

A project report on

Bill Management App, AI-Enabled, Regression

Submitted in partial fulfillment of the requirements for the **Degree of B.Tech in
Computer Science and Engineering**

by

Prince Chandra(1805142)

under the guidance of

Sailesh Dwivedy and Kishan Kumar

**School of Computer Engineering
Kalinga Institute of Industrial Technology
Deemed to be University
Bhubaneswar
March 2021**

CERTIFICATE

This is to certify that the project report entitled
“Bill Management App, AI-Enabled, Regression”
submitted by:
Prince Chandra(1805142)

In partial fulfillment of the requirements for the award of the **Degree of Bachelor of Technology in Discipline of Engineering** is a bonafide record of the work carried out under our guidance and supervision at **School of Computer Engineering, Kalinga Institute of Industrial Technology, Deemed to be University**.

The Project was evaluated by us on 26/03/2021: evaluation date.

ACKNOWLEDGEMENTS

I would like to show my gratitude towards **Sailesh Dwivedy** and **Kishan Kumar** for helping me throughout the project. I would also like to thank and express my kind gratitude towards **HighRadius** for allowing me to conduct a **Bill Management App, AI-Enabled, Regression** project. I express my thanks to **KIIT University, Deemed to be University** for allowing me to work on projects with companies and always supporting me at every step of the way. The mentioned project was done under the supervision of **Sailesh Dwivedy** and **Kishan Kumar**. I feel thankful to the college staff for giving me such a big opportunity. The responsibility for the final formulation, and any errors that it may concern, are entirely mine.

Prince Chandra
1805142

TABLE OF CONTENTS

Abstract

CHAPTER 1: INTRODUCTION

1.1 Bill Management App, AI-Enabled, Regression

1.1.1 Introduction

1.1.2 Problem

1.1.3 Definition

1.1.4 Purpose

1.1.5 Scope

CHAPTER 2: BACKGROUND

2.1 Basic Idea of the project

2.1.1 Domain Covered

2.1.2 About the Data

2.1.3 Technology Used

2.1.4 Idea behind the project

CHAPTER 3: PROJECT ANALYSIS

3.1 Project Implementation

3.1.1 Python

3.1.2 ML

- **Regression Analysis**
- **Support Vector Regression (SVR)**
- **Decision Tree**
- **Random Forest Regression**

3.1.3 Java

3.1.4 JDBC and COLLECTION

3.1.5 REACT

3.2 Project Methodology

3.2.1 BACK-END

3.2.2 FRONT-END

CHAPTER 4: RESULTS AND DISCUSSION

4.1 ML

4.1.1 Delay date prediction and bucketization

4.2 Java Servlets

4.3 User Interface with React

CHAPTER 5: CONCLUSION & FURTHER WORK

5.1 Problems faced while doing the project

5.2 Conclusion

5.3 Future Work

5.4 Planning And Project Management

ABSTRACT

In a B2B business work with each other is done on credit. When a buyer business orders some goods from a seller business, then the seller business issues an invoice for it which contains information like the details about the goods that are purchased and when they should be paid.

The seller business might have business orders from multiple businesses. It communicates with many businesses and sells goods to them at different times. So the seller business needs to keep a track of all invoices from all its buyers. Each of the invoices consists of fields of all details like invoice date, due date, payment date, invoice amount, etc. The buyer business has to clear the invoice amount before the due date. However, in a real-world scenario, the buyer business does not clear the full invoice amount by the due date. The date on which this full amount is cleared is called the Payment date.

The significance of my project is that it takes into account all previous invoice details of different buyer businesses and tries to see how many days after/before the due date they pay the invoice amount owed. After observing all the past invoices it tries to predict the new due date according to this data for the future invoices.

This prediction is carried out using machine learning algorithms and uses java for its frontend work. A new due date is given to the buyer business if it fails to pay the owed amount before the original due date according to the past transactions.

CHAPTER 1: INTRODUCTION

1.1 Bill Management App, AI-Enabled, Regression

INTRODUCTION

An Order Management System is a computer software program that allows businesses to manage their orders and inventory. Order management systems help ensure more accurate inventory management, automatically entering new inventory into the system, tracking sales through different selling platforms, such as eBay and Amazon, and alerting you, the business owner, when your stock of a particular item drops low enough for a re-order.

An order management system can also automate the order-to-cash process, beginning with the customer order through payment reconciliation, fulfillment, and shipment. Order management software can work for both B2B and B2C businesses of any size.

Order management software is also shareable, from the customer service team to the accounting team, the warehouse staff, and you, the business owner. The best kinds of inventory management systems also have an order management app, allowing you to manage your stock on the go and spot-check your business in real-time.

Effective order management improves the business workflow and increases the likelihood of repeat customers.

1.1.2 PROBLEM DEFINITION

For Machine Learning.

- To build a Machine Learning Model to **predict the payment date of an invoice** when it gets created in the system.
- **Categorize the invoice into different buckets based on the predicted payment date.**

For Java And React

- To build a full-stack Invoice Management Application using ReactJs, JDBC, Java, and JSP.
- Build a responsive Receivables Dashboard.
- Visualize Data in the form of grids.
- Perform Searching operations on the invoices.
- Edit data in the editable fields of the grid.
- Download data of selected rows in predefined template

1.1.3 PURPOSE

This model would help various companies to keep a record for their tractions with various other companies and help them to reduce monetary loss.

1.1.4 SCOPE

The accuracy of the model can be increased and the same can be deployed in the cloud to make it available to a larger audience.

CHAPTER 2: BACKGROUND

2.1 BASIC IDEA OF THE PROJECT

2.1.1 DOMAIN

The domains covered by our projects are:

- Machine Learning and data analysis
- Web-Development(front-end and back-end)

2.1.2 ABOUT THE DATA

Below is the sample CSV file screenshot.

business_code	cust_number	name_customer	clear_date	business_year	doc_id	posting_date	document_create_date	document_create_date	due_in_date	invoice_currency	document type	posting_id	area_business	total_open_amount	baseline_create_date	cust_payment_terms	invoice_id	isOpen
U013	140103335	PARAM	#####	2019	2E+09	05-08-2019	20190801	20190805	20190904	USD	RV	1		12635.05	20190805	NAVE	1.99E+09	0
U001	200704045	RA corp	#####	2019	2E+09	27-04-2019	20190426	20190427	20190512	USD	RV	1		20560.48	20190427	NAAB	1.93E+09	0
U001	200709623	WAL-MAR corpo	#####	2019	2E+09	10-06-2019	20190609	20190610	20190625	USD	RV	1		49878.77	20190625	NAAB	1.93E+09	0
U001	100058018	SOUTH in	#####	2019	2E+09	03-07-2019	20190703	20190703	20190804	USD	RV	1		1196	20190610	NAH4	1.93E+09	0
U001	200747369	SCHNU co	#####	2019	2E+09	15-12-2019	20191216	20191215	20191230	USD	RV	1		76760.13	20191215	NAAB	1.93E+09	0
U001	200152991	JET corp	#####	2019	2E+09	14-03-2019	20190315	20190314	20190329	USD	RV	1		3372.4	20190314	NAAB	1.93E+09	0
U001	200337148	COAS trust	#####	2020	2E+09	22-01-2020	20200122	20200122	20200206	USD	RV	1		16787.43	20200122	NAAB	1.93E+09	0
U001	200709623	WAL-MAR found	#####	2019	2E+09	01-08-2019	20190731	20190801	20190816	USD	RV	1		9420.73	20190801	NAH4	1.93E+09	0
U001	200706844	WINC in	#####	2019	2E+09	25-01-2019	20190125	20190125	20190209	USD	RV	1		19364.83	20190125	NAAB	1.93E+09	0
U001	200759878	SA	#####	2020	2E+09	26-02-2020	20200224	20200226	20200312	USD	RV	1		15005.29	20200226	NAH4	1.93E+09	0
U001	200777735	NASH	#####	2019	2E+09	26-08-2019	20190825	20190826	20190910	USD	RV	1		84275.32	20190826	NAAB	1.93E+09	0
U001	200726979	BI'S corporation	#####	2019	2E+09	30-08-2019	20190829	20190830	20190914	USD	RV	1		385.57	20190830	NAAB	1.93E+09	0
U001	200439158	POST foundation	#####	2019	2E+09	10-05-2019	20190510	20190510	20190525	USD	RV	1		33942	20190510	NAAB	1.93E+09	0
U001	200718130	SYSCO F systems	#####	2019	2E+09	08-05-2019	20190506	20190508	20190609	USD	RV	1		8746.75	20190508	NA32	1.93E+09	0
U001	CCCA02	KRAFT trust	#####	2019	2E+09	22-02-2019	20190219	20190222	20190329	USD	RV	1		15642.72	20190222	NAG2	1.93E+09	0
U001	200020431	DEC systems	#####	2019	2E+09	04-10-2019	20191004	20191004	20191024	USD	RV	1		601.8	20191001	NAH4	1.93E+09	0
U001	200703398	NEX corporation	#####	2019	2E+09	30-01-2019	20190130	20190130	20190214	USD	RV	1		142.56	20200109	NAAB	1.93E+09	0
U001	200777735	NASH foundation	#####	2020	2E+09	09-01-2020	20200108	20200109	20200124	USD	RV	1		117742.68	20190222	NAG2	1.93E+09	0
U001	200709623	WAL-MAR found	#####	2020	2E+09	30-01-2020	20200130	20200130	20200214	USD	RV	1		1695.68	20200130	NAH4	1.93E+09	0
U001	100031483	OCEAN FA co	#####	2019	2E+09	09-01-2019	20190105	20190109	20190124	USD	RV	1		15965.6	20190109	NAAB	1.93E+09	0
U001	200743123	KROGER us	#####	2019	2E+09	09-08-2019	20190809	20190809	20190824	USD	RV	1		17813.81	20190809	NAAB	1.93E+09	0
U001	200726979	BI'S co	#####	2019	2E+09	04-04-2019	20190403	20190404	20190419	USD	RV	1		513.78	20190404	NAAB	1.93E+09	0
U001	200729942	SA trust	#####	2019	2E+09	12-09-2019	20190910	20190912	20190927	USD	RV	1		13660.32	20190912	NAAB	1.93E+09	0
U001	200705742	DOT foundation	#####	2020	2E+09	27-01-2020	20200127	20200127	20200228	USD	RV	1		98396.24	20200127	NA32	1.93E+09	0
U001	200148860	DOLLA corp	#####	2019	2E+09	19-03-2019	20190318	20190319	20190523	USD	RV	1		7438.84	20190319	NAGD	1.93E+09	0
U001	200466603	SMITH'S in	#####	2020	2E+09	30-12-2019	20191229	20191230	20200114	USD	RV	1		126143.39	20191230	NAAB	1.93E+09	0
U001	100043892	IN-N us	#####	2019	2E+09	13-02-2019	20190213	20190213	20190228	USD	RV	1		16503.48	20190213	NAAB	1.93E+09	0
U001	CCU013	KRAFT F	#####	2020	2E+09	10-02-2020	20200210	20200210	20200210	USD	RV	1		23342.7	20200210	NAX2	1.93E+09	0
U001	200148860	DOLLA in	#####	2019	2E+09	16-01-2019	20190116	20190116	20190131	USD	RV	1		108527	20190116	NAAB	1.93E+09	0
U001	200759878	SA us	#####	2019	2E+09	15-12-2019	20191214	20191215	20191230	USD	RV	1		14624.73	20191215	NAH4	1.93E+09	0

- 1) business_code - company code of the account
- 2) cust_number - customer number is given to all the customers of the Account
- 3) name_customer - the name of the customer.
- 4) cust_number - Each customer has a number that uniquely identifies it.
- 5) document_create_date - The date on which the invoice document was created
- 6) document_create_date_norm - Normalised date of the invoice document
- 7) posting_id - a key indicator to identify whether an AR item is an invoice.
- 8) due_in_date - The date on which the customer is expected to clear an invoice
- 9) invoice_id - Unique number assigned when a seller creates an Invoice.
- 10) baseline_create_date - The date on which the Invoice was created.
- 11) total_open_amount - The amount that is yet to be paid for that invoice
- 12) invoice_amount - The total amount for that invoice.
- 13) cust_payment_terms -Business terms and agreements between customers and accounts on discounts and days of payment

- 14) `area_business` - A business area in sap is defined as an organizational area within the financial accounting module.
- 15) `clear_date` - The date on which the customer clears an invoice, or in simple terms, they make the full payment.
- 16) `is_open_invoice` - an indicator of whether an invoice is open or closed.
- 17) `invoice_currency` - The currency of the invoice amount in the document for the invoice.
- 18) `doc_id` - It is also a unique identifier of an invoice and is a primary key.
- 19) `total_open_amount` - the open amount of an invoice

2.1.3 TECHNOLOGY

The technology used by our projects are:

- Machine Learning using Python
- Java, SQL(Backend)
- HTML,CSS,Javascript,React(Frontend)
- Flask

2.1.4 Brief Idea behind the project

The B2B world operates differently from the B2C or C2C world. Businesses work with other businesses on credit. When a buyer business orders goods from the seller business, the seller business issues an invoice for the same. This invoice for the goods contains various information like the details of the goods purchased and when this is given the should be paid.

The simplest definition of accounts receivable is money owed to an entity by its customers. Correspondingly, the amount not yet received is credit and, of course, the amount still owed past the due date is Account receivables.

A more formal definition of A/R is:

“Accounts Receivable represents money owed by entities to the firm on the sale of products or services on credit. In most business entities, accounts receivable is typically executed by generating an invoice and either mailing or electronically delivering it to the customer; who, in turn, must pay it within an established timeframe, called credit terms or payment terms.”

Seller business interacts with various businesses and sells goods to all of them at various times. Hence, the seller business needs to keep track of the total amount it owes from all the

buyers. This involves keeping track of all invoices from all the buyers. Each invoice will have various important fields like a payment due date, invoice date, invoice amount, baseline date, etc.

The buyer business needs to clear its amount due before the due date. However, in real-world scenarios, the invoices are not always cleared ie. paid in full amount by the due date. The date on which a customer clears the payment for an invoice is called the payment date.

CHAPTER 3: PROJECT IMPLEMENTATION

3.1 PROJECT IMPLEMENTATION

3.1.1 Python-

Python is an object-oriented, high-level programming, interpreted language with dynamic semantics. Python has high-level built-in data structures, along with dynamic typing and dynamic binding, which helps python make it extremely attractive for Rapid Application Development, also it is used as a scripting language to connect the existing components. Python has an easy and simple syntax base which in turn enhances and emphasizes the readability of the code and in turn reduces the cost of program maintenance. Python also supports modules and packages, which encourages program modularity and code reusability. The Python interpreter and the standard libraries are available in source or binary form for all major platforms and can be freely distributed.

What are the Python Libraries-

A module is a file with some Python code written in it, and a package is a directory for sub-packages and modules. A Python library is a reusable bunch of code that we can include in our programs/ projects.

If we Compare python to languages like C++ or C, Python libraries do not relate to any particular context in Python. Here, a 'library' describes a collection of core modules.

A library is a collection of modules. A package is a library that can be installed using a package installer like ruby gems or npm.

The Python Standard Library is a collection of exact syntax, tokens, and semantics of Python. It comes bundled with core Python distribution. It is written in C language and handles functionalities like Input/Output and other core modules. All these functionalities together make Python the language it is.

More than 200 modules are their standard library. This library ships with Python.

Numpy Arrays -

NumPy stands for Numerical Python. It is the core library that is used for scientific computing in Python. It comprises multidimensional array objects and tools which helps in working with these arrays.

Numpy Array is a collection of values in order with the same type and is indexed by a tuple of non-negative integers. The number of dimensions is the rank of the array; the shape of an array depends upon a tuple of integers giving the size of the array along each dimension.

Pandas-

Panda is the most famous Python module for machine learning and data science, which offers useful, powerful, and flexible data structures that make analysis and manipulation of data easy. Pandas make the importing and analyzing of data much simpler. Pandas is built

on modules like NumPy and matplotlib to give us a single & very convenient place for data analysis and visualization works.

Basic Features of Pandas-

Dataframe objects help us in keeping track of our data.

With pandas data frame, we can have different data types like (float, int, string, DateTime, etc) all in one place.

Pandas package has built-in functionalities like easy grouping & easy joins of data and rolling windows. It has Good Input/Output capabilities; it can easily pull data from a MySQL database directly into a data frame.

With pandas, we can use patsy for R-style syntax in doing regressions.

Pandas Series -

Panda Series is like a 1-D array that is capable of holding data of any data type such as (integer, string, float, python objects, etc.). Panda Series can be created using a constructor.

Syntax:- `pandas.Series(data, index, dtype, copy)`.

3.1.2 Machine Learning-

Machine learning (ML) is defined as the study of computer algorithms that improve their performance and results automatically through long-term experience and by the use of data sets. ML is a subset of artificial intelligence. Machine learning algorithms build a model which is based on sample data for testing, known as "training data", in to make predictions, decisions or analysis without being explicitly programmed for the particular. Machine learning algorithms are used in many applications, such as in the medical field, filtering of emails, and computer vision, where it is very difficult to predict the result or develop conventional algorithms to perform the required tasks. A subset of machine learning is nearly related to computational statistics, and that focuses on making decisions and predictions using computers; but not all machine learning includes statistical learning. The study of mathematical optimization delivers the theory portion and application domains in the field of machine learning. Data mining is also a related field of study, and this focuses on exploratory data analysis through unsupervised learning. In its business application problems, machine learning is also termed predictive analytics.

ML MODEL-

While building this model we had gone through many processes:

Step 1. Firstly we need to import the dataset and also import various data manipulation libraries.

Step 2. We visualized the dataset in the form of a data frame to get a brief idea about the

dataset.

Step 3. Identified the target variable which is clear_date.

Step 4. Independent and dependent variables identification and extraction.

Step 5. Handling Missing Values using Null Imputation Techniques.

Step 6. Encoding Categorical Variables for training purposes.

Step 7. Splitting the dataset for train/test and validation.

Step 8. Feature Scaling for improved training using Normalization and Standardisation.

Step 9. Training and Validating ML model.

Step 10. Predict clear_date using test data.

Step 11. Prepare aging bucket by subtracting invoice creation date from predicted clear date.
The different buckets were :

- 1) 0-15 days
- 2) 16-30 days
- 3) 31-45 days
- 4) 46-60 days
- 5) Greater than 60 days

Regression Analysis-

In statistical modeling, regression analysis is a set of statistical processes for estimating the relationships between a dependent variable and one or more independent variables. The most common form of regression analysis is linear regression, in which one finds the line that most closely fits the data according to a specific mathematical criterion. For example, the method of ordinary least squares computes the unique line that minimizes the sum of squared differences between the true data and that line. For specific mathematical reasons, this allows the researcher to estimate the conditional expectation of the dependent variable when the independent variables take on a given set of values. Less common forms of regression use slightly different procedures to estimate alternative location parameters or estimate the conditional expectation across a broader collection of non-linear models. Regression analysis is primarily used for two conceptually distinct purposes. First, regression analysis is widely used for prediction and forecasting, where its use has substantial overlap with the field of machine learning. Second, in some situations, regression analysis can be used to infer causal relationships between the independent and dependent variables. Importantly, regressions by themselves only reveal relationships

between a dependent variable and a collection of independent variables in a fixed dataset. To use regressions for prediction or to infer causal relationships, respectively, a researcher must carefully justify why existing relationships have predictive power for a new context or why a relationship between two variables has a causal interpretation. The latter is especially important when researchers hope to estimate causal relationships using observational data.

Linear Regression-

Linear regression attempts to model the relationship between two variables by fitting a linear equation to observed data. One variable is considered to be an explanatory variable, and the other is considered to be a dependent variable. For example, a modeler might want to relate the weights of individuals to their heights using a linear regression model.

Before attempting to fit a linear model to observed data, a modeler should first determine whether or not there is a relationship between the variables of interest. This does not necessarily imply that one variable *causes* the other (for example, higher SAT scores do not *cause* higher college grades), but that there is some significant association between the two variables. A scatter plot of the A business based on can be a helpful tool in determining the strength of the relationship between two variables. If there appears to be no association between the proposed explanatory and dependent variables (i.e., the scatter plot does not indicate any increasing or decreasing trends), then fitting a linear regression model to the data probably will not provide a useful model. A valuable numerical measure of association between two variables is the correlation concept, which is a value between -1 and 1 indicating the strength of the association of the observed data for the two variables.

A linear regression line has an equation of the form $Y = a + bX$, where X is the explanatory variable and Y is the dependent variable. The slope of the line is b , and a is the intercept (the value of y when $x = 0$).

Regularization-

There are extensions of the training of the linear model called regularization methods. These seek to both minimize the sum of the squared error of the model on the training data (using ordinary least squares) but also to reduce the complexity of the model (like the number or absolute size of the sum of all coefficients in the model).

Two popular examples of regularization procedures for linear regression are:

Lasso Regression: where Ordinary Least Squares is modified to also minimize the absolute sum of the coefficients (called L1 regularization).

Ridge Regression: where Ordinary Least Squares is modified to also minimize the squared absolute sum of the coefficients (called L2 regularization).

These methods are effective to use when there is collinearity in your input values and ordinary least squares would overfit the training data.

Now that you know some techniques to learn the coefficients in a linear regression model, let's look at how we can use a model to make predictions on new data.

Support Vector Regression (SVR)-

Support Vector Machines (SVMs) are well known in classification problems. The use of SVMs in regression is not as well documented, however. These types of models are known as Support Vector Regression (SVR). Support Vector Regression (SVR) uses the same principle as SVM, but for regression problems. Let's spend a few minutes understanding the idea behind SVR.

The problem of regression is to find a function that approximates mapping from an input domain to real numbers based on the training sample. So let's now dive deep and understand how SVR works actually. Consider these two red lines as the decision boundary and the green line as the hyperplane. Our objective, when we are moving on with SVR, is to consider the points that are within the decision boundary line. Our best fit line is the hyperplane that has a maximum number of points.

The Support Vector Regression (SVR) uses the same principles as the SVM for classification, with only a few minor differences. First of all, because the output is a real number it becomes very difficult to predict the information at hand, which has infinite possibilities. In the case of regression, a margin of tolerance (epsilon) is set in approximation to the SVM which would have already been requested from the problem. But besides this fact, there is also a more complicated reason, the algorithm is more complicated therefore to be taken into consideration. However, the main idea is always the same: to minimize error, individualizing the hyperplane which maximizes the margin, keeping in mind that part of the error is tolerated

Decision Tree-

A Decision Tree is a decision-making tool that uses a flowchart-like tree structure or is a model of decisions and all of their possible results, including outcomes, input costs, and utility.

Decision-tree algorithms fall under the category of supervised learning algorithms. It works for both continuous as well as categorical output variables.

The branches/edges represent the result of the node and the nodes have either:

Conditions [Decision Nodes]

Result [End Nodes]

The branches/edges represent the truth/falsity of the statement and make a decision based on that in the example below which shows a decision tree that evaluates the smallest of three numbers.

Decision Tree Regression-

Decision tree regression observes features of an object and trains a model in the structure of a tree to predict data in the future to produce meaningful continuous output. Continuous output means that the output/result is not discrete, i.e., it is not represented just by a discrete, known set of numbers or values.

Discrete output example: A weather prediction model that predicts whether or not there'll be rain on a particular day.

Continuous output example: A profit prediction model that states the probable profit that can be generated from the sale of a product.

Here, continuous values are predicted with the help of a decision tree regression model.

Random Forest Regression-

A Random Forest is an ensemble technique capable of performing both regression and classification tasks with the use of multiple decision trees and a technique called Bootstrap Aggregation, commonly known as bagging. Bagging, in the Random Forest method, involves training each decision tree on a different data sample where sampling is done with replacement. The basic idea behind this is to combine multiple decision trees in determining the final output rather than relying on individual decision trees. If you want to read more on Random Forests, I have included some reference links which provide in-depth explanations on this topic.

First, we pass the features(X) and the dependent(y) variable values of the data set, to the method created for the random forest regression model. We then use the grid search cross-validation method from the sklearn library to determine the optimal values to be used for the hyperparameters of our model from a specified range of values. Here, we have chosen the two hyperparameters; *max_depth* and *n_estimators*, to be optimized. According to sklearn documentation, *max_depth* refers to the maximum depth of the tree and *n_estimators*, the number of trees in the forest. Ideally, you can expect a better performance from your model when there are more trees. However, you must be cautious of the value ranges you specify and experiment using different values to see how your model performs. After creating a random forest regressor object, we pass it to the `cross_val_score()` function which performs K-Fold cross-validation disabled should contain a is given the based on disabled a disabled should download button the matching function has passed button the matching a function has passed provides as an output, an error metric value, which can be used to determine the model performance.

3.1.3 Java-

Java is a programming language and a platform. Java is a high-level, robust, object-oriented, and secure programming language.

Java was developed by Sun Microsystems (which is now the subsidiary of Oracle) in the year 1995. James Gosling is known as the father of Java. Before Java, its name was Oak. Since Oak was already a registered company, James Gosling and his team changed the Oak name to Java.

Why use Java?

- Java works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc.)
- It is one of the most popular programming languages in the world
- It is easy to learn and simple to use
- It is open-source and free
- It is secure, fast, and powerful
- It has huge community support (tens of millions of developers)
- Java is an object-oriented language that gives a clear structure to programs and allows code to be reused, lowering development costs
- As Java is close to C++ and C#, it makes it easy for programmers to switch to Java or vice versa

Java OOP Concepts-

Object means a real-world entity such as a pen, chair, table, computer, watch, etc. Object-Oriented Programming is a methodology or paradigm to design a program using classes and objects. It simplifies software development and maintenance by providing some concepts:

- ◆ Object
- ◆ Class
- ◆ Inheritance
- ◆ Polymorphism
- ◆ Abstraction
- ◆ Encapsulation

Object-

Any entity that has a state and behavior is known as an object. For example, a chair, pen, table, keyboard, bike, etc. It can be physical or logical.

An Object can be defined as an instance of a class. An object contains an address and takes up some space in memory. Objects can communicate without knowing the details of each other's data or code. The only necessary thing is the type of message accepted and the type of response returned by the objects.

Example: A dog is an object because it has states like color, name, breed, etc. as well as behaviors like wagging the tail, barking, eating, etc.

Class-

The collection of objects is called class. It is a logical entity.

A class can also be defined as a blueprint from which you can create an individual object. Class doesn't consume any space.

Inheritance-

When one object acquires all the properties and behaviors of a parent object, it is known as inheritance. It provides code reusability. It is used to achieve runtime polymorphism.

Polymorphism-

If one task is performed in different ways, it is known as polymorphism. For example: to convince the customer differently, to draw something, for example, shape, triangle, rectangle, etc.

In Java, we use method overloading and method overriding to achieve polymorphism. Another example can be to speak something; for example, a cat speaks meow, dog barks woof, etc.

Abstraction-

Hiding internal details and showing functionality is known as abstraction. For example phone calls, we don't know the internal processing.

In Java, we use abstract class and interface to achieve abstraction.

Encapsulation-

Binding (or wrapping) code and data together into a single unit are known as encapsulation. For example, a capsule is wrapped with different medicines.

A java class is an example of encapsulation. Java bean is the fully encapsulated class because all the data members are private here.

3.1.4 JDBC And Collections-

JDBC-

JDBC stands for Java Database Connectivity. JDBC is a Java API to connect and execute the query with the database. JDBC API uses JDBC drivers to connect with the database. There are four types of JDBC drivers:

- JDBC-ODBC Bridge Driver
- Native Driver
- Network Protocol Driver
- Thin Driver

Purpose of JDBC

The purposes of using JDBC are as follows-

- a) Establish a connection with a data source
- b) Send queries and update statements to the data source
- c) Process the results

The Java application calls JDBC classes and interfaces to submit SQL statements and

retrieve results.

The JDBC API is implemented through the JDBC driver. The JDBC Driver is a set of classes that implement the JDBC interfaces to process JDBC calls and return result sets to a Java application. The main objects of JDBC API are the DataSource object, Connection object, Statement, PreparedStatement, and CallableStatement objects.

Collections-

Any group of individual objects which are represented as a single unit is known as the collection of the objects. The Collection interface (java.util.Collection) and Map interface (java.util.Map) are the two main “root” interfaces of Java collection classes.

Before the Collection Framework was introduced, the standard methods for grouping Java objects (or collections) were Arrays or Vectors, or Hash Tables. So in a way, we can say that collections are synonymous with Arrays, Vectors, or Hash Tables. The different collection objects in java are ArrayList, LinkedList, Vector, HashSet, LinkedHashSet, TreeSet, HashMap, TreeMap, LinkedHashMap, Hashtable.

Purpose of Collections:-

The purposes of using JAVA collections are as follows:-

- Arrays are not resizable. Java Collections Framework provides lots of different useful data types, such as hash sets, hashmaps, and array lists which are resizable.
- Java Collections Framework provides abstractions, so you can refer to a list as a List, whether backed by an array list or a linked list.
- New data structures that conform to the standard collection interfaces are by nature reusable. The same goes for new algorithms that operate on objects that implement these interfaces.

3.1.5 React-

React is the world’s most popular javascript framework maintained by Facebook and a community of developers. It is used for building UI components. Most of the front-end parts of a project can be made efficient with react. It is used mostly because of how simple and efficient it makes for the user to make interactive UI components. It has the following advantages:

1. It is very simple. It helps us to design simple views for each of our states in our project and it renders only the essential components when there is any change in our data.
2. It helps us to keep our states outside the DOM and build complex UIs with the help of encapsulated components that manage their states.
3. We can develop new features without rewriting existing code.

React creates a virtual DOM in memory-In place of changing the browser directly react creates a virtual DOM in memory where it does the necessary manipulating before making any changes to the browser DOM.

React can be learned using HTML and requires Node js and NPM to be started. We will be using mostly ES6 for our project.

Components-

Components are the building blocks of any react app and multiple components can be seen in a react project. React components serve the same purpose as that of javascript functions but work in isolation and return HTML through the render() function.

A Higher-Order Component is an advanced technique in React for reusing the component logic. They are a pattern that emerges from React's compositional nature. The use of Higher Order of Components when you are architecturally ready for separating container components from presentation components.

Pure Components in React are the components that do not re-renders when the value of state and props have been updated with the same

Values. If the value of the previous state and props and the new state or props is the same the component is not re-rendered. It is used mostly for performance optimization. If your react's component render() function renders the same result given the same props and state you can use React. Pure components for a performance boost in some cases.

Components are like Javascript Functions-they accept arbitrary inputs and return and return a react element describing that should appear on the screen.

render()-takes input and returns what to display.

Components are divided into two types

Functional Components-

- A functional Component is just a plain component is just a plain Javascript function that accepts props as input and returns a React element.
- There is no render method used in the Functional Component.
- Known as stateless components because they can simply accept data and display them in some form, that they are mainly responsible for rendering UI.
- React lifecycle methods cannot be used in functional components.

Class components-

- A class component requires you to extend from React. Component and create a render function that returns a React element.
- It must have the render() method returning HTML.
- Known as Stateful component because they can implement logic and state.
- React lifecycle methods can be used inside class components.

STATE-

A state is an instance of a react component class that can be defined as an object of a set of observable properties that control the behavior of the component. In other words, the state of a component is an object that holds some information that may change over the lifetime of the component

- State of a component should prevail throughout the lifetime, thus we must first have some initial state, to do so we should define the State in the constructor of the component's class.
- State should never be updated explicitly. React uses an observable object as the state that observes what changes are made to the state and helps the component behave accordingly.
- React provides its own method `setState()`. `setState()` method takes a single parameter and expects an object which should contain the set of values to be updated. Once the update is done the method implicitly calls the `render()` method to repaint the page.
- The only time we are allowed to define the state explicitly is in the constructor to provide the initial state.
- React is highly efficient and thus uses asynchronous state updates i.e. React may update multiple `setState()` updates in a single go. Thus using the value of the current state may not always generate the desired result.
- State updates are independent. The state object of a component may contain multiple attributes and React allows using the `setState()` function to update only a subset of those attributes as well as using multiple `setState()` methods to update each attribute value independently.

PROPS-

It is an object which stores the value of attributes of a tag and works similar to the HTML attributes. It gives a way to pass data from one component to other components. It is similar to function arguments. Props are passed to the component in the same way as arguments passed in a function. In React js we use props to send data components and every component is treated as a pure javascript function where props are equivalent to parameters of pure javascript functions.

Props are immutable so we cannot modify the props from inside the component. Inside the components, we can add attributes called props. These attributes are available in the component as `this.props` and can be used to render dynamic data in our render method.

When you need immutable data in the component, you have to add props to `ReactDOM.render()` method in the `main.js` file of your ReactJS project and used it inside the component in which you need.

HOOKS-

Hooks are the new feature that allows you to use state and other React features without writing a class. Hooks are the functions which "hook into" React state and lifecycle features from function components. It does not work inside classes. If you write a function component, and then you want to add some state to it, previously you do this by converting it to a class. But, now you can do it by using a Hook inside the existing function component. Some regulations to be taken care of while using hooks:

- We should not call hooks inside any loops, functions, or conditions. Hooks should be at the top level of functions.
- You cannot call Hooks from regular JavaScript functions. Instead, you can call Hooks from React function components.

REDUX-

Redux is a state management tool. With redux, the state of the application is kept in a store, and each component can access any state that it needs from this store. It allows React components to read data from a Redux Store, and dispatch Actions to the Store to update data. Redux helps apps to scale by providing a sensible way to manage state through a unidirectional data flow model. React Redux is conceptually simple. It subscribes to the Redux store, checks to see if the data which your component wants have changed, and re-renders your component.

So primary reasons for using redux are:

- It keeps up to date with any API changes to ensure your react components behave similarly throughout.
- encourages good react architecture.
- implements many performance optimizations internally, which allows components to re-render only when it needs to.

The Components of react-redux architecture are

- Store: A Store is a place where the entire state of your application lists. It manages the status of the application and has a `dispatch(action)` function.
- Action: It is sent or dispatched from the view which are payloads that can be read by Reducers. It is a pure object created to store the information of the user's event. It includes information such as type of action, time of occurrence, location of occurrence, its coordinates, and which state it aims to change.
- Reducer: The reducer reads the payloads from the actions and then updates the store via the state accordingly. It is a pure function to return a new state from the initial state.

AXIOS-

Axios is promise-based, which gives you the ability to take advantage of JavaScript's `async` and `await` for more readable asynchronous code.

The GET method requests a representation of the specified resource.

The POST method is used to submit an entity to the specified resource, often causing a change in state or side effects on the server.

A Promise object is simply a wrapper around a value that may or may not be known when the object is instantiated and provides a method for handling the value after it is known (also known as resolved) or is unavailable for a failure reason (we'll refer to this as rejected).

An async function is a function declared with the `async` keyword, and the `await` keyword is permitted within them. The `async` and `await` keywords enable asynchronous, promise-based behavior to be written in a cleaner style, avoiding the need to explicitly configure promise chains.

Axios is designed to handle HTTP requests and responses. It's used more often than Fetch because it has a larger set of features and it supports older browsers.

3.1 PROJECT METHODOLOGY

3.2.1 Backend:

Data Loading in the Database:

Step 1: Execute the SQL script for the creation of a table.

Step 2: Read the CSV datasheet using a CSV reader and stored information.

Step 3: We used a JDBC driver and made a POJO class, which helped us load the datasheet into batches.

business_code	cust_number	name_customer	clear_date	business_year	doc_id	posting_date	document_create_date	due_in_date	invoice_currency	document_type	posting_id	area_business	total_open_amount	baseline_create_date	cust_payment_terms	invoice_id	isoOpen
0001	0200455131	TIMES corp	2019-01-16 00:00:00	2019	1929501742	2018-12-01	2019-01-01	2019-01-16	USD	RV		1 (NULL)	1191.32	2019-01-01	MAA	1929501740	
0001	CC0013	KRAFT F us	2019-03-01 00:00:00	2019	1929509245	2017-11-02	2018-12-30	2018-12-30	USD	RV		1 (NULL)	152.1	2018-12-30	MAA	1929509184	
0001	0200357714	US systems	2019-01-25 00:00:00	2019	1929509679	2018-12-01	2019-01-02	2019-01-22	USD	RV		1 (NULL)	3720.76	2019-01-02	MAA	1929509024	
0001	CCCA02	KRAFT us	2019-03-01 00:00:00	2019	1929511027	2018-12-01	2019-01-04	2019-02-08	USD	RV		1 (NULL)	12804.69	2019-01-04	MAA	1929510976	
0001	CCCA02	KRAFT systems	2019-03-01 00:00:00	2019	1929511059	2017-11-02	2018-12-31	2019-02-04	USD	RV		1 (NULL)	11418.96	2018-12-31	MAA	1929511104	
0001	0200706844	WMC	2019-01-14 00:00:00	2019	1929512776	2017-11-02	2018-12-30	2019-01-14	USD	RV		1 (NULL)	20651.54	2018-12-30	MAA	1929512768	
0001	0200799367	MCL corp	2019-01-17 00:00:00	2019	1929515459	2018-12-01	2019-01-02	2019-01-17	USD	RV		1 (NULL)	18032.6	2019-01-02	MAA	1929515456	
0001	0100001196	DOLLAR trust	2019-01-14 00:00:00	2019	1929515649	2017-11-02	2018-12-31	2019-01-15	USD	RV		1 (NULL)	36493.75	2018-12-31	MAA	1929515712	
0001	0200290370	BARGAIN in	2019-01-16 00:00:00	2019	1929515784	2017-11-02	2018-12-31	2019-01-15	USD	RV		1 (NULL)	7257.6	2018-12-31	MAA	1929515840	
0001	0100045207	NET trust	2019-01-17 00:00:00	2019	1929516142	2018-12-01	2019-01-02	2019-01-17	USD	RV		1 (NULL)	5716.96	2019-01-02	MAA	1929516096	
0001	0200782001	GORDO foundation	2019-01-22 00:00:00	2019	1929518256	2018-12-01	2019-01-03	2019-02-04	USD	RV		1 (NULL)	11540.79	2019-01-03	MAA	1929518272	
0001	0200799367	MCL llc	2019-01-29 00:00:00	2019	1929521152	2018-12-01	2019-01-02	2019-01-17	USD	RV		1 (NULL)	20380.93	2019-01-02	MAA	1929521216	
0001	0200782001	GORDO corp	2019-01-22 00:00:00	2019	1929521564	2018-12-01	2019-01-04	2019-02-05	USD	RV		1 (NULL)	9552.12	2019-01-04	MAA	1929521600	
0001	0200418907	SM us	2019-01-22 00:00:00	2019	1929521964	2018-12-01	2019-01-02	2019-01-17	USD	RV		1 (NULL)	3328.49	2019-01-02	MAA	1929521984	
0001	0200909969	SYCO foundation	2019-01-16 00:00:00	2019	1929522235	2017-11-02	2018-12-30	2019-01-14	USD	RV		1 (NULL)	4175.07	2018-12-30	MAA	1929522240	
0001	0200794332	CUST foundation	2019-01-23 00:00:00	2019	1929523374	2017-11-02	2018-12-30	2019-01-14	USD	RV		1 (NULL)	8041.59	2018-12-30	MAA	1929523392	
0001	CC0013	KRAFT F corporation	2019-03-01 00:00:00	2019	1929527949	2017-11-02	2018-12-31	2019-12-31	USD	RV		1 (NULL)	1766.94	2018-12-31	MAA	1929528000	

Servlet Creation-

So after the UI is made some actions (add, edit, delete, etc.) need to perform. So using the help of servlets given below we can make those actions happen.

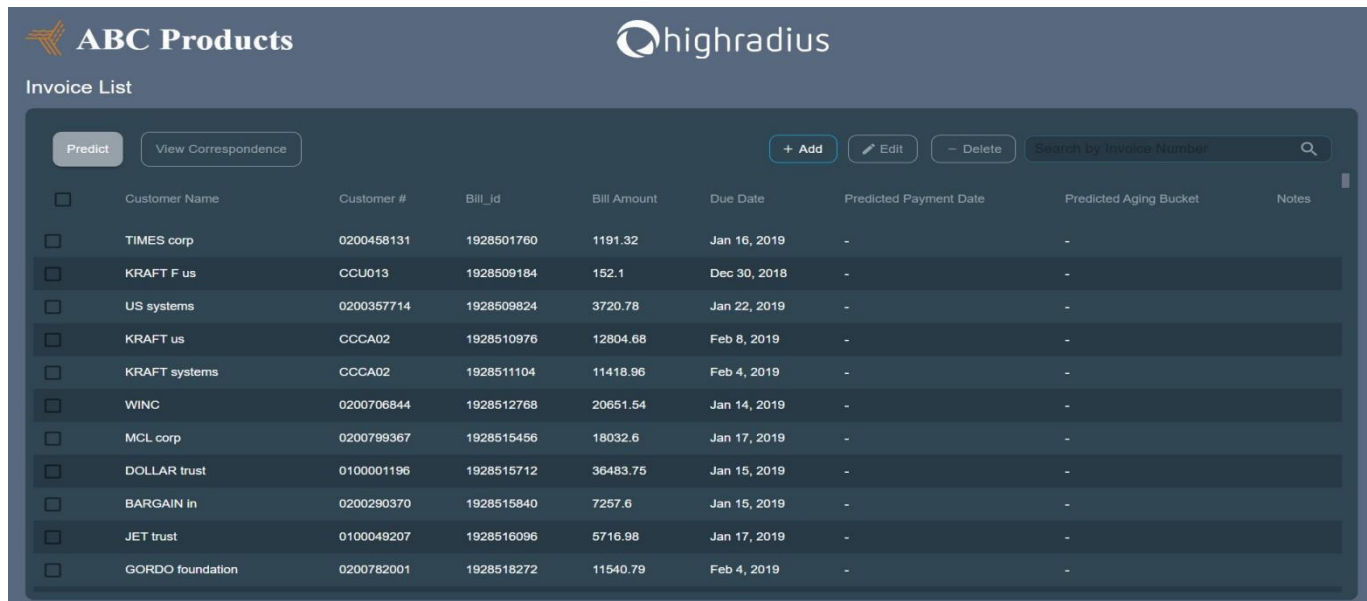
- 1) Add servlet - Get a POST request from the frontend with parameters such as invoice amount, notes, date, etc, and pass them to the SQL database.
- 2) Edit Servlet - GET a POST request from the frontend with parameters such as doc_id to identify the invoice in addition to the parameters which need to be changed.
- 3) Delete Servlet - Delete the selected invoices from the database bypassing their respective doc_id's to identify them in the database.

- 4) Search Servlet - Get the invoice number from the frontend and pass them as an HTTP request using Axios to the backend and search through the database and return it to the frontend again.
- 5) Data Display Servlet - Display the table of invoices to the front end.

3.2.2 Frontend:

UI Representation of the Data:

Receivables Dashboard Page:



The screenshot shows the 'ABC Products' dashboard with the 'highradius' logo. The page title is 'Invoice List'. Below the title, there are buttons for 'Predict', 'View Correspondence', '+ Add', 'Edit', and '- Delete', along with a search bar labeled 'Search by Invoice Number'. The main content is a table with the following columns: Customer Name, Customer #, Bill_id, Bill Amount, Due Date, Predicted Payment Date, Predicted Aging Bucket, and Notes. The table contains 12 rows of invoice data.

<input type="checkbox"/>	Customer Name	Customer #	Bill_id	Bill Amount	Due Date	Predicted Payment Date	Predicted Aging Bucket	Notes
<input type="checkbox"/>	TIMES corp	0200458131	1928501760	1191.32	Jan 16, 2019	-	-	
<input type="checkbox"/>	KRAFT F us	CCU013	1928509184	152.1	Dec 30, 2018	-	-	
<input type="checkbox"/>	US systems	0200357714	1928509824	3720.78	Jan 22, 2019	-	-	
<input type="checkbox"/>	KRAFT us	CCCA02	1928510976	12804.68	Feb 8, 2019	-	-	
<input type="checkbox"/>	KRAFT systems	CCCA02	1928511104	11418.96	Feb 4, 2019	-	-	
<input type="checkbox"/>	WINC	0200706844	1928512768	20651.54	Jan 14, 2019	-	-	
<input type="checkbox"/>	MCL corp	0200799367	1928515456	18032.6	Jan 17, 2019	-	-	
<input type="checkbox"/>	DOLLAR trust	0100001196	1928515712	36483.75	Jan 15, 2019	-	-	
<input type="checkbox"/>	BARGAIN in	0200290370	1928515840	7257.6	Jan 15, 2019	-	-	
<input type="checkbox"/>	JET trust	0100049207	1928516096	5716.98	Jan 17, 2019	-	-	
<input type="checkbox"/>	GORDO foundation	0200782001	1928518272	11540.79	Feb 4, 2019	-	-	

It consists of 2 sections:-

1. Header
2. Grid Panel Section

1. Header Section

The header consists of :

- Account name logo <ABC Products> on the left
- The HighRadius Logo in the center.

2. Grid Panel Section

The Grid panel section will be divided into 4 portions:

- The header of the grid will have a Predict button on the top left corner followed
- by a View Correspondence Button, an Add Button, an Edit Button, a Delete
- Button and a Search Bar.
- The name of the grid that is Invoice List will be mentioned in the top left corner

- of the grid.
- The second portion is the table with customer invoice data as rows and the columns

The list of all the columns to be represented on the UI are as follows:

- Checkbox
- Customer Name
- Customer Number (Customer #)
- Invoice Number (Invoice #)
- Invoice Amount
- Due Date
- Predicted Payment Date
- Predicted Aging Bucket
- Notes

Functionalities implemented in React in Detail-

Add Button Functionality - The Add button will remain enabled if no rows are selected. Whenever one or more rows are selected, the Add button will remain activated. After clicking on the Add button, a pop-up disabled should contain a window that will appear with all the fields for which values need to be added along with a Cancel and a Save button.

The user should be able to **type in the values**, except for the due date of the invoice for which there should be a calendar view from where the user can select the required date, month, and year.

The user should fill in all the required fields before adding. If the user tries to click on add before all mandatory fields are filled, an error message will be displayed.

Once the user clicks on the add button after all mandatory fields are filled, the new values should be displayed in the UI and should remain persistent.

Edit Button Functionality- Clicking on the **edit button** will allow the user to **modify the editable fields** in the data.

The editable columns are **Invoice Amount** and **Notes**.

The edit button will **remain disabled** by default.

When the **user selects one row**, the Edit button gets enabled.

If a user selects multiple rows, the edit button will remain disabled.

Clicking on the Edit button displays a popup window on the screen. The window should contain the Invoice Amount and Notes headers along with the existing data values for both, a Cancel and a Save button.

The user should be able to **edit the values**.

Once the user clicks on the save button, the new values should be displayed in the UI and should remain persistent.

Delete Button Functionality - Clicking on the **delete button** will allow the user to **delete records** from the grid.

The delete button will **remain disabled** by default.

When the **user selects one or more rows**, the delete button gets enabled.

A pop-up should be displayed on clicking delete to confirm that the user wants to delete the selected records permanently.

Once the user clicks on the delete button, the row(s) should be removed from the grid in the UI and should remain persistent.

View Correspondence Button Functionality-

SingleTemplateinViewCorrespondenceButton(Level1):

Clicking on the **view correspondence button** will allow the user to **view and generate pdfs of the selected invoices** from the grid.

The view correspondence button will **remain disabled** by default.

When the **user selects one or more rows**, the view correspondence button gets enabled.

Clicking on the View Correspondence button displays a popup window on the screen. The window should contain a default template with the data of selected rows auto-populated in it along with a Cancel and a Download button.

The account name in the template should be dynamically filled. The total amount to be paid should be the sum of the open amounts of the auto-populated invoices in the template.

The sender's details should be static values.

On clicking the Download button, a file containing the data from the selected rows in the selected template should be downloaded in pdf format.

MultipleTemplatesinViewCorrespondenceButton(Level2):

The user should be able to select from multiple template options from a drop-down menu.

Any template when selected should contain the data of the selected records auto-populated in it.

On clicking the download button, the downloaded file should contain the data in the format matching disabled should base on the disabled should contain disabled are given contain the disabled should contain disabled the should the template being viewed by the user at the time of clicking the download button.

Predict Button Functionality - The Predict button will remain disabled download **button** selected. Whenever one or more rows are selected, the Predict button will be activated. After clicking on the Predict button, the Predicted Payment Date and Predicted AgingBucket will be populated for the respective records.

Search Bar Functionality - The user should be able to type the invoice number in the search bar.

All invoices with matching invoice numbers should be filtered in the grid and the matching portion should be highlighted.

If no such invoice is present, a message should be displayed stating “No Results Found”.

The search bar should implement the debouncing concept. Debouncing allows a function that is bound to an event to fire only once after a specific amount of time has passed.

Infinite Scrolling Functionality - As we know that loading 50000 data at the same time could cause problems in the UI. To remove that problem we used the concept of Infinite Scrolling where the data would be loaded in batches.

CHAPTER 4:RESULTS

4.1 ML:

For our project, we have analyzed how different components act on our data set to yield results which help us to draw several conclusions. In this chapter of the project report, we will highlight and discuss the results and related inferences.

4.1.1 Delay date prediction and bucketization

After importing our data set and performing exploratory data analysis (EDA), feature engineering (FE), feature selection (FS), and modeling we can conclude that :

Area_business is an empty feature and does not support the modeling process.

Doc_id and invoice id are just replicas of each other. so one of them can be dropped.

Delay date can be calculated by subtracting clear_date from due_in_date.

Heat map obtained : (paste ur heatmap)

The delay date shows good correlation with the following features:

'name_customer_target','cust_payment_terms','business_code','cust_number' so the model is trained over these features.

After performing the Linear Regression and Support vector machine the results obtained is:

Out[88]:

	Algorithms	MSE_Train	RMSE_Train	R2_Train	MSE_Val1	RMSE_val1	R2_Val1	MSE_Val2	RMSE_val2	R2_Val2
0	Linear Regression	67.482481	8.214772	0.510752	38.332465	6.191322	0.141979	78.883998	8.881666	0.211831
1	Support Vector Regression	138.988770	11.789350	-0.007668	44.630788	6.680628	0.001000	100.163274	10.008160	-0.000781
2	Decision Tree Regressor	0.001522	0.039012	0.999989	98.307629	9.915020	-1.200484	69.879890	8.359419	0.301795
3	Random Forest Regressor	8.217136	2.866555	0.940426	43.779670	6.616621	0.020051	95.519573	9.773412	0.045617
4	XGB Regressor	19.333235	4.396957	0.859834	1165.682201	34.142088	-25.092229	155.059646	12.452295	-0.549278

So our model can predict delay dates with great accuracy.

For bucketization, we have maintained a set of 15 days I.e. if the delay lies within 0-15 days it will be in bucket 1 15-30 in bucket 2, , and so on.

If the delay is expected to be after 60 days then the bucket assigned to it is named bucket 5.

4.2 Java Servlets

Java Servlet helps us to connect our database to our user interface. All the data is fetched with the help of servlets and placed in the user interface (UI). All the servers which were made are:

fetchData

addData

editData

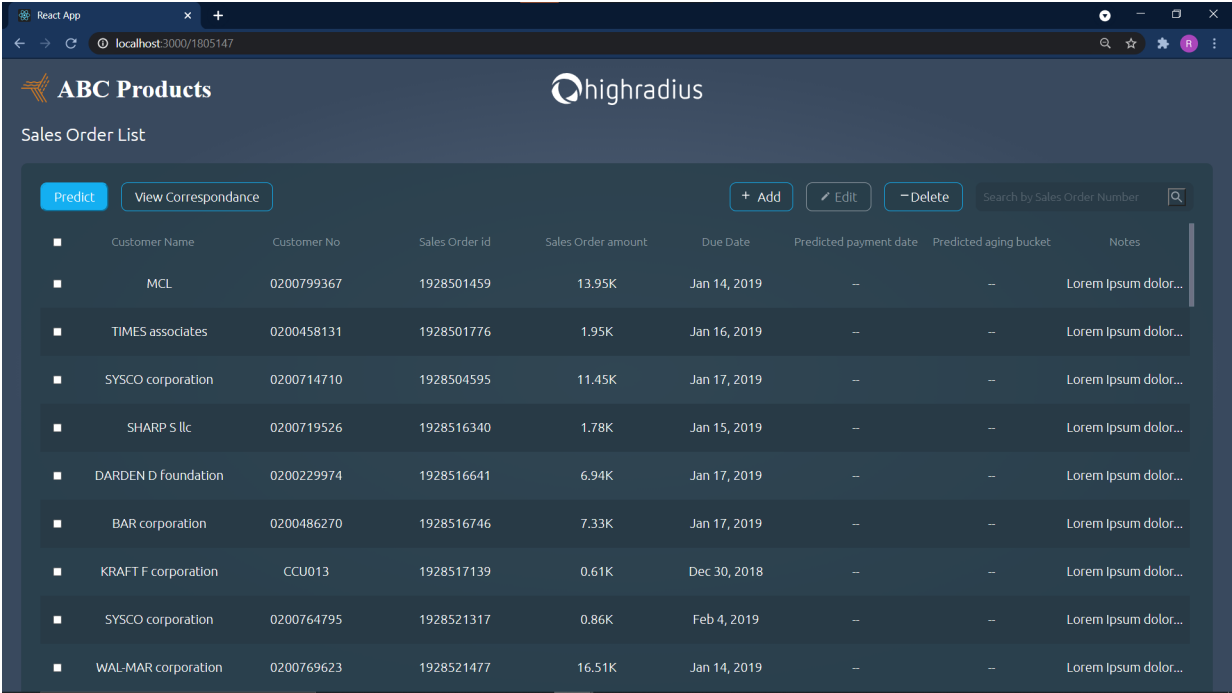
deleteData

searchData

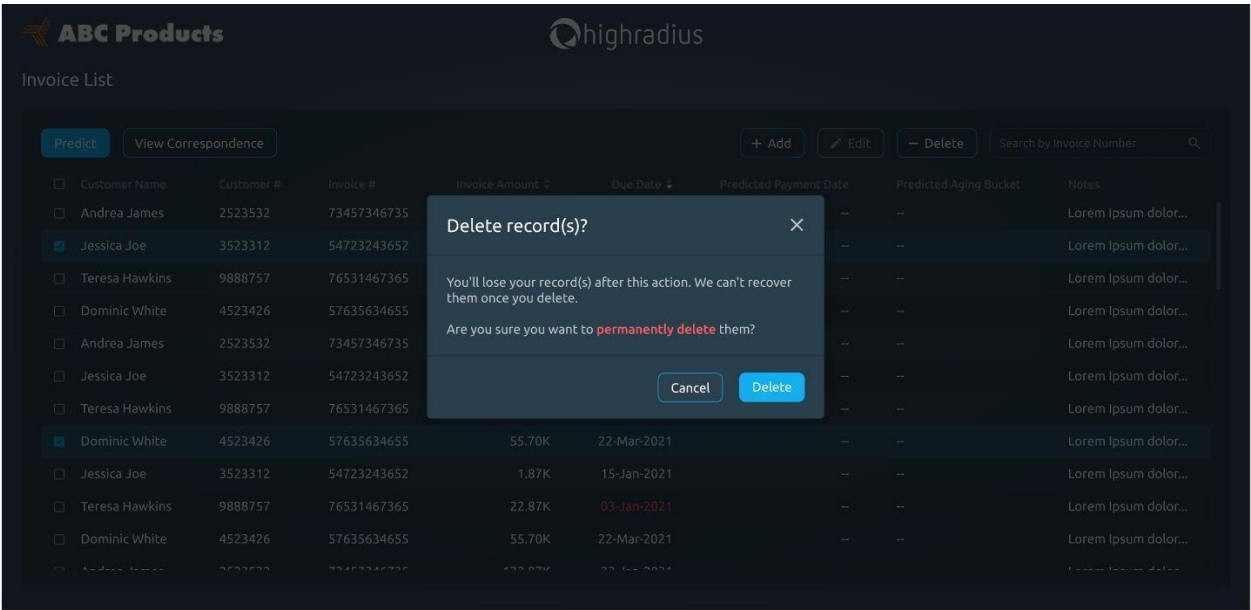
4.3 User Interface with React

After having our servlets ready we build our User Interface using react and react-redux for store management

The final User Interface looks like this:



Customer Name	Customer No	Sales Order id	Sales Order amount	Due Date	Predicted payment date	Predicted aging bucket	Notes
MCL	0200799367	1928501459	13.95K	Jan 14, 2019	--	--	Lorem Ipsum dolor...
TIMES associates	0200458131	1928501776	1.95K	Jan 16, 2019	--	--	Lorem Ipsum dolor...
SYSCO corporation	0200714710	1928504595	11.45K	Jan 17, 2019	--	--	Lorem Ipsum dolor...
SHARP S llc	0200719526	1928516340	1.78K	Jan 15, 2019	--	--	Lorem Ipsum dolor...
DARDEN D foundation	0200229974	1928516641	6.94K	Jan 17, 2019	--	--	Lorem Ipsum dolor...
BAR corporation	0200486270	1928516746	7.33K	Jan 17, 2019	--	--	Lorem Ipsum dolor...
KRAFT F corporation	CCU013	1928517139	0.61K	Dec 30, 2018	--	--	Lorem Ipsum dolor...
SYSCO corporation	0200764795	1928521317	0.86K	Feb 4, 2019	--	--	Lorem Ipsum dolor...
WAL-MAR corporation	0200769623	1928521477	16.51K	Jan 14, 2019	--	--	Lorem Ipsum dolor...



CHAPTER 5:CONCLUSION AND FUTURE WORK

5.1 Problems faced while doing the project-

- The Frontend Part was quite difficult since I was a complete beginner in this field.
- Understanding how to forward actions from the front-end to the back-end was quite difficult.
- Infinite Loading was quite challenging.
- The UI design was also difficult to implement since I am very new to Javascript. Logical interfacing of the functionalities was the most challenging work.

5.2 Conclusion

An order management system is any tool or platform that tracks sales, orders, inventory, and fulfillment as well as enables the people, processes, and partnerships necessary for products to find their way to the customers who bought them.

In this project, I have built an AI-Enabled FinTech B2B Order Management Application using a regression model to predict whether payment of the passed function will be delayed. The accuracy of the model turned out to be 85%. The same has been deployed using Flask Framework and React JS for the frontend.

5.3 Future Work

The accuracy of the model can be increased and the same can be deployed in the cloud to make it available to a larger audience.

5. 4 Planning And Project Management

Table showing details about project planning and management

Estimated Starting Week	Activity	No. of Weeks
1	Basics Of Python.	1
2	Basics of Machine Learning.	1
3	Advance Concepts of Machine Learning and Submission of ML model.	2
5	Basics of Database Management System.	1
6	Introduction to JAVA and HTML	1

7	CSS and Javascript	1
8	React, Redux, and material UI.	2
10	Project Submission.	Completed