A PROJECT REPORT

on

"A.I. Enabled Fin Tech B2B Invoice Management Application"

Submitted to

KIIT Deemed to be University

In Partial Fulfillment of the Requirement for the Award of

BACHELOR'S DEGREE IN
COMPUTER SCIENCE
AND
SYSTEMS ENGINEERING

BY

DRISHTI

Roll No. : 1928228

Reg. No. : 19592272019

UNDER THE GUIDANCE OF

Chandramouli Das and Akash Kumar Yadav

SCHOOL OF COMPUTER ENGINEERING

KALINGA INSTITUTE OF INDUSTRIAL TECHNOLOGY

BHUBANESWAE, ODISHA -751024

May 2022

KIIT Deemed to be University

School of Computer Engineering
Bhubaneswar, ODISHA 751024

CERTIFICATE

This is certify that the project entitled

"A.I. Enabled Fin Tech B2B Invoice Management Application"

submitted by

DRISHTI
Roll No. : 1928228
Reg. No. : 19592272019

is a record of bonafide work carried out by her, in the partial fulfillment of the requirement for the award of Degree of Bachelor of Engineering (Computer Science and Systems Engineering) at KIIT deemed to be university, Bhubaneswar. This work is done during year 2022-2023, under our guidance.

Date:15/04/2022

Chandramouli Das &
Akash Kumar Yadav
(Project Mentor)

ACKNOWLEDGEMENT

On this auspicious occasion of the completion of my “A.I. Enabled Fin Tech B2B Invoice Management Application” project, I'd like to convey my heartfelt appreciation to Mr. Chandramouli Das and Mr. Akash Kumar Yadav for their excellent direction and constant support in ensuring that this project met its objectives since its commencement to its completion.

I would also like to express my gratitude to our Director General of School of Computer Engineering (KIIT-DU) Prof. Biswajeet Sahoo for giving me this opportunity and all of the necessary resources to accomplish my project.

I would like to convey my deep appreciation to Deans of the School of Computer Engineering (KIIT-DU), Prof. Bhabani Shankar Prasad Mishra, Dr. Amulya Ratna Swain, and Dr. Arup Abhinna Acharya, for their encouragement in completion of my project.

Finally, I would like to thank my parents and friends, without them this project would not have been completed.

DRISHTI
Roll No.: 1928228
Reg. No.: 19592272019

ABSTRACT

In the business-to-business sector, where large-scale firms engage in supply chain commerce, invoice management technologies cause fundamental transformations in the understanding of effective working practices and play a critical role in ensuring the smooth operation of the business. Any software that organizes invoice documents from vendors and suppliers and helps firms automate operations connected to invoice processing is known as an invoice management system.
When businesses or accounting departments receive invoices, they often go through an approval procedure before being linked with sales and purchase orders, as well as payments.

The goal of this project is to develop an Artificial Intelligence-enabled invoice management system. The system has a Dashboard that can estimate invoice payment dates and group them into ageing buckets, as well as search, advance search, edit, remove, and add invoices. After comparing the best fit to forecast the delay in payments, the A.I. is based on Extreme Gradient Boost Regression. We employed Machine learning, Java, JDBC, Servlet, and ReactJS principles in a cohesive manner. This programme assists users in better understanding their clients based on ageing buckets and performing procedures accordingly.

Keywords: Invoice management Machine Learning Java ReactJS Prediction

Contents

1 Introduction…… 1

2 Basic Concepts…… 4

2.1 Tools Employed……4

2.2 Techniques Employed…… 5

3 Problem Statement / Requirement Specifications……8

3.1 Project Planning.....…….. 8

3.2 Project Analysis.....…… 9

3.3 System Design …… 9

3.3.1 Design Constraints …… 9

3.3.2 System Architecture / Block Diagram …….. 10

4 Implementation…… 11

.

4.1 Methodology / Proposal .....…….…… 11

4.2 Testing / Verification Plan ……17

4.3 Result Analysis / Screenshots …… 21

5.1 Design Standards . . . . . ……..…… 5

6 Conclusion and Future Scope…….. 25

6.1 Conclusion ……25

6.2 Future Scope …… 25

References……26

List of Figures

1.1 Manual Invoice Procesing……..... 2

1.2 Automatic Invoice Processing……..... 3

3.1 System Architecture …… 10

4,1 Web Application UI landing page…… 21

4.2 Web Application UI Add model…….. 21

4.3 Web Application UI Edit model…….. 22

4.4 Web Application UI Delete model…….. 22

4.5 Web Application UI Adv Search model…… 23

4.6 Web Application UI Predict model……. 23

4.7 Web Application UI Analytic View model……. 24

4.8 Web Application UI Analytic View 02 model……. 24

List of Tables

4.1 Testing Case……..... 17

4.2 Application Testing……..... 19

CHAPTER 1

Introduction

What is Invoice management? It is the process through which businesses invoice their customers for the items and services they buy, as well as track invoices and pay suppliers and vendors. In a nutshell, invoice management entails dealing with both accounts payable and receivable.

What is the purpose of it? While handwritten invoice handling is possible, it is an antiquated method of doing business. Manual invoicing may quickly become a tiresome and time-consuming procedure, with tasks such as writing down each invoice on an Excel spreadsheet and double-checking for inaccuracies. We don't have to confine our business to pen and paper. Invoice management systems are now used by the majority of enterprises as a faster, more accurate, and cost-effective option. Most aspects of the invoice management process, such as producing invoices and managing accounts payable, may be automated using invoice management software.

And why wouldn't we want to automate the invoicing process? It has several advantages, including consistent rapid invoice processing, prevention of errors and data loss, elimination of payment delays and associated complications such as vendor relationship jeopardy, work pause, etc., rerouting of manpower on more rewarding tasks, supply chain movement maintenance, fraud prevention, cost savings, greater visibility, control, and compliance, eco-friendliness and avoidance of paper clutter, and better document preservation.

What is the significance of invoice management?

For starters, inaccurate invoicing exposes our company to a slew of problems we'd want to avoid. Cash flow problems, strained client and supplier relationships, late interest costs, and even legal action are all possible repercussions.

Invoice management, on the other hand, not only protects us from errors, but also aids in the development of positive connections with our suppliers and customers. It serve as the foundation for prompt payment. It's critical to keep track of invoices to avoid delays and mistakes when receiving or paying for products and services. It maintains track of sales and suppliers, avoiding waste and delays, inventory management can be aided by it. Statistics from invoice management may aid in the planning and analysis of a company's financial and performance data, tax paperwork, audits, and taxation tasks are made easier are only some facts which supports the statement.

Lastly, how is it done? This may be accomplished in a number of ways, the first of which is the manual method, and the second of which is the automated approach.

When a new transaction happens in a manual process, an invoice is issued so that it may be traced afterwards. The invoice managers are then given a copy of it, which might be a physical or digital copy. Then they enter the data that they manually collect, filling out all of the columns in their respective data format one by one, which may be tedious and time consuming.

They then carry out the operations, as they must determine who has paid them and who has not in the subsequent phases. As a result, they must conduct actions such as add, remove, search, advanced search, modify, and even forecast the results. Then they keep track of all upcoming transactions with a due date in the near future and keep informing the firm about them; if the form responds and submits the amount, they remove it and archive it for later use, which can be done as a paper file or as a digital file; if not, they give the company an ultimatum and renew it according to the customer payment terms.

Now, this can be a bit tiresome in a manual process because everything is done by hand and it takes a long time. Because the world is moving fast, new data is being generated at a rapid rate, and it is required that the prediction, searching, and all other operations required for invoice management be completed at a rapid rate, which is difficult for a normal human speed, an automated invoice management system is installed.

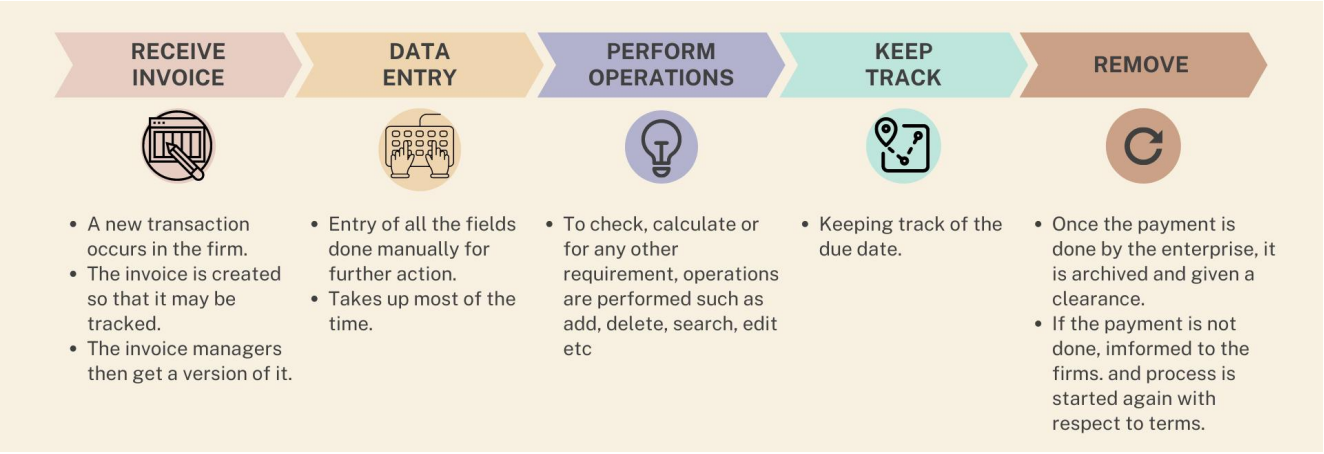

Figure 1.1 : Manual Invoice Processing

In automated invoice management system when a new transaction happens in the company, an invoice is generated so that it can be monitored. The invoice can be a physical copy or a digital copy; if it is a physical copy, it is scanned and converted to a digital copy. The document is next processed by OCR, which stands for optical character recognition and is a type of NLP software in which the data in the document is analyzed and extracted based on the structure of the data using a vector technique. When the data is extracted, it is saved as a CSV file in the appropriate columns with the appropriate format. It is then forwarded to the checking department to be confirmed, validated, and enhanced. This is to ensure that the data is right, and it differs from completing everything manually in that we had to manually input all of the data, but here we are simply checking the input data.

The artificial intelligence work follows, when machine learning takes control and completes the task. It forecasts, sets, and performs all of the processes necessary for invoice management. Whatever data we gather in real time in a CSV file is transmitted to the machine learning model, which uses that data to continually anticipate the ageing bucket. An ageing bucket is simply a period of time during which a company will return money. Machine learning is used to anticipate payment dates, analyse all forthcoming bills, and even prior invoices in terms of company or customer payment terms, date, or quantity, among other things.

We used to track manually by contacting the company or changing the entire terms and manually archiving it all, but now the tracking will be done by the personal assistant that will be embedded in the system using machine learning. The personal assistant will handle any disagreements that arise, as well as all tracking, processing, and communication with the businesses. It will extract parts of all the essential phrases and dates, emails, and phone numbers that have been discussed in the conversations and emails and construct a gist out of it before sending it back to the processing unit to be analyzed by people while speaking.

The entire process requires human input, but just from start to finish, and that is how it is effective and fast. Prediction may take long for people if they do it independently using any formulation, but prediction can be done in a matter of seconds with only a few codes.

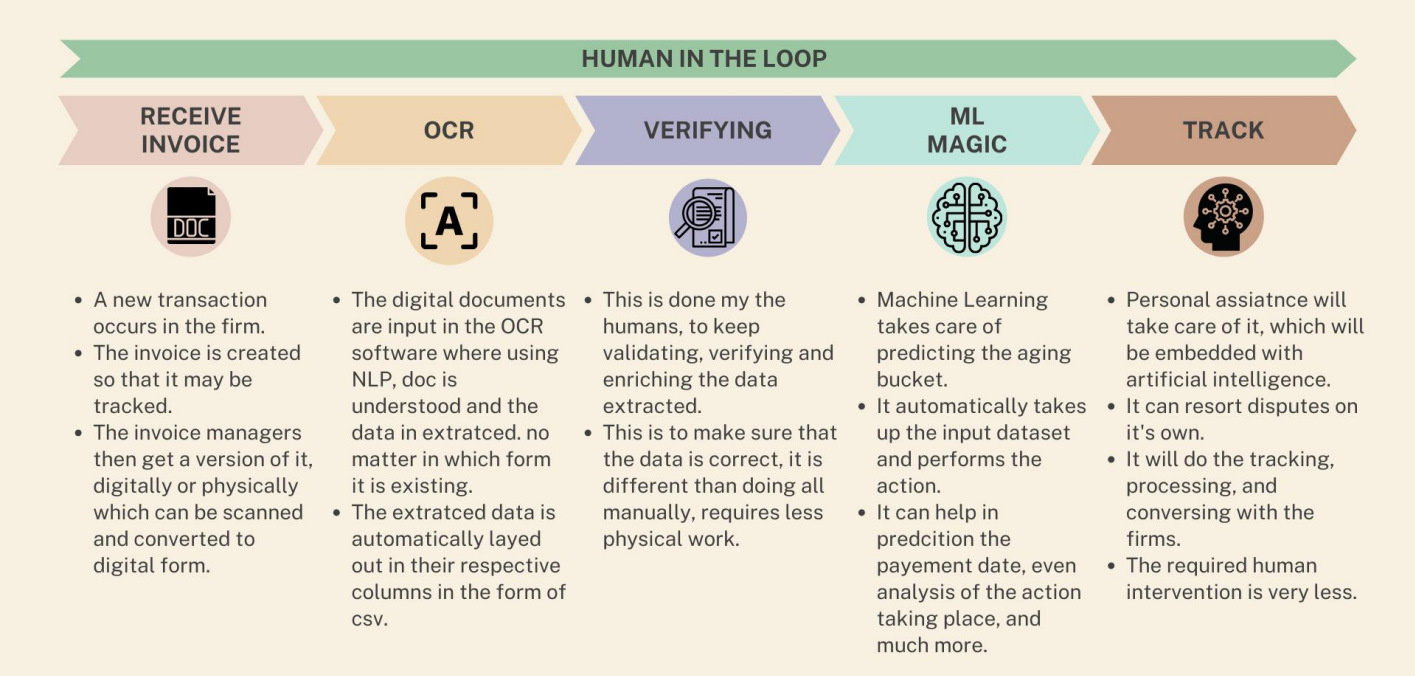

Figure 1.2 : Automated Invoice Processing

CHAPTER 2

Basic Concepts

2.1 Tools Employed

2.1.1 Jupyter Notebook (via Anaconda Navigator)

The Jupyter Notebook is a free and open source web tool that lets us create and share documents with live code, equations, visualisations, and text. I utilised Anaconda Navigator to use it in this project. It's a desktop graphical user interface (GUI) that lets us effortlessly handle conda packages, environments, and channels without having to use command-line instructions. As a result, I didn't have to import every time I used a new type of function in my notebook when coding for machine learning.

2.1.2 VScode

Visual Studio Code is a streamlined code editor that includes features for debugging, task execution, and version management. It seeks to provide developers only the tools they need for a rapid code-build-debug cycle, leaving more sophisticated processes to full-featured IDEs like Visual Studio IDE. In this project, I used it to build the whole front end after importing a few things, such as axios and mui, to mention a few. This includes a live server function that I enabled so that I could view the changes I was making in the code in real time.

2.1.3 MySql

MySQL is a relational database management system that is free and open source. It is relational because it keeps information in different "tables" and relates it using "keys."In this project, I used it to upload my data and keep track of all the critical operations I'll be performing later on. I uploaded the data in two formats: first, a simple upload, and second, a sql query form, . It also has a text field where I can write the query to perform the operation in the database.

2.1.4 Eclipse

Eclipse is a free Java-based development environment recognised for its plugins, which allow programmers to build and test code written in other languages. In this project, I utilised it to code all of the servelts required to perform the database's crud operations. This is the backbone of the whole backend, which is implemented using Tomcat 9.0. on a localhost live server, because of which I was able to publish my output on a live server.

2.2 Techniques Employed

2.2.1 Machine Learning

Machine learning (ML) is a subset of artificial intelligence (AI) that allows software to increase prediction accuracy without being specifically intended to do so. Machine learning algorithms use previous data as input to anticipate new output values. It isn't extremely precise, but we use a variety of techniques to understand and improve it.

2.2.1.1 Python

Python is a scripting language that is high-level, interpreted, interactive, and object-oriented. Python is the greatest programming language for machine learning because of its independent platform and many built-in libraries. The Python programming language is thought to be the finest method for automating jobs since it is simpler and more consistent than other programming languages.

2.2.1.2 Scikit Learn

For the Python programming language, Scikit-learn is a machine learning library. Support-vector machines, random forests, gradient boosting, and other classification, regression, and clustering techniques are included.

2.2.1.2.1 Linear Regressor

Linear regression is a supervised learning machine learning technique. It carries out a regression job. Based on independent variables, regression models a goal prediction value. Its a regression model that estimates the relationship between one independent variable and one dependent variable using a straight line. [1]

2.2.1.2.2 Support Vector Regressor

The supervised learning technique Support Vector Regression is used to predict discrete values. SVMs and Support Vector Regression are both based on the same premise. SVR's main premise is to locate the best-fitting line.[2]

2.2.1.2.3 Decision Tress Regressor

In the shape of a tree structure, a decision tree constructs regression or classification models. It gradually cuts down a dataset into smaller and smaller sections while also developing an associated decision tree. A tree containing decision nodes and leaf nodes is the end result. The data is continually separated based on a certain parameter. Two entities, decision nodes and leaves, can be used to explain the tree. [3]

2.2.1.2.4 Random Forest Regressor

Random Forest Regression is a supervised learning approach for regression that use the ensemble learning method. The ensemble learning approach combines predictions from several machine learning algorithms to get a more accurate forecast than a single model.

2.2.1.2.5 XGB Regressor

XGBoost is a fast gradient boosting solution that may be used for regression predictive modelling. Extreme Gradient Boosting (XGBoost) is a distributed gradient-boosted decision tree (GBDT) machine learning toolkit that is scalable. It is the top machine learning library for regression, classification, and ranking tasks, and it includes parallel tree boosting. [4]

2.2.1.3 Pandas

Pandas is mostly used for data analysis and related data manipulation in dataframes. Pandas supports importing data from comma-separated values (CSV), JSON, Parquet, SQL database tables or queries, and Microsoft Excel.

2.2.1.4 NumPy

NumPy may be used to conduct a wide range of array-based mathematical operations. It extends Python with sophisticated data structures that ensure fast computations with arrays and matrices, as well as a large library of high-level mathematical functions that work with these arrays and matrices.

2.2.1.5 Seaborn

Seaborn is a Python module for creating statistical visuals. It aids in data exploration and comprehension. Seaborn is a Python module for creating statistical visuals. It is based on matplotlib and tightly interacts with pandas data structures. Seaborn assists you in exploring and comprehending your data.

2.2.1.6 Matplotlib

Matplotlib is a charting library for the Python programming language and NumPy, the Python numerical mathematics extension. It aids in data visualisation and a better comprehension of the data.

2.2.2 Web Application

2.2.2.1 Back End

2.2.2.1.1 Java Servlets

A web application is created using the Servlet technology. It's an API with a lot of different interfaces and classes, as well as documentation. Servlet is a class that enhances the server's capabilities and reacts to incoming requests. It is capable of responding to any request. It's operating on a Tomcat server powered by Apache.

2.2.2.1.2 Flask

Flask is a simple and lightweight Python backend framework that provides essential tools and functionalities for building web applications in Python. It was utilised to integrate the machine learning model that was used to estimate the ageing bucket for invoices.

2.2.2.2 Front End

2.2.2.2.1 NodeJS

Node.js is a cross-platform, open-source back-end JavaScript runtime environment that uses the V8 engine to execute JavaScript code outside of a web browser.

2.2.2.2.2 ReactJS

It's a free and open-source front-end JavaScript framework for creating UI components-based user interfaces. Meta (previously Facebook) and a community of individual developers and businesses manage it.

2.2.2.2.3 MaterialUI

Material UI is a front-end framework for React components that is open-source.

CHAPTER 3

Problem Statement / Requirement Specifications

To create a full-stack, A.I-enabled invoice management application that forecasts when the buyer will pay the invoice.

3.1 Project Planning

3.1.1 We uploaded the dataset to MySQL Workbench and used the text editor to access it, allowing us to conduct the necessary functions in query format.

3.1.2 We develop a machine learning model that automates account receivables on a given dataset. To predict account receivables, we need to first understand the data, then pre-process it, perform feature engineering, and then train a model.

3.1.3 We built servlets in eclipse for each and every function we wanted to do, such as add, remove, and edit, and then ran them on the server Tomcat after connecting them to our database. We obtained the user and password for our local system's sql workrbench and connected it.

3.1.4 We will construct a full stack Invoice Management Application utilizing ReactJS, JSP in this phase, which will enable data visualization in the form of grids, searching operations on invoices, editing data in the editable fields of the grid, and deleting invoices as needed, and for the prediction we will be using Flask.

3.2 Project Analysis

The data is imported and preprocessed, as per the requirements. The features in the data are picked using lasso regression and encoding methods such as one-hot encoding and label encoding. Then, using models such as linear regression, Support Vector Regression, Decision Tree Regression, Random Forest Regression, and XGB Regression, the given features are trained. The accuracy will be determined, and the model with the highest accuracy will be chosen. The model will be preserved and reused in the future.

Here, a Java Servlet will be used to establish an Application Programming Interface. A SQL script is used to load the data into a local MySQL database. A java servlet will be linked to a database using the JDBC package and used to conduct CRUD (create, read, update, and delete) actions.
Another Flask backend will be built, and the stored ML model will be used to forecast when the customer will pay the invoice.
For the customer to see the data, a frontend application will be constructed where they can search, add, amend, and remove invoice information as well as anticipate the time period.

3.3 System Design

3.3.1 Design Constraints

- Software Used:
  - Python: Python 3
  - Libraries: Numpy, Pandas, MatplotLib, Seaborn, Sklearn, etc.
  - Operating System: Windows
  - IDE: Visual Studio Code, Jupyter Notebook, Eclipse
- Hardware Used: Laptop with basic hardware

3.3.2 System Architecture **OR** Block Diagram

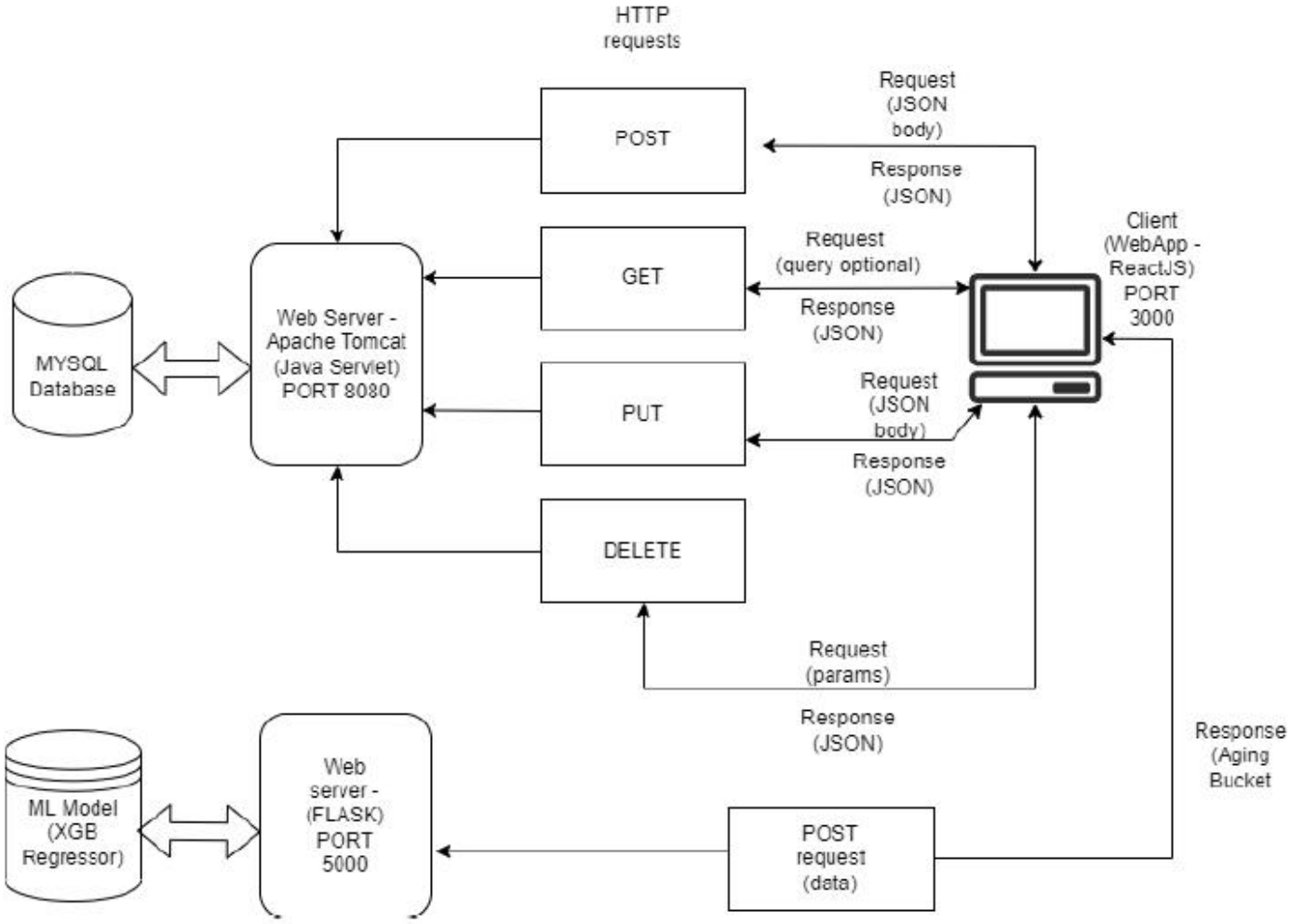

Figure 3.1 System Architecture of the whole project

CHAPTER 4

Implementation

4.1 Methodology OR Proposal

1. <u>Machine Learning</u>

a. Importing related libraries like
- i. Pandas
- ii. Seaborn
- iii. Matplotlib
- iv. Numpy
- v. Scikit Learn
- vi. Xgboost

b. Importing the dataset and storing it as a dataframe using the pandas library and then displaying it.

c. Checking for the shape of the dataframe and getting detailed information about it along with the Maximum, minimum values of the numerical variables and their Standard Deviation, Mean and median values.

d. Data cleaning
- i. Checking for the null values in the dataframe and checking for their percentage.
- ii. Checking for any abnormalities in the dataset and cleaning those data.
- iii. Checking for the unique values of the columns.
- iv. Dropping the columns those are irrelevant or those that are quite similar
  1. 'document_create_date' and 'document_create_date1' columns are the same, hence the 'document_create_date1' column has been dropped.
  2. IDs are not required so 'invoice_id' and 'doc_id' have been dropped.
  3. 'area_business' column contains full null values, so it has been dropped.
  4. 'document_type' and 'isOpen' are also not required, so those columns have been dropped

v. Checking for the duplicate values and hence dropping those.

vi. Again checking for the column and shape of the dataframe.

vii. Now, checking for the data types of each column.

1. The date columns are not in proper data types. Converting those values into date type format.

a. Changing these columns 'clear_date', 'posting_date', 'due_in_date', 'baseline_create_date' into date time data type and into proper display format.

viii. Checking for the 'invoice_currency'. It has only two unique values - 'CAD' and 'USD'

1. In order to keep the whole 'total_open_amount' in 'USD' currency, all the 'CAD' currency has been converted into 'USD' by multiplying with 0.7.

2. Creating a new feature 'converted_usd' that will only contain the final 'total_open_amount'

ix. Checking for the 'buisness_year' as it has only two unique values - 2019 and 2020. The value counts are checked for each of them.

x. As the currencies have been changed to only one currency, the 'invoice_currency' column is no longer required and neither the 'total_open_amount' column is required. Therefore, these two columns 'invoice_currency' and 'total_open_amount' have been dropped.

e. Splitting of the dataset

i. Splitting the dataset into two parts

1. Maindata - It consists of the data from the actual dataframe where the 'clear_date' column does not contain null values.

2. Nulldata - It consists of the data from the actual dataframe where the 'clear_date' column contains null values.

3. train_test_split function from sklearn has been used for splitting.

ii. Checking for the columns and shape of the two derived dataframes.

f. Considering the maindata for feature engineering

i. A new feature 'Delay' is generated from the existing 'clear_date' column and 'due_in_date' column i.e. maindata['Delay'] = maindata['clear_date'] - maindata['due_in_date']

ii. Again another new feature 'avgdelay' is generated by grouping the 'Delay' column by 'name_customer' column and thus converting into a dictionary i.e. avgdelay = maindata.groupby('name_customer')['Delay'].mean(numeric_only=False).to_dict()

iii. Now, the "avg_delay" column is added with the maindata, mapped with "name_customer" column. The maindata[‘avgdelay’] is displayed.

iv. Since the ‘avg_delay’ column is in days format, the format is changed into seconds. maindata['avg_delay'] = maindata['avg_delay'].dt.total_seconds()

v. Since ‘avgdelay’ column is derived from ‘clear_date’ column, there is no need of these two columns ‘clear_date’ and ‘Delay’ anymore. Hence these two columns have been dropped.

g. Splitting the maindata into train and test data

i. X (train dataframe) - It will contain the whole dataframe excluding the ‘clear_date’ column.

ii. Y (test dataframe) - It will contain only the ‘clear_date’ column.

iii. Similarly, from X and y, splitting into sets like X_train, X_loc_test, y_train, y_loc_test.

iv. And from X_loc_test and y_loc_test, it is splitted into train and validation sets X_val, X_test, y_val, y_test.

h. Exploratory Data Analysis (EDA)

i. Visualization of different columns has been done using several graphs like distplot, countplot and boxplot from seaborn library.

i. Feature Engineering

a. Categorical variables need to be encoded before training into an ML model. ‘business_code’ has been Label Encoded using LabelEncoder from sklearn. A new feature will be created ‘business_code_enc’ and hence ‘business_code’ is deleted

i. Same for the validation set.

ii. ‘cust_numer’ is manually encoded as 1,2 and so.

iii. ‘name_customer’ is also label encoded.

b. Passing dataframe into ML model

i. In order to pass it into model, converting it into float format.

1. Extracting days from "posting_date" column and storing it into a new column "day_of_postingdate" for train, test and validation dataset .

2. Extracting months from "posting_date" column and storing it into a new column "month_of_postingdate" for train, test and validation dataset.

3. Extracting year from "posting_date" column and storing it into a new column "year_of_postingdate" for train, test and validation dataset.

4. Repeating the above 3 steps for 'baseline_create_date', 'due_in_date'.

c. Feature selection

i. Any one of the method has been used to select the features among the following implemented methods

1. Filter method

2. Correlation - variables which have closer to 1 or -1 correlation is selected

3. Wrapper method

4. Lasso

ii. Modelling

1. Different blank list has been made for different evaluation matrix

a. MSE (Mean Square Error)

b. R2

c. Algorithm

2. All the ML models are implemented out of which XGB Regressor is chosen with accuracy 67.44% for train set, 12.46% for validation set and 18.5% for test set.

iii. Lastly, all previous the steps are applied to the nulldata dataframe

iv. The 'final_result' is predicted and added with the 'avg_delay' and mapped into existing data

v. 'Due_in_date' is added with 'avg_delay' to obtain the clear date.

Then the 'avg_delay' is converted into days and then sorted into bins called 'aging_bucket'

2. Backend API

a. An SQL dump has been provided as a script which was runned on the MYSQL workbench on the localhost connection.

b. Java Servlet

i. Creating a 'Dynamic Web Project' and attaching 'Apache Tomcat 10.0.6' as the runtime environment

ii. Creating three packages

1. 'hrc.com.web.model' - This package consists of a POJO (Plain Old Java Object) class with the necessary data objects, constructors, and getters and setters functions

2. 'hrc.com.web.dao' - This package consists of DAO (Data Access Object) Class. This class has functions to connect to MYSQL DB with JDBC and logic for inserting invoices, editing invoices, fetching invoices, and deleting invoices

3. 'hrc.com.web.servelet' - This package consists of a Servlet class where the functions doPost, doGet, doPut, and doDelete. The function from the DAO class is called here in their respective functions.

iii. Now running the server at port 8080 of localhost and testing in Postman

c. Flask API: This API is responsible for prediction and using the model saved from ML

i. Creating a Flask server

ii. Adding an endpoint '/predict' with HTTP method as "POST" and writing a function below where the data obtained from the request is cleaned, engineered, and selected using the steps mentioned above and then sending a response back with the 'aging_bucket' field in JSON format.

3. Frontend Application (ReactJS app)

a. Creating a new project with ‘npx create-react-app . ‘ command to create a react project in that folder.

b. Installing the dependencies/modules from npm using ‘npm i’ command.

c. Cleaning the pre-built code

d. Adding new folders like ‘components’ to store the components that will be created later

e. The following flow was used to create the app

- i. Topbar was created
  1. Used “topbar.jsx” and inserted the images and the styling was done
- ii. Utilbar was created
  1. Two button groups were created and a textfield was added accordion to the Product Requirement Specification
  2. The functionalities were also added
- iii. Data grid View was created
  1. It was done using the ‘data-grid-view’ component of Material UI
  2. Row selection features was added
- iv. Footer was created.
- v. Modals were created
  1. Add invoice Modal - Textfields were added to receive data from user and was connected with ‘/payment’ POST API.

  2. Edit Invoice Modal - 2 text field were added to receive data from user and was connected with ‘/payment’ PUT API.

  3. Delete Invoice Modal - A confirmation message was shown upon which options were provided to surely delete the row(s) or cancel the operations. If delete was selected, ‘/payment’ DELETE was called.

4. Advanced Search Modal - Consists of textfields where data will be provided and '/payment' GET with the queries.
5. Analytics View Modal - Consists textfield to take the parameters for query from user and show the obtained data in form of charts in a different modal.

4.2 Testing OR Verification Plan

Test ID	Test Case Title	Test Condition	System Behavior	Expected Result
T01	POST /payment	{ "business_code" : "U002", "cust_number": 200769623, "clear_date": "2022-04-23", "buisness_year" : "2003", "doc_id": "100222", "posting_date": "2021-04-09", "document_create_date": "2021-04-09", "due_in_date": "2021-04-09", "invoice_currency" : "CAD", "document_type" : "U7222", "posting_id" : 1222111 , "total_open_amount" : "232222", "baseline_create_date": "2022-08-21", "cust_payment_terms" : "NHJWW", "invoice_id" : "9020022" }	Waiting for response	200 OK

Table 4.1 Testing Case

T02	POST /payment	{ "business_code" : "NHA2", "cust_number": 200769623, "clear_date": "2022-04-23", "buisness_year" : "2003", "doc_id": "100222", "posting_date": "2021-04-09", "document_create_date": "2021-04-09", "due_in_date": "2021-04-09", "invoice_currency" : "CAD", "document_type" : "U7222", "posting_id" : 1222111 , "total_open_amount" : "232222", "baseline_create_date": "2022-08-21", "cust_payment_terms" : "NHJWW", "invoice_id" : "9020022" }	Waiting for response	500 SERVER ERROR
T03	GET /payment	No Condition	Waiting Response	Json array consistin g of data present in DB
T04	GET /payment?cust_number=”102993”	Query can be added	Waiting for response	JSON array consisting data that m the query

T05	PUT /payment	{ "sl_no" : 1, "invoice_currency" : "CAD", "cust_payment_terms" : "NH12" }	Waiting for response	If sl_no exists, data will get updated (200 OK), else 404 NOT FOUND error. If DB error, then 500 SERVER ERROR
T06	DELETE /payment?sl_no=1	No condition	Waiting for response	If sl_no exists, data will get updated (200 OK), else 404 NOT FOUND error. If DB error, then 500 SERVER ERROR

APPLICATION

Test ID	Test Case Title	Test Condition	System Behavior	Expected Result
T01	ADD	Adding new column	Dialog box appears, takes input	Row is added in datagrid

Table 4.2 : All applications testing

T02	DELETE	Deleting a column	Pop-up to confirm deletion	Selected column is deleted
T03	EDIT	Editing a column	Dialog box appears, takes input	The editable field Values are edited
T04	Advanced Search	To perform an advanc search on the data	Dialog box appear, Takes 4 field as input (Document Id, Customer No, Invoice No,Business Year	Selected column is deleted
T05	Predict	Predict the payment date of selected Invoices	Calls the predict API	The predicted Payment Date column on the Ul with the predicted dates
T06	Analytics	Provide the user with an illustration of a bar graph and pie chart	Takes the parameter	Visualize data i the form o graphs

4.3 Result Analysis OR Screenshots

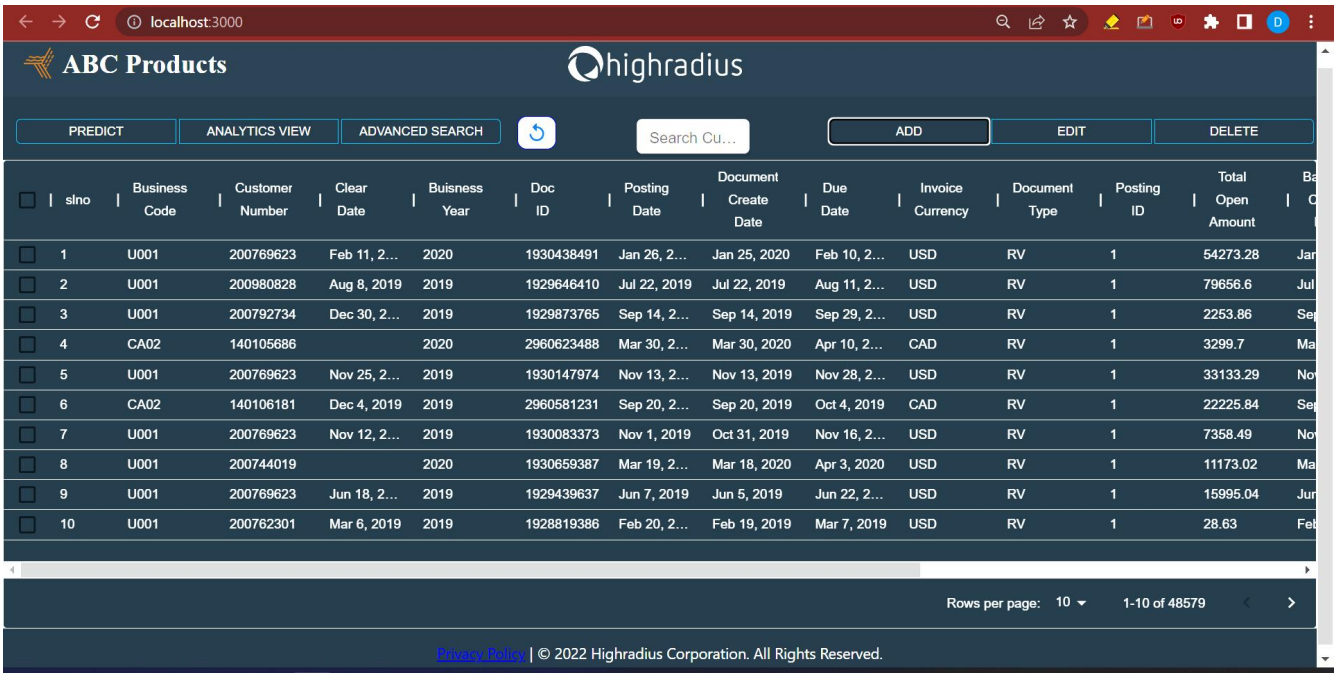

Figure 4.1 : Web Application UI landing page

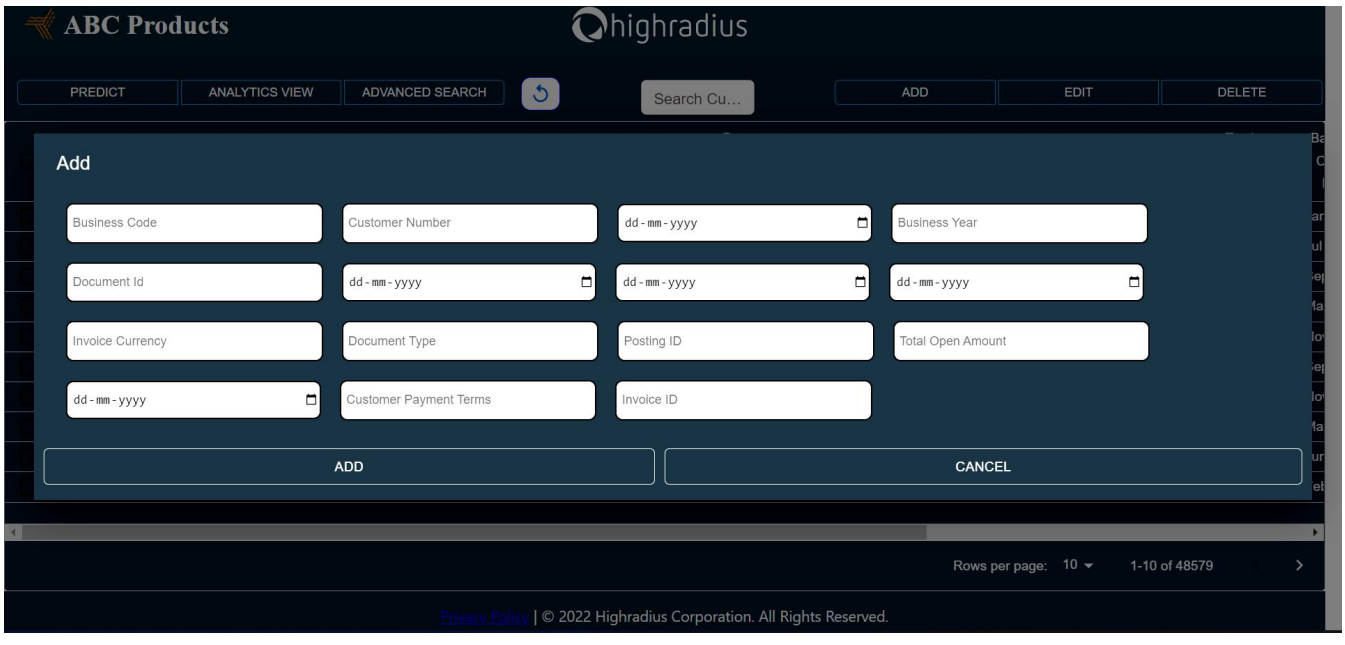

Figure 4.2 : Web Application UI- Add model

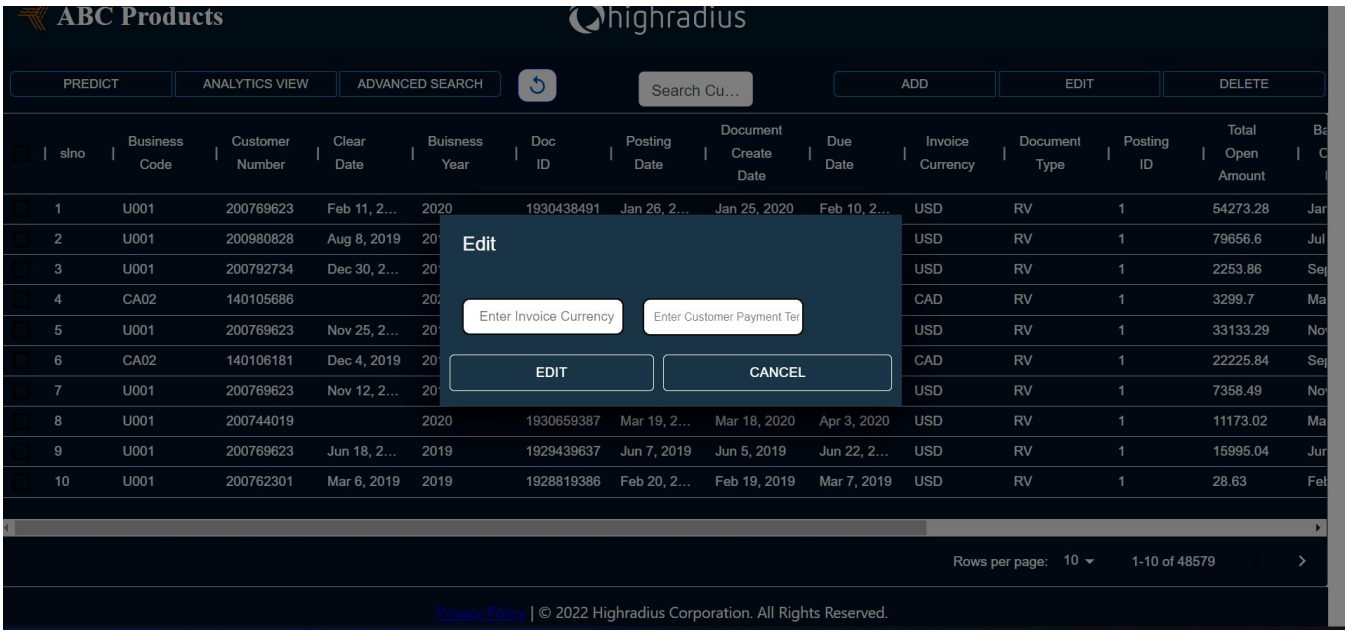

Figure 4.3: Web Application UI- Edit model

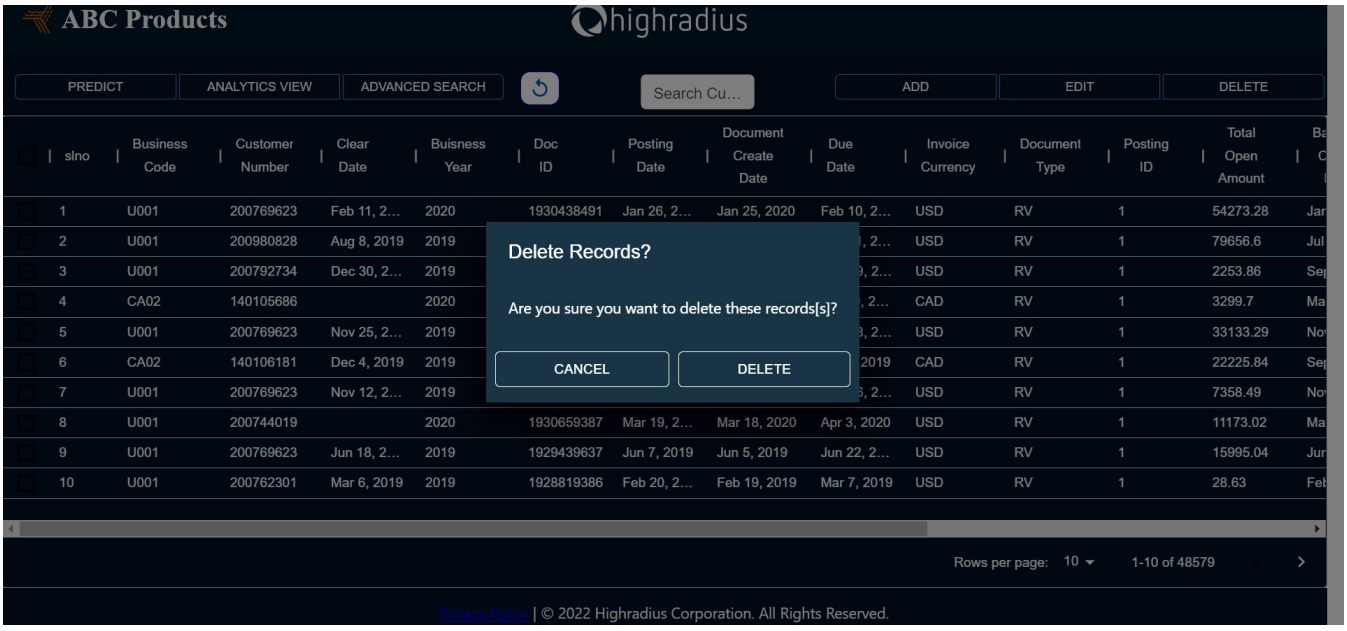

Figure 4.4: Web Application UI- Delete model

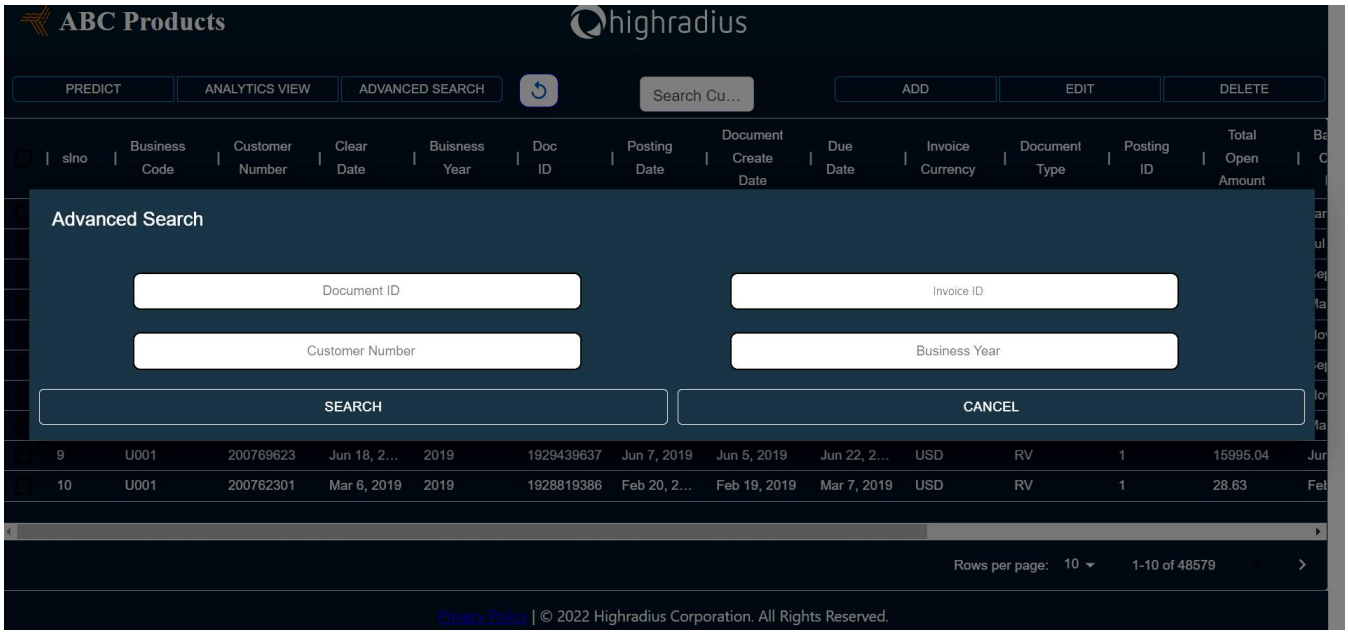

Figure 4.5: Web Application UI- Advance Search model

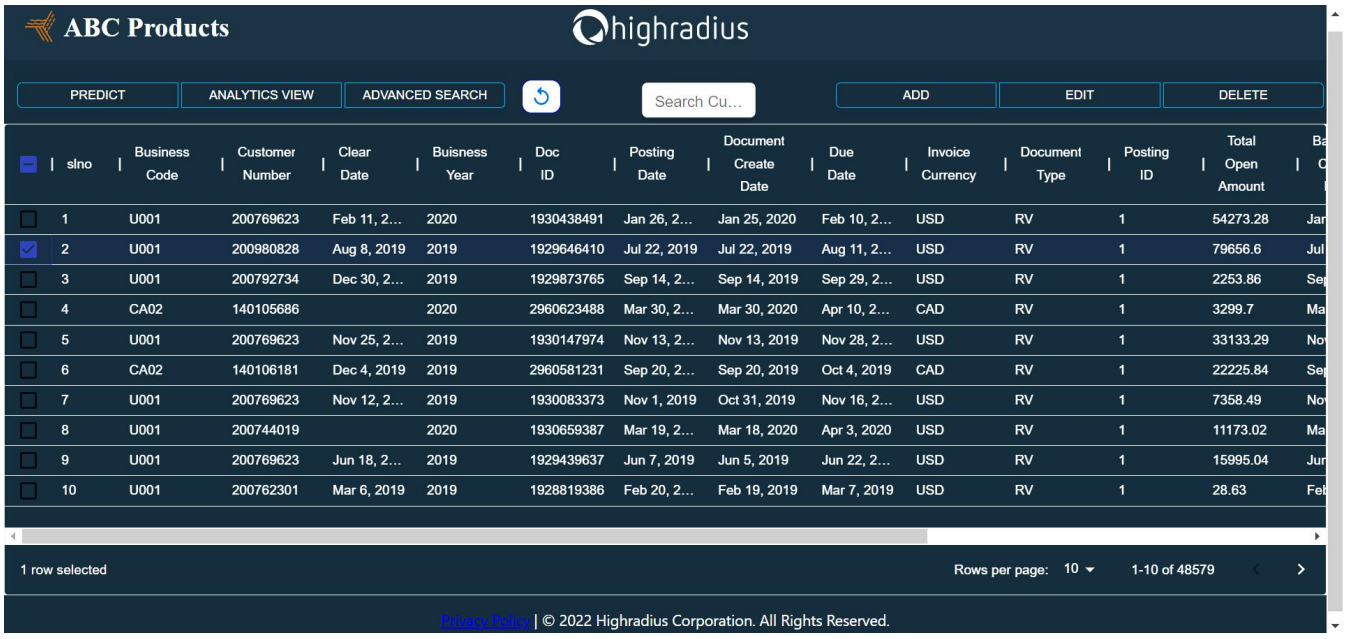

Figure 4.6: Web Application UI- Predict model

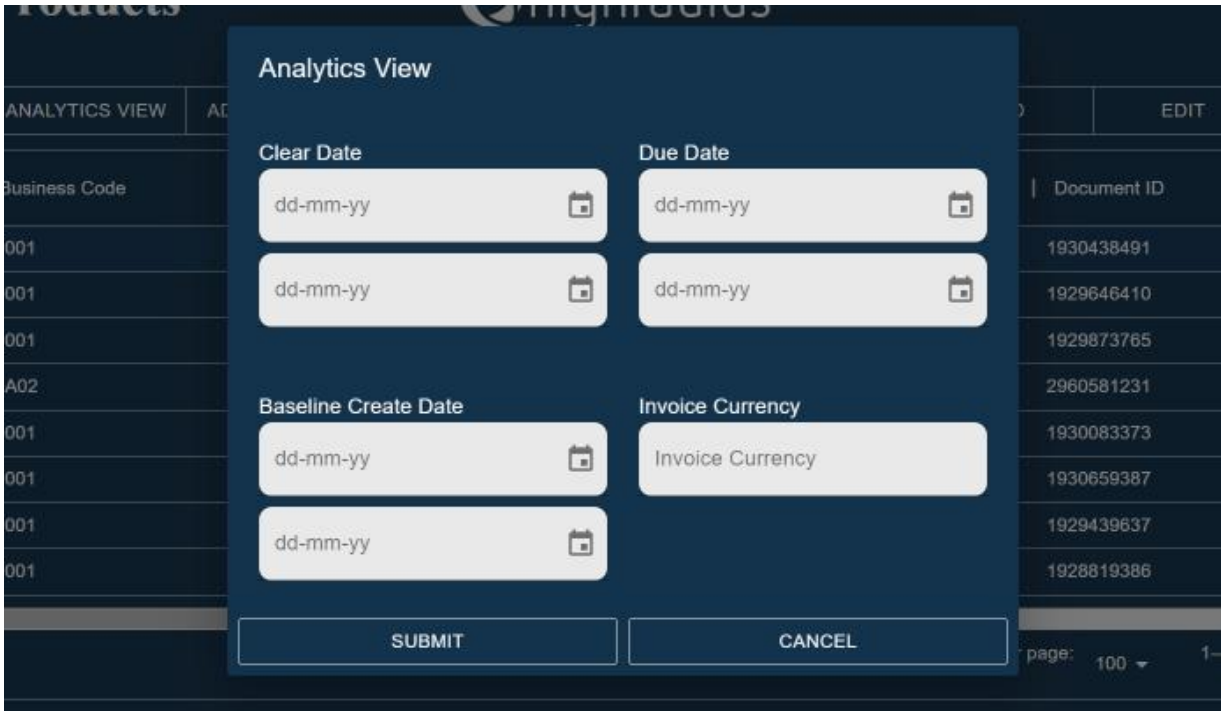

Figure 4.7: Web Application UI- Analytics View model

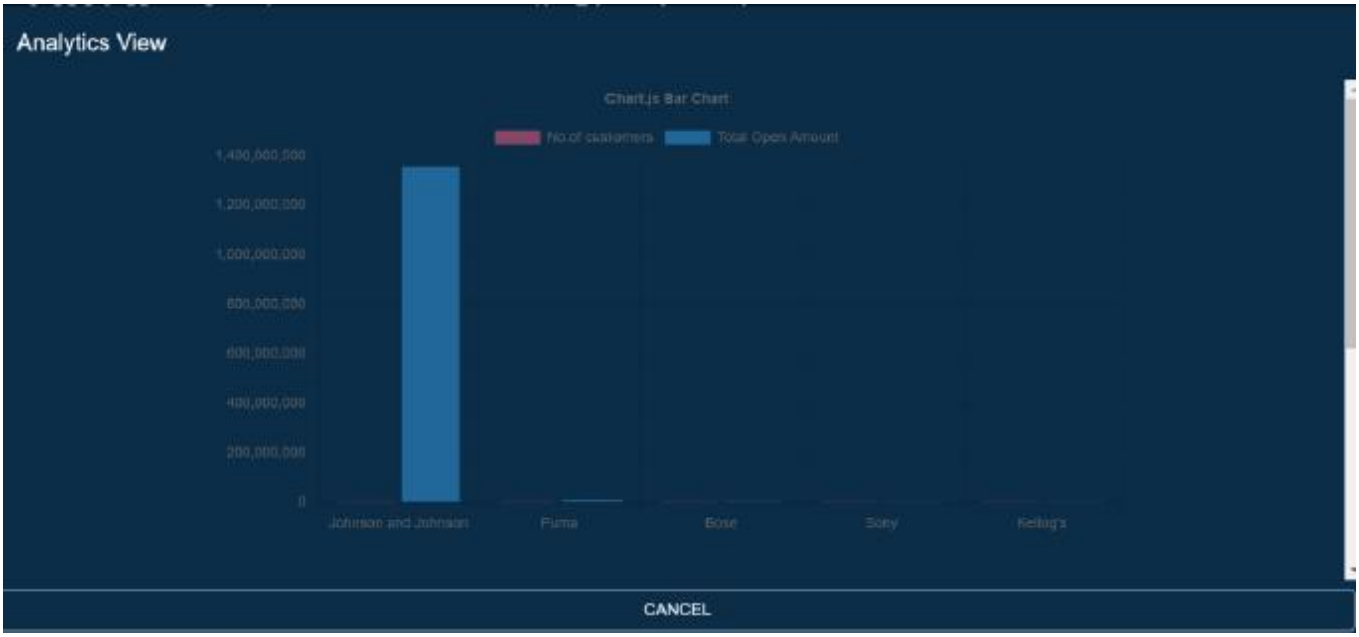

Figure 4.8: Web Application UI- Analytics View 02 model

CHAPTER 5

Conclusion & Future Scope

6.1 Conclusion

The project offers a comprehensive approach to receipt management, allowing sellers' receipt information to be stacked, viewed, edited, and deleted as needed. The internet application may predict the time frame during which the customer will pay the receipt amount thanks to AI features.

6.2 Future Scope

Implementing a computerized receipt processing system now can help support future commerce growth. If one of the invoices hasn't been paid, be prepared to send mail updates and alerts to remind clients. For the time being, our initiative is limited to USD/CAD; however, we are prepared to change the billable currency to allow any customer to accept more currencies in the future. RPA-based arrangements can help to automate the entire procedure. For customers, we may provide a self-service membership administration portal. Clients may use this portal to add, change, delay, or cancel memberships, retrieve previous solicitations, manage instalment methods, and manage their shipping/billing address.

References

[1] 1.https://www.scribbr.com/statistics/simple-linear-regression/
[2]https://towardsdatascience.com/unlocking-the-true-power-of-support-vector-regression-847fd123a4a0
[3] https://www.xoriant.com/blog/product-engineering/decision-trees-machine-learning-algorithm.html
[4].https://www.nvidia.com/en-us/glossary/data-science/xgboost/
[5] Barrett, Paul, et al. "matplotlib--A Portable Python Plotting Package." *Astronomical data analysis software and systems XIV*. Vol. 347. 2005.
[6]Porcu, Valentina. "Matplotlib." *Python for Data Mining Quick Syntax Reference*. Apress, Berkeley, CA, 2018. 201-234.
[7] Kadiyala, Akhil, and Ashok Kumar. "Applications of Python to evaluate environmental data science problems." *Environmental Progress & Sustainable Energy* 36.6 (2017): 1580-1586.
