Reportbyte : An Advanced Business Analytics System

A Project
Submitted in partial fulfillment of the requirements for the Degree of
Bachelor of Science in Computer Science and Engineering

Submitted by

Samiul Islam Niloy	**170204016**
Shoaib Ahmed	**170204032**
MD Samir Uddin	**170204047**
MD Riyadul Islam	**170204054**

Supervised by

Mohammad Moinul Hoque

Department of Computer Science and Engineering
Ahsanullah University of Science and Technology

Dhaka, Bangladesh

June 30, 2022

CANDIDATES' DECLARATION

We, hereby, declare that the Project presented in this report is the outcome of the investigation performed by us under the supervision of Mohammad Moinul Hoque, Department of Computer Science and Engineering, Ahsanullah University of Science and Technology, Dhaka, Bangladesh. The work was spread over two final year courses, CSE4100: Project and Thesis I and CSE4250: Project and Thesis II, in accordance with the course curriculum of the Department for the Bachelor of Science in Computer Science and Engineering program.

It is also declared that neither this Project nor any part thereof has been submitted anywhere else for the award of any degree, diploma or other qualifications.

Samiul Islam Niloy
170204016

Shoaib Ahmed
170204032

MD Samir Uddin
170204047

MD Riyadul Islam
170204054

CERTIFICATION

This Project titled, **"Reportbyte : An Advanced Business Analytics System"**, submitted by the group as mentioned below has been accepted as satisfactory in partial fulfillment of the requirements for the degree B.Sc. in Computer Science and Engineering in June 30, 2022.

Group Members:

Samiul Islam Niloy	**170204016**
Shoaib Ahmed	**170204032**
MD Samir Uddin	**170204047**
MD Riyadul Islam	**170204054**

Mohammad Moinul Hoque
Associate Professor & Supervisor
Department of Computer Science and Engineering
Ahsanullah University of Science and Technology

Professor Dr. Mohammad Shafiul Alam
Professor & Head
Department of Computer Science and Engineering
Ahsanullah University of Science and Technology

ACKNOWLEDGEMENT

First and foremost, We are grateful to the Almighty Allah for the good health and well being that were necessary to complete this project work. Then we have to thank our project supervisor, Mohammad Moinul Hoque - without whose encouragement, this project would have never been accomplished. His constant support, understanding and constructive critiques over the final year have enriched our project work to a great extent.

We place on record, our sincere thanks to Prof. Dr. Mohammad Shafiul Alam, honourable Head of the department, for his continuous encouragement.

We take this opportunity to express gratitude to all the respected faculty members of our department for their help and support. Also, we thank our parents for the unceasing encouragement, support and attention. Finally, we are grateful to everyone who helped us in every possible ways to make our project fruitful.

Dhaka
June 30, 2022

Samiul Islam Niloy

Shoaib Ahmed

MD Samir Uddin

MD Riyadul Islam

ABSTRACT

Artificial intelligence is rapidly establishing itself as a major player in the fields of business and data analytics. We can now use artificial intelligence to not only evaluate data but also foresee and predict the future, which can aid in making key management and commercial choices. In our project, we're creating four of these technologies to act as a smart business assistant. The first one, called ShopBot, is a conversational chatbot with intelligence that will assist users of e-commerce websites in shopping and selecting items that best fit their needs by navigating the inventory more quickly and easily. The second one, called QueryBot, is a question-answering technology based on table parsing that will assist managers in making crucial managerial decisions. The third one, Predictor, will aid in the examination of predictive data. The fourth One is a tool for data visualization that goes by the moniker Visualy.

Keywords: NLP, NLU, Table Parsing, Chatbot, Monte Carlo, Data Mining.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Reportbyte is an advanced business analytic and assistant system which will provide four kind of services built with machine learning and artificial intelligence tools.

1.1 Shopbot

Shopbot is a chatbot which can be used in e-commerce sites. It is built with Rasa. Rasa is an open-source machine learning framework for automated text and voice-based conversations. Understand messages, hold conversations and connect to messaging channels and APIs. Shopbot will help customers of E-Commerce sites to buy and recommend products.

1.2 QueryBot

QueryBot is a question answering bot built using TAPAS question answering model. QueryBot will help managers or financial advisors by giving key insights from a given data. User will basically ask questions & the answer will be fetched using machine learning algorithms and displayed.

1.3 Visualy

Visualy is a data visualization tool. Clients will have a customized dashboard and options to choose from, how they want to visualize a portion of data from their given datasets. Various data mining algorithms such as K-Means, KNN etc will be used to find a group of customers with similar interest to promote a product. By using association rules, we'll find products

that are frequently bought together using Market Basket Analysis, Apriori algorithm so that we can improve efficiency of offer promotion.

1.4 Predicto

Predicto is a predictive data analysis tool.It will do business analysis or market analysis using Monte Carlo Simulation and then generate report/summary using Natural Language Generation (NLG). Monte Carlo is a technique used to understand the impact of risk and uncertainty in prediction and forecasting models. The T5 model, pre-trained on C4, achieves state-of-the-art results on many NLP benchmarks while being flexible enough to be fine-tuned to a variety of important downstream tasks.

Chapter 2

Project General Context

2.1 Problem

In today's world where E-Commerce sites are blooming faster than before, there's always an increasing need for customer support and care. While many websites offer a chatbot service nowadays to make shopping experience easier but their capability and performance is very limited and often needs human intervention. That's why we aim to create a chatbot (Shopbot) built with modern machine learning and artificial intelligence based techniques and tools that can help user by recommending and aiding while shopping in more ways than that was possible before.

In today's world marketing, key to success for any business is short and small. That's why importance of data and data analysis is getting more popular than ever before, that's why managers need various insights from existing data to make present and future decisions we aim to create a question answering bot (QueryBot) that can answer question from any given data by table parsing method and generating query by itself.

Data visualization is essential to assist businesses in quickly identifying data trends, which would otherwise be a hassle. The pictorial representation of data sets allows analysts to visualize concepts and new patterns. With the increasing surge in data every day, making sense of the quintillion bytes of data is impossible without Data Proliferation, which includes data visualization. Visualy will help in these kinds of situations.

By examining patterns in large amounts of data, predictive analytics professionals can identify trends and behaviors in an industry. These predictions provide valuable insights that can lead to better-informed business and investment decisions. Predicto will do these for the business leaders for betterment of their businesses.

2.2 Objective

The purpose of business analytic is to gain a complete understanding of the "how" and "why" of past events by identifying, collecting and analyzing key performance data, which can improve the decision making process going forward.

Business analytic solutions make it possible to synthesize historical data across an entire business. Applying business analytic can guide - and accelerate - business decisions and improve performance, helping organizations deliver better customer experiences, improve products, optimize marketing and enhance business processes.

Business analytic turns data into actionable insights that can inform strategic and tactical decisions, such as improving business planning, understanding and enhancing customer loyalty, and improving the performance of a contact center or help desk.

For example, Gatwick Airport uses business data to build a stronger customer experience. By monitoring data from its own systems and social media activity, the airport can more accurately predict passenger flow ahead of time.

Business analysis software tools, from data platforms (such as relational databases) to data visualization (like dashboards), collectively help business stakeholders manage their business operations more effectively.

Business analytic is an important tool for organization seeking a competitive edge in fast-moving industries. Active data analysis for faster, better decision making is seen as crucial to business success. Business analytic software makes it easier for nontechnical users to glean insights from performance data. The benefits of business analytic include cost savings, resource allocation, process improvement, products and customer experience, and the ability to better meet future needs. Business analytic solutions make it possible to synthesize key data collected across an entire business, regardless of the source. This lets organizations find the answers they need so that they can make smart decisions quickly.

2.3 Functionality

2.3.1 Shopbot

Shopbot runs using action servers that can operate independently separate from main website. It can be integrated to any website irrespective of the frameworks or technology it is built with. Whether the site runs on Python or PHP or any modern JavaScript based frameworks it can be connected in more ways than one. First the model is built with Rasa Framework then trained after that the trained model is served to fulfill the real time requests

made by user.

2.3.2 Querybot

QueryBot is a service hosted in Reportbyte's main website which takes a question from user then feeds the input to a question answering model named tapas then it returns the answer back to the user.

2.3.3 Visualy

Visualy is a data visualization tool. It will let the user know sales analysis overview, regional product performance, customer product recommendation based on purchase history clustering, market basket analysis.

2.3.4 Pedicto

Predicto is a sales forecasting tool. It uses Monte Carlo Simulation. Monte Carlo Simulations are used to model the probability of different outcomes in a process that cannot easily be predicted due to the intervention of random variables. It is a technique used to understand the impact of risk and uncertainty in prediction and forecasting models.

2.4 Feasibility

2.4.1 Technical Feasibility

Technical feasibility evaluates the technical complexity of the expert system and often involves determining whether the expert system can be implemented with state-of-the-art techniques and tools. In the case of expert systems, an important aspect of technical feasibility is determining the shell in which the system will be developed. The shell used to develop an expert system can be an important determinant to its quality and makes it vital to the system's success. Although the desirable characteristics of an expert system shell will depend on the task and domain requirements, the shell must be flexible enough to build expert reasoning into the system effectively. It must also be easily integrated with existing computer-based systems. Furthermore, a shell providing a user-friendly interface encourages end users to use the system more frequently.

2.4.2 Financial feasibility

Financial feasibility focuses specifically on the financial aspects of the study. It assesses the economical viability of a proposed venture by evaluating the startup costs, operating expenses, cash flow and making a forecast of future performance.

The project is technically and financially feasible. Parties who will use Shopbot, Querybot, Visualy or Predicto service for the growth of their business will be charged a subscription fee on monthly basis. Most of the tech used in this project are open source and free of cost; Uses secure technologies on both the front-end and the back-end and any requests sent by the users are passed by multiple middle-wares on the back-end. Fast and secure data transaction between host website and our services will be ensured.

Chapter 3

Methodology

3.1 Shopbot

We used Rasa to implement the chatbot feature of Reportbyte. The bot is trained with this NLU framework and takes appropriate actions and gives satisfactory responses to customer's query.

3.1.1 Rasa

Rasa [2] is an open source machine learning framework for automated text and voice-based conversations. Understand messages, hold conversations and connect to messaging channels and APIs. Basic parts of Rasa are:

1. NLU data
2. Responses
3. Stories
4. Rules
5. Look-Up Table
6. Actions

3.1.2 NLU Data

For an assistant to recognize what a user is saying no matter how the user phrases their messages, we need to provide example messages from which the assistant can learn from. We group these examples according to the idea or the goal the message is expressing, which is also called the 'intent'. In the code block on the right, we have added an intent called 'greet', which contains example messages like "Hi", "Hey", and "Good morning".
Intents and their examples are used as training data for the assistant's Natural Language Understanding (NLU) model.
NLU training data consists of example user utterances categorized by intent. Training examples can also include entities. Entities are structured pieces of information that can be extracted from a user's message. Admin can also add extra information such as regular expressions and lookup tables to admin's training data to help the model identify intents and entities correctly.
NLU training data is defined under the NLU key. Items that can be added under this key are:

- Training examples grouped by user intent e.g. optionally with annotated entities

```
nlu:
- intent: check_balance
  examples: |
   - What's my [credit](account) balance?
   - What's the balance on my [credit card account]
     {"entity":"account","value":"credit"}
```

- Synonyms

```
nlu:
- synonym: credit
  examples: |
   - credit card account
   - credit account
```

- Regular expressions

```
nlu:
- regex: account_number
  examples: |
   - \d{10,12}
```

- Lookup tables

```
nlu:
- lookup: banks
  examples: |
    - JPMC
    - Comerica
    - Bank of America
```

3.1.3 Training Examples

Training examples are grouped by intent and listed under the examples key. Usually, admin will list one example per line as follows:

```
nlu:
- intent: greet
  examples: |
    - hey
    - hi
    - whats up
```

However, it's also possible to use an extended format if admin have a custom NLU component and need metadata for his examples:

```
nlu:
- intent: greet
  examples:
    - text: |
        hi
      metadata:
        sentiment: neutral
    - text: |
        hey there!
```

The metadata key can contain arbitrary key-value data that is tied to an example and accessible by the components in the NLU pipeline. In the example above, the sentiment metadata

could be used by a custom component in the pipeline for sentiment analysis.
Admin can also specify this metadata at the intent level:

```
nlu:
- intent: greet
  metadata:
    sentiment: neutral
  examples:
  - text: |
      hi
  - text: |
      hey there!
```

In this case, the content of the metadata key is passed to every intent example.

If admin want to specify retrieval intents, then admin's NLU examples will look as follows:

```
nlu:
- intent: chitchat/ask_name
  examples: |
    - What is your name?
    - May I know your name?
    - What do people call you?
    - Do you have a name for yourself?
- intent: chitchat/ask_weather
  examples: |
    - What's the weather like today?
    - Does it look sunny outside today?
    - Oh, do you mind checking the weather for me please?
    - I like sunny days in Berlin.
```

All retrieval intents have a suffix added to them which identifies a particular response key for admin's assistant. In the above example, ask_name and ask_weather are the suffixes. The suffix is separated from the retrieval intent name by a '/' delimiter.

3.1.4 Entities

Entities are structured pieces of information that can be extracted from a user's message. Entities are annotated in training examples with the entity's name. In addition to the entity name, admin can annotate an entity with synonyms, roles, or groups.
In training examples, entity annotation would look like this:

```
nlu:
- intent: check_balance
  examples: |
    - how much do I have on my [savings](account) account
    - how much money is in my [checking]{"entity": "account"} account
    - What's the balance on my [credit card account]
      {"entity":"account","value":"credit"}
```

The full possible syntax for annotating an entity is:

```
[<entity-text>]{"entity": "<entity name>", "role": "<role name>",
 "group": "<group name>", "value": "<entity synonym>"}
```

The keywords role, group, and value are optional in this notation. The value field refers to synonyms. To understand what the labels role and group are for, see the section on entity roles and groups.

3.1.5 Synonyms

Synonyms normalize admin's training data by mapping an extracted entity to a value other than the literal text extracted. Admin can define synonyms using the format:

```
nlu:
- synonym: credit
  examples: |
    - credit card account
    - credit account
```

Admin can also define synonyms in-line in user's training examples by specifying the value of the entity:

```
nlu:
- intent: check_balance
  examples: |
    - how much do I have on my [credit card account]
      {"entity": "account", "value": "credit"}
    - how much do I owe on my [credit account]
      {"entity": "account", "value": "credit"}
```

3.1.6 Regular Expressions

Admin can use regular expressions to improve intent classification and entity extraction using the RegexFeaturizer and RegexEntityExtractor components.

The format for defining a regular expression is as follows:

```
nlu:
- regex: account_number
  examples: |
    - \d{10,12}
```

Here account_number is the name of the regular expression. When used as as features for the RegexFeaturizer the name of the regular expression does not matter. When using the RegexEntityExtractor, the name of the regular expression should match the name of the entity admin want to extract.

3.1.7 Responses

Now that the assistant understands a few messages users might say, it needs responses it can send back to the user.
"Hello, how can I help you?" and "what's your email address?" are some of the responses our assistant will use. Admin will see how to connect user messages and responses in the next steps.
In the code block below, we have listed some responses and added one or more text options for each of them. If a response has multiple text options, one of these options will be chosen at random whenever that response is predicted.

```
responses:
 utter_greet:
   - text: |
       Hello! How can I help you?
   - text: |
       Hi!
 utter_ask_email:
   - text: |
       What is your email address?
 utter_subscribed:
   - text: |
       Check your inbox at {email} in order to finish subscribing
       to the newsletter!
   - text: |
       You're all set! Check your inbox at {email} to confirm your
       subscription.
```

3.1.8 Stories

Stories are example conversations that train an assistant to respond correctly depending on what the user has said previously in the conversation. The story format shows the intent of the user message followed by the assistant's action or response.
Admin's first story should show a conversation flow where the assistant helps the user accomplish their goal in a straightforward way. Later, admin can add stories for situations where the user doesn't want to provide their information or switches to another topic.
In the code block below, we have added a story where the user and assistant exchange greetings, the user asks to subscribe to the newsletter and the assistant starts collecting the information it needs through the newsletter_form. Admin will learn about forms in the next step.

```
stories:
 - story: greet and subscribe
 steps:
  - intent: greet
  - action: utter_greet
  - intent: subscribe
```

```
  - action: newsletter_form
  - active_loop: newsletter_form
```

3.1.9 Rules

Rules describe parts of conversations that should always follow the same path no matter what has been said previously in the conversation.
We want our assistant to always respond to a certain intent with a specific action, so we use a rule to map that action to the intent.
In the code block below, we have added a rule that triggers the newsletter_form whenever the user expresses the intent "subscribe". We've also added a rule that triggers the utter_subscribed action once all the required information has been provided. The second rule only applies when the newsletter_form is rules:

```
- rule: activate subscribe form
  steps:
  - intent: subscribe
  - action: newsletter_form
  - active_loop: newsletter_form
- rule: submit form
  condition:
  - active_loop: newsletter_form
  steps:
  - action: newsletter_form
  - active_loop: null- action: utter_subscribedactive to begin with;
  once it is no longer active (active_loop: null), the form
  is complete.
```

Rules for the Conversation Start

To write a rule which only applies at the beginning of a conversation, add a **conversation_start: true** to the code-block's rule:

```
rules:
- rule: Say `hello` when the user starts a conversation with
    intent `greet`
```

```
  conversation_start: true
  steps:
  - intent: greet
  - action: utter_greet
```

If a user sends a message with the intent **greet** later in the conversation, the rule will not match.

Rules with Conditions

Conditions describe requirements which have to be fulfilled for a rule to be applicable. To do so, add any information about the prior conversation under the **condition** key:

```
rules:
- rule: Only say `hello` if the user provided a name
  condition:
  - slot_was_set:
  - user_provided_name: true
  steps:
  - intent: greet
  - action: utter_greet
```

Possible information that admin can include under **condition** includes **slot_was_set** events and **active_loop** events.

Skip Waiting for User Input at the End of a Rule

By default, rules will wait for the next user message when finished with the last step:

```
rules:
- rule: Rule which will wait for user message when it was applied
  steps:
  - intent: greet
  - action: utter_greet
  # - action: action_listen
  # Every rule implicitly includes a prediction for `action_listen`
    as last step.
  # This means that Rasa Open Source will wait for the next user message.
```

If admin want to hand over the next action prediction to another story or rule, add **wait_for_user_input: false** to admin's rule:

```
rules:
- rule: Rule which will not wait for user message once it was applied
steps:
- intent: greet
- action: utter_greet
wait_for_user_input: false
```

This indicates that the assistant should execute another action before waiting for more user input.

3.1.10 Actions

After each user message, the model will predict an action that the assistant should perform next. This page gives admin an overview of the different types of actions admin can use. There are different types of action depending on the needs.

Responses

A response is a message the assistant will send back to the user. This is the action admin will use most often, when admin wants the assistant to send text, images, buttons or similar to the user. start with utter_ and send a specific message to the user. e.g.

```
stories:
- story: story with a response
  steps:
  - intent: greet
  - action: utter_greet
```

Custom Actions

A custom action is an action that can run any code admin wants. This can be used to make an API call, or to query a database for example. start with action_, run arbitrary code and send any number of messages (or none).

```
stories:
- story: story with a custom action
  steps:
  - intent: feedback
  - action: action_store_feedback
```

Forms

Forms are a special type of custom action, designed to handle business logic. If admin have any conversation designs where admin expects the assistant to ask for a specific set of information, admin should use forms.
A form is a specific kind of custom action that contains the logic to loop over a set of required slots and ask the user for this information. Admin defines a form in the forms section in his domain. Once defined, admin should specify the happy path for a form as a rule. Admin should include interruptions of forms or other "unhappy paths" in stories so that the model can generalize to unseen conversation sequences. As a step in a story, a form takes the following format:

Default Actions

Default actions are actions that are built into the dialogue manager by default. Most of these are automatically predicted based on certain conversation situations. Admin may want to customize these to personalize admin's assistant.

```
stories:
- story: story with a form
  steps:
  - intent: find_restaurant
  - action: restaurant_form # Activate the form
  - active_loop: restaurant_form # This form is currently active
  - active_loop: null # Form complete, no form is active
  - action: utter_restaurant_found
```

The action step activates the form and begins looping over the required slots. The active_loop: restaurant_form step indicates that there is a currently active form. Much like a slot_was_set step, a form step doesn't set a form to active but indicates that it should already

be activated. In the same way, the active_loop: null step indicates that no form should be active before the subsequent steps are taken.
A form can be interrupted and remain active; in this case the interruption should come after the action: <form to activate> step and be followed by the active_loop: <active form> step. An interruption of a form could look like this:

```
stories:
- story: interrupted food
  steps:
   - intent: request_restaurant
   - action: restaurant_form
   - intent: chitchat
   - action: utter_chitchat
   - active_loop: restaurant_form
   - active_loop: null - action: utter_slots_values
```

Slot Validation Actions

A slot validation action is a special type of custom action, designed to handle custom extraction and/or validation of slot values. This can be used to validate slots with predefined mappings or extract slots with custom mappings.
A slot event is specified under the key slot_was_set: with the slot name and optionally the slot's value.
Slots act as the bots memory. Slots are set by either the default action action_extract_slots according to the slot mappings specified in the domain, or by custom actions. They are referenced by stories in slot_was_set steps. For example:

```
stories:
- story: story with a slot
  steps:
  - intent: celebrate_bot
  - slot_was_set:
    - feedback_value: positive
  - action: utter_yay
```

This means the story requires that the current value for the feedback_value slot be positive for the conversation to continue as specified. Whether or not admin need to include the slot's value depends on the slot type and whether the value can or should influence the

dialogue. If the value doesn't matter, as is the case for e.g. text slots, admin can list only the slot's name:

```
stories:
- story: story with a slot
  steps:
  - intent: greet
  - slot_was_set:
    - name
  - action: utter_greet_user_by_name
```

The initial value for any slot by default is null and admin can use it to check if the slot was not set:

```
stories:
- story: French cuisine
  steps:
  - intent: inform
  - slot_was_set:
    - cuisine: null
```

3.2 Querybot

Querybot was built usng the TAPAS model which is Weakly Supervised Table Parsing via Pre-training.

3.2.1 Tapas: Introduction

TAPAS [3] was pre-trained on a set of 6.2 million data tables extracted from Wikipedia, with associated questions extracted from the article title, article description, table caption, and other related text snippets. The Google team used three benchmark datasets: SQA, WTQ and Salesforce's WikiSQL. For SQA, TAPAS achieved 67.2% accuracy. The TAPAS model was proposed in TAPAS: Weakly Supervised Table Parsing via Pre-training by Jonathan Herzig, Pawel Krzysztof Nowak, Thomas Muller, Francesco Piccinno and Julian Martin Eisenschlos. It's a BERT-based model specifically designed (and pre-trained) for answering questions about tabular data. Compared to BERT, TAPAS uses relative position embeddings and has 7 token types that encode tabular structure. TAPAS is pre-trained on the masked language

modeling (MLM) objective on a large dataset comprising millions of tables from English Wikipedia and corresponding texts. For question answering, TAPAS has 2 heads on top: a cell selection head and an aggregation head, for (optionally) performing aggregations (such as counting or summing) among selected cells. TAPAS has been fine-tuned on several datasets: SQA (Sequential Question Answering by Microsoft), WTQ (Wiki Table Questions by Stanford University) and WikiSQL (by Salesforce). It achieves state-of-the-art on both SQA and WTQ while having comparable performance to SOTA on WikiSQL, with a much simpler architecture.

3.2.2 What is TAPAS

Answering natural language questions over tables is usually seen as a semantic parsing task. To alleviate the collection cost of full logical forms, one popular approach focuses on weak supervision consisting of denotations instead of logical forms. However, training semantic parsers from weak supervision poses difficulties and in addition, the generated logical forms are only used as an intermediate step prior to retrieving the denotation. In this book, we present TAPAS, an approach to question answering over tables without generating logical forms. TAPAS trains from weak supervision, and predicts the denotation by selecting table cells and optionally applying a corresponding aggregation operator to such selection. TAPAS extends BERT's architecture to encode tables as input, initializes from an effective joint pre-training of text segments and tables crawled from Wikipedia and is trained end-to-end. We experiment with three different semantic parsing datasets and find that TAPAS outperforms or rivals semantic parsing models by improving state-of-the-art accuracy on SQA from 55.1 to 67.2 and performing on par with the state-of-the-art on WIKISQL and WIKITQ, but with a simpler model architecture. We additionally find that transfer learning, which is trivial in our setting, from WIKISQL to WIKITQ, yields 48.7 accuracy, 4.2 points above the state-of-the-art.

3.2.3 How TAPAS works

The TAPAS model is Weakly Supervised Table Parsing via Pre-training. It's a BERT-based model specifically designed (and pre-trained) for answering questions about tabular data. Compared to BERT, TAPAS uses relative position embeddings and has 7 token types that encode tabular structure. TAPAS is pre-trained on the masked language modeling (MLM) objective on a large dataset comprising millions of tables from English Wikipedia and corresponding texts. For question answering, TAPAS has 2 heads on top: a cell selection head and an aggregation head, for (optionally) performing aggregations (such as counting or summing) among selected cells. TAPAS has been fine-tuned on several datasets: SQA (Se-

quential Question Answering by Microsoft), WTQ (Wiki Table Questions by Stanford University) and WikiSQL (by Salesforce). It achieves state-of-the-art on both SQA and WTQ, while having comparable performance to SOTA on WikiSQL, with a much simpler architecture.

3.2.4 TAPAS in Depth

TAPAS is a model that uses relative position embeddings by default (restarting the position embeddings at every cell of the table). Note that this is something that was added after the publication of the original TAPAS paper. According to the authors, this usually results in a slightly better performance, and allows admin to encode longer sequences without running out of embeddings. This is reflected in the reset_position_index_per_cell parameter of TapasConfig, which is set to True by default. The default versions of the models available in the model hub all use relative position embeddings. Admin can still use the ones with absolute position embeddings by passing in an additional argument revision="no_reset" when calling the .from_pretrained() method. Note that it's usually advised to pad the inputs on the right rather than the left.

TAPAS is based on BERT, so TAPAS-base for example corresponds to a BERT-base architecture. Of course, TAPAS-large will result in the best performance (the results reported in the paper are from TAPAS-large). Results of the various sized models are shown on the original Github repository.

TAPAS has checkpoints fine-tuned on SQA, which are capable of answering questions related to a table in a conversational set-up. This means that admin can ask follow-up questions such as "what is his age?" related to the previous question. Note that the forward pass of TAPAS is a bit different in case of a conversational set-up: in that case, admin have to feed every table-question pair one by one to the model, such that the prev_labels token type ids can be overwritten by the predicted labels of the model to the previous question. See "Usage" section for more info.

TAPAS is similar to BERT and therefore relies on the masked language modeling (MLM) objective. It is therefore efficient at predicting masked tokens and at NLU in general, but is not optimal for text generation. Models trained with a causal language modeling (CLM) objective are better in that regard.

3.2.5 Fine-tuning TAPAS

Basically, there are 3 different ways in which one can fine-tune TapasForQuestionAnswering, corresponding to the different datasets on which Tapas was fine-tuned:

1. **SQA:** If admin is interested in asking follow-up questions related to a table, in a conversational set-up. For example if admin first asks "what's the name of the first actor?" then admin can ask a follow-up question such as "how old is he?". Here, questions do not involve any aggregation (all questions are cell selection questions).

2. **WTQ:** If admin is not interested in asking questions in a conversational set-up, but rather just asking questions related to a table, which might involve aggregation, such as counting a number of rows, summing up cell values or averaging cell values. Admin can then for example ask "what's the total number of goals Cristiano Ronaldo made in his career?". This case is also called weak supervision, since the model itself must learn the appropriate aggregation operator (SUM/COUNT/AVERAGE/NONE) given only the answer to the question as supervision.

3. **WikiSQL-supervised:** This dataset is based on WikiSQL with the model being given the ground truth aggregation operator during training. This is also called strong supervision. Here, learning the appropriate aggregation operator is much easier.

4. Admin can also experiment by defining any hyperparameters admin wants when initializing TapasConfig, and then create a TapasForQuestionAnswering based on that configuration. For example, if admin has a dataset that has both conversational questions and questions that might involve aggregation, then admin can do it this way. Here's an example:

```
>>> from transformers import TapasConfig, TapasForQuestionAnswering

>>> # you can initialize the classification heads any way you want
>>> config = TapasConfig(num_aggregation_labels=3,
    average_logits_per_cell=True)
>>> # initializing the pre-trained base sized model with our custom
    classification heads
>>> model = TapasForQuestionAnswering.from_pretrained
    ('google/tapas-base', config=config)
```

And here is the equivalent code for TensorFlow:

```
>>> from transformers import TapasConfig,
    TFTapasForQuestionAnswering

>>> # you can initialize the classification heads any way you want
>>> config = TapasConfig(num_aggregation_labels=3,
    average_logits_per_cell=True)
>>> # initializing the pre-trained base sized model with our custom
    classification heads
>>> model = TFTapasForQuestionAnswering.from_pretrained
    ('google/tapas-base', config=config)
```

What admin can also do is start from an already fine-tuned checkpoint. A note here is that the already fine-tuned checkpoint on WTQ has some issues due to the L2-loss which is somewhat brittle.

3.3 Visualy

3.3.1 Market Basket Analysis [1]

Data analysis allows us to obtain valuable insights while saving time and effort. Without it, we'd be left searching for needles in the proverbial haystack of unstructured data. Thanks to data science and advanced analytic, however, data mining is greatly simplified, so that sifting through a large amount of data is made relatively simple. This, in turn, has a wide range of benefits for businesses and consumers.

With data analytic, business analysts can obtain a much better understanding of what customers need based on high-quality historical data from a plethora of data sources, allowing them to make better-informed decisions.

In this project we apply some machine learning algorithms and techniques that will help businesses take important decisions. Some key goals of this project are:

- Sales analysis for Product promotions.
- Sales analysis for Inventory management.
- Product recommendations.
- Market Basket Analysis for customer experience improvement.

- Sales Forecasting using Monte Carlo Simulation

3.3.2 Apriori

The Apriori algorithm is used for mining frequent item-sets and devising association rules from a transactional database. The parameters "support" and "confidence" are used. Support refers to items frequency of occurrence; confidence is a conditional probability.

Items in a transaction form an item set. The algorithm begins by identifying frequent, individual items (items with a frequency greater than or equal to the given support) in the database and continues to extend them to larger, frequent item-sets.

Fundamental concepts of the algorithm:

1. Calculate the support of item sets (of size k = 1) in the transactional database (note that support is the frequency of occurrence of an item-set). This is called generating the candidate set.
2. Prune the candidate set by eliminating items with a support less than the given threshold.
3. Join the frequent item-sets to form sets of size k + 1, and repeat the above sets until no more item-sets can be formed. This will happen when the set(s) formed have a support less than the given support.

3.3.3 Algorithm

A key concept in Apriori algorithm is the anti-monotonicity of the support measure. It assumes that all subsets of a frequent item-set must be frequent. Similarly, for any infrequent item-set, all its super-sets must be infrequent too.

Step 1: Create a frequency table of all the items that occur in all the transactions.

Step 2: We know that only those elements are significant for which the support is greater than or equal to the threshold support.

Step 3: The next step is to make all the possible pairs of the significant items keeping in mind that the order doesn't matter, i.e., AB is same as BA.

Step 4: We will now count the occurrences of each pair in all the transactions.

Step 5: Again only those itemsets are significant which cross the support threshold

Step 6: Now let's say we would like to look for a set of three items that are purchased together. We will use the itemsets found in step 5 and create a set of 3 items.

	lhs		rhs	support	confidence	lift
[1]	{liquor,red/blush wine}	=>	{bottled beer}	0.001931876	0.9047619	11.235269
[2]	{curd,cereals}	=>	{whole milk}	0.001016777	0.9090909	3.557863
[3]	{yogurt,cereals}	=>	{whole milk}	0.001728521	0.8095238	3.168192
[4]	{butter,jam}	=>	{whole milk}	0.001016777	0.8333333	3.261374
[5]	{soups,bottled beer}	=>	{whole milk}	0.001118454	0.9166667	3.587512
[6]	{napkins,house keeping products}	=>	{whole milk}	0.001321810	0.8125000	3.179840
[7]	{whipped/sour cream,house keeping products}	=>	{whole milk}	0.001220132	0.9230769	3.612599
[8]	{pastry,sweet spreads}	=>	{whole milk}	0.001016777	0.9090909	3.557863
[9]	{turkey,curd}	=>	{other vegetables}	0.001220132	0.8000000	4.134524
[10]	{rice,sugar}	=>	{whole milk}	0.001220132	1.0000000	3.913649

Figure 3.1: Top 10 Rules Derived from the Dataset

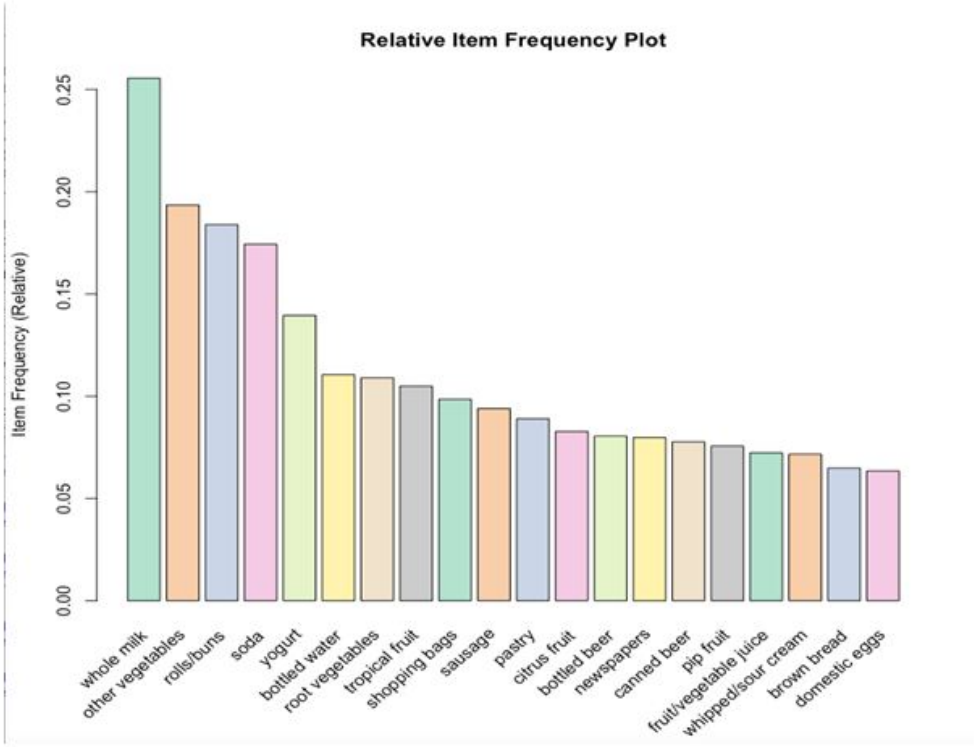

Figure 3.2: Item Frequency Histogram

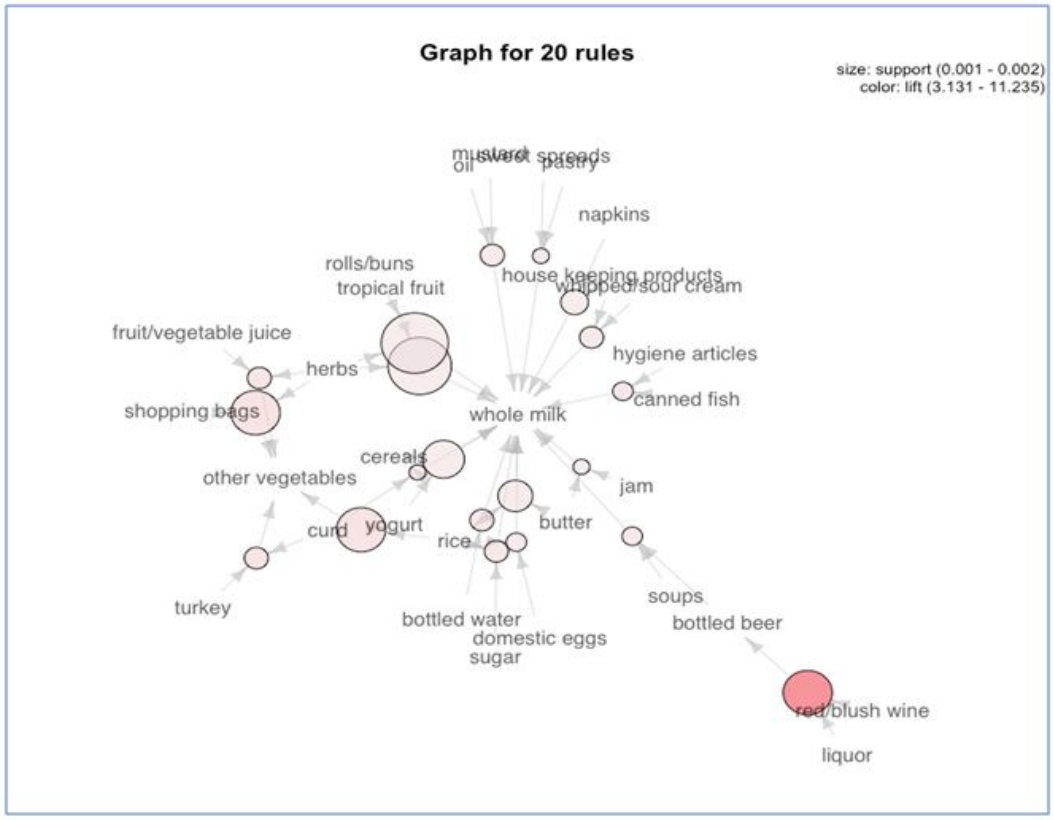

Figure 3.3: Graphical Representation of 20 Rules

3.3.4 Sales Analysis

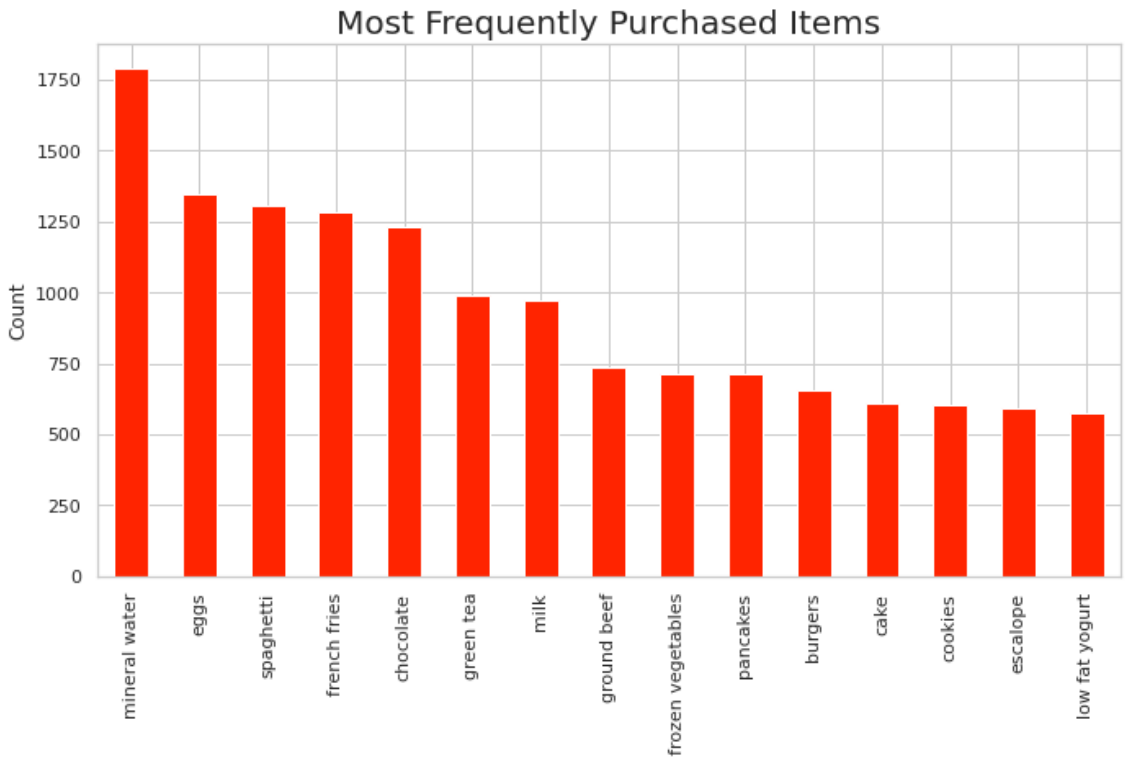

Figure 3.4: Most Commonly Bought Items, Presuming That Each Item Was Bought in One Unit Per Transaction

Out of all the things that are sold in a certain time period, the graph displays the top 15 most popular items. We can see that individuals most commonly purchase mineral water (1800 units on average), followed by eggs (1350 units), spaghetti (1300 units), and so forth.

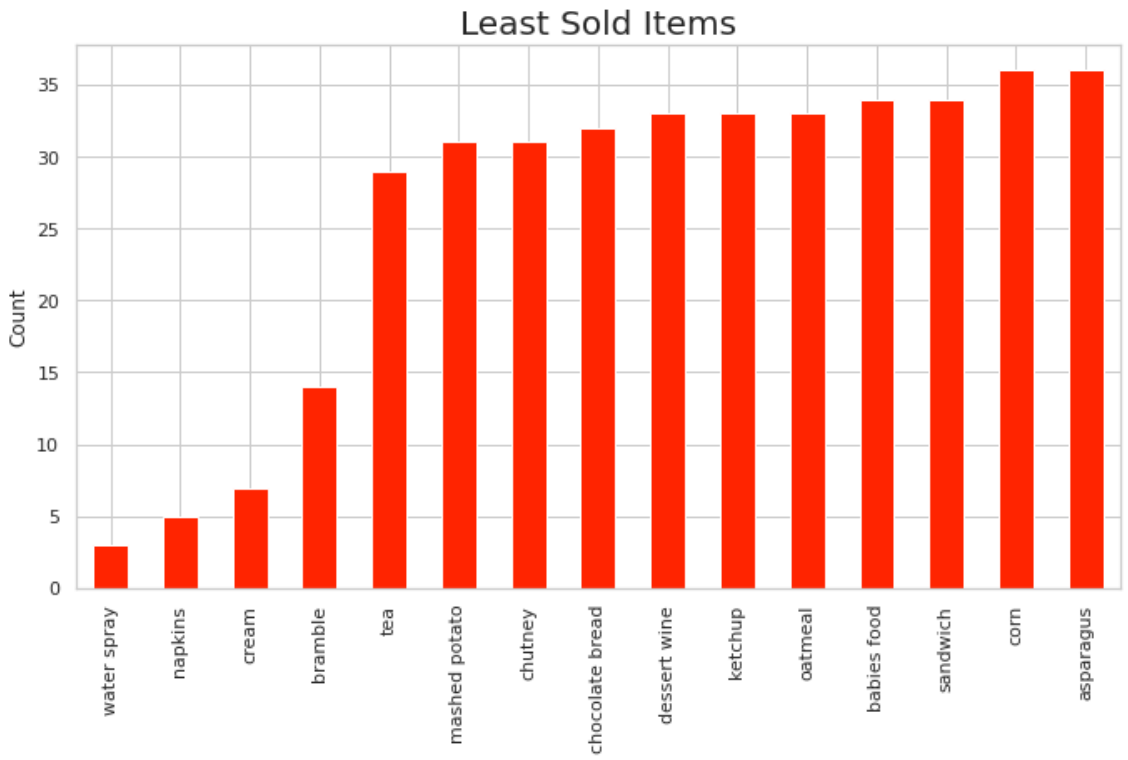

Figure 3.5: Least Sold Items, Considering That Each Transaction Only Purchased One Unit of Each Item

This graph displays the minimum number of goods that are sold during a given time frame—15. The least popular item, a water spray, sells for around 3 units, followed by napkins, which sell for about 5 units, cream, which sells for about 7 units, and so on.

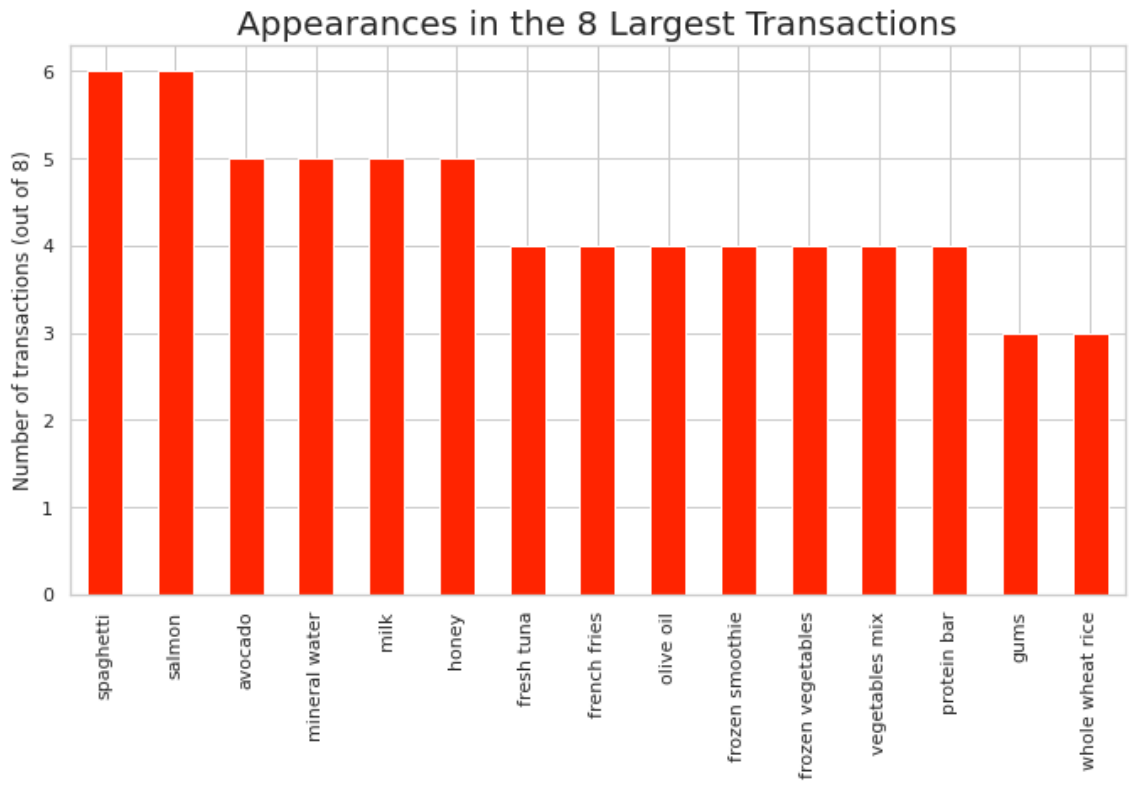

Figure 3.6: Items Appeared in the Eight Largest Transactions

Salmon and Spaghetti have both combined to feature six times in the top eight deals. The next four items—Avocado, mineral water, milk, and honey—appeared in the eight largest transactions a total of five times. In the eight greatest transactions, four different items—fresh tuna, French fries, olive oil, frozen smoothie, frozen veggies, vegetable mix, and protein bar—appeared. Finally, gums and whole grain rice made three appearances.

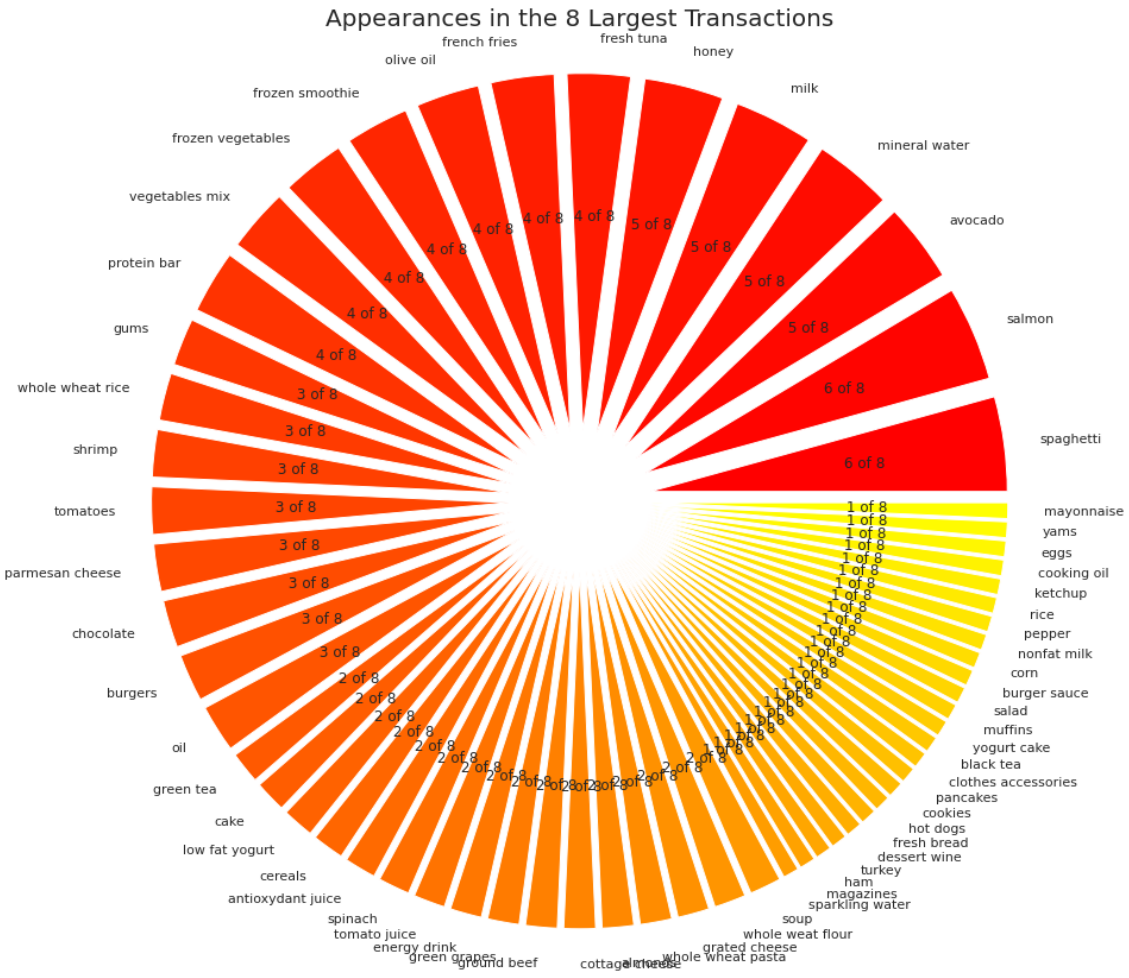

Figure 3.7: All Items Appeared in the Eight Biggest Transactions

Similar to the preceding image, this pie chart displays the goods that were a part of the eight greatest transactions. The pie chart, as opposed to the preceding graph, displays all of the objects that were involved in the eight largest transactions. The most common foods were spaghetti and salmon, whereas the least common foods were mayonnaise, yams, eggs, etc.

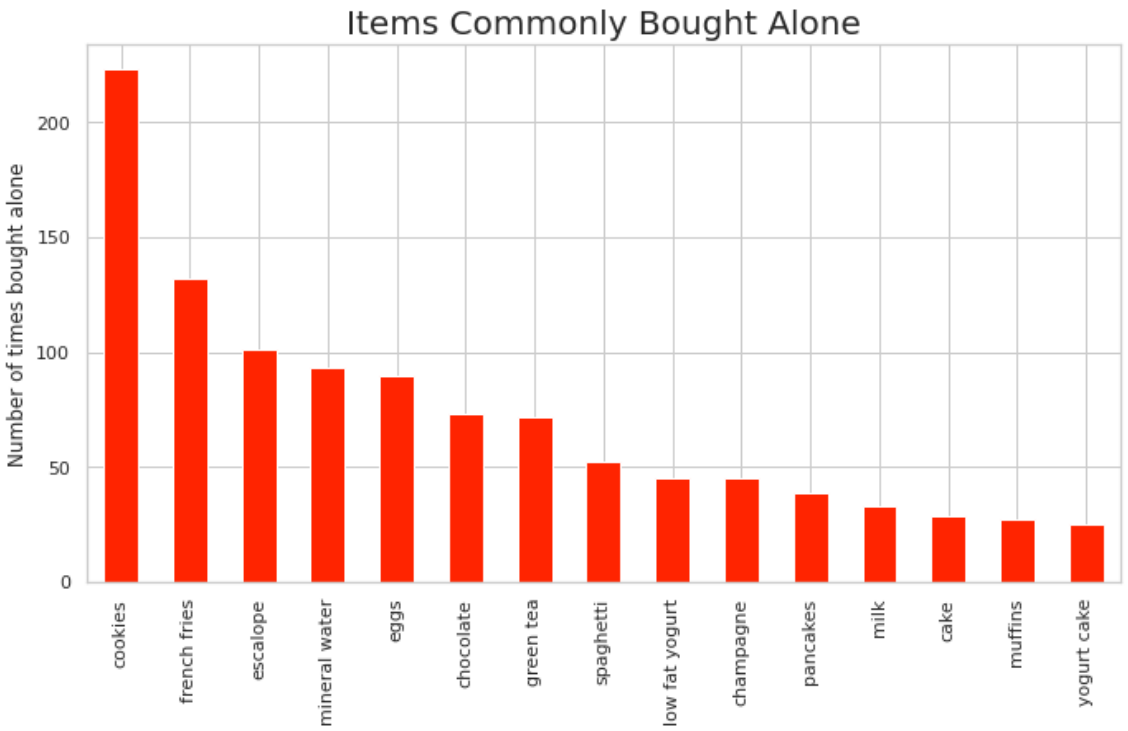

Figure 3.8: Lowest Value Transactions

Customers occasionally purchase pairs of products (such as tea and biscuits) or single items, as seen in the graph. We can notice that the most frequently sold alone item is a package of cookies. Then escalope is the third most popular food, followed by french fries in second place, and so on.

3.3.5 Data Mining

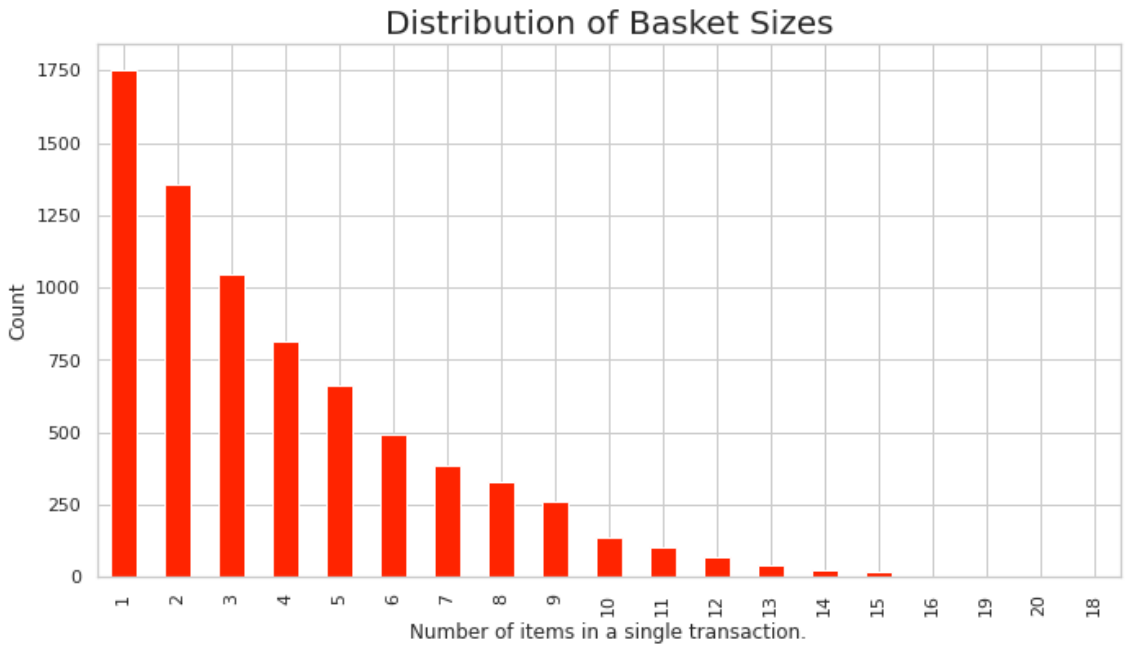

Figure 3.9: Distribution of Basket Sizes Under the Assumption that Each Item Was Purchased Once Only in Each Transaction

In this sense, the term "basket" refers to the collection of distinctive products bought in a single transaction.

The majority of the transactions only involved one item. In general, the frequency of basket (transaction) transactions dropped as basket size increased.

Therefore, the graph shows that one item is the most frequently transacted in a single transaction, followed by two things and so on.

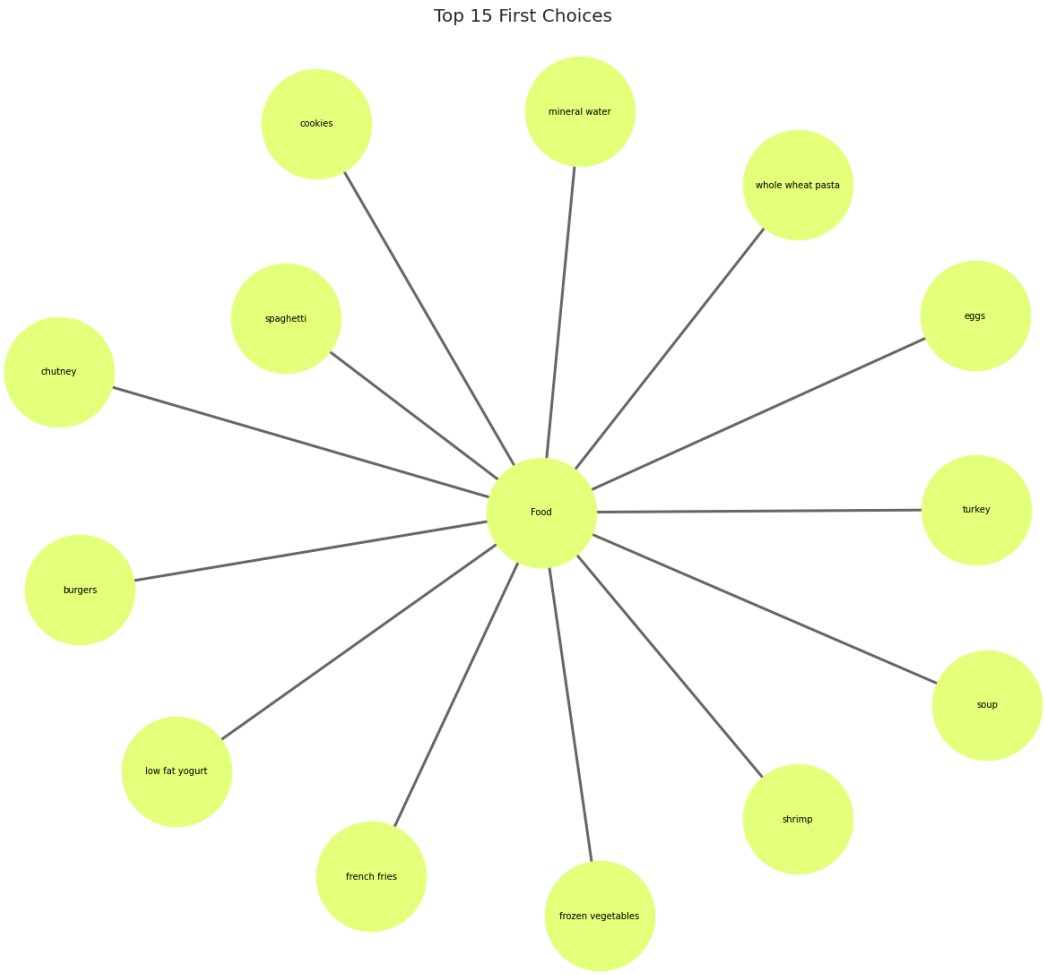

Figure 3.10: The Top 15 Choices (Food)

The top 15 food products according to user preference are displayed in this network diagram. When purchasing food from stores outdoors, people frequently start by purchasing these 15 foods.

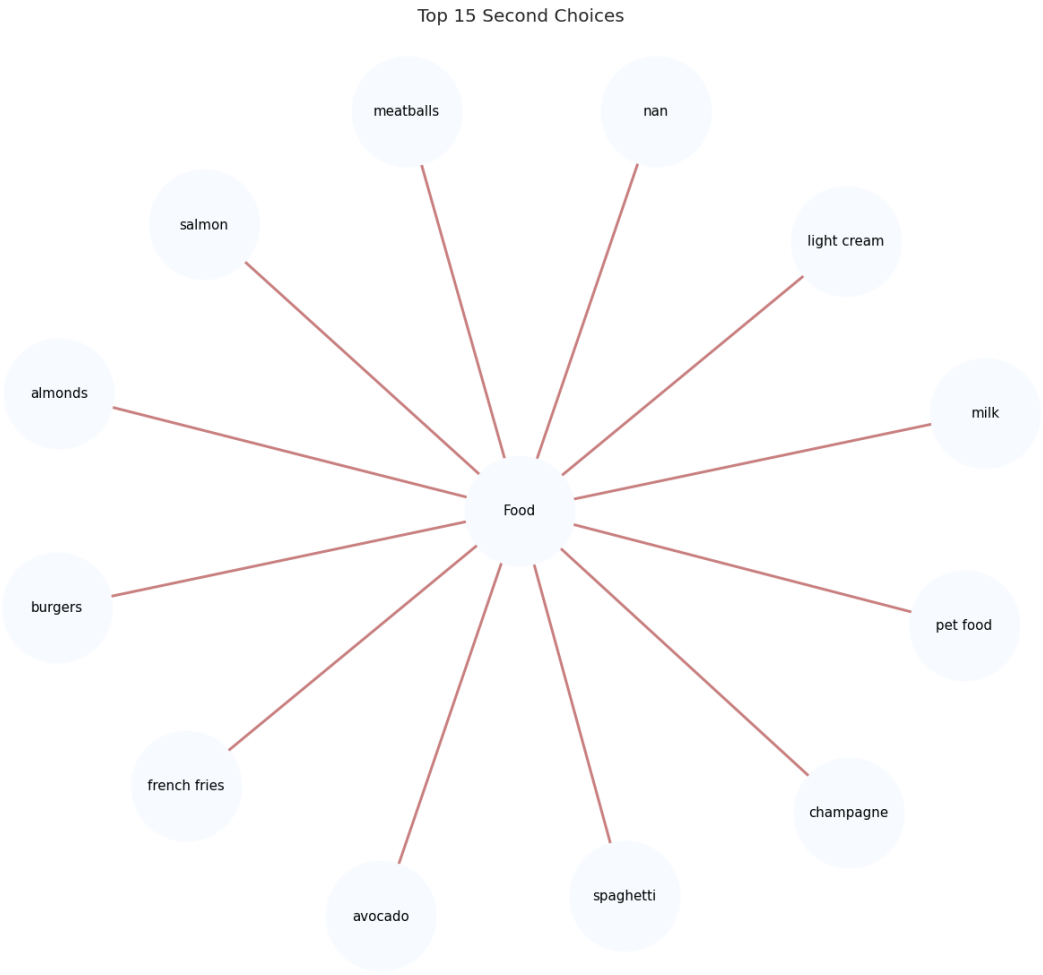

Figure 3.11: The Top 15 Second Choices (Food)

The top 15 second-place food products are displayed in this network graph. After purchasing their first choices, people will purchase these things.

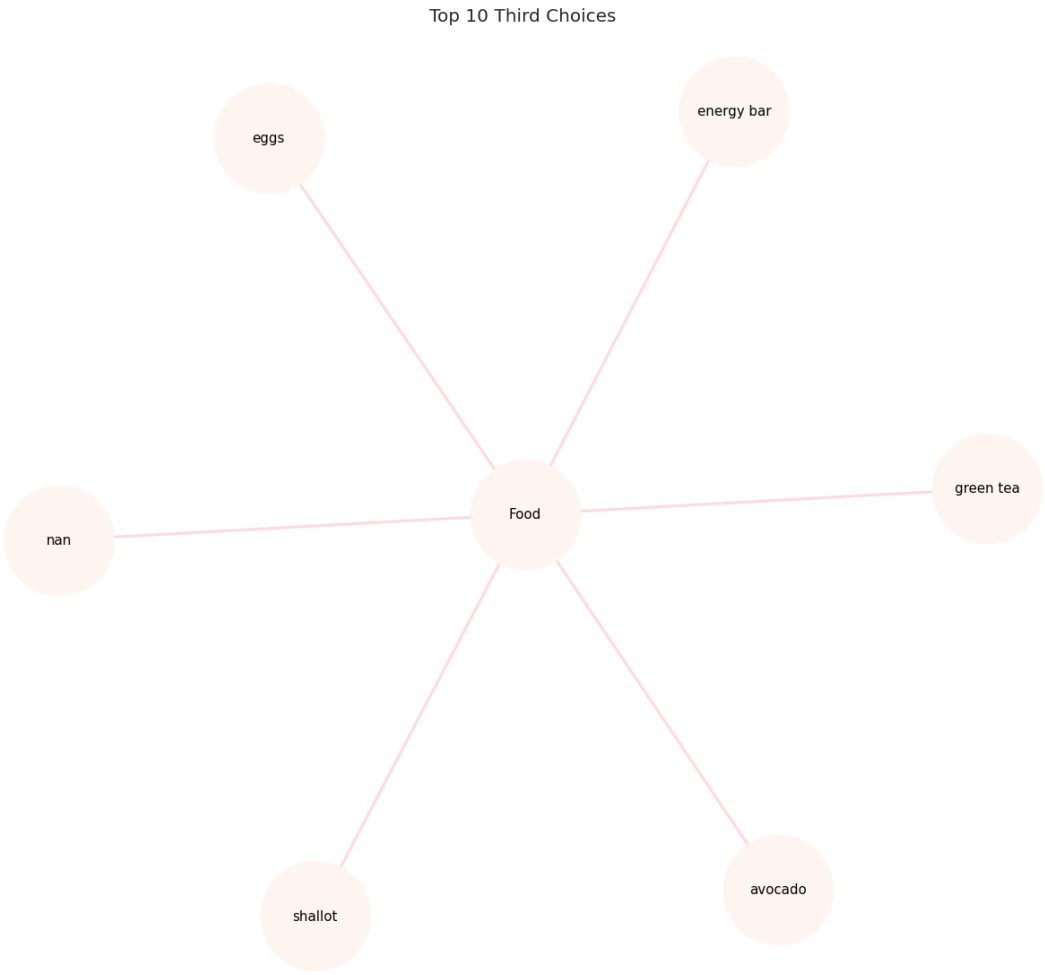

Figure 3.12: The Top 10 Third Choices (Food)

This graph shows the top 10 third meal items. In other words, customers won't buy these things until they've bought their first and second choices.

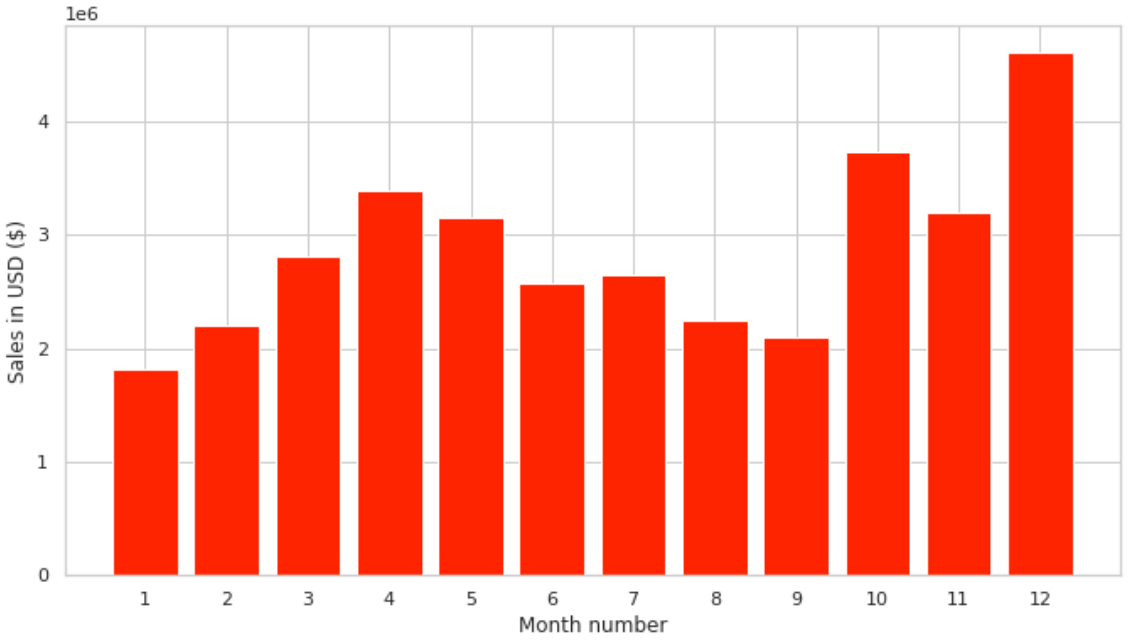

Figure 3.13: Sales in USD over the last 12 months

Sales for the entire year by month may be seen in the graph. Items are sold in US Dollar units, and as can be seen, total sales are highest in month number 12, December, and lowest in month number 1, January. The graph's second-highest and second-lowest months are October and September, respectively.

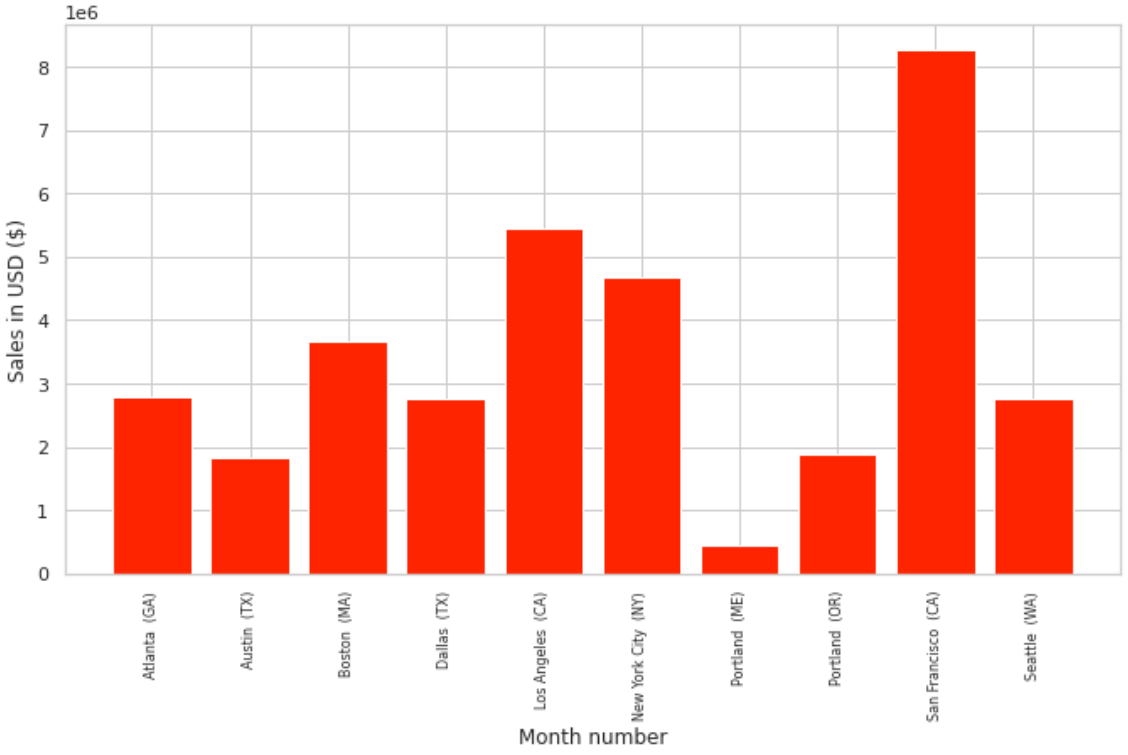

Figure 3.14: Sales According to Several US States and Cities

The graph displays the total sales in US dollars for each state. In the United States, there are 50 states and over ten thousand cities. Ten of those cities' sale statuses are visible. And among the ten cities, San Francisco in the state of California has the highest total sales, while Portland in the state of Maine has the lowest total sales. Average sales in other cities are mild.

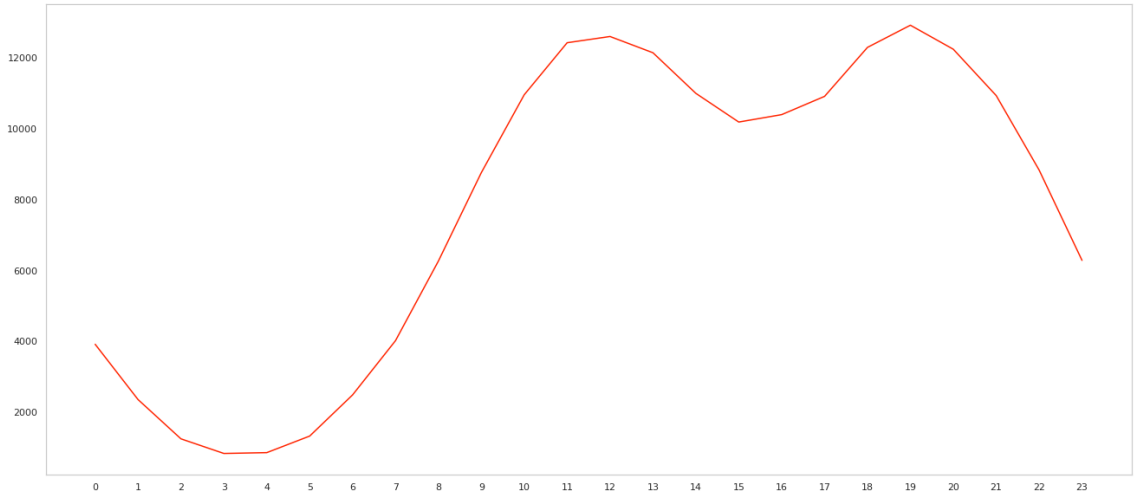

Figure 3.15: Total Sales (USD) for the Day

This graph helps us understand what percentage of the overall sale occurs throughout which hour. We can see that overall sales are at their highest between 10 AM and 12 PM and 6 PM and 8 PM, whereas total sales decline after these hours. Total sales are lowest between 3 and 4 AM, although they progressively increase with time.

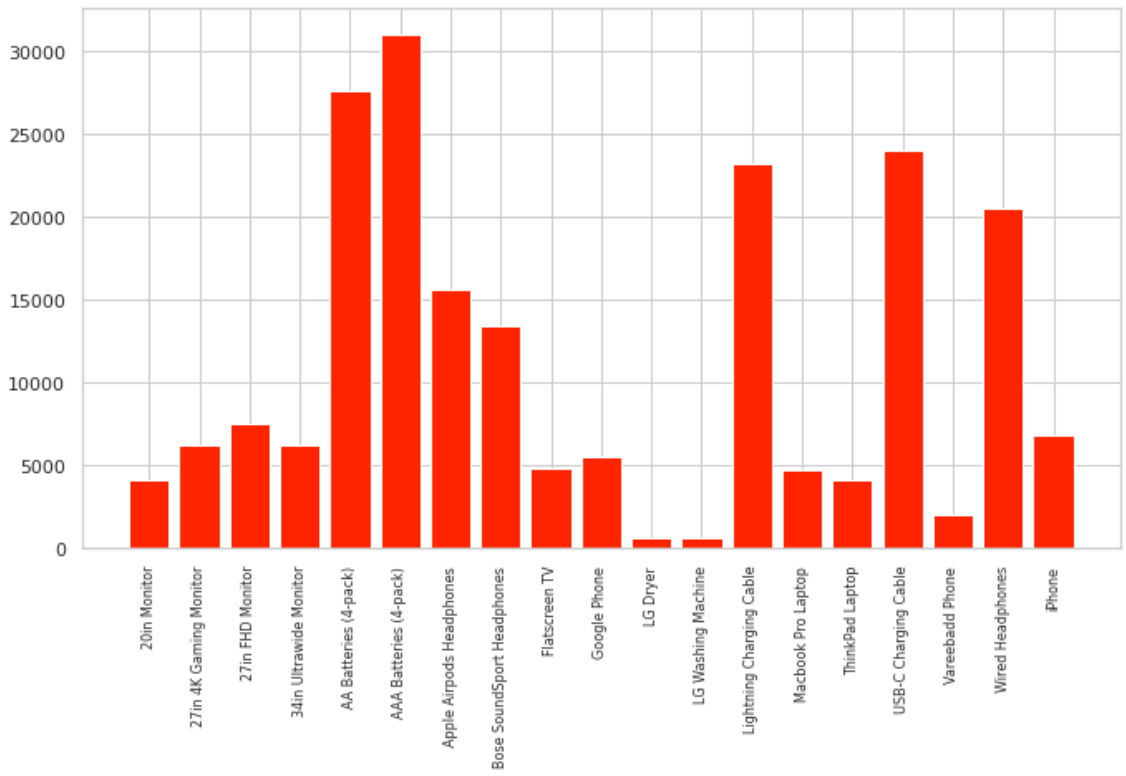

Figure 3.16: Product Name versus. Ordered Quantity

We can examine the overall volume of several products sold in various US cities from this graph. The majority of the products in the graph—nearly 31000 units—are sold as AAA Batteries (4-Pack). The least popular LG appliances, with little under 500 units sold each, are the dryer and washing machine.

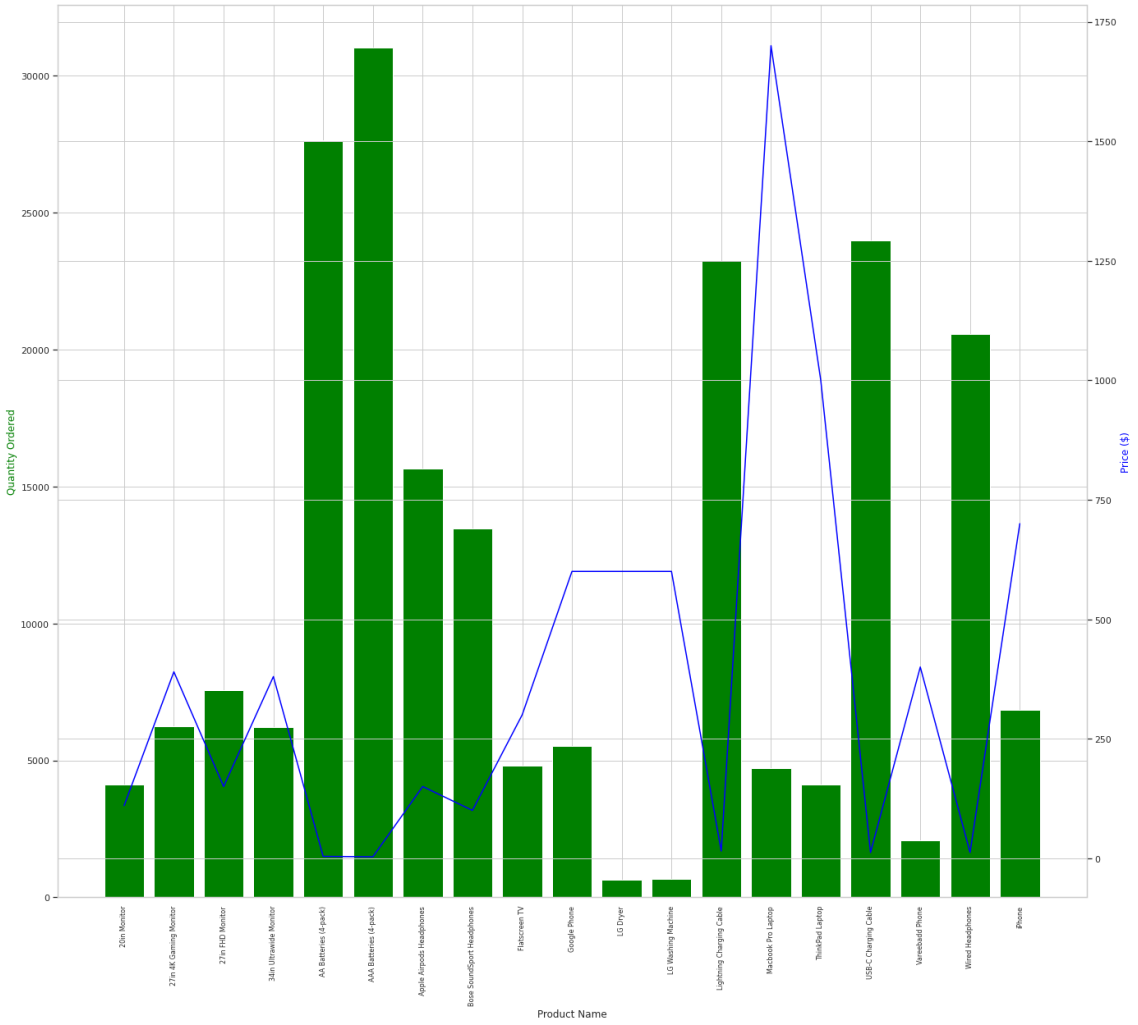

Figure 3.17: Product Name in Relation to Ordered Quantity and Price

The only difference between this graph and the one before it is that every product's price is displayed here. We can see that typically, things with cheap prices sell the most, whilst products with higher prices sell in less quantities. Examples are the 4-packs of AAA and AA batteries, the Macbook Pro laptop, the LG dryer, etc.

3.3.6 Product Recommendation

Product recommendation engines analyze data about shoppers to learn exactly what types of products and offerings interest them. Based on search behavior and product preferences, they serve up contextually relevant offers and product options that appeal to individual shoppers and help drive sales.

A product recommendation engine is a technology that uses machine learning and artificial intelligence (AI) to generate product suggestions and predictive offers, such as special deals and discounts, tailored to each customer. An effective product recommendation engine an-

alyzes data and uses the results to create accurate, individualized customer profiles. These profiles help the engine generate the exact kind of content or products a specific customer might be interested in. That's why you may get frequent follow-up emails from your favorite brands based on recent purchases and site searches.

Product recommendation engines analyze the following types of customer data:

1. Browser history
2. Current purchasing behavior
3. Feedback
4. Most-viewed products
5. Preferences
6. Previous purchases
7. Recently viewed items
8. Search history
9. Shopping carts
10. Wish lists

Based on that data, the technology can surface relevant products the customer might like. It can intelligently anticipate customer intent and include the recommended products in marketing materials, on an app, in site searches, and on ads featured on other web pages.

Recommendation system part I: Product popularity based system targeted at new customers

Popularity based are a great strategy to target the new customers with the most popular products sold on a business's website and is very useful to cold start a recommendation engine.

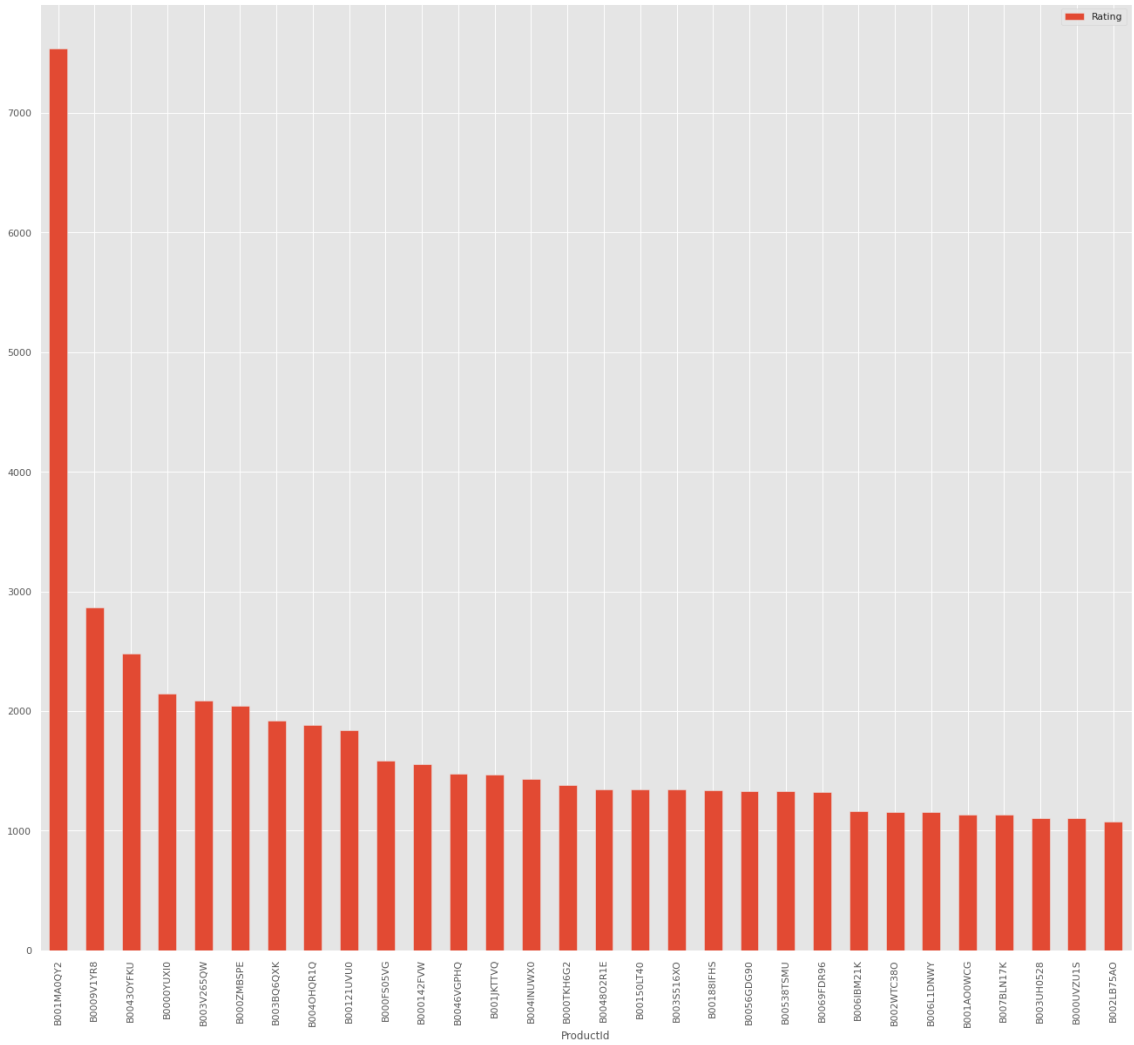

Figure 3.18: Product Rating and ProductID Indicate Product Popularity

Using product IDs, this graph displays the top-selling items out of all the products in the dataset. This graph was produced using a technique that targets brand-new clients. This implies that new customers are more likely to purchase a product the more well-liked it is.

Recommendation system part II: Model-based collaborative filtering system based on customer's purchase history and ratings provided by other users who bought items similar items

We used SVD to implement this feature using customer purchase history.

The Singular Value Decomposition(SVD)

The Singular Value Decomposition (SVD) of a matrix is a factorization of that matrix into three matrices. It has some interesting algebraic properties and conveys important geo-

metrical and theoretical insights about linear transformations. It also has some important applications in data science. In this article, we will explain the mathematical intuition behind SVD and its geometrical meaning.

- Recommend items to users based on purchase history and similarity of ratings provided by other users who bought items to that of a particular customer.
- A model based collaborative filtering technique is chosen here as it helps in making predicting products for a particular user by identifying patterns based on preferences from multiple user data.

Recommendation system part III: When a business is setting up its E-Commerce website for the first time without any product rating

For a business without any user-item purchase history, a search engine based recommendation system can be designed for users. The product recommendations can be based on textual clustering analysis given in product description. We used k-means to implement this feature.

3.3.7 K-Means Algorithm

K-Means Clustering [4] is an Unsupervised Learning algorithm, which groups the unlabeled dataset into different clusters. Here K defines the number of pre-defined clusters that need to be created in the process, as if K=2, there will be two clusters, and for K=3, there will be three clusters, and so on.

It is an iterative algorithm that divides the unlabeled dataset into k different clusters in such a way that each dataset belongs only one group that has similar properties. It allows us to cluster the data into different groups and a convenient way to discover the categories of groups in the unlabeled dataset on its own without the need for any training.

It is a centroid-based algorithm, where each cluster is associated with a centroid. The main aim of this algorithm is to minimize the sum of distances between the data point and their corresponding clusters.

The algorithm takes the unlabeled dataset as input, divides the dataset into k-number of clusters, and repeats the process until it does not find the best clusters. The value of k should be predetermined in this algorithm.

The k-means clustering algorithm mainly performs two tasks:

Determines the best value for K center points or centroids by an iterative process. Assigns

each data point to its closest k-center. Those data points which are near to the particular k-center, create a cluster. Hence each cluster has datapoints with some commonalities, and it is away from other clusters.

Algorithm

1. We randomly pick K (centroids). We name them c1,c2,..... ck, and we can say that ,

$$\boxed{C = c_1, c_2, ..., c_k} \tag{3.1}$$

   Where C is the set of all centroids.

2. We assign each data point to its nearest center, which is accomplished by calculating the euclidean distance. Select random K points or centroids. (It can be other from the input dataset).

$$\boxed{argmin_{c_i \in C} dist(c_i, x^2)} \tag{3.2}$$

   Where dist() is the Euclidean distance. Here, we calculate each x value's distance from each c value, i.e. the distance between x1-c1, x1-c2, x1-c3and so on. Then we find which is the lowest value and assign x1 to that particular centroid. Similarly, we find the minimum distance for x2, x3, etc.

3. We identify the actual centroid by taking the average of all the points assigned to that cluster.

$$\boxed{c_i = \frac{1}{S_i} \sum_{\pi_i \in S_i} x_i} \tag{3.3}$$

   Where S_i is the set of all points assigned to the ith cluster. It means the original point, which we thought was the centroid, will shift to the new position, which is the actual centroid for each of these groups.

4. Keep repeating step 2 and step 3 until convergence is achieved.

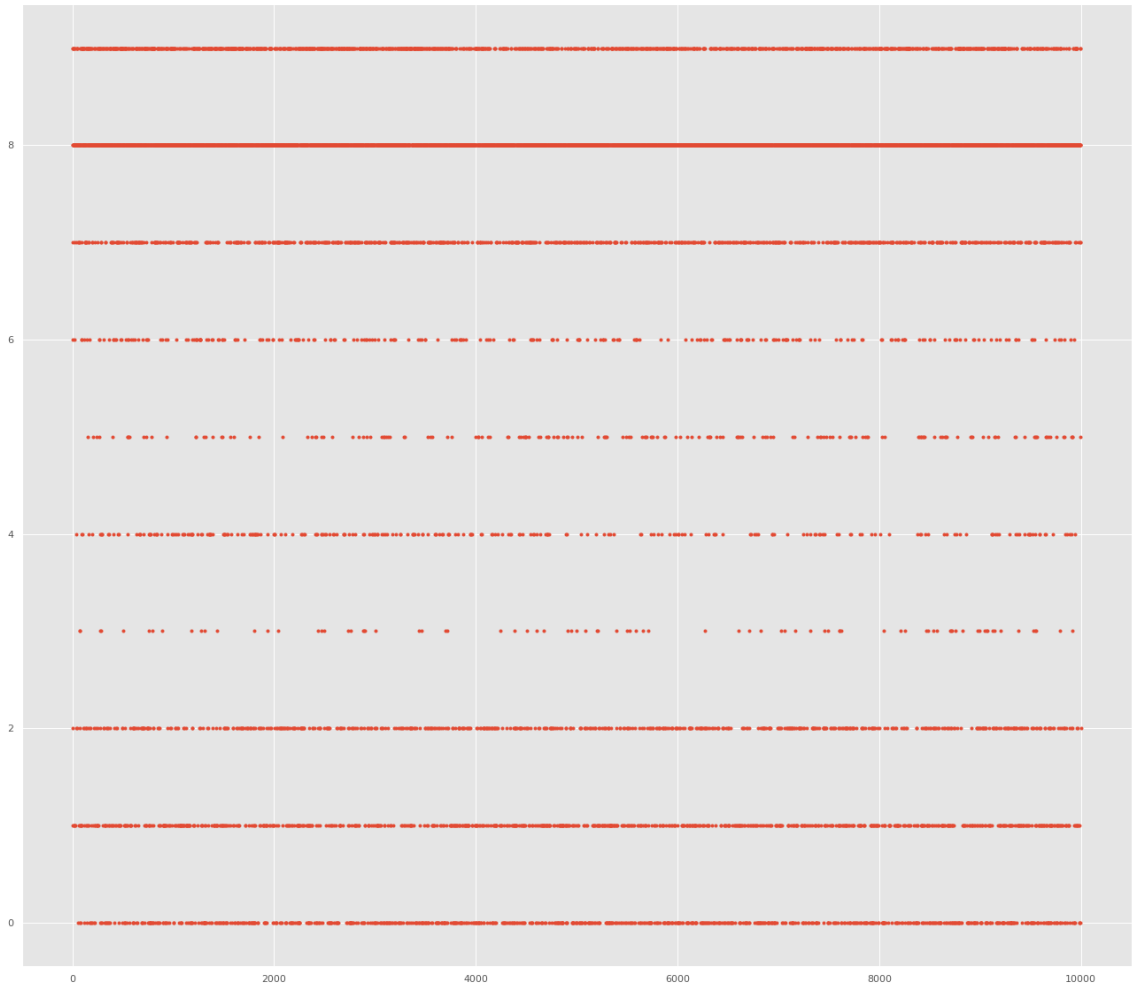

Figure 3.19: Visualizing Product Clusters in Subset of Data

3.4 Predicto

Sales forecasting is done using Monte Carlo simulation. Monte Carlo Simulations are used to model the probability of different outcomes in a process that cannot easily be predicted due to the intervention of random variables. It is a technique used to understand the impact of risk and uncertainty in prediction and forecasting models.

A Monte Carlo Simulation can be used to tackle a range of problems in virtually every field such as finance, engineering, supply chain, and science. It is also referred to as a multiple probability simulation.

3.4.1 Monte Carlo

The basis of a Monte Carlo simulation [5] is that the probability of varying outcomes cannot be determined because of random variable interference. Therefore, a Monte Carlo Simulation focuses on constantly repeating random samples to achieve certain results.

A Monte Carlo Simulation takes the variable that has uncertainty and assigns it a random value. The model then run and a result is provided. This process is repeated again and again while assigning the variable in question with many different values. Once the simulation is complete, the results are averaged together to provide an estimate.

1. A Monte Carlo simulation is a model used to predict the probability of different outcomes when the intervention of random variables is present.
2. Monte Carlo simulations help to explain the impact of risk and uncertainty in prediction and forecasting models.
3. A variety of fields utilize Monte Carlo simulations, including finance, engineering, supply chain, and science.
4. The basis of a Monte Carlo simulation involves assigning multiple values to an uncertain variable to achieve multiple results and then averaging the results to obtain an estimate. Monte Carlo simulations assume perfectly efficient markets.

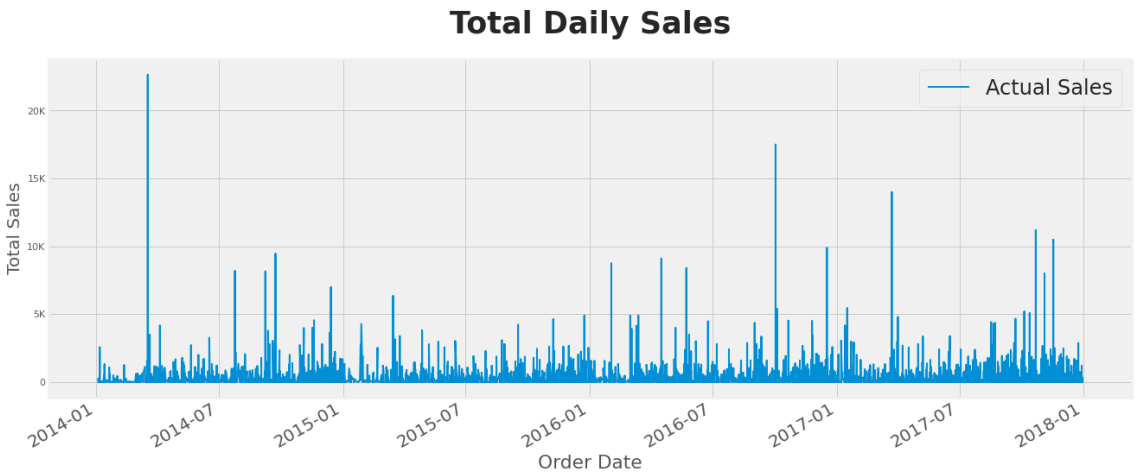

Figure 3.20: Total Daily Sales in 4 years Span

From given data in between a range years we forecast sales for next year. It shows daily sales data between 2014 and 2018 for a global superstore. For our purposes we will make some changes to the raw data to better fit the problem a little better.

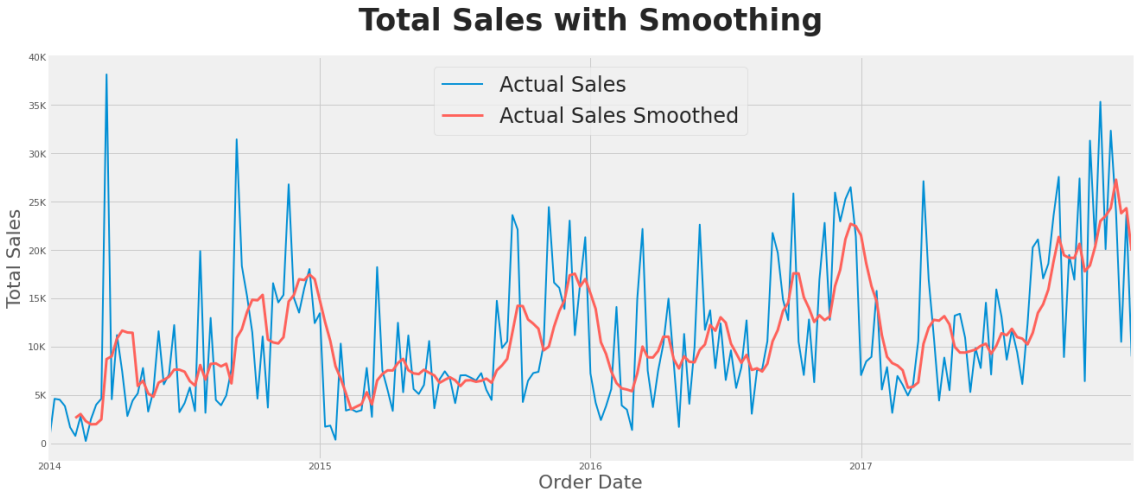

Figure 3.21: Total Daily Sales in 4 years Span Smoothed

Volatility is a measure of how much a value changes over time. The more volatile the more drastic individual changes can be. Models based on high volatility tend to be less reliable. There are big swings in daily sales between 0 and $25K.

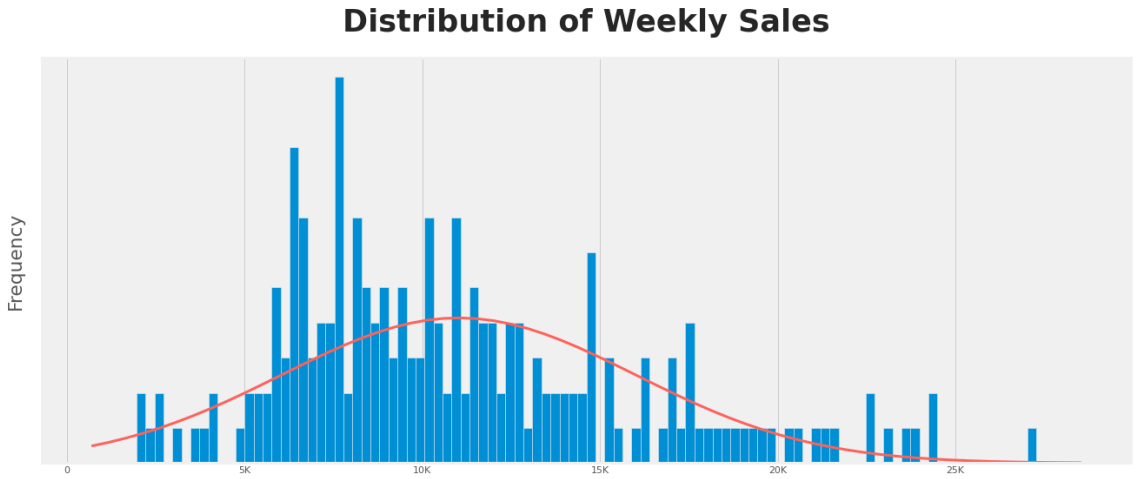

Figure 3.22: Distribution of Weekly Sales

For modelling purposes start off by looking at weekly sales. The weekly sales still have quite a bit of volatility. It can be dealt with by smoothing. The red line shows smoothing on a 6 week moving average. Looking at the individual week's smoothed sales, we can plot it on a standard distribution to get more insight into what they look like. For instance, when we look at the data set, the majority of the sales happen between $5K and $15K. There are sales that happen outside these ranges but not as many. It means we can assume that this is likely to be the case in the forecasting year as well. Based on how sales are likely to change over the year, we can simulate what could happen over the forecast year. Each of the weekly sales will be drawn from a probability distribution like the Distribution of Weekly Sales above. Numbers in the middle of the distribution (around the 10K mark) will be drawn more frequently than numbers on the sides.

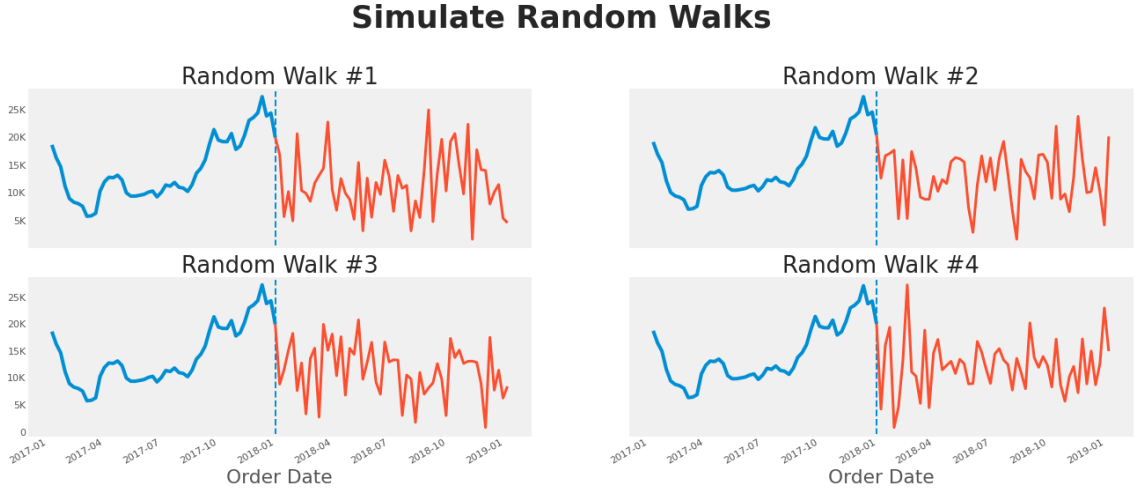

Figure 3.23: Simulating Random Walk 4 Times

Plotted on a graph, it gives you a unique path every time a simulation is run. In the Monte Carlo simulation we run this process many times in order to capture predictable randomness.

Each path is not predictable but all paths are. Above is shown simulation of random walk 4 times.

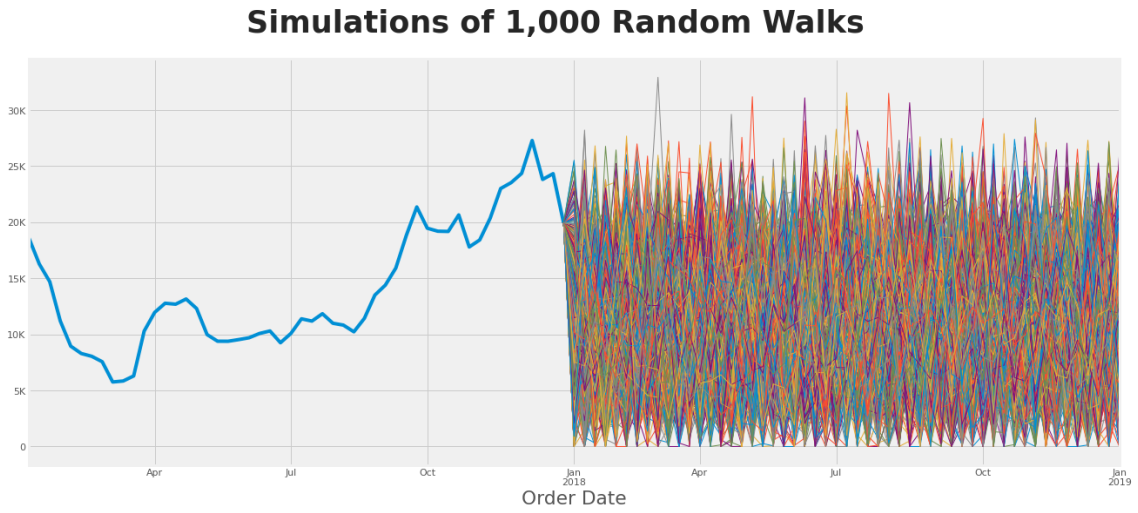

Figure 3.24: Simulating Random Walk 1000 Times on Actual Sales Data

Like before, above is shown simulation of random walk 1000 times and it is applied on an actual sales data. Every 1000 path is unique and it helps to capture us the predictable randomness.

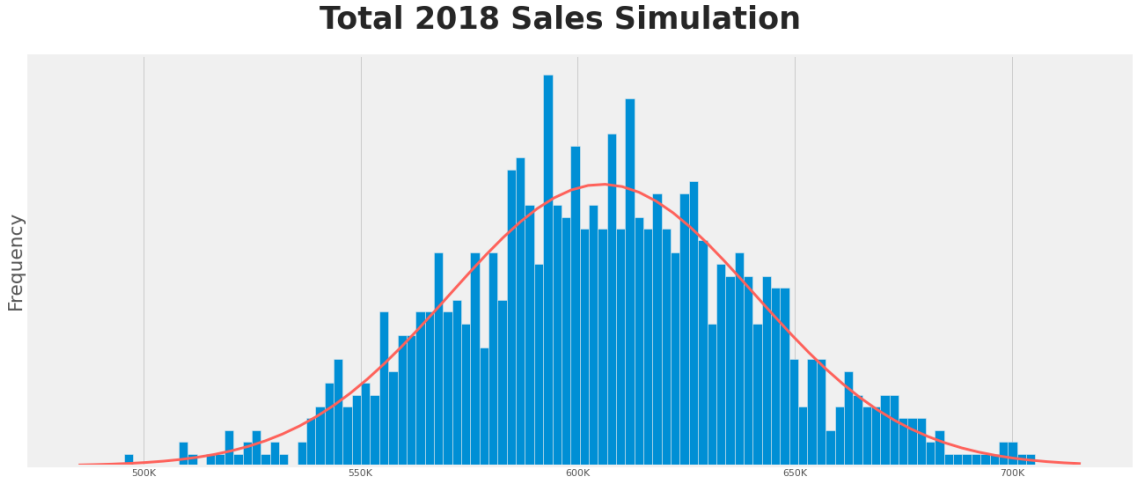

Figure 3.25: Simulation of 2018 Sales Data

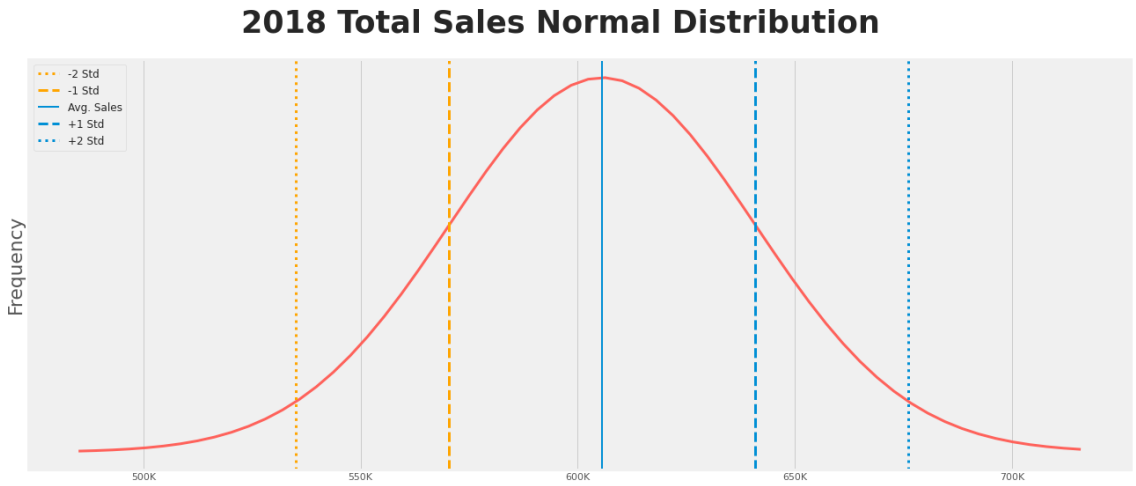

Figure 3.26: Normal Distribution of 2018 Sales Data

From the both graphs, we can conclude that it's highly likely that sales will be between $530,931 and $678,475. There's a 25% chance that sales will be less than 579,824. For instance, there is a 25% chance that sales will be below a particular amount. If the chance of this happening is too high, then it needs to be looked at.

Chapter 4

Analysis And Design

4.1 Requirement Analysis

4.1.1 Functional Requirements:

Table 4.1: Login/Access Account

Requirement No.	Functional Requirements
FR01-01	The system shall allow the users to register an account and login to their account with matching their Email ID and password.
FR01-02	The system shall enable users to edit their login information and passwords and retrieve passwords if being forgotten.
FR01-03	The user can upload profile picture also.

Table 4.2: User Account

Requirement No.	Functional Requirements
FR02-01	The system will have individual user profile which will contain their name, email, phone number, gender, age and profile picture etc.
FR02-02	The user can edit their profile information and save them.

Table 4.3: Administrator Account

Requirement No.	Functional Requirements
FR03-01	The system will have individual administrator profile which will contain their name, email, phone number, gender, age and profile picture etc.
FR03-02	The administrator can edit their certain profile information and save them.

Table 4.4: Search Option in Natural Language for Users

Requirement No.	Functional Requirements
FR04-01	Users can search in natural language for specific items along with their prices and delivery information etc.
FR04-02	The search result will show in BOT the asked query result in a natural language form.

Table 4.5: Viewing Top Products and Top Deals

Requirement No.	Functional Requirements
FR05-01	Users can view top products or top deals by typing in the BOT in their own way.
FR05-02	If they click on the top products, they can view all the details of the top products.
FR05-03	If they click on the top deals, they can view all the details of the top deals of that day.

4.1.2 Non-Functional Requirements

Table 4.6: Performance

Requirement No.	Functional Requirements
NFR01-01	Average load time of the BOT must be less than 2 seconds.
NFR01-02	The log in information shall be verified within five seconds.
NFR01-03	Queries shall return results within five seconds.
NFR01-04	The BOT should be able to do smooth searching.

Table 4.7: Security

Requirement No.	Functional Requirements
NFR02-01	Database connection close. After data use the database connection will close.
NFR02-02	No general user can get access to update the database.
NFR02-03	The users should get up to date information and emergency contact from the system.

Table 4.8: Portability

Requirement No.	Functional Requirements
NFR03-01	System must run on any device of windows based operating system ; And from any web browser also.

Table 4.9: User Friendly Interface

Requirement No.	Functional Requirements
NFR04-01	The system should have a decent interface and should provide good experience to the users.

Table 4.10: Defects-Maintenance

Requirement No.	Functional Requirements
NFR05-01	Post Release defects of the system must not exceed 1 critical bug per month.
NFR05-02	Post Release bug fixing should not take more than 5 hours.

4.2 Functional Analysis

Functional analysis is a methodology for analyzing the mission and performance requirements of a system, and translating them into discrete activities or tasks which must be performed by the system. Determination of the system's functionality is the first step in obtaining a conceptual view of a system that is to be designed.

4.2.1 ShopBot

Use-Case Diagram

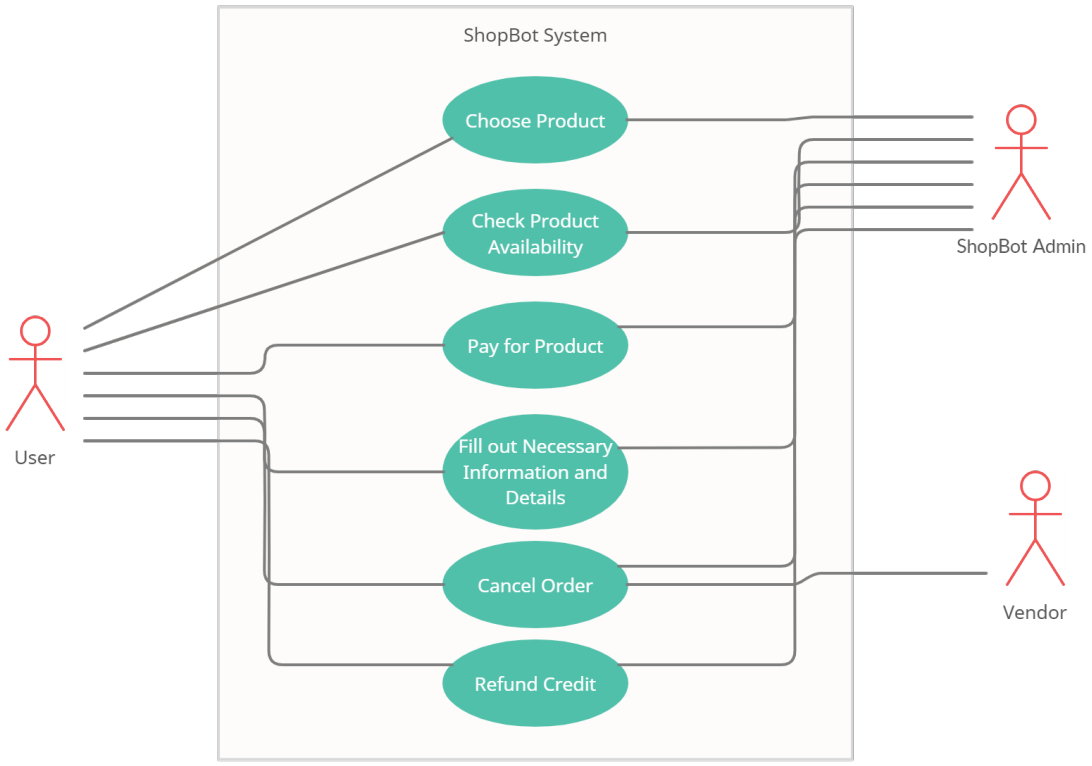

Figure 4.1: Use-Case Diagram for ShopBot

4.2.2 QueryBot

Use-Case Diagram

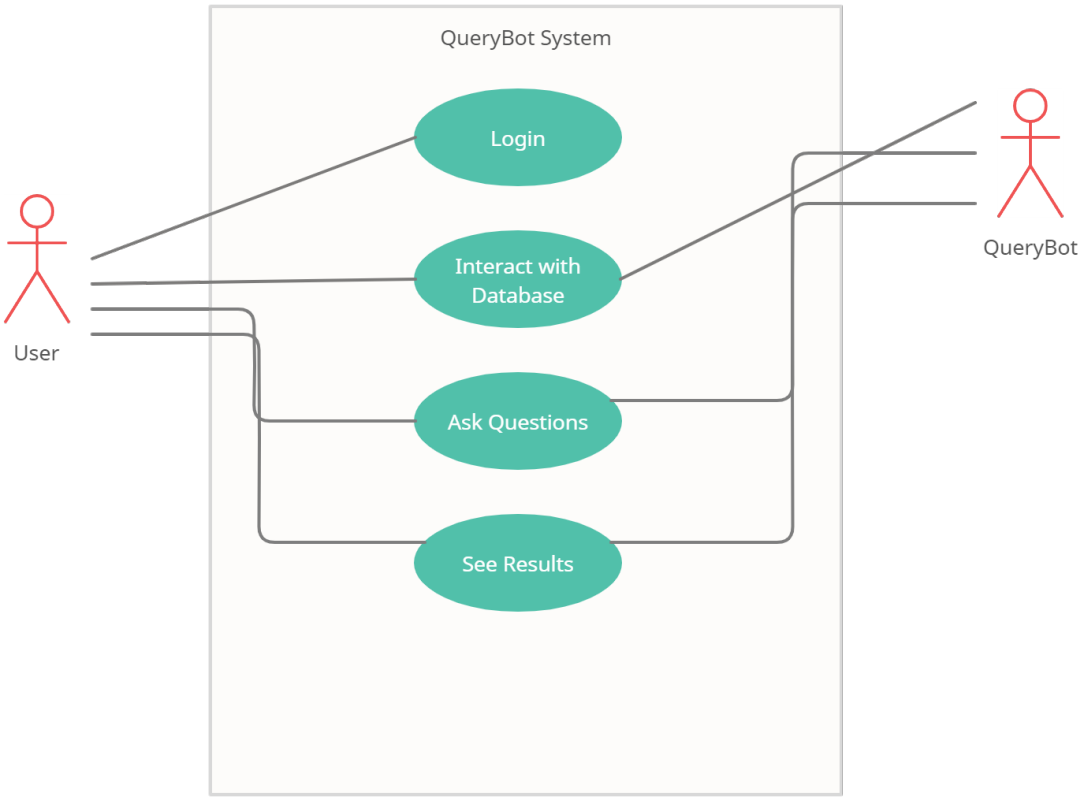

Figure 4.2: Use-Case Diagram for QueryBot

4.3 Dynamic Analysis

Dynamic analysis is the testing and evaluation of a program by executing data in real-time. The objective is to find errors in a program while it is running, rather than by repeatedly examining the code offline. By debugging a program in all the scenarios for which it is designed, dynamic analysis eliminates the need to artificially create situations likely to produce errors. Other advantages include reducing the cost of testing and maintenance, identifying and eliminating unnecessary program components.

Activity Diagram

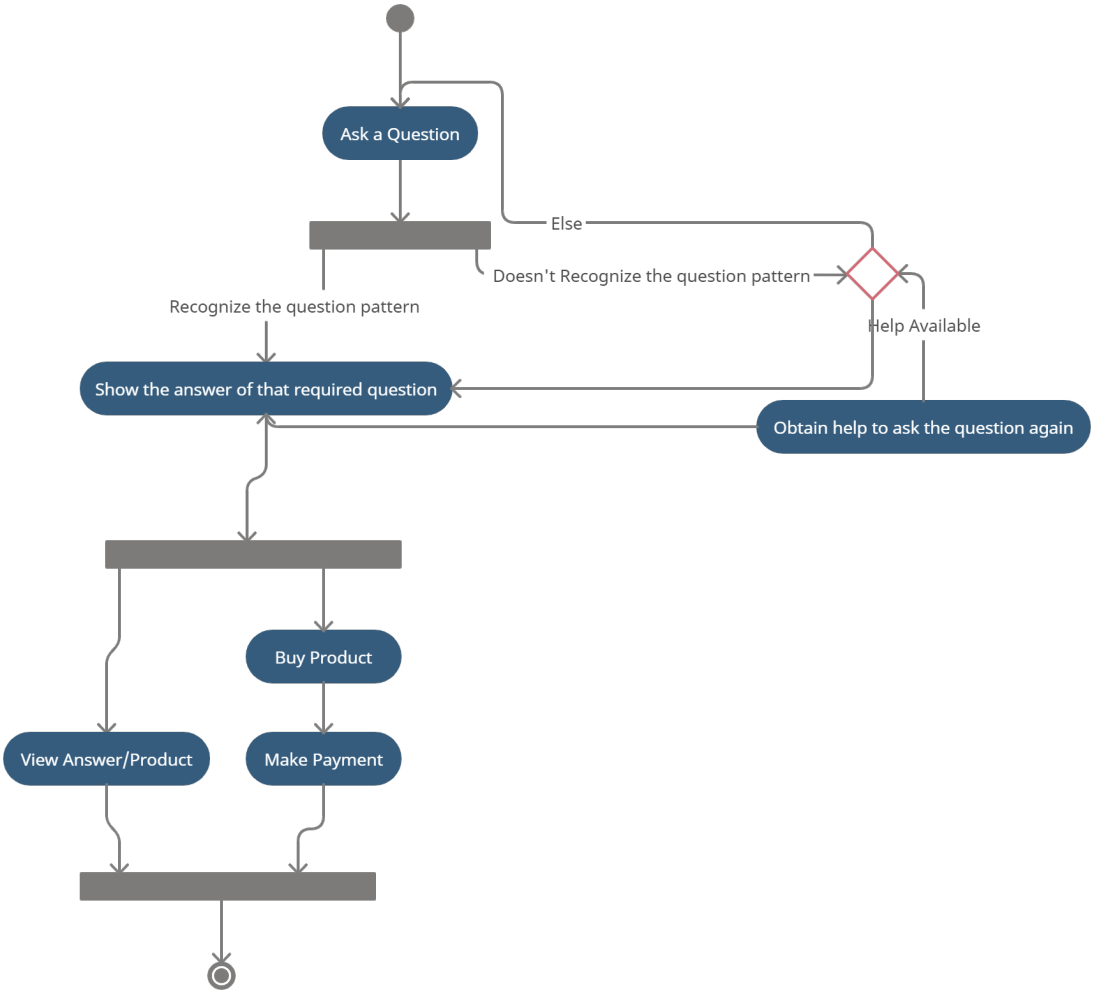

Figure 4.3: Activity Diagram for ShopBot

Activity Diagram

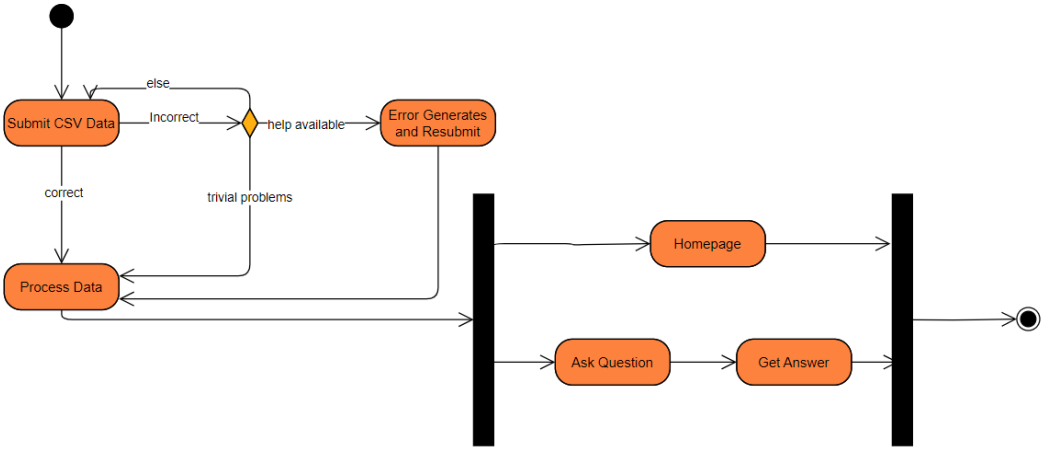

Figure 4.4: Activity Diagram for QueryBot

4.4 Risk Analysis

Risk analysis is part of every decision we make. We are constantly faced with uncertainty, ambiguity, and variability. And even though we have unprecedented access to information, we can't accurately predict the future. Monte Carlo Simulation (also known as the Monte Carlo Method) lets one see all the possible outcomes of one's decisions and assess the impact of risk, allowing for better decision making under uncertainty.

4.4.1 Monte Carlo Simulation

Monte Carlo Simulation is a computerized mathematical technique that allows people to account for risk in quantitative analysis and decision making. The technique is used by professionals in such widely disparate fields as finance, project management, energy, manufacturing, engineering, research and development, insurance, oil & gas, transportation, and the environment.

Monte Carlo simulation furnishes the decision-maker with a range of possible outcomes and the probabilities they will occur for any choice of action. It shows the extreme possibilities–the outcomes of going for broke and for the most conservative decision–along with all possible consequences for middle-of-the-road decisions.

The technique was first used by scientists working on the atom bomb; it was named for Monte Carlo, the Monaco resort town renowned for its casinos. Since its introduction in

World War II, Monte Carlo simulation has been used to model a variety of physical and conceptual systems.

4.4.2 How Monte Carlo Works

Monte Carlo simulation performs risk analysis by building models of possible results by substituting a range of values–a probability distribution–for any factor that has inherent uncertainty. It then calculates results over and over, each time using a different set of random values from the probability functions. Depending upon the number of uncertainties and the ranges specified for them, a Monte Carlo Simulation could involve thousands or tens of thousands of recalculations before it is complete. Monte Carlo simulation produces distributions of possible outcome values.

By using probability distributions, variables can have different probabilities of different outcomes occurring. Probability distributions are a much more realistic way of describing uncertainty in variables of a risk analysis.

Monte Carlo simulation provides a number of advantages over deterministic, or "single-point estimate" analysis:

1. **Probabilistic Results:** Results show not only what could happen, but how likely each outcome is.
2. **Graphical Results:** Because of the data a Monte Carlo simulation generates, it's easy to create graphs of different outcomes and their chances of occurrence. This is important for communicating findings to other stakeholders.
3. **Sensitivity Analysis:** With just a few cases, deterministic analysis makes it difficult to see which variables impact the outcome the most. In Monte Carlo simulation, it's easy to see which inputs had the biggest effect on bottom-line results.
4. **Scenario Analysis:** In deterministic models, it's very difficult to model different combinations of values for different inputs to see the effects of truly different scenarios. Using Monte Carlo simulation, analysts can see exactly which inputs had which values together when certain outcomes occurred. This is invaluable for pursuing further analysis.
5. **Correlation of Inputs:** In Monte Carlo simulation, it's possible to model interdependent relationships between input variables. It's important for accuracy to represent how, in reality, when some factors goes up, others go up or down accordingly.

Chapter 5

Development And Testing

5.1 Frameworks and Tools

5.1.1 Django

Django [6] is a high-level Python web framework that enables rapid development of secure and maintainable websites. Built by experienced developers, Django takes care of much of the hassle of web development, so we can focus on writing our app without needing to reinvent the wheel. It is free and open source, has a thriving and active community, great documentation, and many options for free and paid-for support. We used django to built a demo e-commerce site to demonstrate how ShopBot works with realtime database.

5.1.2 React.js

React.js [7] was released by a software engineer working for Facebook - Jordane Walke in 2011. React is a JavaScript library focused on creating declarative user interfaces (UIs) using a component-based concept. It's used for handling the view layer and can be used for web and mobile apps. React's main goal is to be extensive, fast, declarative, flexible, and simple.

React is not a framework, it is specifically a library. The explanation for this is that React only deals with rendering UIs and reserves many things at the discretion of individual projects. The standard set of tools for creating an application using ReactJS is frequently called the stack.

Our base website reportbyte is built using react.

5.1.3 Node.js

Node.js [8] is an open-source, and a cross-platform runtime environment for developing server-side and networking applications. Node.js applications are written in JavaScript, and can be run within the Node.js runtime on OS X, Microsoft Windows and Linux. Node.js also provides a rich library of various JavaScript modules which simplifies the development of web applications using Node.js to a great extent.
The following are some of the important features that make Node.js the first choice of software architects. We used node.js as backend framework for Reportbyte's QueryBot service website.

1. Asynchronous and Event-Driven
2. Very Fast
3. Single-Threaded but Highly Scalable
4. No Buffering
5. Node.js is released under the MIT license

5.1.4 Express.js

ExpressJS [9] is a web application framework that provides us with a simple API to build websites, web apps, and back ends. With ExpressJS, we need not worry about low-level protocols, processes, etc. Express provides a minimal interface to build our applications. It provides us the tools that are required to build our app. It is flexible as there are numerous modules available on npm, which can be directly plugged into Express. Express was developed by TJ Holowaychuk [10] and is maintained by the Node.js Foundation and numerous open source contributors. We used express.js as backend framework for Reportbyte's QueryBot service website.

5.1.5 Mongodb

MongoDB [11] is a cross-platform, document-oriented database that provides, high performance, high availability, and easy scalability. MongoDB works on the concept of collection and document. We used mongdodb Reportbyte's QueryBot service website.

1. **Database:** The database is a physical container for collections. Each database gets its own set of files on the file system. A single MongoDB server typically has multiple databases.

2. **Collection:** A collection is a group of MongoDB documents. It is the equivalent of an RDBMS table. A collection exists within a single database. Collections do not enforce a schema. Documents within a collection can have different fields. Typically, all documents in a collection are of similar or related purposes.

3. **Document:** A document is a set of key-value pairs. Documents have a dynamic schema. Dynamic schema means that documents in the same collection do not need to have the same set of fields or structures, and common fields in a collection's documents may hold different types of data.

5.1.6 Mongoose

Mongoose is an Object Data Modeling (ODM) library for MongoDB and NodeJS. It manages relationships between data, provides schema validation, and is used to translate between objects in code and the representation of those objects in MongoDB.

5.2 Programming Languages

5.2.1 Python

Python is an interpreted high-level general-purpose programming language. It's design philosophy emphasizes code readability with its use of significant indentation. It's language constructs as well as its object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects. Many of the frameworks and tools to complete this project is based on python.

5.2.2 Javascript

JavaScript, often abbreviated as JS, is a programming language that conforms to the ECMAScript specification. JavaScript is high-level, often just-in-time compiled and multi-paradigm. It has dynamic typing, prototype-based object-orientation and first-class functions. We used many javascript frameworks for web developing part of this project.

5.2.3 HTML

Hypertext Markup Language HTML is the standard markup language for documents designed to be displayed in a web browser. It can be assisted by technologies such as Cascading

Style Sheets (CSS) and scripting languages such as JavaScript.

5.2.4 CSS

Cascading Style Sheets CSS is a style sheet language used for describing the presentation of a document written in a markup language like HTML. CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript.

5.3 Development Environment

5.3.1 VS Code

Visual Studio Code is a source code editor developed by Microsoft for Windows, Linux, and macOS. It includes support for debugging, embedded Git control and GitHub, syntax highlighting, intelligent code completion, snippets, and code refactoring.

5.3.2 Jupyter Notebook

Project Jupyter is a project and community whose goal is to "develop open-source software, open-standards, and services for interactive computing across dozens of programming languages".

5.3.3 Anaconda

Anaconda is a distribution of the Python and R programming languages for scientific computing, that aims to simplify package management and deployment. The distribution includes data-science packages suitable for Windows, Linux, and macOS.

5.3.4 Google Colab

Colab notebooks allow us to combine executable code and rich text in a single document, along with images, HTML, LaTeX and more. When we create our own Colab notebooks, they are stored in our Google Drive account. We can easily share our Colab notebooks with co-workers or friends, allowing them to comment on our notebooks or even edit them.

5.3.5 Postman

Postman is an interactive and automatic tool for verifying the APIs of our project. Postman is a Google Chrome app for interacting with HTTP APIs. It presents us with a friendly GUI for constructing requests and reading responses. It works on the backend, and makes sure that each API is working as intended.

5.3.6 MongoDB Atlas

The core of MongoDB Cloud is MongoDB Atlas, a fully managed cloud database for modern applications. Atlas is the best way to run MongoDB [12], the leading modern database. MongoDB's document model is the fastest way to innovate, bringing flexibility and ease of use to the database.

5.3.7 Overleaf

Overleaf is a collaborative cloud-based LaTeX editor used for writing, editing and publishing scientific documents. It partners with a wide range of scientific publishers to provide official journal LaTeX templates and direct submission links.

5.4 Version Control

When building software, it's always important to track your changes. This is especially critical when collaborating on projects where multiple people will be updating the same code. Software that can keep track of all these changes are called Version Control.

5.4.1 Git

Git is a distributed revision control and source code management system that allows several people to work on the same codebase at the same time on different computers and networks. These can be pushed together, with all changes stored and recorded. It's also possible to roll back to an earlier state if necessary.

5.4.2 Github

At a high level, GitHub is a website and cloud-based service that helps developers store and manage their code, as well as track and control changes to their code.

Github Desktop

GitHub Desktop is a fast and easy way to contribute to projects, and is designed to simplify all processes and workflow in our GitHub.

5.5 Web Interface

Below is shown the front-end version of our project with necessary detailing and figures.

5.6 Shopbot

We have created a small E-Commerce site to demonstrate how shopbot works.

5.6.1 E-commerce Site (Demo)

We tried to show four major types of E-Commerce site ,

1. Grocery
2. Electronics
3. Furniture
4. Automobile

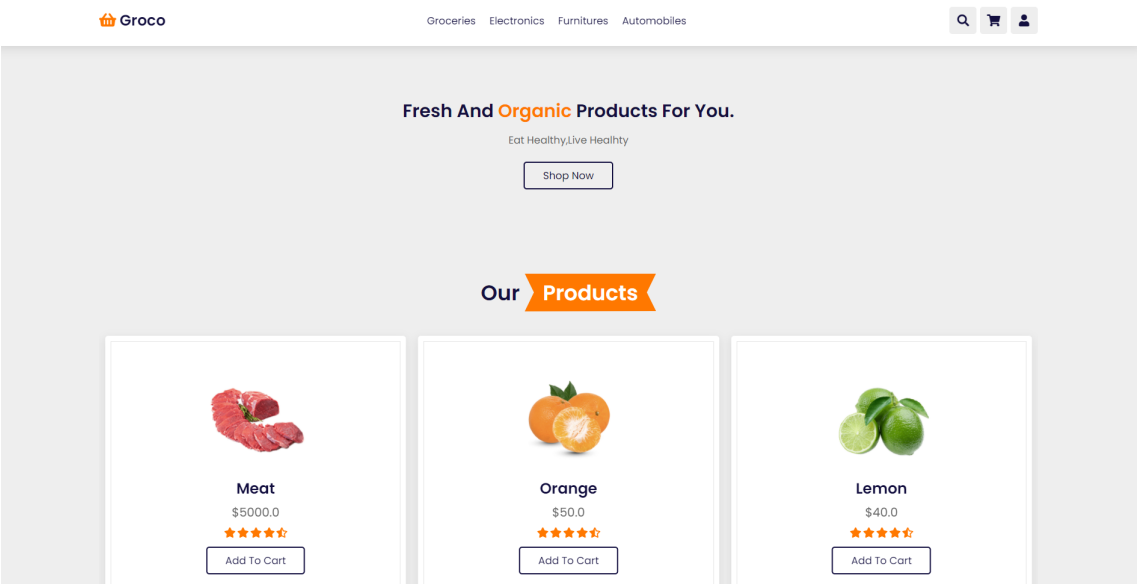

Figure 5.1: E-Commerce Site View (Grocery) Top View

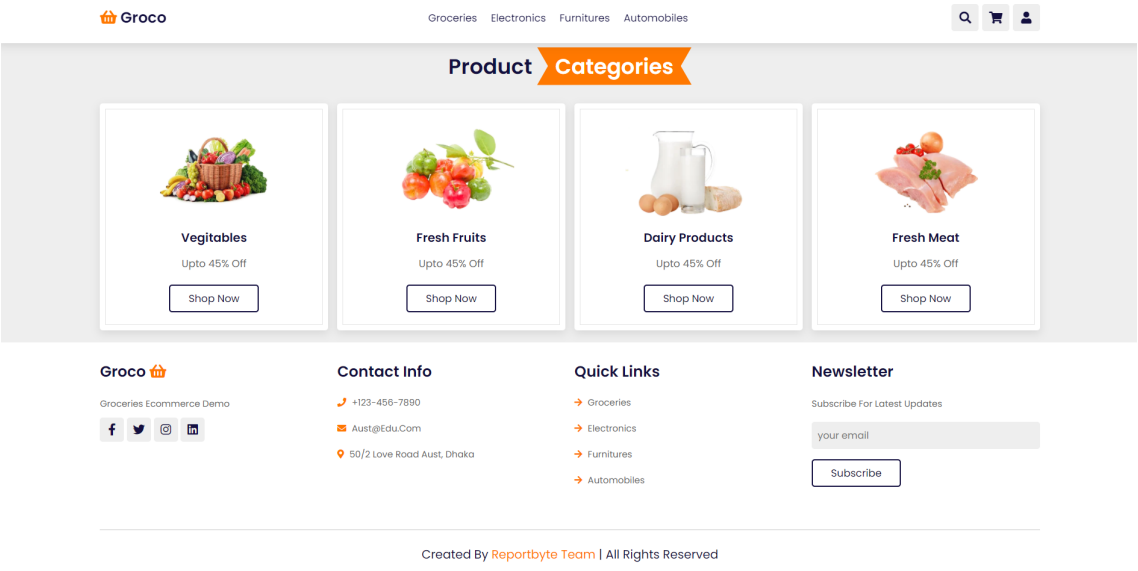

Figure 5.2: E-Commerce Site View (Grocery) Bottom View

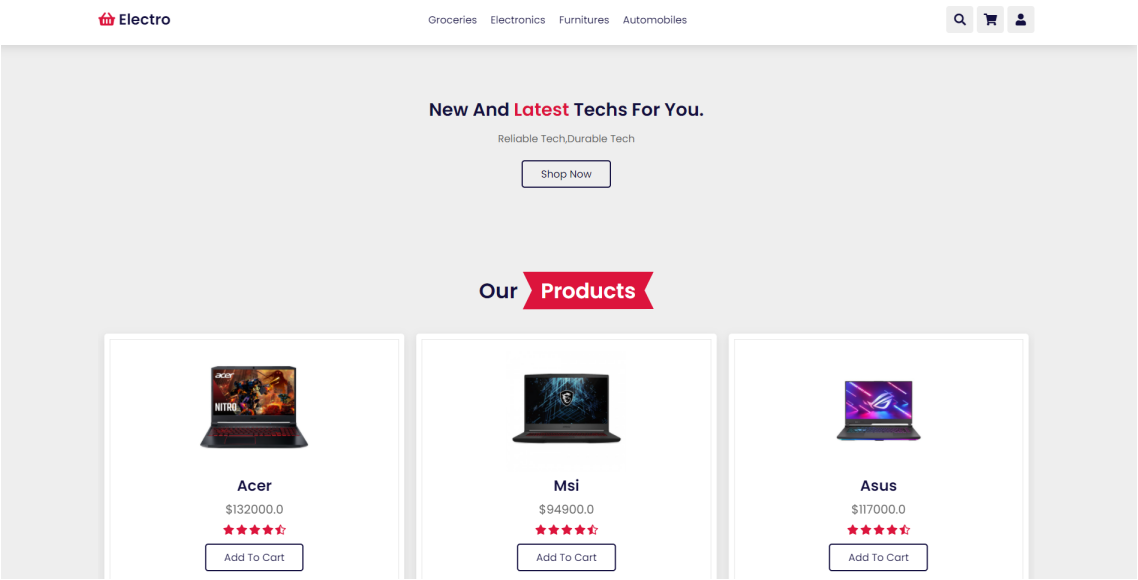

Figure 5.3: E-Commerce Site View (Electronics) Top View

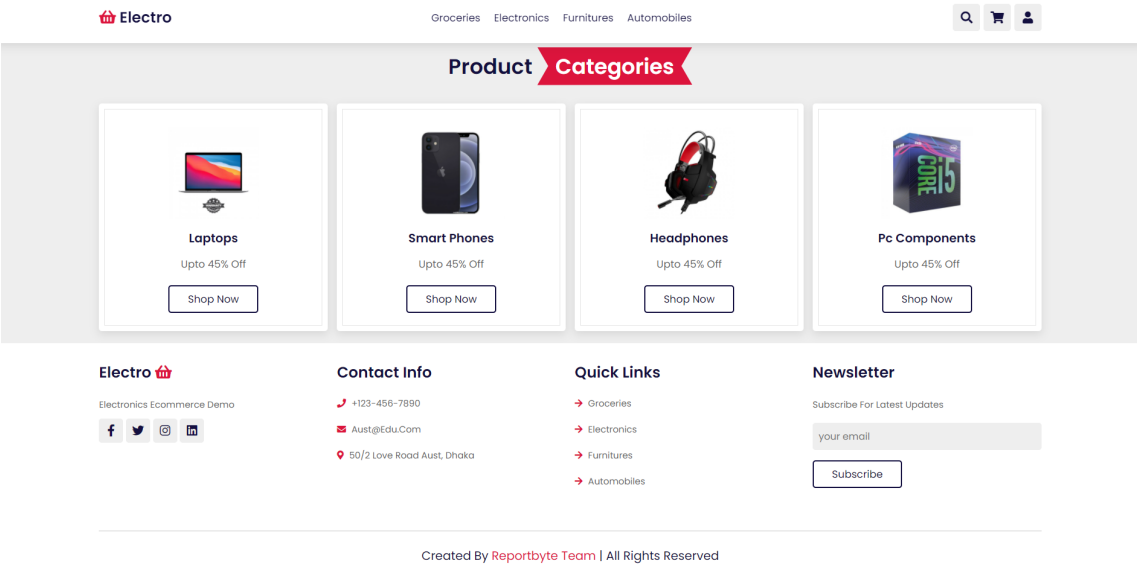

Figure 5.4: E-Commerce Site View (Electronics) Bottom View

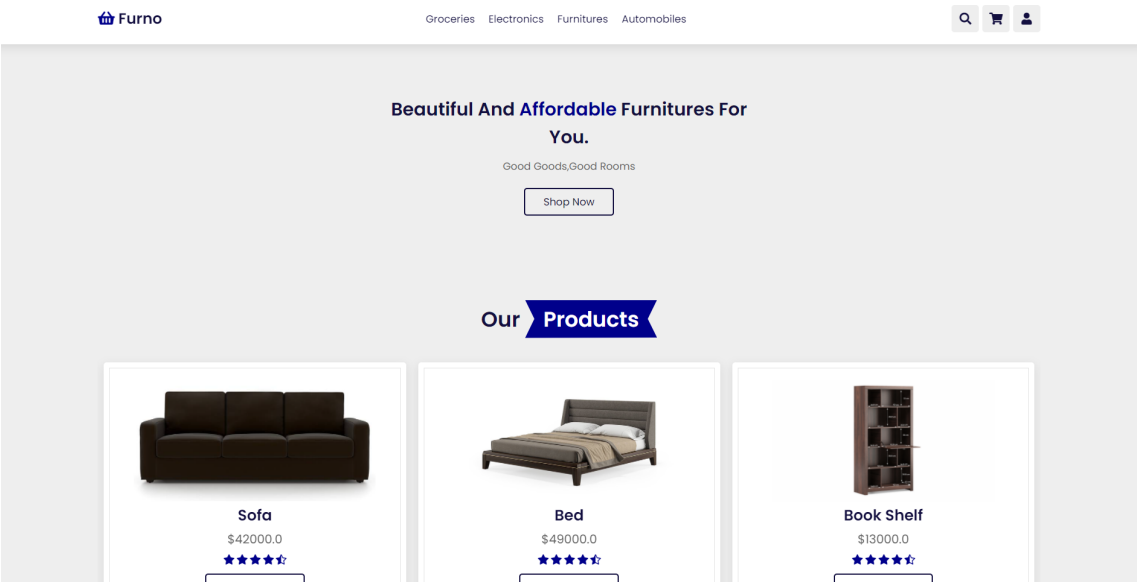

Figure 5.5: E-Commerce Site View (Furniture) Top View

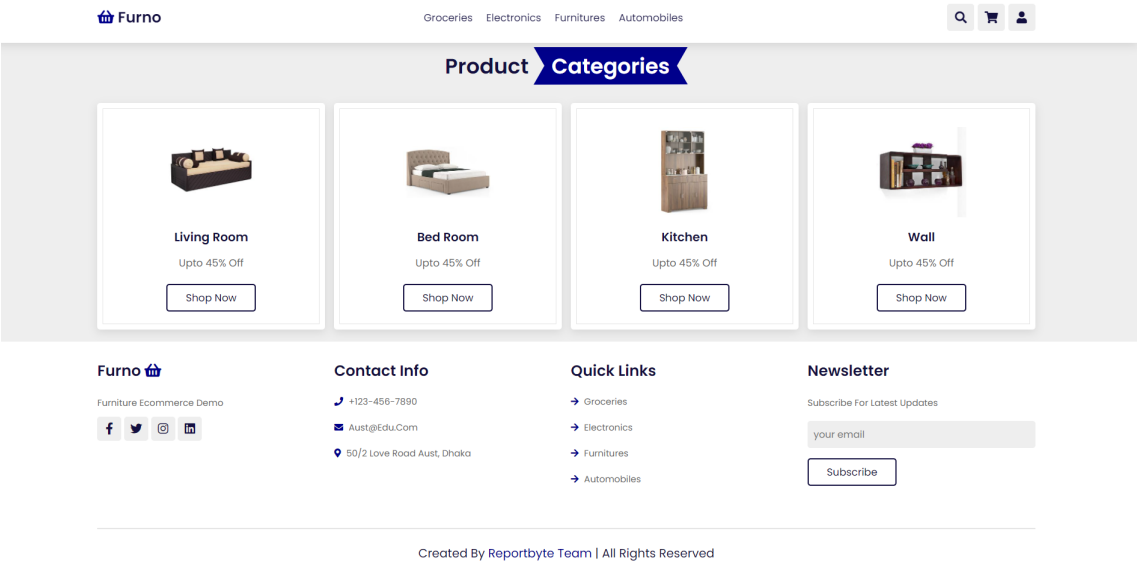

Figure 5.6: E-Commerce Site View (Furniture) Bottom View

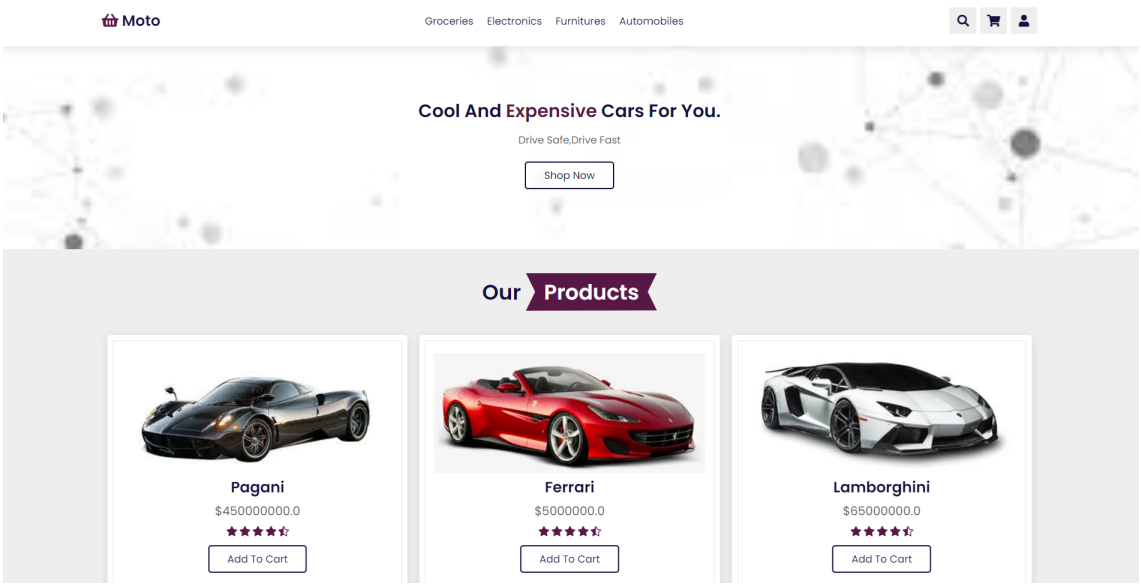

Figure 5.7: E-Commerce Site View (Automobile) Top View

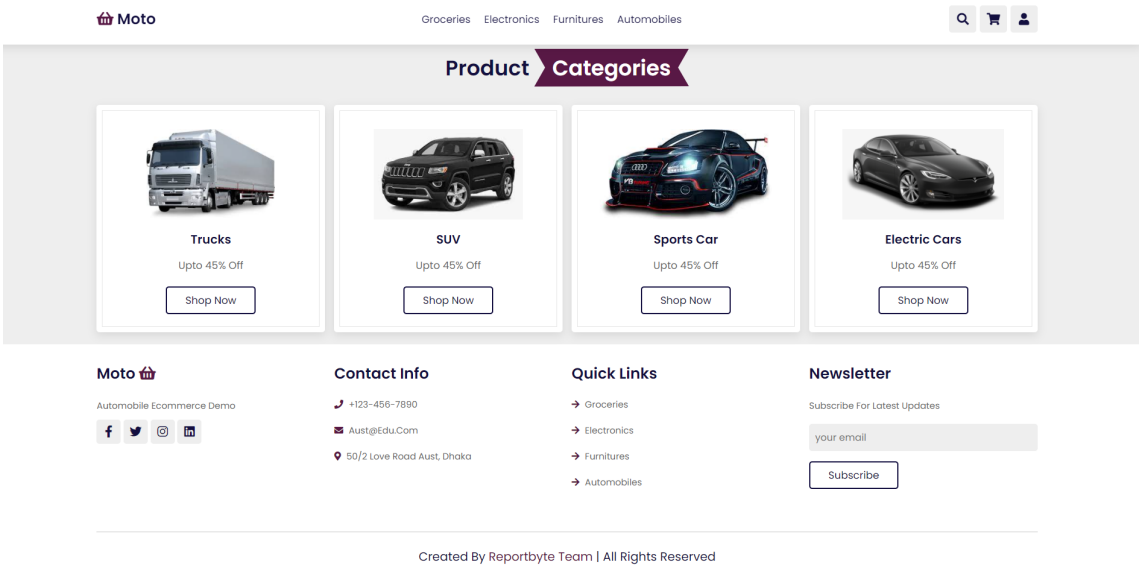

Figure 5.8: E-Commerce Site View (Automobile) Bottom View

5.6.2 Webchat Widget (Shopbot)

We implemented the bot with a chat widget which can be seen from Figure: 5.9 to Figure: 5.11

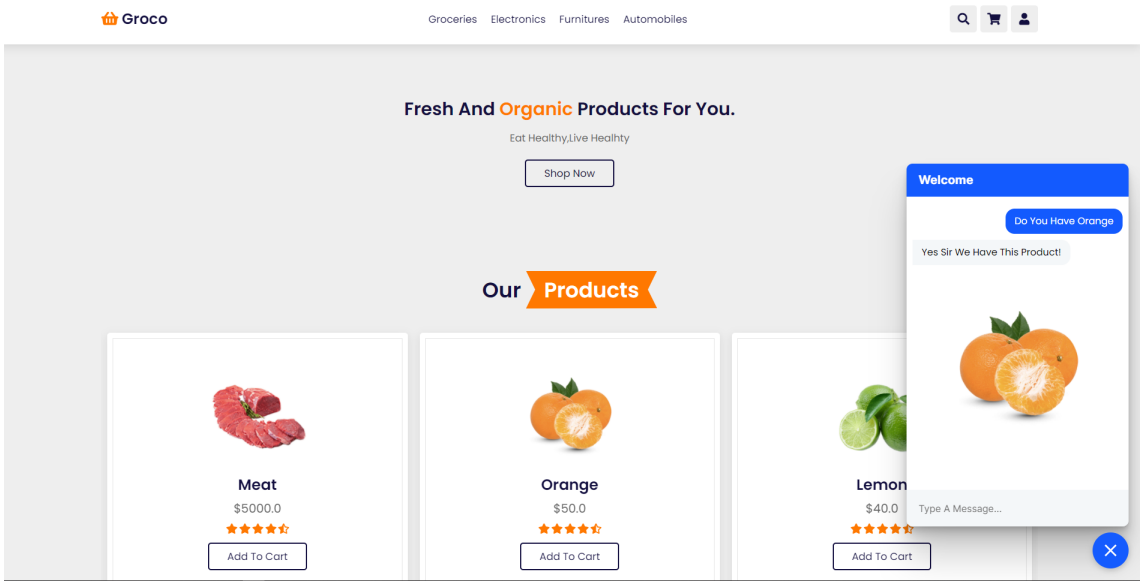

Figure 5.9: Showing Product Availability in Shopbot Chat

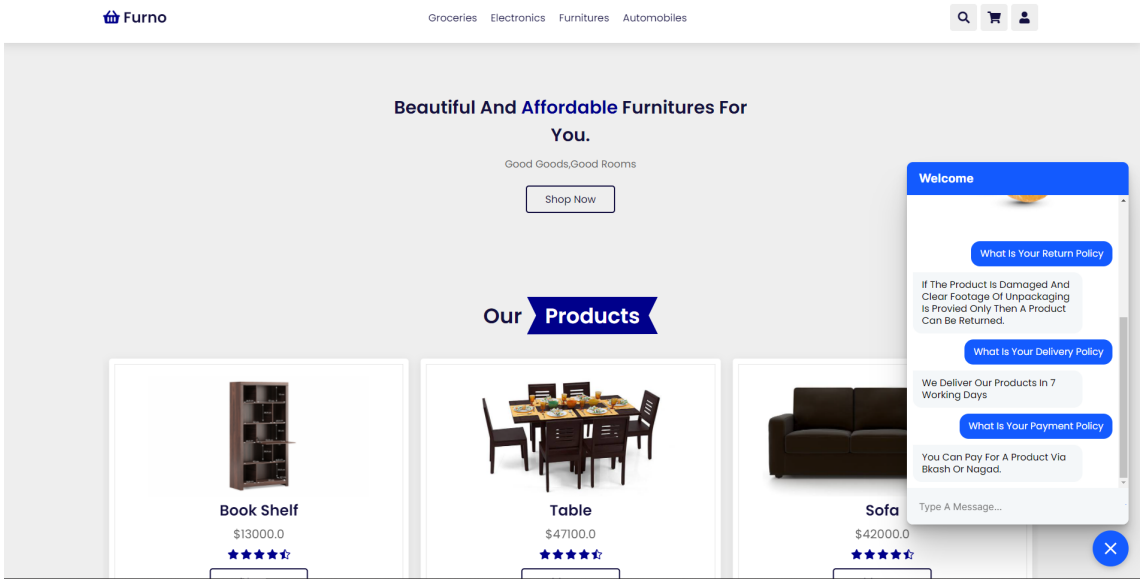

Figure 5.10: Answering Customer Questions with Shopbot

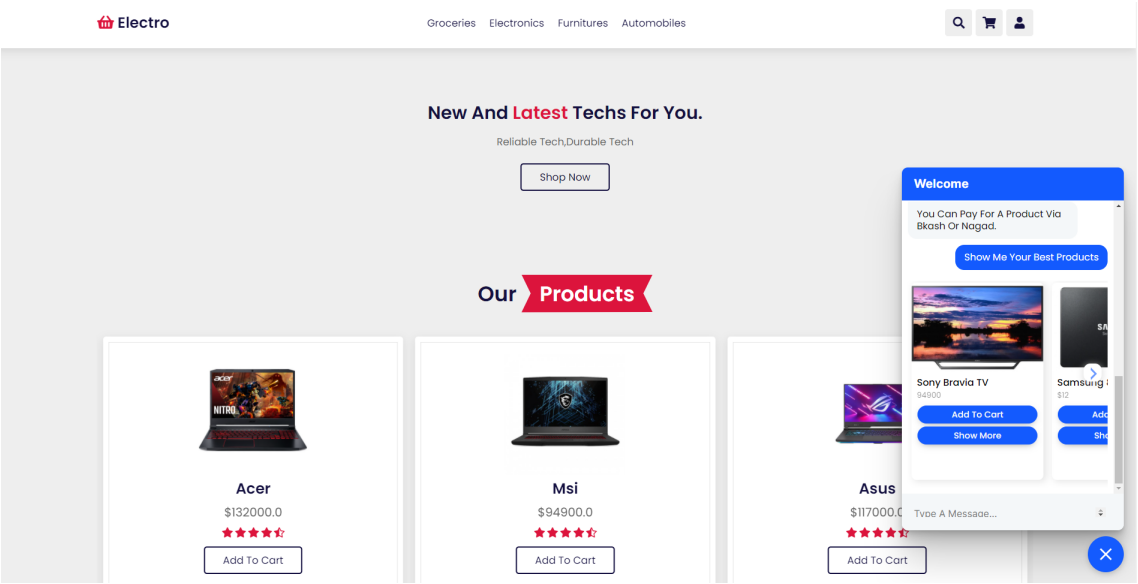

Figure 5.11: Using Shopbot to Display the Best Products in a Store

5.7 Querybot

5.7.1 Dashboard

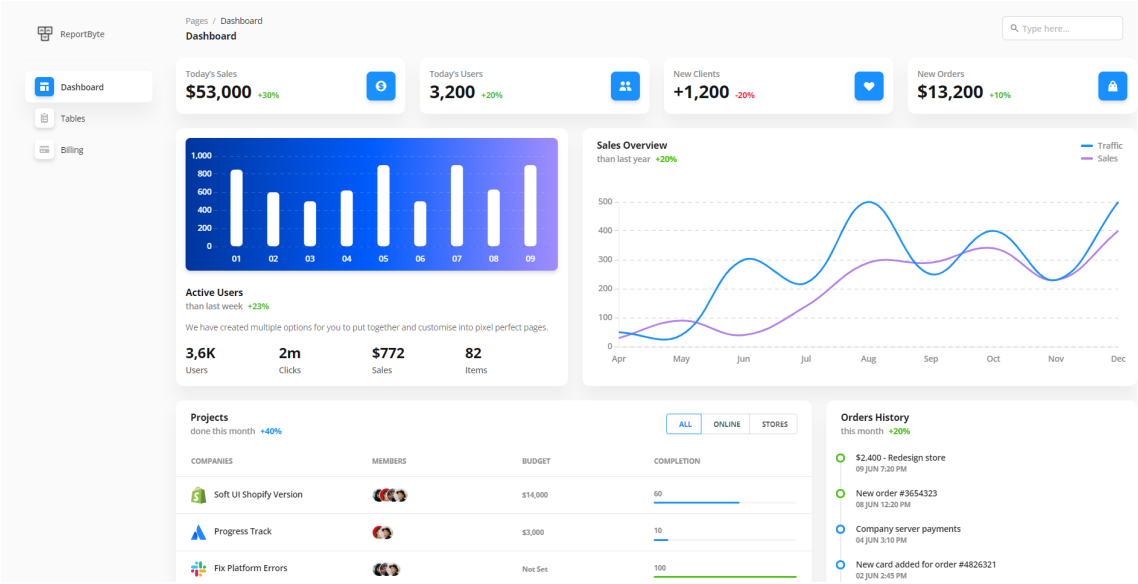

Figure 5.12: ReportByte Daashboard View

5.7.2 Querybot Input Area

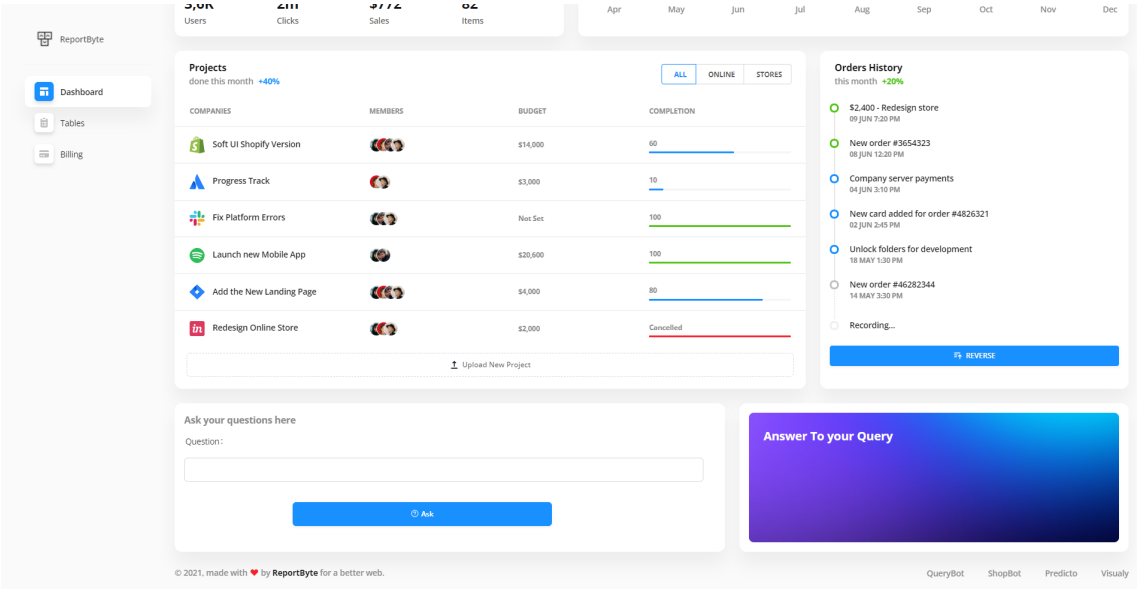

Figure 5.13: Querybot Input Section

5.8 Reportbyte

Main website where all the four features are implemented together.

5.8.1 Reportbyte: Landing Page

Home page will give overview of all the services reportbyte can offer. It will also represent the project timeline.

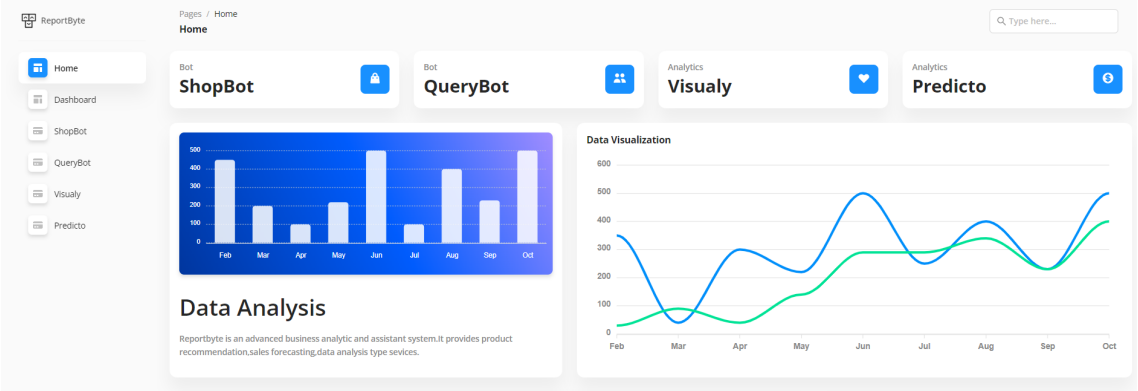

Figure 5.14: Landing Page

5.8.2 Reportbyte: Home Page

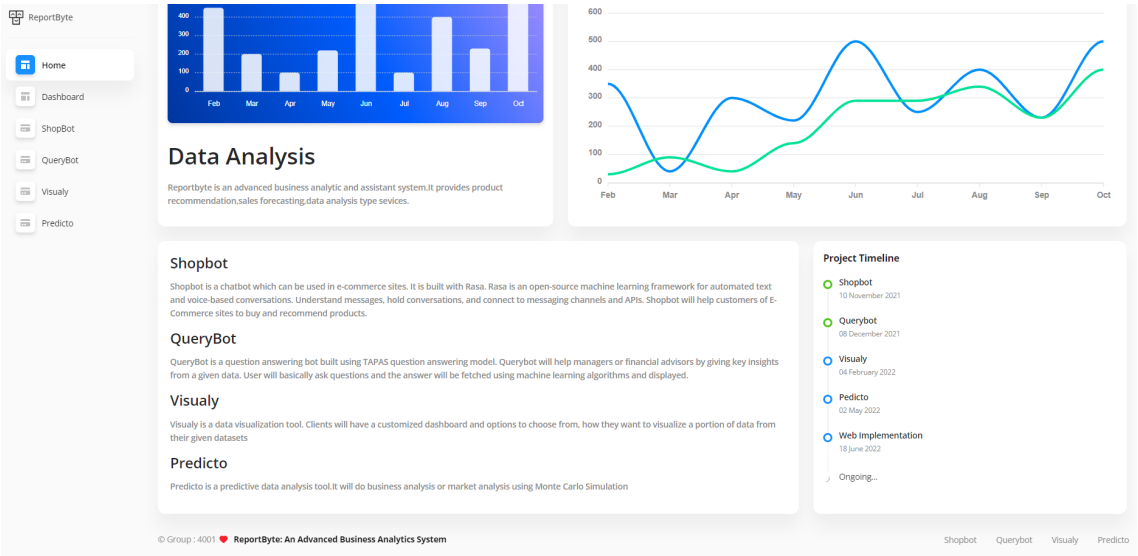

Figure 5.15: Homepage

5.8.3 Reportbyte: Shopbot feature

Since shopbot is used in client website we tried to show is demonstration video in main website.

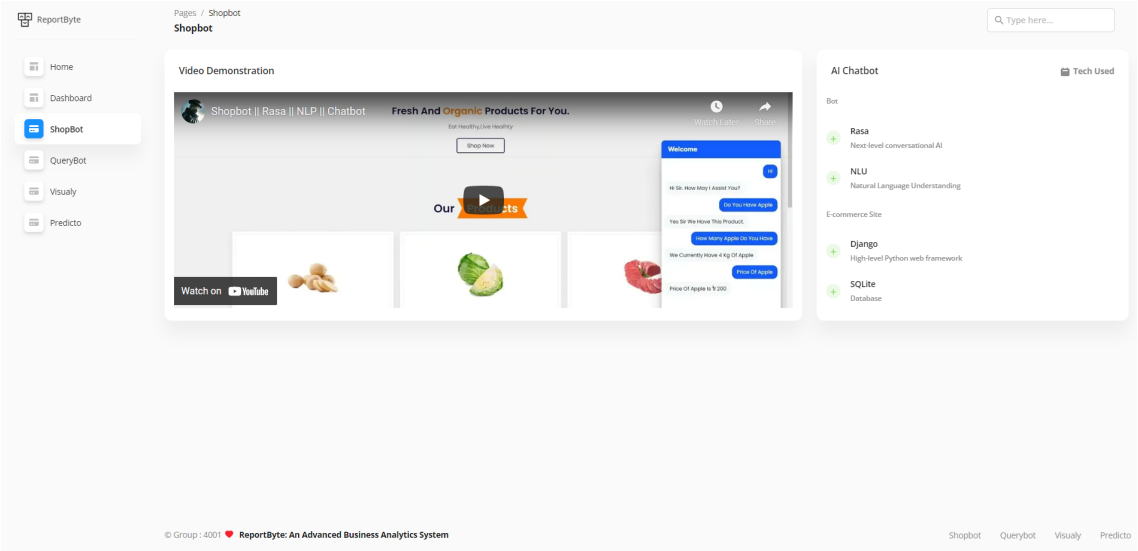

Figure 5.16: Feature Page (Shopbot)

5.8.4 Reportbyte: Querybot feature

Here we can see how a user can ask questions from data and get answers.

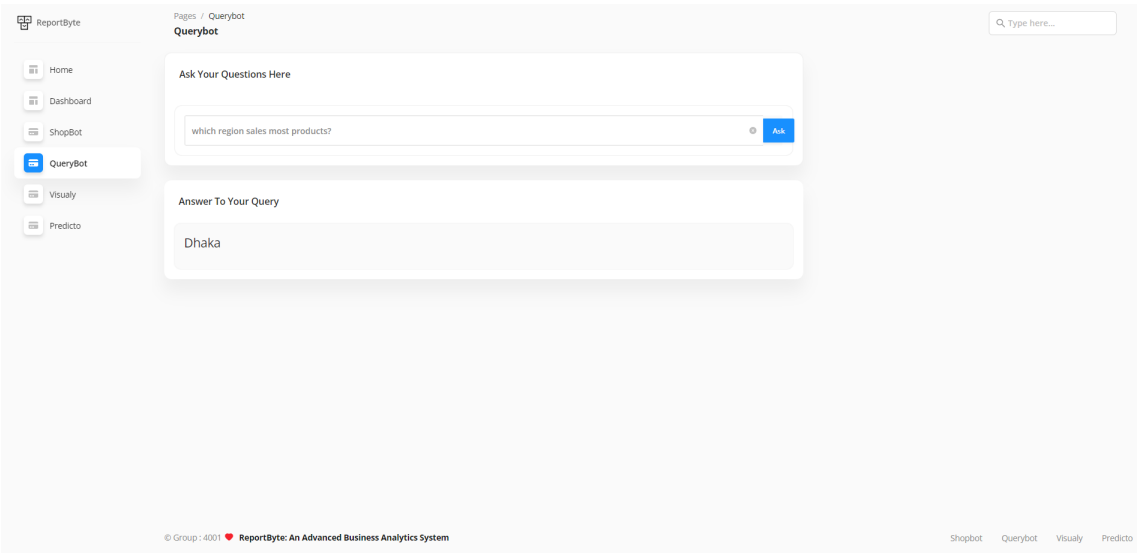

Figure 5.17: Feature Page (QueryBot)

5.8.5 Reportbyte: Visualy feature

This page will show important sales analysis and product recommendation reports which were generated using various machine learning algorithms.

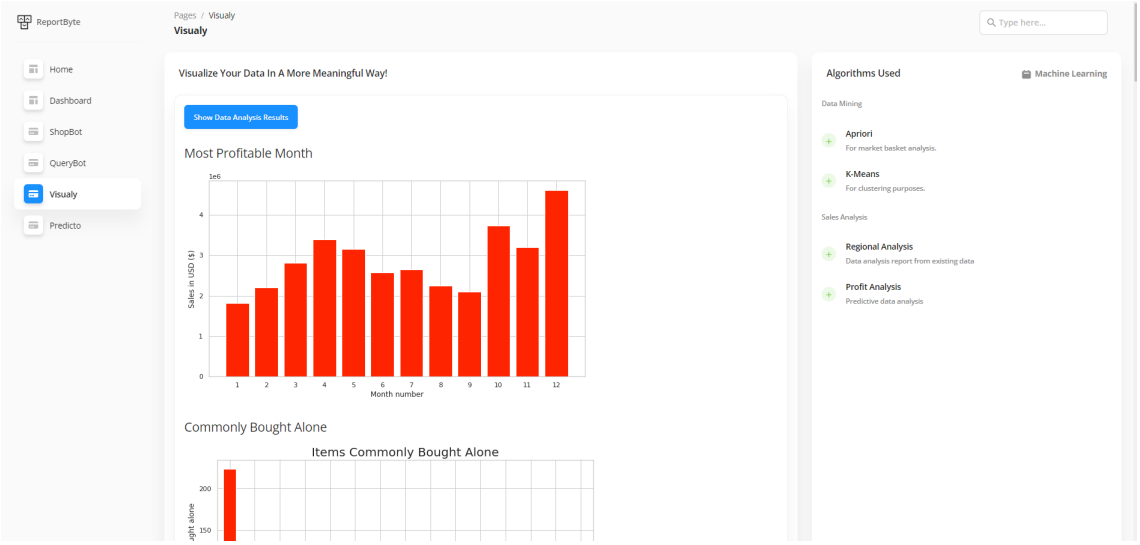

Figure 5.18: Feature Page (Visualy)

5.8.6 Reportbyte: Data Loading

This page will demonstrate how user can upload their data to reportbyte and use it's features.

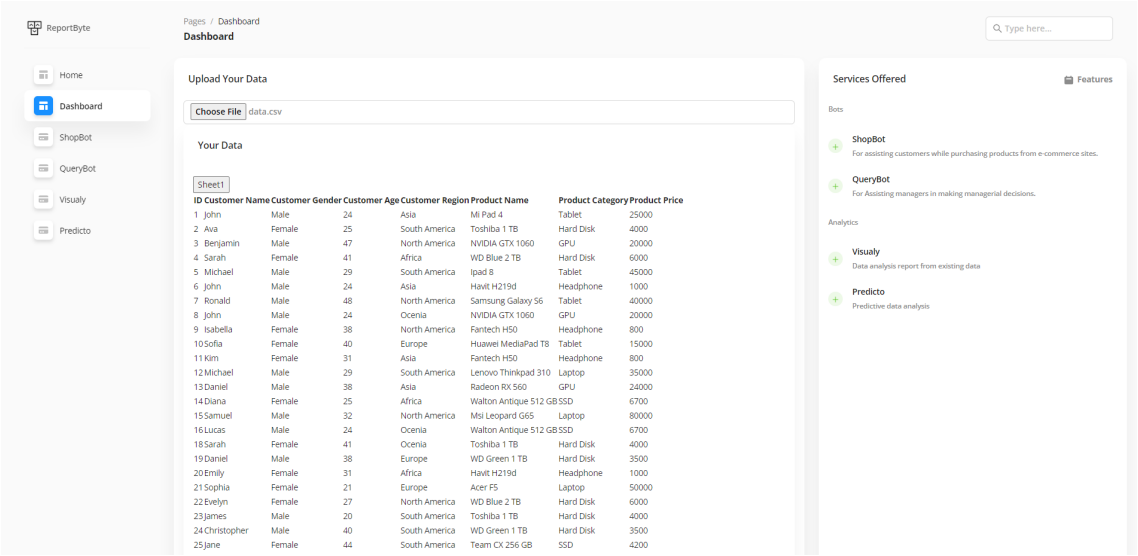

Figure 5.19: Data and Their Features

5.9 Deployment

Today, there exist multiple PaaS on the internet, that gives a lot of rich environments with NodeJS included. Sadly, most of them are paid, and those that provide the free tier services, puts a lot of constraints on the application, such as Heroku, that we will be using to deploy our application, it does delete all the static files on every server restart, and shuts down the server when it does not receive any request for some time. But one of the positive points was that we could deploy the application by linking it to the development repository on Github.

5.9.1 Heroku

Heroku is a cloud platform as a service supporting several programming languages. One of the first cloud platforms, Heroku has been in development since June 2007, when it supported only the Ruby programming language, but now supports Java, Node.js, Scala, Clojure, Python, PHP and Go.

5.10 Testing

Software Testing is a method to check whether the actual software product matches expected requirements and to ensure that software product is Defect free. It involves execution of software/system components using manual or automated tools to evaluate one or more properties of interest. The purpose of software testing is to identify errors, gaps or missing requirements in contrast to actual requirements.

Some prefer saying Software testing definition as a White Box and Black Box Testing. In simple terms, Software Testing means the Verification of Application Under Test (AUT). This Software Testing course introduces testing software to the audience and justifies the importance of software testing.

5.11 Challenges

Software Development is the collective process of some computer science activities dedicated to the process of developing software applications. Software Development process proceeds according to Software Development Life Cycle (SDLC). While building the project we had to deal with many challenges:

1. Changing Requirements
2. Providing complete Security
3. Misinterpreted requirements
4. System and Application integration
5. Maintenance and Upgradation
6. Adapting latest Technology
7. Time limitations
8. Communication and Coordination

5.11.1 ShopBot

1. **Fallback and Human Handoff**
   Even if one design his bot perfectly, users will inevitably say things to his assistant that he did not anticipate. In these cases, his assistant will fail, and it's important he ensures it does so gracefully.

   - **Fallback:** Although Rasa will generalize to unseen messages, some messages might receive a low classification confidence. Using Fallbacks will help ensure that these low confidence messages are handled gracefully, giving your assistant the option to either respond with a default message or attempt to disambiguate the user input.
   - **Two Stage Fallback:** To give the bot a chance to figure out what the user wants, you will usually want it to attempt to disambiguate the user's message by asking clarifying questions. The Two-Stage Fallback is made to handle low NLU confidence in multiple stages.
   - **Human Handoff:** As part of your fallback action, you may want the bot to hand over to a human agent e.g. as the final action in Two-Stage-Fallback, or when the user explicitly asks for a human. A straightforward way to achieve human handoff is to configure your messaging or voice channel to switch which host it listens to based on a specific bot or user message.

     For example, as the final action of Two-Stage-Fallback, the bot could ask the user, "Would you like to be transferred to a human assistant?" and if they say yes, the bot sends a message with a specific payload like e.g. "handoff_to_human" to the channel. When the channel sees this message, it stops listening to the Rasa

server, and sends a message to the human channel with the transcript of the chat conversation up to that point.

2. **Handling Unexpected Input**
   One thing you can count on when building a conversational assistant is that users will say unexpected things. This page is a guide on handling unexpected input. Unexpected input is a deviation from the happy path that you have defined. For example:

   A user walks away in the middle of a conversation about their subscription, then comes back and says "hi!"

   A user asks "Why do you need to know that?" when the bot asks for their email address.

3. **Contextual Conversations**
   Taking context into account is often key to providing a good user experience. This page is a guide to creating contextual conversation patterns.
   In a contextual conversation, something beyond the previous step in the conversation plays a role in what should happen next. For example, if a user asks "How many?", it's not clear from the message alone what the user is asking about. In the context of the assistant saying, "You've got mail!", the response could be "You have five letters in your mailbox". In the context of a conversation about outstanding bills, the response could be, "You have three overdue bills". The assistant needs to know the previous action to choose the next action.
   To create a context-aware conversational assistant, you need to define how the conversation history affects the next response.

4. **Reaching Out to the User**
   Sometimes you want your assistant to reach out to the user without the user's prompting. For example, you might want the assistant to send a message when the user opens the chat window, or you might want to prompt the user if they haven't sent a message for a while.

5.11.2 Querybot

1. Large Dataset : Since we are working with open source model only limited edition is available to us ,whise tapas is a very powerful model and can deal with lots of data it sometimes fails if there's too many row and column.

2. Website Integration: Since resources about dealing with machine learning models and web integration are scarce and transformers is a very latest concept in the world of machine learning and artificial intelligence we had to adapt and learn.

5.11.3 Visualy

Since visualy is mostly dependent on data mining algorithms, the challenges of data mining hampers it's performance too. Transforming data into organized information is not an easy process. There are many challenges in data mining.

Below are some of these Challenges listed and briefly explained:

1. Security and Social Challenges: Dynamic techniques are done through data assortment sharing, so it requires impressive security. Private information about people and touchy information is gathered for the client's profiles, client standard of conduct understanding-illicit admittance to information and the secret idea of information turning into a significant issue.

2. Noisy and Incomplete Data: Data Mining is the way toward obtaining information from huge volumes of data. This present reality information is noisy, incomplete, and heterogeneous. Data in huge amounts regularly will be unreliable or inaccurate. These issues could be because of human mistakes blunders or errors in the instruments that measure the data.

3. Distributed Data: True data is normally put away on various stages in distributed processing conditions. It very well may be on the internet, individual systems, or even on the databases. It is essentially hard to carry all the data to a unified data archive principally because of technical and organizational reasons.

4. Complex Data: True data is truly heterogeneous, and it very well may be media data, including natural language text, time series, spatial data, temporal data, complex data, audio or video, images, etc. It is truly hard to deal with these various types of data and concentrate on the necessary information. More often than not, new apparatuses and systems would need to be created to separate important information.

5. Performance: The presentation of the data mining framework basically relies upon the productivity of techniques and algorithms utilized. On the off chance that the techniques and algorithms planned are not sufficient; At that point, it will influence the presentation of the data mining measure unfavorably.

6. Scalability and Efficiency of the Algorithms: The Data Mining algorithm should be scalable and efficient to extricate information from tremendous measures of data in the data set.

7. Improvement of Mining Algorithms: Factors, for example, the difficulty of data mining approaches, the enormous size of the database, and the entire data flow inspire the distribution and creation of parallel data mining algorithms.

8. Incorporation of Background Knowledge: In the event that background knowledge can be consolidated, more accurate and reliable data mining arrangements can be found. Predictive tasks can make more accurate predictions, while descriptive tasks can come up with more useful findings. Be that as it may, gathering and including foundation knowledge is an unpredictable cycle.

5.11.4 Predicto

Since predicto is a predictive data analysis tool we faced some challenges building this, those are mentioned below:

Incompleteness

The accuracy of predictive analytics models is limited by the completeness and accuracy of the data being used. Because the analytical algorithms attempt to build models based on the available data, deficiencies in the data may lead to deficiencies in the model.

Correspondingly, the developed model might not encompass enough information to be able to recognize enough sentinel predictive patterns to be of any value. For example, a customer retention model might be built using customer service event histories and transactions, but the most accurate prediction models might require sales and returns transactions to provide patterns that yield value.

Data myopia

Customer profiles are engineered using guidelines based on people's expectations, which come with certain biases. Limitations in the range of different demographic variables in the model may force customers to be classified in ways that are too limited. As an example, individuals might be classified using salary averages calculated within the boundaries of defined census tracts. However, certain urban areas may have census tract regions in

which there are multiple discrete micro-communities with significantly different salary demographics. In this case, refining the size of the area of focus for average salary will improve the precision of the customer classification model.

Narrow-ization

This refers to what happens when the reliance on predictive analytics models to shape the business processes that influence customer behavior creates artificial boundaries that narrow the range of a customer's anticipated behaviors. In this case, there may be business opportunities – such as product bundling or up-selling – that are not even considered because the analytics-driven business process does not expect those opportunities to arise.

Spookiness

For a long time, automated systems have been capable of simple ad tracking in which sites drop cookies that provide information that can be accessed by partners within an ad network. Systems are becoming increasingly capable of scanning customer actions within a hierarchical semantic context to provide increased information about customer interests. A person's search terms coupled with product page visits may provide enough information to make inferences about what the customer is really looking for. However, as these bits of information are being employed to present advertisements and product placements, customers are becoming unnerved by automated systems attempting to anticipate their intent and influence their activities.

5.12 Limitations

5.12.1 ShopBot

1. **Named Entity Extraction**
   Text appears as unstructured data in several formats such as document files, spreadsheets, web pages and social media. Any system's ability to identify the entities within the documents - people, places, organizations, concepts, numerical expressions (e.g., dates, times, currency amounts, phone numbers, etc.) as well as temporal expressions (e.g., dates, time, duration, frequency, etc.) - enables that system to understand the information they contain and put them to good use.

   Entity extraction can provide a useful view of unknown data sets by immediately revealing who and what the information focuses on - at a minimum. This enables analysts to view all the names of people, companies, brands, cities, countries, and even phone numbers in a (structured) corpus that they can use as a point of departure for further analysis and investigation.

   The system currently lack on this feature but we are working on this to improve named entity extraction feature.

2. **Unexpected Input**
   If a user input a misspelled product or spell nearly similar to the product he or she is searching for, the result may or may not be correct. The accuracy of the result will vary depending on the spelling mistake. Too much spelling mistake will not show any result at all. We are also currently working on this feature to improve the accuracy of the returned result based on spelling mistake.

5.12.2 QueryBot

1. **Complex Query**
   Dealing with complex queries that involves sub-query and multi table relation is producing lots of errors and bugs ,we are currently working on fixing those issues and debugging the errors so that the model works properly.

5.12.3 Visualy

We used market basket analysis to impelement visualy .There were some limitations ,those are mentioned below:

1. **Iffy correlations**
   Averages tend to lie. If admin is trying to duplicate a conclusion drawn on chain wide data to merchandise a single store, admin will hit some speed bumps. An infamous version of this pitfall in the retail world is that of the 'beer and diapers' correlation. Way back before 'big data' (the '90's), a retail company ran SQL queries against its store data and discovered that beer was often purchased with diapers. This discovery quickly caught wind, and stores started to put diapers and beer nearby on store shelves. Naturally, sales of the two together went up.

   The issue is this... Of course sales went up for the combination of items; stores were merchandising them next to one another in a highly trafficked area. The root of the problem is that you can't seek out and validate correlations in data that you had a hand in creating. Thus, although market basket analysis may help you spot a trend, once you act on it, it's difficult to assess the validity of the correlation.

2. **No clear calls to action**
   Even if there's a true link between the two products, it takes time and skill to figure out how to act on this information.

   For example, let's say you learn that 50% of customers who buy bread buy rice within two weeks. You could use your market basket analysis tool to have a recommendation sent to each online shopper who bought bread within two weeks of their purchase. But when do you send it to ensure you capitalize on the correlation? The tool can't tell you. And how can you use this insight in your physical stores? In a best case scenario, your customer has an app or loyalty card that lets you track their purchases. With that, you could send an app-based promotion for the rice. But given that most Americans are enrolled in an average of 29 loyalty programs at once, the odds of your message being received is low.

3. **Test and learn lag times**
   Because there are no clear calls to action, retailers who use these solutions must dedicate time to A/B test any actions they take as a result of the data. But how do you decide which correlation to A/B test? Testing even just the strongest correlations takes enormous time and effort.

   Let's say you manage to pick a product pairing that you think will generate the most revenue. There's a lot of work to make the test happen. For instance, if you're going to run an in-store cross-promotion, you'll need to re-organize your shelves, rework your planograms and send directives down to all stores. From there, you'll need to train staff on the new locations and make them aware of the promotion. You'll then need an adequate amount of time (weeks? months?) to test whether or not you were right.

Costs aside, from learning the correlations to making changes and then testing their efficacy and using what you learned moving forward, this entire process is slow.

Clustering

We used K-means algorithm to implement some product recommendation features. Some limitations we faced are discussed below:

1. Choosing k manually.
2. Being dependent on initial values.
3. Clustering data of varying sizes and density.
4. Clustering outliers.
5. Scaling with number of dimensions.

5.12.4 Predicto

We used Monte Carlo Simulation to implement predicto. Some limitations of Monte Carlo are mentioned below:

1. **Benchmarking**
   In benchmarking, we found the #1 problem was not doing Monte Carlo but instead hoping a single number could be a proxy for the full range of uncertainty. A variation of this problem is building a Monte Carlo Simulation but doing it poorly so that it runs slowly. This leads to running only one Monte Carlo trail - essentially a deterministic simulation, not Monte Carlo.
2. **Interpretation of the Monte Carlo results**
   Truly understanding the spans of uncertainty and how multiple outcomes are related is hard for most of us to grasp. Good visualization and thoughtful presentation is important.
3. **Speed**
   It is easy to create slow Monte Carlo simulations. Big simulations can be blazing fast, but most people don't know how to do Monte Carlo right. This includes some vendors do Monte Carlo tools and platforms.
4. **Errors in choice of input distributions**
   The world is not a symmetrical Gaussian sort of place. But in benchmarking, we found many practitioners using deeply flawed inputs to describe uncertainty.

Chapter 6

Future Work

6.1 Further Improvement of Shopbot

We plan to train the bot using more customer and sales Representative conversation data and solve the named entity recognition problem using spacy model.

6.2 Further Improvement of QueryBot

We plan to improve tapas model by using various ensemble and deep learning techniques such that the bot can answer more complex and complicate questions.

6.3 Further Improvement of Visualy

We are planning to use more association rule and data mining algorithms to further extend our project. Such as Eclat and F-P Growth. Eclat algorithm stands for Equivalence Class Transformation. This algorithm uses a depth-first search technique to find frequent item sets in a transaction database. It performs faster execution than Apriori algorithm. The F-P growth algorithm stands for Frequent Pattern and it is the improved version of the Apriori Algorithm. It represents the database in the form of a tree structure that is known as a frequent pattern or tree. The purpose of this frequent tree is to extract the most frequent patterns.

6.4 Further Improvement of Predicto

Predictive analytic tools are powered by several different models and algorithms that can be applied to wide range of use cases. Determining what predictive modeling techniques are best for customer's company is key to getting the most out of a predictive analytics solution and leveraging data to make insightful decisions. We plan to use more predictive data mining techniques such as Classification Model, Clustering Model, Outliers Model, Time Series Model to further improve the performance of this tool.

Chapter 7

Conclusion

Business analytic provides a way for businesses to plan for the future. By modeling the trends in a business's sales, profits and other key metrics, these indicators can be projected into the future. Understanding the changes that are likely to occur seasonally, annually or on any scale allow businesses to better prepare. This may mean decreasing spending in preparation for a slow season or investing in new marketing campaigns to compensate. Large suppliers can use this data to predict order volume and minimize waste in their warehouses. Planning for future events provides a huge advantage to all businesses.

Business analytic can also enable new types of marketing campaigns. The data collected by businesses give insights into customer behavior which helps businesses understand the effectiveness of advertising campaigns with different audiences. Targeting audiences that are more likely to respond to specific campaigns or products increases efficiency overall. In addition, understanding consumer habits can help businesses improve customer retention. By identifying customers who are less likely to return, businesses can offer targeted promotions. This provides a cost-effective way to gain customer loyalty.

References

[1] M. Kaur and S. Kang, "Market basket analysis: Identify the changing trends of market data using association rule mining," *Procedia computer science*, vol. 85, pp. 78–85, 2016.

[2] "Rasa framework." https://rasa.com/open-source. Accessed: 12-12-2021.

[3] "Tapas model." https://huggingface.co/docs/transformers/model_doc/tapas. Accessed: 12-12-2021.

[4] H. H. Ali and L. E. Kadhum, "K-means clustering algorithm applications in data mining and pattern recognition," *International Journal of Science and Research (IJSR)*, vol. 6, no. 8, pp. 1577–1584, 2017.

[5] A. Alamsyah and B. Nurriz, "Monte carlo simulation and clustering for customer segmentation in business organization," in *2017 3rd International Conference on Science and Technology-Computer (ICST)*, pp. 104–109, IEEE, 2017.

[6] "Django." https://www.djangoproject.com/. Accessed: 12-12-2021.

[7] "React.js." https://reactjs.org/. Accessed: 02-04-2022.

[8] "Node.js." https://en.m.wikipedia.org/wiki/Node.js. Accessed: 12-12-2021.

[9] "Express.js." https://expressjs.com/. Accessed: 12-12-2021.

[10] "Express.js wiki." https://en.m.wikipedia.org/wiki/Express.js. Accessed: 12-12-2021.

[11] "Mongodb." https://en.m.wikipedia.org/wiki/MongoDB. Accessed: 12-12-2021.

[12] "Mongodb atlas operational efficiency." https://www.mongodb.com/cloud/atlas/efficiency?utm_source=google&utm_campaign=

gs_footprint_row_search_core_brand_atlas_mobile&utm_term=mongodb&utm_medium=cpc_paid_search&utm_ad=e&utm_ad_campaign_id=12212624587&adgroup=115749713983. Accessed: 12-12-2021.

Generated using Undegraduate Thesis LaTeX Template, Version 1.4. Department of Computer Science and Engineering, Ahsanullah University of Science and Technology, Dhaka, Bangladesh.

This thesis was generated on Saturday 2nd July, 2022 at 10:57pm.