. The negative R2 values indicate that the model's predictions are not better than a basic mean model and the regression analysis is not performing well on the given data. A positive R2 value is noticed for the Decision Tree and Random Forest Regressors, and they have been chosen for further analysis.

Table: 10.2 Sample Input

BRAND	MODEL	YEAR	FUEL TYPE	OWNER TYPE	KM DRIVEN (in km)	TRANSMISSION	MILEAGE	ENGINE(in cc)	POWER(bhp)	NO OF SEATS
Honda	Amaze I-VTEC SX	2016	Petrol	Second	45000	Manual	17	1199	78	5

Table: 10.3 Sample Results

	SAMPLE RUN	MSE (in rupees^2)	R^2	PREDICTED OUTPUT (in Rupees)
DECISION TREE	1	4567500000.0	-0.250	358000.0
	2	4567500000.0	-0.250	358000.0
	3	4567500000.0	-0.250	358000.0
RANDOM FOREST with 100 trees	1	3703983750.0	-0.013	385460.0
	2	3727430300.0	-0.020	391780.0
	3	3773148175.0	-0.032	394190.0
RANDOM FOREST with 30 trees	1	3903037222.22222	-0.068	383500.0
	2	3345164166.66667	0.085	387533.34
	3	4145676666.66667	-0.134	398900.0

Table: 10.4 Comparison of Predicted Values

Attribute changed	Value (in km)	Predicted Value of Decision Tree (in Rupees)	Predicted Value of Random Forest (in Rupees)
KM_driven	25,000	4,05,000.0	4,50,000-4,60,000
	35,000	4,05,000.0	4,25,000-4,45,000
	45,000	4,05,000.0	3,90,000-4,10,000
	55,000	4,05,000.0	3,37,000-3,35,000

We can see that the Decision Tree predicts the same value even if a specific attribute in the user's data changes by a small margin but the Random Forest Regressor accounts for it.

After multiple runs of the same models on datasets for different car models, it has been concluded that the Random Forest model with 30 trees or estimators performs relatively better as it is an ensemble method of multiple decision trees. Our web scraped datasets are highly prone to overfitting which is mitigated by using a lesser number of estimators and therefore we have tuned the number of trees in our Random Forest Regressor to 30. This helps reduce overfitting compared to individual decision trees and increases the model's robustness making it a good choice for small datasets.

Chapter 11

Conclusion and Future work

We have successfully developed a powerful staging that can provide the end price of the product in a matter of few minutes while showcasing impressive results in its current form. When compared to other pricing tools that are designed for categories, it holds up well and sometimes outperforms them. The front end of the project is made to be user-friendly, offering a smooth experience. Looking ahead, we wish to improve the time taken by the entire process and introduce multi-threading into our process. Our goal is to also allow users to find prices for niche products.

Our current workflow enables a user for demonstration search their exact vehicle model and explore the posts of various websites, namely cars24.com and carwale. Based on the continuously changing data from those websites, we were able to successfully pull relevant posts in an average time span of 4-6 minutes. In this duration the program was capable of inputting, retrieving, and computing the processed data for making a reliable prediction.

The future work for the current program, but not limited to, should be simplifying the user-interface enabling various other product inputs, handling multiple searches in parallel for faster data collection. Accessibility for the web-scraper to scrape more than one website simultaneously and representing the users posts and queries as a public community with encapsulated data, created an environment for sharing the status of their searches enabling other users to assess the legitimacy or reliability of the price they have received.

CHAPTER 12

REFERENCES/BIBLIOGRAPHY

[1] Auto-Prep: Efficient and Automated Data Preprocessing Pipeline-Mehwish Bilal, Ghulam Ali, Muhammed Waseem Iqbal, Muhammed Anwar January 2022 https://www.researchgate.net/publication/362718554_Auto-Prep_Efficient_and_Robust_Automated_Data_Preprocessing_Pipeline

[2]LSTM Online Training and Prediction: Non-Stationary Real Time Data Stream Forecasting https://www.researchgate.net/profile/Jack-Press-2/publication/328228359_LSTM_Online_Training_and_Prediction_Non-Stationary_Real_Time_Data_Stream_Forecasting/links/5bbf9b83458515a7a9e29568/LSTM-Online-Training-and-Prediction-Non-Stationary-Real-Time-Data-Stream-Forecasting.pdf

[3]New Evaluation Metric for Demand Response-Driven Real-Time Price Prediction Towards Sustainable Manufacturing https://www.researchgate.net/profile/Lingxiang-Yun/publication/363578087_A_New_Evaluation_Metric_for_Demand_Response-driven_Real-time_Price_Prediction_Towards_Sustainable_Manufacturing/links/63287bb670cc936cd31daa05/A-New-Evaluation-Metric-for-Demand-Response-Driven-Real-Time-Price-Prediction-Towards-Sustainable-Manufacturing.pdf

[4]Automation of Analytical Data Processing-Alexander Serov September 2019 https://www.researchgate.net/publication/335890028_Automation_of_Analytical_Data_Processing

[5] "Monitoring of Stocks using LSTM Model and Prediction of Stock Prices," 2022 International Conference on Edge Computing and Applications (ICECAA), Tamilnadu, India, 2022, pp. 1689-1695, doi: 10.1109/ICECAA55415.2022.9936204.

[6] Ambekar, S. (2023) *Smartphone specifications and prices in India*, *Kaggle*. Available at: https://www.kaggle.com/datasets/shrutiambekar/smartphone-specifications-and-prices-in-india (Accessed: May 1, 2023).

[7] Vaddoriya, M. (2022) *Old car price prediction*, *Kaggle*. Available at: https://www.kaggle.com/datasets/milanvaddoriya/old-car-price-prediction (Accessed: May 1, 2023).

[8] Selenium (2023) *The selenium browser automation project*, *Selenium*. Available at: https://www.selenium.dev/documentation/ (Accessed: May 1, 2023).

[9] Msuch, owenthcarey (2018) *Used_guitar_price_prediction*, *GitHub*. Available at: https://github.com/msuch/used_guitar_price_prediction (Accessed: May 1, 2023).

Appendix A: Definitions, Acronyms and Abbreviations

Web Scraper - a tool that extracts data from one or more websites.

Web Crawler - a tool that finds or discovers URLs or links on the web.

Data augmentation - a technique of artificially increasing the training set by creating modified copies of a dataset using existing data.

Overfitting - an undesirable machine learning behavior that occurs when the machine learning model gives accurate predictions for training data but not for new data.

Genetic Algorithms - The genetic algorithm is a method for solving both constrained and unconstrained optimization problems that is based on natural selection, the process that drives biological evolution. The genetic algorithm repeatedly modifies a population of individual solutions.

DBMS - Database Management System