GLOBAL SALES DATA ANALYTICS

1. INTRODUCTION

1.1 Project Overview

Shopping online is currently the need of the hour. Because of this COVID, it's not easy to walk in a store randomly and buy anything you want. So, try to understand a few things like, Customer Analysis and Product Analysis of this Global Super Store. Sales refers to all activities involved in selling a product or service to a consumer or business.

1.2 Purpose

It is important for sales and marketing teams to review their strategies and performance in order to make improvements. One way to measure performance is with sales analytics.

2. LITERATURE SURVEY

2.1 Existing problem

2.1.1 Field Proxy

Field proxy can automate the field operations and manage sales tracking & report generation. It also lets you manage everything by making custom apps for your field team to manage the day-to-day struggles of working on the field. It can easily

Create unique dashboards

- Identify areas of improvement
- Track resources with greater efficiency
- Create tasks for different employees and manage them seamlessly.

2.1.2 Glew

Glew make it easy for businesses of all sizes to access and act on data-driven insights by providing everything you need to run business intelligence in one place: data pipeline, data warehouse and powerful, scalable reporting and analytics.

A single platform that would allow anyone to easily connect disparate data sources, instantly access out-of-the-box metrics and visualizations and create powerful custom reports - all without writing a single line of code.

2.2 References

[1] K. Singh and R. Wajgi, "Data analysis and visualization of sales data," 2016 World Conference on Futuristic Trends in Research and Innovation for Social Welfare (Startup Conclave), 2016, pp. 1-6, doi: 10.1109/STARTUP.2016.7583967.

[2] T. Liang, S. Lu and Q. Liu, "Data Visualization System Based on Big Data Analysis," 2020 International Conference on Robots & Intelligent System (ICRIS), 2020, pp. 76-79, doi: 10.1109/ICRIS52159.2020.00027.

[3] G. Ferreira, P. Alves and S. de Almeida, "Platform for real-time data analysis and visualization based on Big Data methods," 2021 16th Iberian Conference on Information Systems and Technologies (CISTI), 2021, pp. 1-6, doi: 10.23919/CISTI52073.2021.9476628.

2.3 Problem Statement Definition

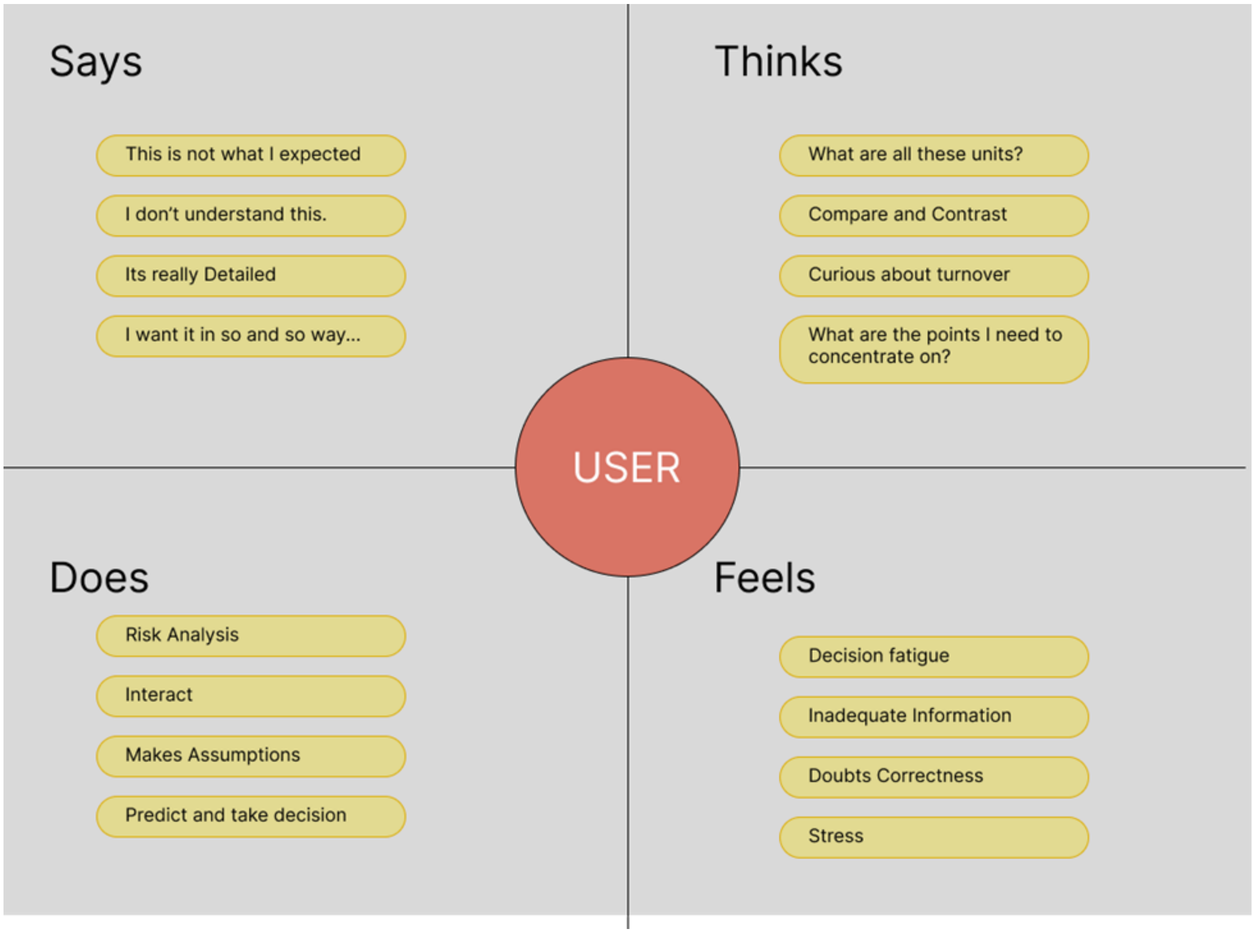
Sales Data Analytics usually has few main categories

- **Descriptive:** What happened? Descriptive analytics entails tracking historical sales data—revenue, number of users, etc.—so you can make comparisons and better understand what's currently happening.
- **Diagnostic:** Why did it happen? Diagnostic analytics is examining and drilling down into the data to determine exactly why something occurred
- **Predictive:** What's going to happen? Predictive analytics is taking what you've learned about past sales and using it to gauge patterns and trends. This allows you to make educated predictions.
- **Prescriptive:** What's the best solution or action? Prescriptive analytics involves assessing all the data and recommending the best plan of action.

3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas

This map is created with view of the project in user's perspective, to find pain & gain points and to summarize it with a list of problem statements.



Derived Problem Statements:

- Create a simple easy to understand analytics of Sales data, by plotting different visualization.
- Use of familiar metrics to analyze data.
- Easy to find Insights of data with clear and legible color coding.
- Pleasing UI design.
- Detailed information gathering
- Localization of areas of interest, and complete analysis on them.

3.2 Ideation & Brainstorming

Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem solving. Prioritizing volume over value, out-of-the-box ideas are welcome and built upon, and all participants are encouraged to collaborate, helping each other develop a rich amount of creative solutions. Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

Step-1: Team Gathering, Collaboration and Select the Problem Statement:

Global Sales analytics refers to the use of technology to collect and use sales data to derive actionable insights. It is used to identify, optimize, and forecast sales. It uses different metrics and KPIs to plan an efficient sales model that generates higher revenue for the business. It is important for sales and marketing teams to review their strategies and performance in order to make improvements. One way to measure performance is with sales analytics.

Step-2: Brainstorm, Idea Listing and Grouping:

[Team Lead] 311619104067 Sivabalan J

- Creating an API for open access of the Analytics data.
- Avoid cluttered UI components.
- Responsive Design for every screen size

[Member 1] 311619104073 Sriram Kathick K

- Choosing a modern and legible color theme.
- Cleaning the input data properly.
- Deciding the graphs/analytical components to work with.
- Do not bombard user with Information, and make it easy to swallow.

[Member 2] 311619104051 Padmaprabhan M

- Creating a presentation of the whole analytics.
- Proper understanding of given data and its structure.
- Making the analytics modular by using data files with same format, and comparing 2 files of same structure.
- Making the analytics available for multiple devices and platforms.

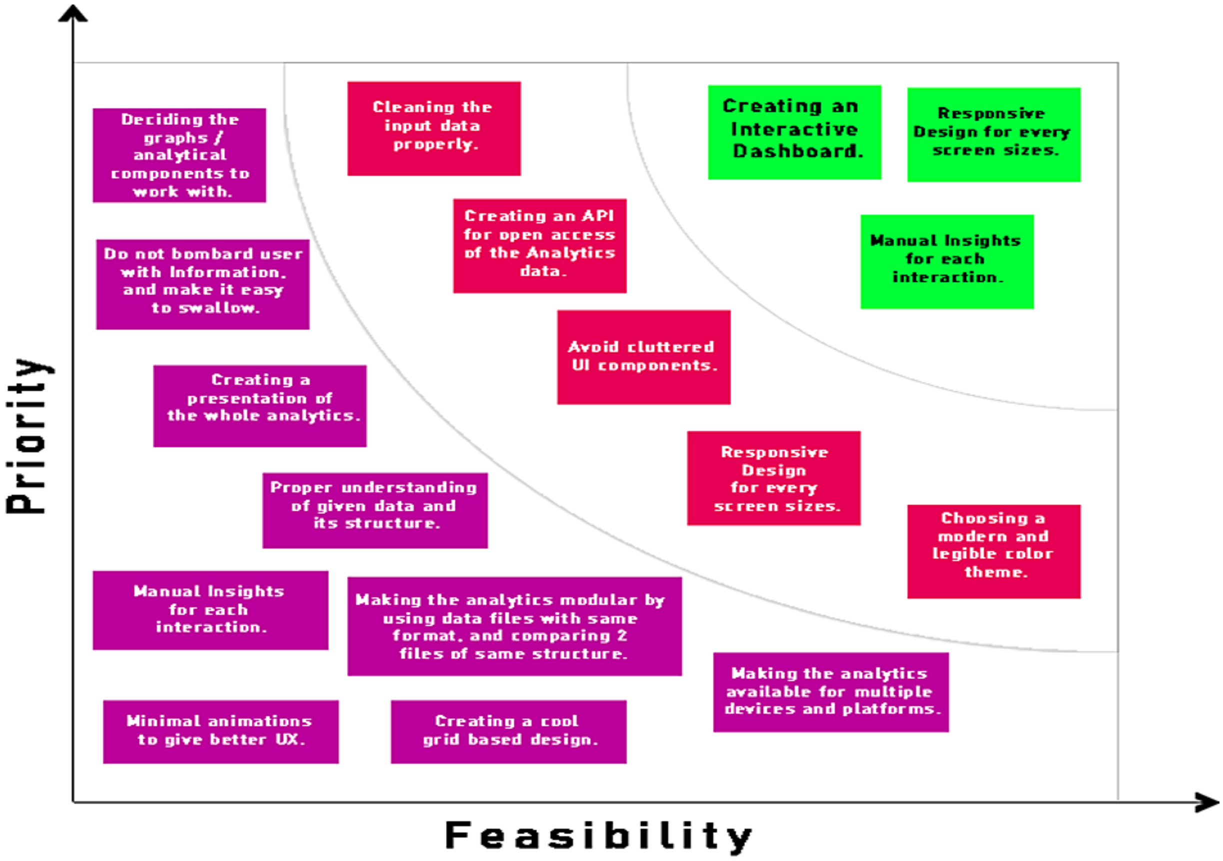
[Member 3] 311619104301 Ashwin V

- Creating a Detailed Report.
- Manual Insights for each interaction.
- Minimal animations to give better UX.
- Creating a cool grid based design.

Selected 3 IDEA

- Creating an Interactive Dashboard.
- Responsive Design for every screen sizes.
- Manual Insights for each interaction.

Step-3: Idea Prioritization:

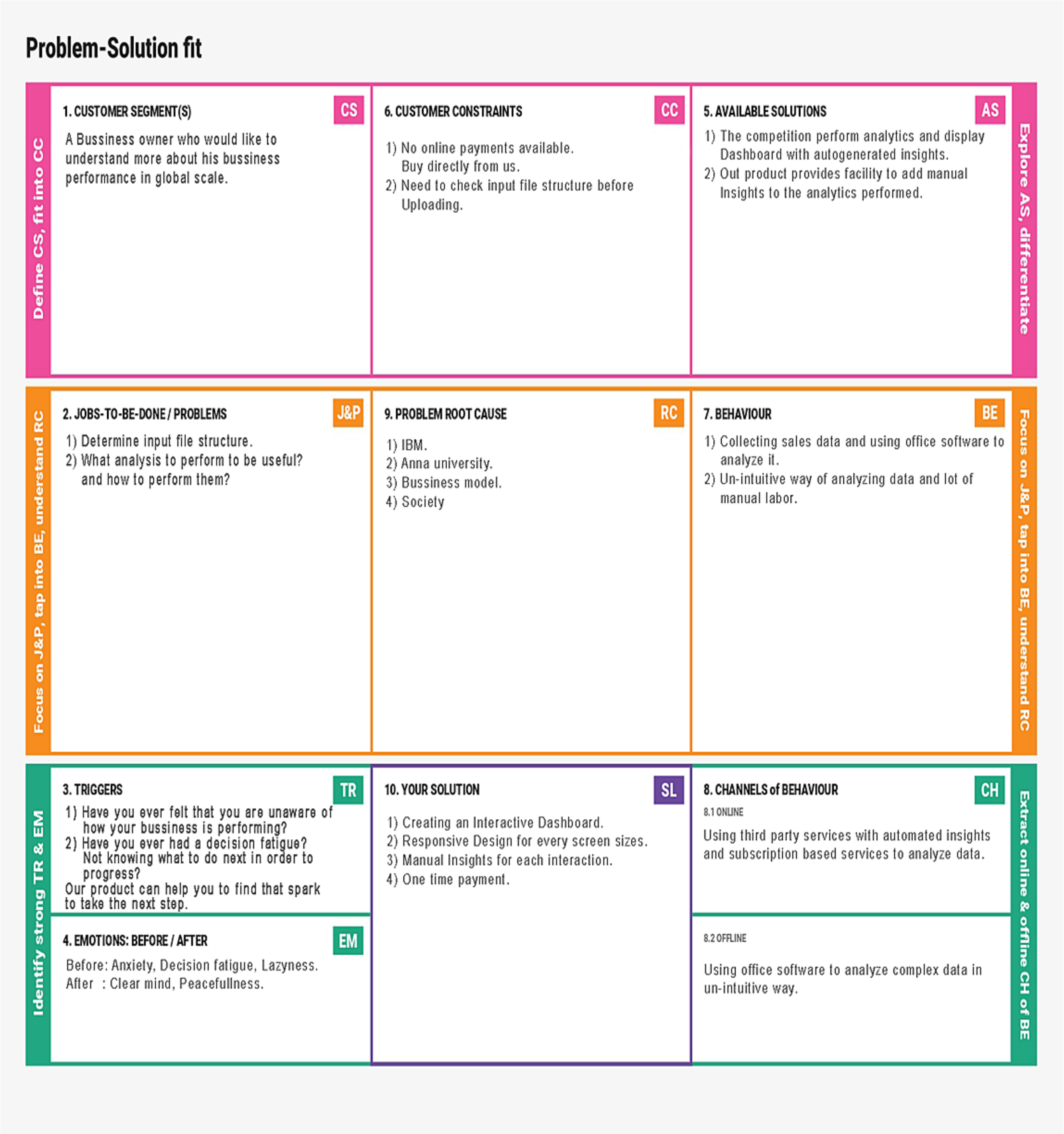

Deciding the graphs / analytical components to work with.
Cleaning the input data properly.
Creating an Interactive Dashboard.
Responsive Design for every screen sizes.
Creating an API for open access of the Analytics data.
Manual Insights for each interaction.
Do not bombard user with Information, and make it easy to swallow.
Avoid cluttered UI components.
Creating a presentation of the whole analytics.
Responsive Design for every screen sizes.
Proper understanding of given data and its structure.
Choosing a modern and legible color theme.
Manual Insights for each interaction.
Making the analytics modular by using data files with same format, and comparing 2 files of same structure.
Making the analytics available for multiple devices and platforms.
Minimal animations to give better UX.
Creating a cool grid based design.
Priority
Feasibility

3.3 Proposed Solution

S. No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	**Global Sales Data Analytics** - Create a simple easy to understand analytics of Sales data, by plotting different visualization. - Use of familiar metrics to analyze data. - Easy to find Insights of data with clear and legible color coding. - Pleasing UI design. - Detailed information gathering - Localization of areas of interest, and complete analysis on them.
2.	Idea / Solution description	Creating an Interactive Dashboard. Responsive Design for every screen sizes. Modular file based analytics. Manual Insights for each interaction
3.	Novelty / Uniqueness	Analytics are modular with the help of exporting and importing files. Has ability to add manual insights for later viewing.

4.	Social Impact / Customer Satisfaction	Customer gets instant analytical diagrams when they input the file to the software, as long as the file is in the correct format. Customer can reuse the same any number of times.
5.	Business Model (Revenue Model)	Onetime payment – for a user. Free Trial 30 days.
6.	Scalability of the Solution	The solution scales well by default, as its file based. Any number of similarly formatted files can be submitted and the analytics will be drawn for that particular file.

3.4 Problem Solution fit


Define CS, fit into CC
Explore AS, differentiate
Focus on J&P, tap into BE, understand RC
Focus on J&P, tap into BE, understand RC
Identify strong TR & EM
Extract online & offline CH of BE

Problem-Solution fit

1. CUSTOMER SEGMENT(S) CS

A Bussiness owner who would like to understand more about his bussiness performance in global scale.

6. CUSTOMER CONSTRAINTS CC

1) No online payments available.
Buy directly from us.
2) Need to check input file structure before Uploading.

5. AVAILABLE SOLUTIONS AS

1) The competition perform analytics and display Dashboard with autogenerated insights.
2) Out product provides facility to add manual Insights to the analytics performed.

2. JOBS-TO-BE-DONE / PROBLEMS J&P

1) Determine input file structure.
2) What analysis to perform to be useful? and how to perform them?

9. PROBLEM ROOT CAUSE RC

1) IBM.
2) Anna university.
3) Bussiness model.
4) Society

7. BEHAVIOUR BE

1) Collecting sales data and using office software to analyze it.
2) Un-intuitive way of analyzing data and lot of manual labor.

3. TRIGGERS TR

1) Have you ever felt that you are unaware of how your bussiness is performing?
2) Have you ever had a decision fatigue? Not knowing what to do next in order to progress?
Our product can help you to find that spark to take the next step.

4. EMOTIONS: BEFORE / AFTER EM

Before: Anxiety, Decision fatigue, Lazyness.
After : Clear mind, Peacefullness.

10. YOUR SOLUTION SL

1) Creating an Interactive Dashboard.
2) Responsive Design for every screen sizes.
3) Manual Insights for each interaction.
4) One time payment.

8. CHANNELS of BEHAVIOUR CH

8.1 ONLINE

Using third party services with automated insights and subscription based services to analyze data.

8.2 OFFLINE

Using office software to analyze complex data in un-intuitive way.

4. REQUIREMENT ANALYSIS

4.1 Functional requirement

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	Token generation	Token Generation using UUID.
FR-2	Application Activation	● Confirmation/Validation of Token. ● Activation of Software.
FR-3	Getting user Input as CSV format	● Get Input. ● Validate Input. ● Clean Input.
FR-4	Generate Visualization	● Convert Input to Processing Format. ● Generate Visualization Using Matplotlib.
FR-5	Add Manual insights	Get Manual Insights from user.
FR-6	Save in DB/Filesystem	Save Necessary data in user filesystem or in mysql database.

4.2 Non-Functional requirements

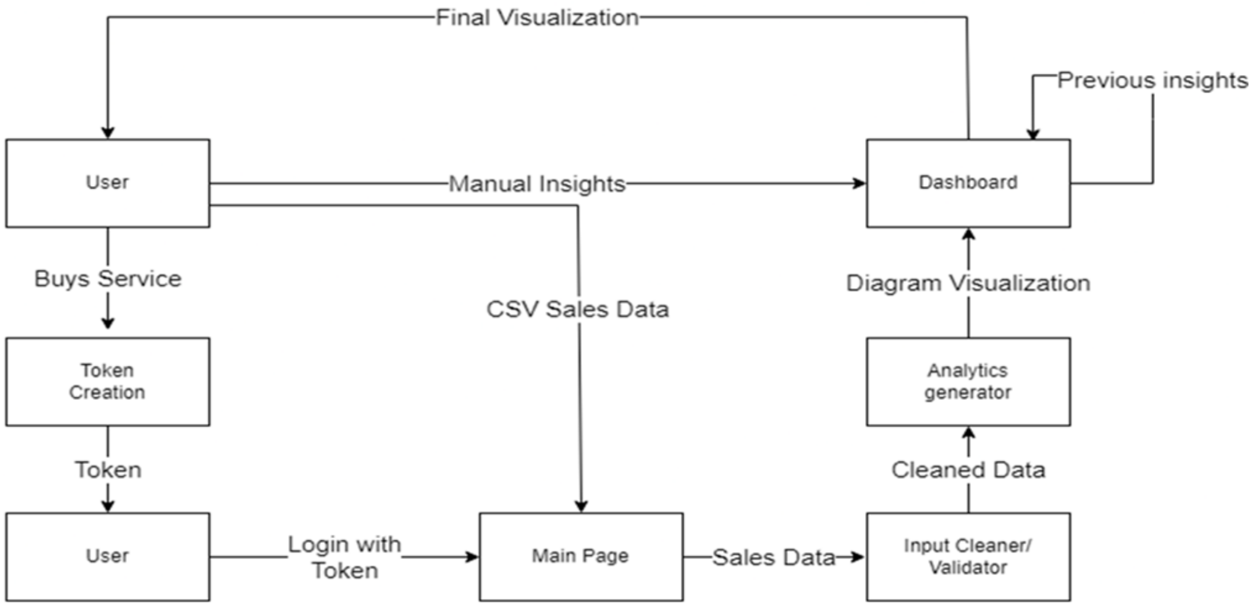
FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	Needs to be actually functional and beneficial for the user.
NFR-2	**Security**	UUID token validation for one time.
NFR-3	**Reliability**	Must have reasonable fault tolerance.
NFR-4	**Performance**	Must take less System Resource and must be able to generate visualization on all modern hardware.
NFR-5	**Availability**	24/7 available as it runs on client side only.
NFR-6	**Scalability**	Automatically scalable as it runs on client’s computer.

5. PROJECT DESIGN

5.1 Data Flow Diagrams

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored

5.2 Solution & Technical Architecture



5.3 User Stories

Use the below template to list all the user stories for the product.

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer	Registration	USN-1	As user I need a token to activate my software	Generate unique Token	High	Sprint-1
	Activation	USN-2	As user I want the token to work property	Activate Software	High	Sprint-1

	Dashboard	USN-3	As a user I need my application to take in a CSV file	Import CSV file	High	Sprint-3
		USN-4	As user I want visualization to be generated	Generate	High	Sprint-1
		USN-5	As user I want to enter Manual insights	Get Insights from user	Low	Sprint-2

6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As user I need a token to activate my software	3	High	J Sivabalan
Sprint-1	Activation	USN-2	As user I want the token to work property	3	High	J Sivabalan
Sprint-2	Dashboard	USN-3	As a user I need my application to take in a CSV file	12	High	Sriram Karthick K J Sivabalan
Sprint-1	---	USN-4	As user I want visualization to be generated	6	High	Sriram Karthick K J Sivabalan
Sprint-3	---	USN-5	As user I want to enter Manual insights	12	Low	Padmaprabhan M Ashwin V
Sprint-4	---	USN-6	As user I want to software to learn from previous insights and try to give one on own	12	Low	J Sivabalan Sriram Karthick K Padmaprabhan M Ashwin V

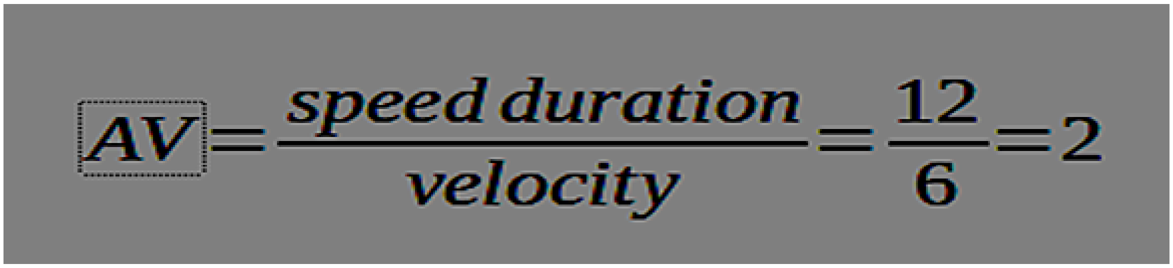
6.2 Sprint Delivery Schedule

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	12	6 Days	24 Oct 2022	29 Oct 2022		
Sprint-2	12	6 Days	31 Oct 2022	05 Nov 2022		
Sprint-3	12	6 Days	07 Nov 2022	12 Nov 2022		
Sprint-4	12	6 Days	14 Nov 2022	19 Nov 2022		

Velocity:

Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day)

Velocity = 12 (points per sprint)
Sprint Duration = 6 days

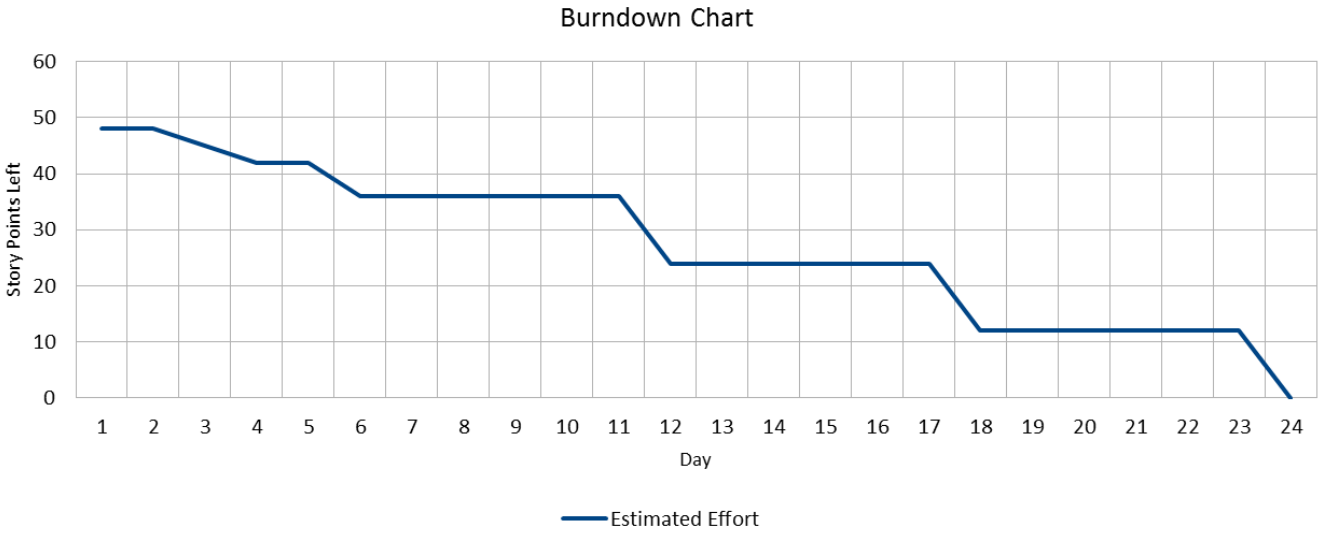


$$AV = \frac{speed\ duration}{velocity} = \frac{12}{6} = 2$$

Burndown Chart:

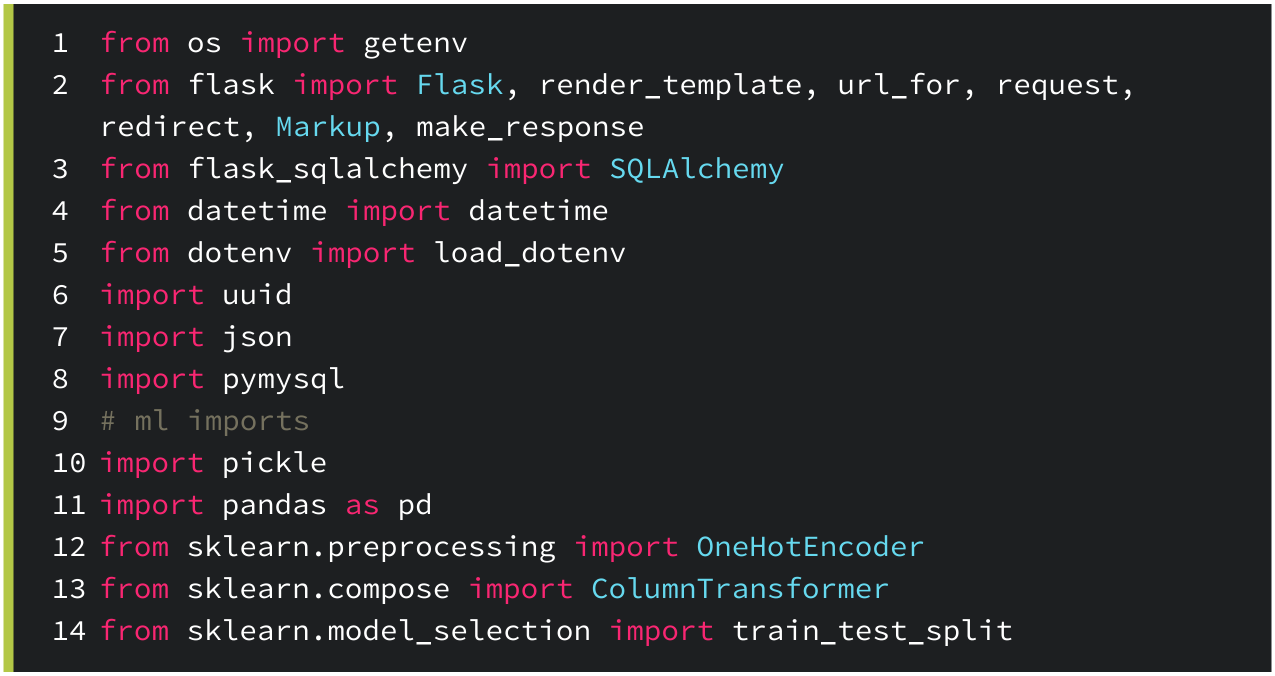
A burn down chart is a graphical representation of work left to do versus time. It is often used in agile software development methodologies such as Scrum. However, burn down charts can be applied to any project containing measurable progress over time.

Expected Burndown Chart:



7. CODING & SOLUTIONING (Explain the features added in the project along with code)

7.1 Server Imports



```python
from os import getenv
from flask import Flask, render_template, url_for, request, redirect, Markup, make_response
from flask_sqlalchemy import SQLAlchemy
from datetime import datetime
from dotenv import load_dotenv
import uuid
import json
import pymysql
# ml imports
import pickle
import pandas as pd
from sklearn.preprocessing import OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.model_selection import train_test_split
```

7.2 Loading Environment Variables using python-dotenv

```python
load_dotenv()
mysql_user = getenv('MYSQL_USER')
mysql_pass = getenv('MYSQL_PASS')
deploy_url = getenv('DEPLOYMENT_URL')
```

7.3 Jokes API

```javascript
async function joke() {
    await fetch("https://v2.jokeapi.dev/joke/Programming?blacklistFlags=nsfw,religious,political,racist,sexist,explicit&type=single", {
      method: "GET",
    }).then(async (res) => {
      let t= await res.json();
      document.getElementById('joke').innerHTML=t.joke
    }).catch((err) => { console.log(err) })
  }
```

7.4 Base HTML file

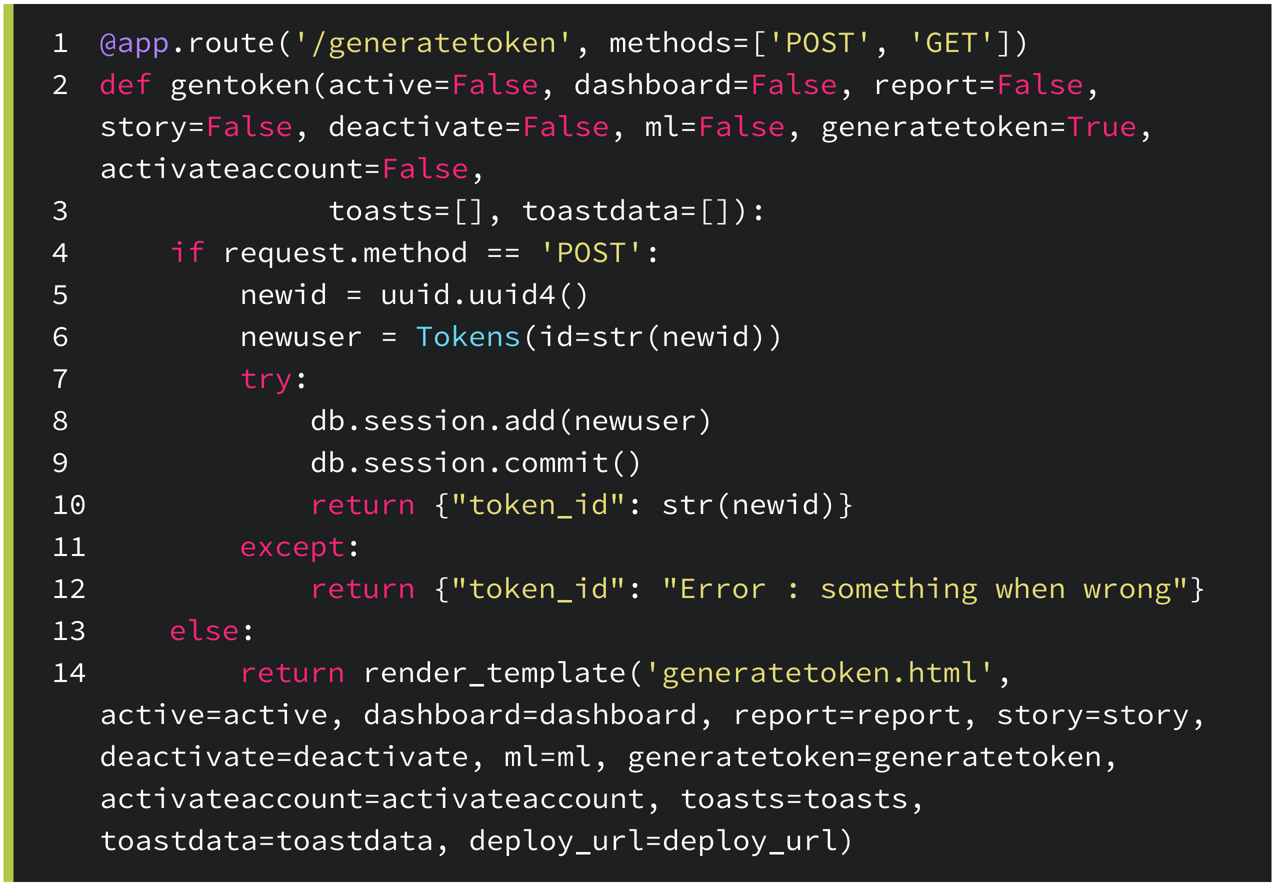
```html
<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link rel="shortcut icon" href="{{url_for('static',filename='img/favicon.ico')}}" type="image/x-icon">
  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/bootstrap/5.2.0/css/bootstrap.min.css"
    integrity="sha512-
```

```
XWTTruHZEYJsxV3W/lSXG1n3Q39YIWOstqvmFsdNEEQfHoZ6vm6E9GK2OrF6DSJSpIbRbi+Nn0WDPID9O7xB2Q=="
    crossorigin="anonymous" referrerpolicy="no-referrer" />
  <link rel="stylesheet" href="{{url_for('static',filename='css/main.css')}}">
  {% block head %}{% endblock %}
</head>

<!--
bool active
bool dashboard
bool report
bool story
bool deactivate
bool ml
bool generatetoken
bool activateaccount
list(String) toasts
list(String) toastdata
-->

<body>
  <nav class="navbar navbar-dark navbar-expand-lg bg-dark fixed-top" id="the-nav">
    <div class="container-fluid">
      <a class="navbar-brand brand" href="/home">GlobalSalesData</a>
      <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarNavDropdown"
        aria-controls="navbarNavDropdown" aria-expanded="false" aria-label="Toggle navigation">
        <span class="navbar-toggler-icon"></span>
      </button>
      <div class="collapse navbar-collapse nav-item-set" id="navbarNavDropdown">
        <ul class="navbar-nav items-container">
          {% if active %}
          <li class="nav-item">
            <a class="nav-link" href="{{deploy_url}}/dashboard">Dashboard</a>
          </li>
          <li class="nav-item">
            <a class="nav-link" href="{{deploy_url}}/report">Report</a>
          </li>
          <li class="nav-item">
            <a class="nav-link" href="{{deploy_url}}/story">Story</a>
          </li>
          <li class="nav-item">
            <a class="nav-link" href="{{deploy_url}}/ml">Machine Learning</a>
          </li>
          <li class="nav-item">
            <a class="nav-link link-danger" href="{{deploy_url}}/deactivate">Deactivate</a>
          </li>
          {% else %}
          <li class="nav-item">
            <a class="nav-link" href="{{deploy_url}}/generatetoken">Generate Token</a>
          </li>
          <li class="nav-item"><a class="nav-link"href="{{deploy_url}}/activateacc"">Activate Account</a>
          </li>
          {% endif %}
        </ul>
      </div>
    </div>
  </nav>

  {% if toasts!=None %}
  {% for i in range(toasts|length): %}
  <div class="toast-container position-fixed bottom-0 end-0 p-3">
    <div class="toast align-items-center text-bg-{{toasts[i]}} border-0" role="alert" aria-live="assertive"
      aria-atomic="true">
      <div class="d-flex">
        <div class="toast-body">
          {{toastdata[i]}}
    </div>
    <button type="button" class="btn-close btn-close-white me-2 m-auto" data-bs-dismiss="toast"
     aria-label="Close"></button>
   </div>
  </div>
 </div>
 {% endfor %}
 {% endif %}


 {% block body %}{% endblock %}
 <script src="https://cdnjs.cloudflare.com/ajax/libs/bootstrap/5.2.0/js/bootstrap.bundle.min.js"
  integrity="sha512-9GacT4119eY3AcosfWtHMsT5JyZudrexyEVzTBWV3viP/YfB9e2pEy3N7WXL3SV6ASXpTU0vzzSxsbfsuUH4sQ=="
  crossorigin="anonymous" referrerpolicy="no-referrer">
  </script>
<script>
 const popoverTriggerList = document.querySelectorAll('[data-bs-toggle="popover"]')
 const popoverList = [...popoverTriggerList].map(popoverTriggerEl => new bootstrap.Popover(popoverTriggerEl))
 const tooltipTriggerList = document.querySelectorAll('[data-bs-toggle="tooltip"]')
const tooltipList = [...tooltipTriggerList].map(tooltipTriggerEl => new bootstrap.Tooltip(tooltipTriggerEl))

 function myFunction(x) {
  if (x.matches) { // If media query matches
   document.getElementById('the-nav').className = 'navbar navbar-dark navbar-expand-lg bg-dark fixed-bottom'; for (a
    in document.getElementsByClassName('img-fluid rounded-start')) { a.className = "card-img-top img-card" }
  }
  else {
   document.getElementById('the-nav').className = 'navbar
navbar-dark navbar-expand-lg bg-dark fixed-top'; for
        (a in document.getElementsByClassName('card-img-top')) { a.className = "img-fluid rounded-start img-card" }
    }
  } var x = window.matchMedia("(max-width: 992px)"); myFunction(x); x.addListener(myFunction); </script>
</body>
<script>
  const toasts = document.getElementsByClassName('toast')
  for (i = 0; i < toasts.length; i++) {
    let elem = toasts.item(i)
    let elem_tost = new bootstrap.Toast(elem)
    elem_tost.show()
  }
</script>

</html>
```

7.5 Generate Token - Frontend

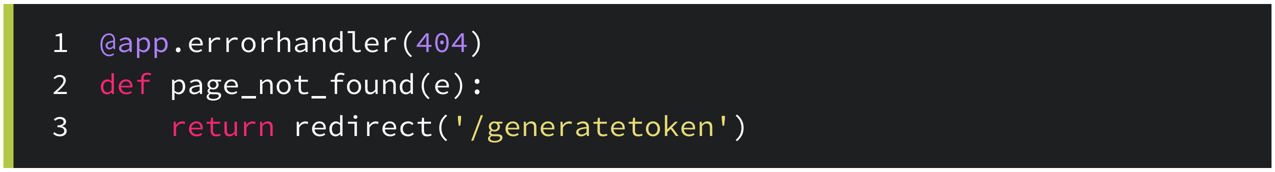
```
{% extends 'base.html' %}

{% block head %}
<title>Generate Token</title>
<link rel="stylesheet" href="{{url_for('static',filename='css/generatetoken.css')}}">
{% endblock %}

{% block body %}
<div class="main-container">

  <div class="sub-container p-5">
    <h1>Generate Token</h1>
    <div class="input-cont">
      <input class="form-control input-tf shadow-none" type="text" value="" aria-label="tokenid" id="tokenid"
        readonly>
      <button type="submit" class="btn btn-dark butt2 shadow-none" data-bs-container="body" data-bs-toggle="popover"
        data-bs-placement="right" data-bs-content="Copied" onclick="copyfn()">
        <svg xmlns="http://www.w3.org/2000/svg" class="icon icon-tabler icon-tabler-clipboard" width="36" height="36" viewBox="0 0 24 24" stroke-width="1.5" stroke="#ffffff" fill="none" stroke-linecap="round" stroke-linejoin="round">
          <path stroke="none" d="M0 0h24v24H0z" fill="none"/>
          <path d="M9 5h-2a2 2 0 0 0 -2 2v12a2 2 0 0 0 2 2h10a2 2 0 0 0 2 -2v-12a2 2 0 0 0 -2 -2h-2" />
          <rect x="9" y="3" width="6" height="4" rx="2" />
        </svg>
      </button>
    </div>
    <div class="generate-btn-group mt-1">
      <button type="button" class="btn btn-dark butt shadow-none" onclick="generate()">Generate</button>
      <a href="{{deploy_url}}/activateacc" class="link-text mt-0">already have a token?</a>
    </div>
  </div>

</div>
<script>
  function copyfn() {
    var copyText = document.getElementById("tokenid");
    copyText.select();
    copyText.setSelectionRange(0, 99999); // For mobile devices
    navigator.clipboard.writeText(copyText.value);
  }
  async function generate() {
    await fetch("{{deploy_url}}" + "/generatetoken", {
      method: "POST",
    }).then(async (res) => {
      let t= await res.json();
      document.getElementById('tokenid').value=t.token_id
    }).catch((err) => { console.log(err) })
  }
</script>

{% endblock %}
```

7.6 Generate Token - Backend



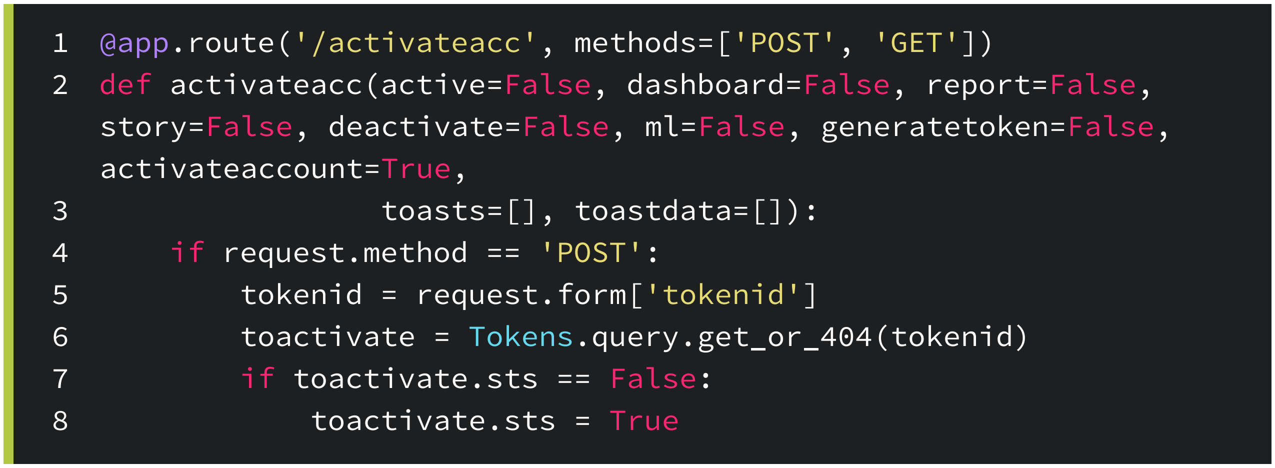
```python
@app.route('/generatetoken', methods=['POST', 'GET'])
def gentoken(active=False, dashboard=False, report=False, story=False, deactivate=False, ml=False, generatetoken=True, activateaccount=False,
             toasts=[], toastdata=[]):
    if request.method == 'POST':
        newid = uuid.uuid4()
        newuser = Tokens(id=str(newid))
        try:
            db.session.add(newuser)
            db.session.commit()
            return {"token_id": str(newid)}
        except:
            return {"token_id": "Error : something when wrong"}
    else:
        return render_template('generatetoken.html', active=active, dashboard=dashboard, report=report, story=story, deactivate=deactivate, ml=ml, generatetoken=generatetoken, activateaccount=activateaccount, toasts=toasts, toastdata=toastdata, deploy_url=deploy_url)
```

7.7 404 redirect



```python
@app.errorhandler(404)
def page_not_found(e):
    return redirect('/generatetoken')
```

7.8 Activate Account - Backend



```python
@app.route('/activateacc', methods=['POST', 'GET'])
def activateacc(active=False, dashboard=False, report=False, story=False, deactivate=False, ml=False, generatetoken=False, activateaccount=True,
                toasts=[], toastdata=[]):
    if request.method == 'POST':
        tokenid = request.form['tokenid']
        toactivate = Tokens.query.get_or_404(tokenid)
        if toactivate.sts == False:
            toactivate.sts = True
            sesid = uuid.uuid4()
            toactivate.session = str(sesid)
            db.session.commit()
            toasts = ['success']
            toastdata = [Markup('Successfully activated your account')]
            active = True
            activateaccount = False
            dashboard = True
            res = make_response(render_template('dashboard.html', active=active, dashboard=dashboard, report=report, story=story, deactivate=deactivate,
                                ml=ml, generatetoken=generatetoken, activateaccount=activateaccount, toasts=toasts, toastdata=toastdata, deploy_url=deploy_url))
            res.set_cookie('tokenid', tokenid)
            res.set_cookie('sessionid', str(sesid))
            return res
        else:
            toasts = ['warning']
            toastdata = [Markup('The Token has already been utilized <br> Please <a href="' +
                                deploy_url+'/generatetoken" class="link-text">generate new one</a>')]
            return render_template('activateacc.html', active=active, dashboard=dashboard, report=report, story=story, deactivate=deactivate, ml=ml, generatetoken=generatetoken, activateaccount=activateaccount, toasts=toasts, toastdata=toastdata, deploy_url=deploy_url)
    else:
        return render_template('activateacc.html', active=active, dashboard=dashboard, report=report, story=story, deactivate=deactivate, ml=ml, generatetoken=generatetoken, activateaccount=activateaccount, toasts=toasts, toastdata=toastdata, deploy_url=deploy_url)
```

7.9 Dashboard - Frontend

```
{% extends 'base.html' %}

{% block head %}
<title>Generate Token</title>
<link rel="stylesheet" href="{{url_for('static',filename='css/dashboard.css')}}">
{% endblock %}

{% block body %}
<div class="main-container">

  <div class="p-5 mt-3">
    <iframe src="https://us1.ca.analytics.ibm.com/bi/?perspective=dashboard&amp;pathRef=.public_folders%2FDevelopment%2BPhase%2FGlobalSales%2BDash&amp;closeWindowOnLastView=true&amp;ui_appbar=false&amp;ui_navbar=false&amp;shareMode=embedded&amp;action=view&amp;mode=dashboard&amp;subView=model000001844c073244_00000000" width="1280" height="900" frameborder="0" gesture="media" allow="encrypted-media" allowfullscreen=""></iframe>
  </div>
  <div class="sub-container quote m-3">
    <h3>Programmer Jokes API</h3>
    <p id="joke">
    </p>
  </div>
</div>
<script>
  async function joke() {
    await fetch("https://v2.jokeapi.dev/joke/Programming?blacklistFlags=nsfw,religious,political,racist,sexist,explicit&type=single", {
      method: "GET",
    }).then(async (res) => {
      let t= await res.json();
      document.getElementById('joke').innerHTML=t.joke
    }).catch((err) => { console.log(err) })
  }
  joke();
</script>
{% endblock %}
```

7.10 Dashboard, Story, Report - Backend

```python
@app.route('/dashboard')
def dashboard(active=False, dashboard=True, report=False, story=False, deactivate=False, ml=False, generatetoken=False, activateaccount=False,
              toasts=[], toastdata=[]):
    sesid = request.cookies.get('sessionid', '')
    if sesid != '':
        row = Tokens.query.filter_by(session=sesid).first()
        if row is None:
            return gentoken(toasts=['danger'], toastdata=[Markup('bad session')])
        else:
            active = True
            return render_template('dashboard.html', active=active, dashboard=dashboard, report=report, story=story, deactivate=deactivate, ml=ml, generatetoken=generatetoken, activateaccount=activateaccount, toasts=toasts, toastdata=toastdata, deploy_url=deploy_url)
    else:
        return gentoken(toasts=['warning'], toastdata=[Markup('no session available <br> please activate account or <a href="'+deploy_url+'/generatetoken" class="link-text">generate new token</a>')])
@app.route('/story')
def story(active=False, dashboard=False, report=False, story=True, deactivate=False, ml=False, generatetoken=False, activateaccount=False,
          toasts=[], toastdata=[]):
    sesid = request.cookies.get('sessionid', '')
    if sesid != '':
        row = Tokens.query.filter_by(session=sesid).first()
        if row is None:
            return gentoken(toasts=['danger'], toastdata=[Markup('bad session')])
        else:
            active = True
            return render_template('story.html', active=active, dashboard=dashboard, report=report, story=story,
deactivate=deactivate, ml=ml, generatetoken=generatetoken, activateaccount=activateaccount, toasts=toasts, toastdata=toastdata, deploy_url=deploy_url)
    else:
        return gentoken(toasts=['warning'], toastdata=[Markup('no session available <br> please activate account or <a href="'+deploy_url+'/generatetoken" class="link-text">generate new token</a>')])


@app.route('/report')
def report(active=False, dashboard=False, report=True, story=False, deactivate=False, ml=False, generatetoken=False, activateaccount=False,
           toasts=[], toastdata=[]):
    sesid = request.cookies.get('sessionid', '')
    if sesid != '':
        row = Tokens.query.filter_by(session=sesid).first()
        if row is None:
            return gentoken(toasts=['danger'], toastdata=[Markup('bad session')])
        else:
            active = True
            return render_template('report.html', active=active, dashboard=dashboard, report=report, story=story, deactivate=deactivate, ml=ml, generatetoken=generatetoken, activateaccount=activateaccount, toasts=toasts, toastdata=toastdata, deploy_url=deploy_url)
    else:
        return gentoken(toasts=['warning'], toastdata=[Markup('no session available <br> please activate account or <a href="'+deploy_url+'/generatetoken" class="link-text">generate new token</a>')])

```

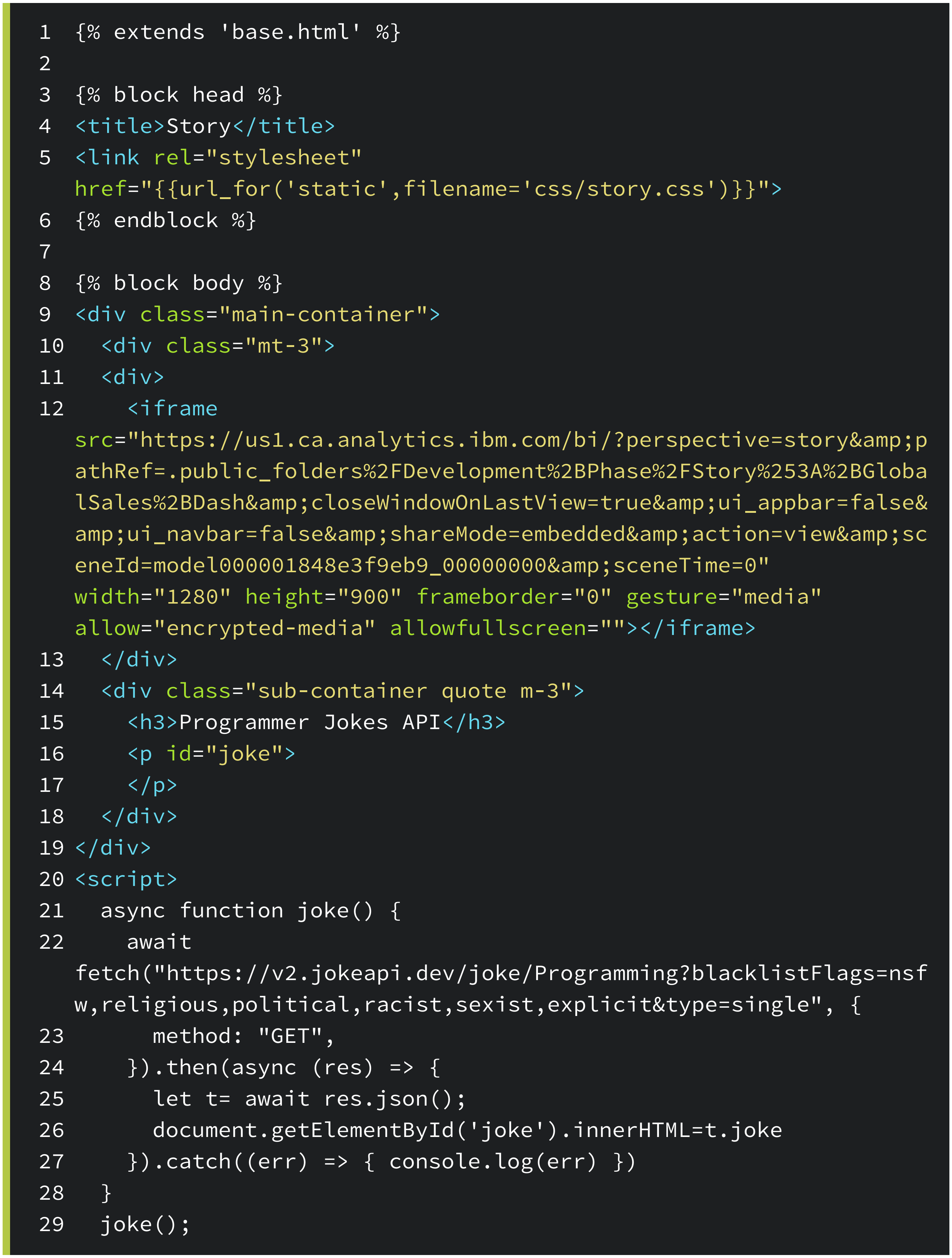
7.11 Report Frontend

```
{% extends 'base.html' %}

{% block head %}
```

```
<title>Report</title>
<link rel="stylesheet" href="{{url_for('static',filename='css/report.css')}}">
{% endblock %}

{% block body %}
<div class="main-container">
  <div>
    <div>
      <iframe src="https://us1.ca.analytics.ibm.com/bi/?pathRef=.public_folders%2FDevelopment%2BPhase%2FGlobalSales%2Breport1&amp;closeWindowOnLastView=true&amp;ui_appbar=false&amp;ui_navbar=false&amp;shareMode=embedded&amp;action=run&amp;prompt=false" width="1280" height="900" frameborder="0" gesture="media" allow="encrypted-media" allowfullscreen=""></iframe>
    </div>
    <div>
      <iframe src="https://us1.ca.analytics.ibm.com/bi/?pathRef=.public_folders%2FDevelopment%2BPhase%2FGlobalSales%2Breport2&amp;closeWindowOnLastView=true&amp;ui_appbar=false&amp;ui_navbar=false&amp;shareMode=embedded&amp;action=run&amp;prompt=false" width="1280" height="900" frameborder="0" gesture="media" allow="encrypted-media" allowfullscreen=""></iframe>
    </div>
    <div>
      <iframe src="https://us1.ca.analytics.ibm.com/bi/?pathRef=.public_folders%2FDevelopment%2BPhase%2FGlobalSales%2Breport%2B3-p&amp;closeWindowOnLastView=true&amp;ui_appbar=false&amp;ui_navbar=false&amp;shareMode=embedded&amp;action=run&amp;prompt=false" width="1280" height="900" frameborder="0" gesture="media" allow="encrypted-media" allowfullscreen=""></iframe>
    </div>
    <div>
      <iframe src="https://us1.ca.analytics.ibm.com/bi/?pathRef=.public_folders%2FDevelopment%2BPhase%2FGlobal%2BSales%2Breport4&amp;closeWindowOnLastView=true&amp;ui_appbar=false&amp;ui_navbar=false&amp;share
```

```
Mode=embedded&amp;action=run&amp;prompt=false" width="1280" height="900" frameborder="0" gesture="media" allow="encrypted-media" allowfullscreen=""></iframe>
    </div>
    <div>
      <iframe src="https://us1.ca.analytics.ibm.com/bi/?pathRef=.public_folders%2FDevelopment%2BPhase%2FGlobal%2BSales%2Breport%2B5&amp;closeWindowOnLastView=true&amp;ui_appbar=false&amp;ui_navbar=false&amp;shareMode=embedded&amp;action=run&amp;prompt=false" width="1280" height="900" frameborder="0" gesture="media" allow="encrypted-media" allowfullscreen=""></iframe>
    </div>
    <div>

    </div>
    <div class="sub-container quote m-3">
      <h3>Programmer Jokes API</h3>
      <p id="joke">
      </p>
    </div>
  </div>
</div>
<script>
  async function joke() {
    await fetch("https://v2.jokeapi.dev/joke/Programming?blacklistFlags=nsfw,religious,political,racist,sexist,explicit&type=single", {
      method: "GET",
    }).then(async (res) => {
      let t= await res.json();
      document.getElementById('joke').innerHTML=t.joke
    }).catch((err) => { console.log(err) })
  }
  joke();
</script>
{% endblock %}
```

7.12 Story - Frontend

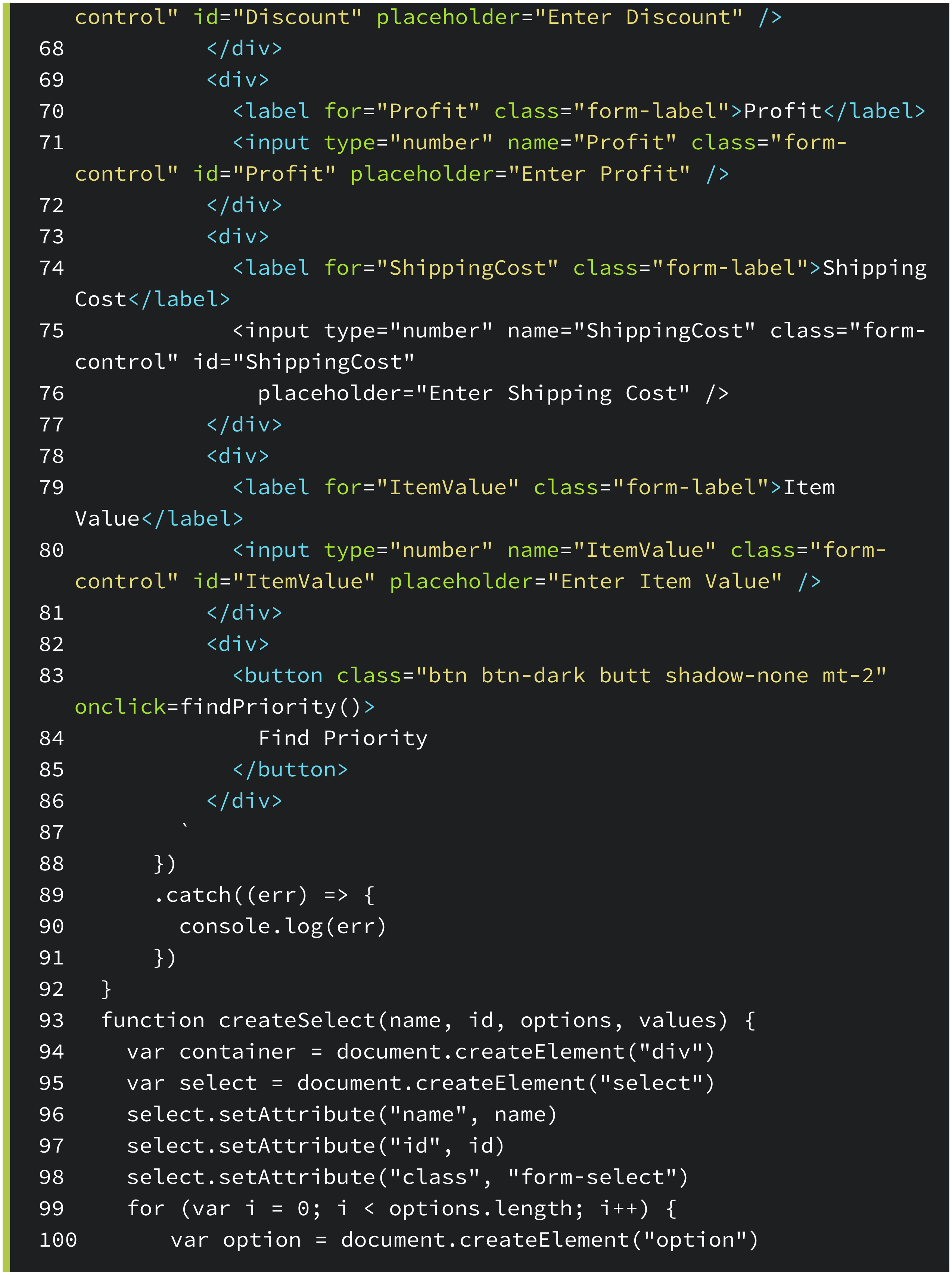


```
{% extends 'base.html' %}

{% block head %}
<title>Story</title>
<link rel="stylesheet" href="{{url_for('static',filename='css/story.css')}}">
{% endblock %}

{% block body %}
<div class="main-container">
  <div class="mt-3">
  <div>
    <iframe src="https://us1.ca.analytics.ibm.com/bi/?perspective=story&amp;pathRef=.public_folders%2FDevelopment%2BPhase%2FStory%253A%2BGlobalSales%2BDash&amp;closeWindowOnLastView=true&amp;ui_appbar=false&amp;ui_navbar=false&amp;shareMode=embedded&amp;action=view&amp;sceneId=model000001848e3f9eb9_00000000&amp;sceneTime=0" width="1280" height="900" frameborder="0" gesture="media" allow="encrypted-media" allowfullscreen=""></iframe>
  </div>
  <div class="sub-container quote m-3">
    <h3>Programmer Jokes API</h3>
    <p id="joke">
    </p>
  </div>
</div>
<script>
  async function joke() {
    await fetch("https://v2.jokeapi.dev/joke/Programming?blacklistFlags=nsfw,religious,political,racist,sexist,explicit&type=single", {
      method: "GET",
    }).then(async (res) => {
      let t= await res.json();
      document.getElementById('joke').innerHTML=t.joke
    }).catch((err) => { console.log(err) })
  }
  joke();
```

```
</script>
{% endblock %}
```

7.13 Machine Learning Frontend

```
{% extends 'base.html' %}

{% block head %}
<title>Machine Learning</title>
<link rel="stylesheet" href="{{url_for('static',filename='css/ml.css')}}">
{% endblock %}

{% block body %}
<div class="main-container">
  <div>
    <div id="input_div" style="margin-top:60px;color: white;">
    </div>
    <div class="sub-container quote m-3">
      <h3>Programmer Jokes API</h3>
      <p id="joke">
      </p>
    </div>
    <div id="prio">
    </div>
  </div>
</div>
<script>
  async function joke() {
    await fetch("https://v2.jokeapi.dev/joke/Programming?blacklistFlags=nsfw,religious,political,racist,sexist,explicit&type=single", {
      method: "GET",
    }).then(async (res) => {
      let t = await res.json();
      document.getElementById('joke').innerHTML = t.joke
    }).catch((err) => { console.log(err) })
  }
  async function renderInputs() {
    await fetch("{{deploy_url}}/mljson", { method: "GET" })
      .then(async (res) => {
        var response_data = await res.json()
        var inputDiv = document.getElementById("input_div")
        var headings = Object.keys(response_data)
        for (var i = 0; i < headings.length; i++) {
          var x = () => {
            var out = ""
            var strings = headings[i].split(" ")
            for (var j = 0; j < strings.length; j++) {
              out += strings[j]
            }
            return out
          }
          inputDiv.append(
            createSelect(
              headings[i],
              x(),
              Object.keys(response_data[headings[i]]),
              Object.values(response_data[headings[i]])
            )
          )
        }

        inputDiv.innerHTML += `
          <div>
            <label for="Sales" class="form-label">Sales</label>
            <input type="number" name="Sales" class="form-control" id="Sales" placeholder="Enter Sales" />
          </div>
          <div>
            <label for="Quantity" class="form-label">Quantity</label>
            <input type="number" name="Quantity" class="form-control" id="Quantity" placeholder="Enter Quantity" />
          </div>
          <div>
            <label for="Discount" class="form-label">Discount</label>
            <input type="number" name="Discount" class="form-
```



```
control" id="Discount" placeholder="Enter Discount" />
          </div>
          <div>
            <label for="Profit" class="form-label">Profit</label>
            <input type="number" name="Profit" class="form-control" id="Profit" placeholder="Enter Profit" />
          </div>
          <div>
            <label for="ShippingCost" class="form-label">Shipping Cost</label>
            <input type="number" name="ShippingCost" class="form-control" id="ShippingCost"
              placeholder="Enter Shipping Cost" />
          </div>
          <div>
            <label for="ItemValue" class="form-label">Item Value</label>
            <input type="number" name="ItemValue" class="form-control" id="ItemValue" placeholder="Enter Item Value" />
          </div>
          <div>
            <button class="btn btn-dark butt shadow-none mt-2" onclick=findPriority()>
              Find Priority
            </button>
          </div>
        `
      })
      .catch((err) => {
        console.log(err)
      })
  }
  function createSelect(name, id, options, values) {
    var container = document.createElement("div")
    var select = document.createElement("select")
    select.setAttribute("name", name)
    select.setAttribute("id", id)
    select.setAttribute("class", "form-select")
    for (var i = 0; i < options.length; i++) {
       var option = document.createElement("option")
      option.setAttribute("value", values[i])
      option.innerText = options[i]
      if (i == 0) {
        option.selected = true
      }
      select.append(option)
    }
    var label = document.createElement("label")
    label.setAttribute("for", name)
    label.className = "form-label"
    label.innerText = name
    container.append(label)
    container.append(select)
    return container
  }
  async function findPriority() {
    var col_name = ['Ship Mode', 'Segment', 'City', 'State', 'Country', 'Market', 'Region',
      'Category', 'Sub-Category', 'Product Name', 'Sales', 'Quantity',
      'Discount', 'Profit', 'Shipping Cost', 'Item Value']
    var form_data = new FormData()
    var final_data = {}
    for (var i = 0; i < col_name.length; i++) {
      var x = () => {
        var out = ""
        var strings = col_name[i].split(" ")
        for (var j = 0; j < strings.length; j++) {
          out += strings[j]
        }
        return out
      }
      form_data.append(col_name[i], Number(document.getElementById(x()).value))
      final_data[col_name[i]] = Number(document.getElementById(x()).value)
    }
    await fetch("{{deploy_url}}/ml-evaluate", { method: "POST", body: JSON.stringify(final_data), headers: new Headers({ 'content-type': 'application/json' }) })
```

```
      .then(async (res) => {
        let t=await res.json()
        let pri=await t.accuracy
        let priority=""
        if(pri==0){
          priority="Critical"
        }else if(pri==1){
          priority="Medium"
        }else if(pri==2){
          priority="High"
        }else if(pri==3){
          priority="Low"
        }
        document.getElementById('prio').innerHTML=`<div class="sub-container quote m-3" >
      <h3>Priority</h3>
      <h2>`+priority+`</h2>
    </div>`
      })
      .catch((err) => {
        console.log(err)
      })
  }
  renderInputs()
  joke()
</script>
{% endblock %}
```

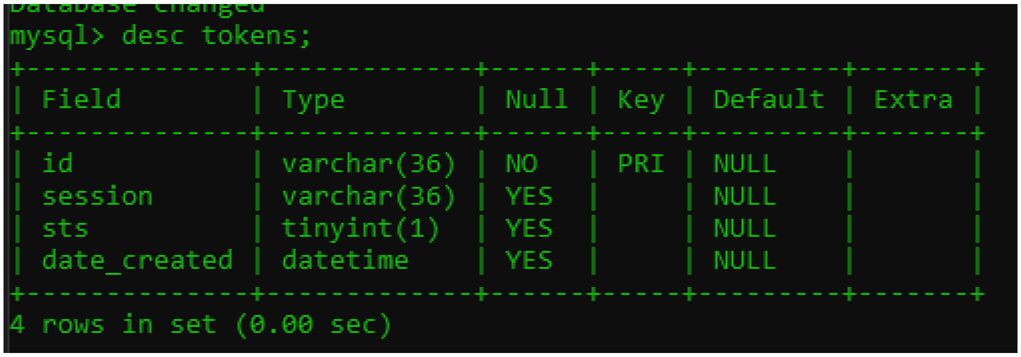
7.14 Machine Learning Backend

```

@app.route('/ml')
def ml(active=False, dashboard=False, report=False, story=False, deactivate=False, ml=True, generatetoken=False, activateaccount=False,
        toasts=[], toastdata=[]):
    sesid = request.cookies.get('sessionid', '')
    if sesid != '':
        row = Tokens.query.filter_by(session=sesid).first()
        if row is None:
            return gentoken(toasts=['danger'], toastdata=[Markup('bad session')])
        else:
            active = True

            return render_template('ml.html', active=active, dashboard=dashboard, report=report, story=story, deactivate=deactivate, ml=ml, generatetoken=generatetoken, activateaccount=activateaccount, toasts=toasts, toastdata=toastdata, deploy_url=deploy_url)
    else:
        return gentoken(toasts=['warning'], toastdata=[Markup('no session available <br> please activate account or <a href="'+deploy_url+'/generatetoken" class="link-text">generate new token</a>')])


# ml functions

@app.route('/mljson')
def mljson():
    with open("static/pkl/data.json", encoding="utf8") as f:
        final_out_data = json.load(f)
        return final_out_data

# ml evalution function


def findAccuracy(value):
    col_name = ['Ship Mode', 'Segment', 'City', 'State', 'Country', 'Market', 'Region',
                'Category', 'Sub-Category', 'Product Name', 'Sales', 'Quantity',
                'Discount', 'Profit', 'Shipping Cost', 'Item Value']
    final_data = {}
    for i in range(len(col_name)):
        final_data[col_name[i]] = [value[col_name[i]]]
    # data frame
    dataFrame = pd.DataFrame(final_data)

    file = "static/pkl/finalModel.pkl"
    fileObject2 = open(file, "rb")
    model = pickle.load(fileObject2, encoding='binary')
    return model.predict(dataFrame)[0]


@app.route('/ml-evaluate', methods=['POST'])
def mlEvaluate():
    return {"accuracy": int(findAccuracy(request.get_json(force=True)))}

```

7.15 Other Flask and backend boiler plate

```
app = Flask(__name__)
app.config['SQLALCHEMY_DATABASE_URI'] = 'mysql+pymysql://' + \
    mysql_user+':'+mysql_pass+'@localhost/globalsalesdata'
db = SQLAlchemy(app)
app.app_context().push()

if __name__ == "__main__":
    app.run(debug=True)

```

7.16 Database Schema (if Applicable)



```
Database changed
mysql> desc tokens;
+--------------+-------------+------+-----+---------+-------+
| Field        | Type        | Null | Key | Default | Extra |
+--------------+-------------+------+-----+---------+-------+
| id           | varchar(36) | NO   | PRI | NULL    |       |
| session      | varchar(36) | YES  |     | NULL    |       |
| sts          | tinyint(1)  | YES  |     | NULL    |       |
| date_created | datetime    | YES  |     | NULL    |       |
+--------------+-------------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```

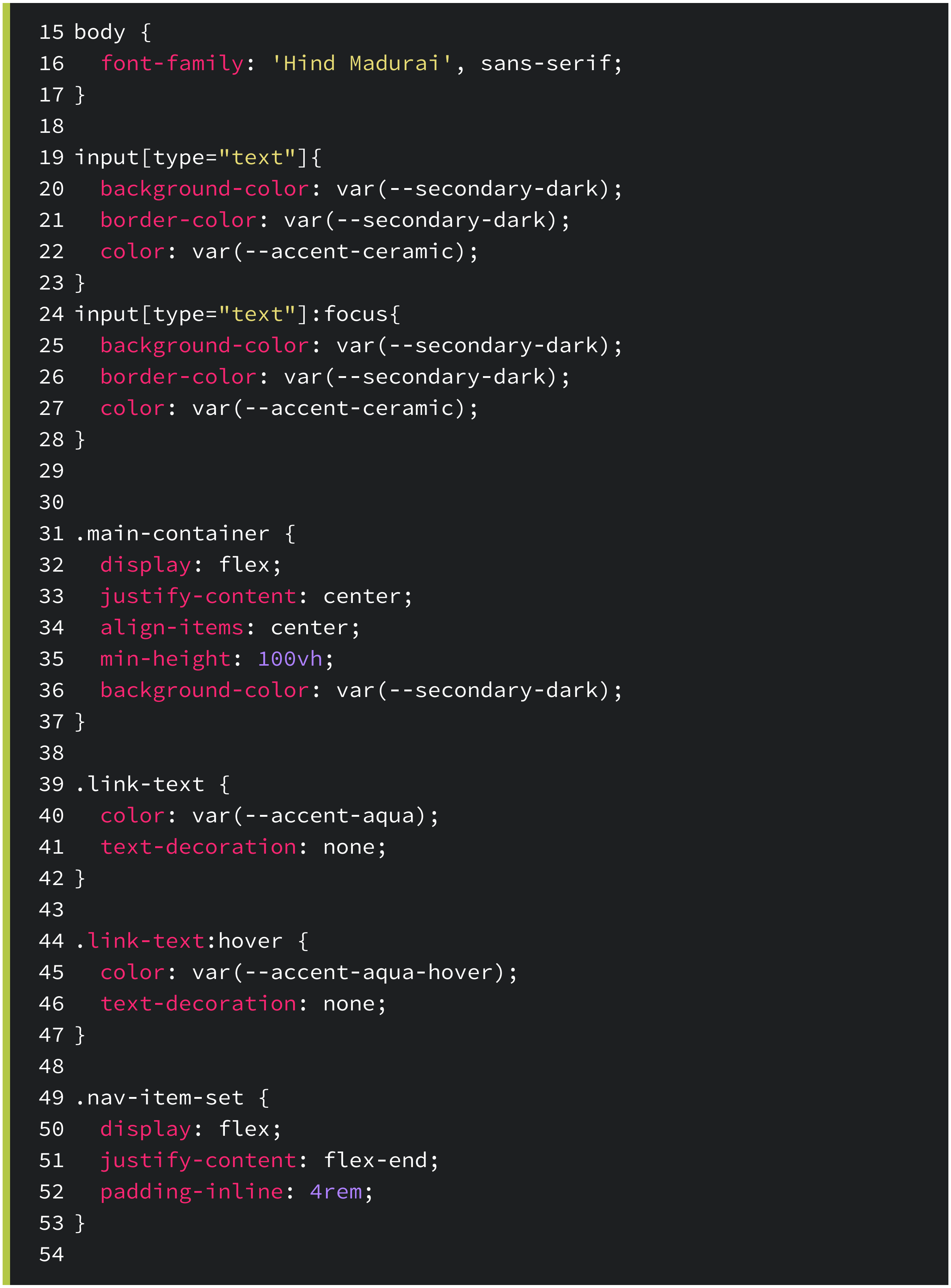
```
class Tokens(db.Model):
    id = db.Column(db.String(36), primary_key=True)
    session = db.Column(db.String(36))
    sts = db.Column(db.Boolean, default=False)
    date_created = db.Column(db.DateTime,
default=datetime.utcnow)

    def __repr__(self):
        return '<Task %r>' % self.id
```

7.17 Locustfile - performance testing

```
import time
from locust import HttpUser, task, between
class QuickstartUser(HttpUser):
    wait_time = between(1, 5)

    @task
    def gethtml(self):
        self.client.get("/generatetoken")
        self.client.get("/activateacc")

    @task(3)
    def view_items(self):
        self.client.post("/generatetoken")
```

7.18 Main.css

```
@import url('https://fonts.googleapis.com/css2?family=Hind+Madurai:wght@300;400;500;600;700&display=swap');

:root {
  --primary-dark: #222831;
  --secondary-dark: #393E46;
  --accent-aqua: #00ADB5;
  --accent-aqua-hover: #00cfda;
  --accent-ceramic: #EEEEEE;
}

.hidden{
  display:none;
}

```



```css
body {
  font-family: 'Hind Madurai', sans-serif;
}

input[type="text"]{
  background-color: var(--secondary-dark);
  border-color: var(--secondary-dark);
  color: var(--accent-ceramic);
}
input[type="text"]:focus{
  background-color: var(--secondary-dark);
  border-color: var(--secondary-dark);
  color: var(--accent-ceramic);
}


.main-container {
  display: flex;
  justify-content: center;
  align-items: center;
  min-height: 100vh;
  background-color: var(--secondary-dark);
}

.link-text {
  color: var(--accent-aqua);
  text-decoration: none;
}

.link-text:hover {
  color: var(--accent-aqua-hover);
  text-decoration: none;
}

.nav-item-set {
  display: flex;
  justify-content: flex-end;
  padding-inline: 4rem;
}

```

```css
.brand {
  padding-inline: 2rem;
}

@media screen and (max-width: 992px) {
  .nav-item-set {
    justify-content: flex-start;
    padding: 0;
  }

  .brand {
    padding-inline: 0;
  }

}

.dropdown-menu-dark {
  --bs-dropdown-link-active-bg: rgba(255, 255, 255, 0.15);
}

.butt {
  background-color: var(--accent-aqua);
}


```

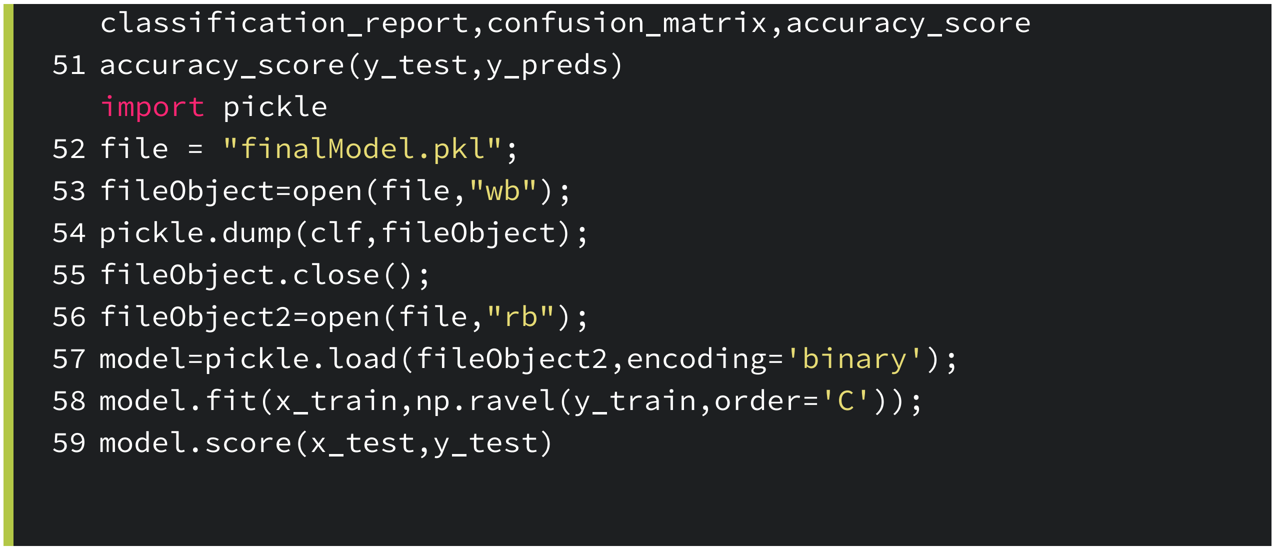
7.19 generate token css

```css
h1 {
  color: white;
  font-weight: 400;
  font-size: 4ch;
}

.sub-container {
  width: 90vw;
  max-width: 550px;
  min-height: 32vh;
  border-radius: 0.5rem;
  background-color: #222831;
  display: flex;
  flex-direction: column;
  justify-content: space-between;
}

.input-tf {
  border-top-left-radius: 0.3rem;
  border-bottom-left-radius: 0.3rem;
  border-top-right-radius: 0rem;
  border-bottom-right-radius: 0rem;
  height:3.4rem;
}


.butt {
  background-color: #00ADB5;
  width: 13rem;
  align-self: center;
}

.link-text {
  align-self: center;
}
.butt2 {
  background-color: #00ADB5;
  border-top-right-radius: 0.3rem;
  border-bottom-right-radius: 0.3rem;
  border-top-left-radius: 0rem;
  border-bottom-left-radius: 0rem;
  width: fit-content;
  height:3.4rem;
}
.input-cont{
  display: flex;
  flex-direction: row;
}
.generate-btn-group{
  display: flex;
  flex-direction: column;
}
```

7.20 Machine Learning IPYNB

```
import os
os.environ['KAGGLE_USERNAME'] = "sriramkarthickk"
os.environ['KAGGLE_KEY'] = "498d9367562336e1b9076a3c240575eb"

from kaggle.api.kaggle_api_extended import KaggleApi
api = KaggleApi()
api.authenticate()
api.dataset_download_files('apoorvaappz/global-super-store-dataset', path=".")
import pandas as pd
import numpy as np
dataFrame=pd.read_csv("./global-super-store-dataset/Global_Superstore2.csv")
dropingFields=["Row ID","Order ID","Order Date","Ship Date","Customer ID","Customer Name","Postal Code","Product ID"]
dataFrame.drop(dropingFields,axis=1,inplace=True)
dataFrame["Order Priority"].unique()
dataFrame["Order Priority"].replace("Critical",0,inplace=True)
dataFrame["Order Priority"].replace("Medium",1,inplace=True)
dataFrame["Order Priority"].replace("High",2,inplace=True)
dataFrame["Order Priority"].replace("Low",3,inplace=True)
dataFrame
dataFrame["Item Value"]=dataFrame["Sales"]/dataFrame["Quantity"]
list(dataFrame.iloc[0])
dataFrame["Product Name"].unique()
x=dataFrame.drop("Order Priority",axis=1)
y=dataFrame["Order Priority"]
dataFrame.columns
x.iloc[85].values
#Turn the categories into numbers
from sklearn.preprocessing import OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.model_selection import train_test_split
import json

categorical_features=["Ship
```

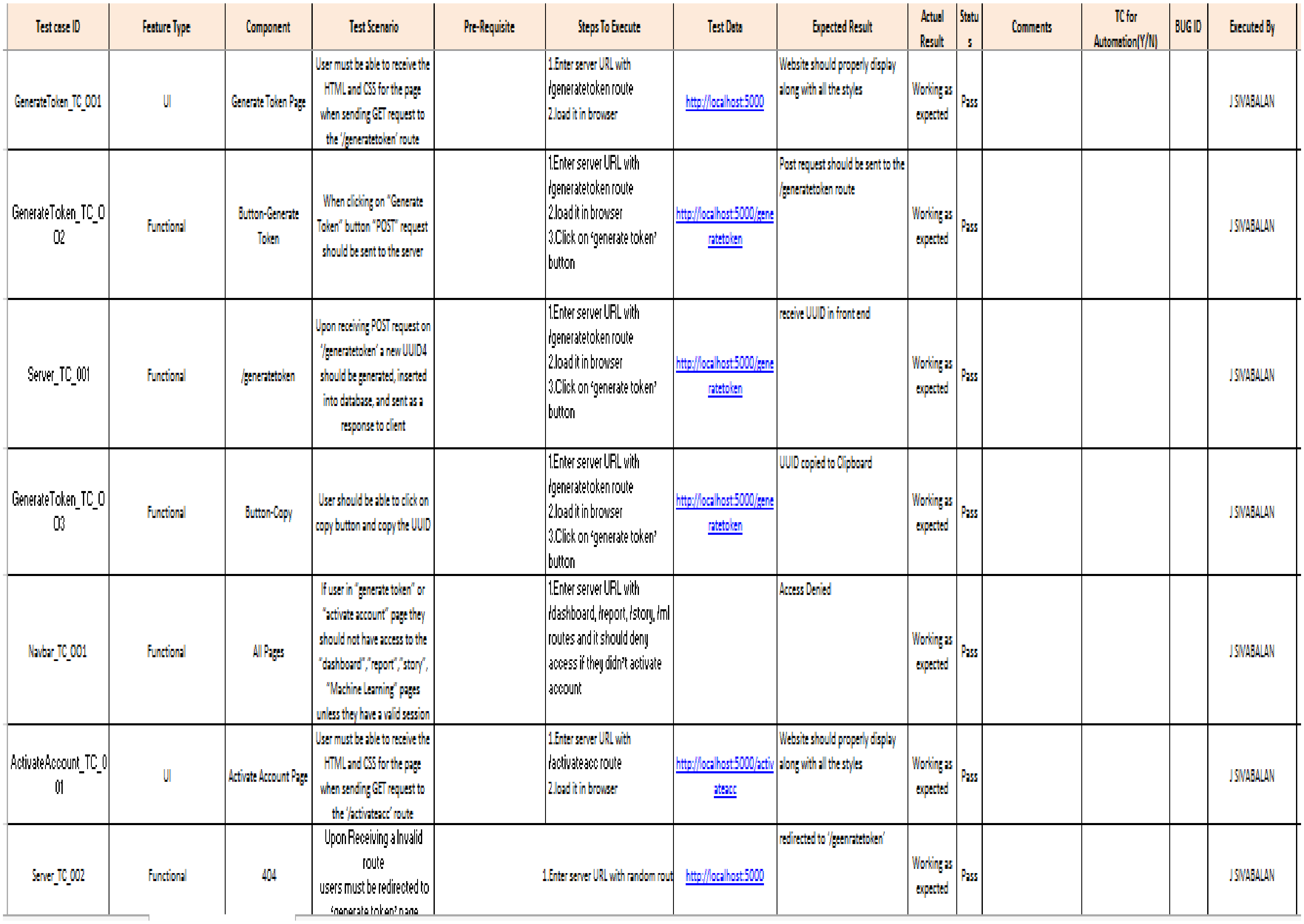
```python
Mode","Segment","City","State","Country","Market","Region","Product Name"]
# one_hot=OneHotEncoder()
# transformer=ColumnTransformer([(
#                                 "one_hot",
#                                 one_hot,
#                                 categorical_features
#                              )],
#                                 remainder="passthrough"
#                             )
# transformed_x = transformer.fit_transform(x)
# pd.DataFrame(transformed_x)
final_out_data={}
for i in categorical_features:
    unique_values=list(x[i].unique())
    for j in unique_values:
        if(i in final_out_data):
            final_out_data[i][j]=unique_values.index(j)
        else:
            final_out_data[i]={}
            final_out_data[i][j]=unique_values.index(j)
x[i].replace(to_replace=j,value=int(unique_values.index(j)),inplace=True)

import json
with open('data.json', 'w', encoding='utf-8') as f:
    json.dump(final_out_data, f, ensure_ascii=False, indent=4)
print("finished")
np.random.seed(40)
x_train,x_test,y_train,y_test=train_test_split(x,y,train_size=0.2)
x_train.shape, x_test.shape, y_train.shape, y_test.shape
from sklearn.ensemble import RandomForestClassifier
clf=RandomForestClassifier(n_estimators=100)

clf.fit(x_train,y_train);
y_preds=clf.predict(x_test)
y_preds
clf.score(x_test,y_test)
from sklearn.metrics import
```



```
classification_report,confusion_matrix,accuracy_score
accuracy_score(y_test,y_preds)
import pickle
file = "finalModel.pkl";
fileObject=open(file,"wb");
pickle.dump(clf,fileObject);
fileObject.close();
fileObject2=open(file,"rb");
model=pickle.load(fileObject2,encoding='binary');
model.fit(x_train,np.ravel(y_train,order='C'));
model.score(x_test,y_test)
```

8. TESTING

8.1 Test Cases



Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Comments	TC for Automation(Y/N)	BUG ID	Executed By
GenerateToken_TC_001	UI	Generate Token Page	User must be able to receive the HTML and CSS for the page when sending GET request to the '/generatetoken' route		1.Enter server URL with /generatetoken route 2.load it in browser	http://localhost:5000	Website should properly display along with all the styles	Working as expected	Pass				J SIVABALAN
GenerateToken_TC_002	Functional	Button-Generate Token	When clicking on "Generate Token" button "POST" request should be sent to the server		1.Enter server URL with /generatetoken route 2.load it in browser 3.Click on 'generate token' button	http://localhost:5000/generatetoken	Post request should be sent to the /generatetoken route	Working as expected	Pass				J SIVABALAN
Server_TC_001	Functional	/generatetoken	Upon receiving POST request on '/generatetoken' a new UUID4 should be generated, inserted into database, and sent as a response to client		1.Enter server URL with /generatetoken route 2.load it in browser 3.Click on 'generate token' button	http://localhost:5000/generatetoken	receive UUID in front end	Working as expected	Pass				J SIVABALAN
GenerateToken_TC_003	Functional	Button-Copy	User should be able to click on copy button and copy the UUID		1.Enter server URL with /generatetoken route 2.load it in browser 3.Click on 'generate token' button	http://localhost:5000/generatetoken	UUID copied to Clipboard	Working as expected	Pass				J SIVABALAN
Navbar_TC_001	Functional	All Pages	If user in "generate token" or "activate account" page they should not have access to the "dashboard","report","story", "Machine Learning" pages unless they have a valid session		1.Enter server URL with /dashboard, /report, /story, /ml routes and it should deny access if they didn't activate account		Access Denied	Working as expected	Pass				J SIVABALAN
ActivateAccount_TC_001	UI	Activate Account Page	User must be able to receive the HTML and CSS for the page when sending GET request to the '/activateacc' route		1.Enter server URL with /activateacc route 2.load it in browser	http://localhost:5000/activateacc	Website should properly display along with all the styles	Working as expected	Pass				J SIVABALAN
Server_TC_002	Functional	404	Upon Receiving a Invalid route users must be redirected to 'generate token' page		1.Enter server URL with random rout	http://localhost:5000	redirected to '/geenratetoken'	Working as expected	Pass				J SIVABALAN

Test Scenarios

1 Verify user is able to receive Web pages on request

2 Is the user able to generate token upon request?

3 Is the user able to Activate Account using the generated token and receive a valid session?

4 Are the Invalid tokens rejected properly?

5 Is the access denied for protected pages (‘/dashboard’, ’/report’, ’/story’, ’/ml’, ‘/mljson’, ‘/ml-evaluate’, ’/deactivate’) sessions which are not activated?

6 Are the IBM cognos iframes working properly?

7 Is the deactivate button deleting the account, and resetting the session?

Machine Learning Form

1 Is the from generated after receiving the json response from /mljson?

2 Is the from submitting to /ml-evaluate as post request?

3 Is the ML model loaded properly from the Pickle?

4 Is the Order Priority properly Predicted?

5 Is the Order Priority displayed in the UI?

8.2 User Acceptance Testing

1. Purpose of Document

The purpose of this document is to briefly explain the test coverage and open issuesof the Global Sales Data Analytics project at the time of the release to User Acceptance Testing (UAT).

2. Defect Analysis

This reportshows the numberof resolved or closed bugs at each severity level, and how they were resolved

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	1	0	0	0	1
Duplicate	1	1	0	0	2

External	0	0	0	0	0
Fixed	2	1	0	0	3
Not Reproduced	0	0	0	0	0
Skipped	0	0	0	0	0
Won't Fix	1	0	0	0	1
Totals	5	2	0	0	7

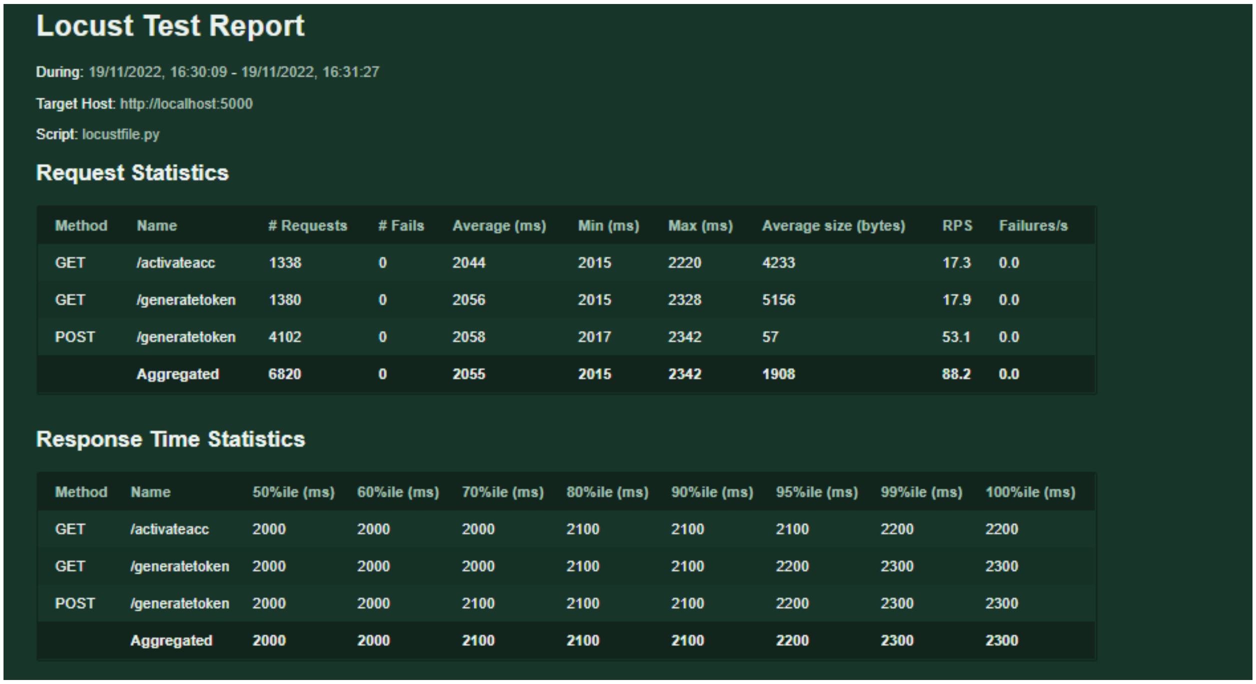
3. Test Case Analysis

This report shows the numberof test cases that have passed, failed,and untested

Section	Total Cases	Not Tested	Fail	Pass
Print Engine	0	0	0	0
Client Application	25	0	0	25
Security	8	0	0	8

9. RESULTS

Performance Metrics



Locust Test Report

During: 19/11/2022, 16:30:09 - 19/11/2022, 16:31:27

Target Host: http://localhost:5000

Script: locustfile.py

Request Statistics

Method	Name	# Requests	# Fails	Average (ms)	Min (ms)	Max (ms)	Average size (bytes)	RPS	Failures/s
GET	/activateacc	1338	0	2044	2015	2220	4233	17.3	0.0
GET	/generatetoken	1380	0	2056	2015	2328	5156	17.9	0.0
POST	/generatetoken	4102	0	2058	2017	2342	57	53.1	0.0
	Aggregated	**6820**	**0**	**2055**	**2015**	**2342**	**1908**	**88.2**	**0.0**

Response Time Statistics

Method	Name	50%ile (ms)	60%ile (ms)	70%ile (ms)	80%ile (ms)	90%ile (ms)	95%ile (ms)	99%ile (ms)	100%ile (ms)
GET	/activateacc	2000	2000	2000	2100	2100	2100	2200	2200
GET	/generatetoken	2000	2000	2000	2100	2100	2200	2300	2300
POST	/generatetoken	2000	2000	2100	2100	2100	2200	2300	2300
	Aggregated	**2000**	**2000**	**2100**	**2100**	**2100**	**2200**	**2300**	**2300**

10. ADVANTAGES & DISADVANTAGES

Advantages:

- Super Fast and Reliable (Can handle more than 200RPS with ease)
- Secure
- Uses Tokenization
- No Account Necessary
- Privacy Guarenteed.
- One time Activation (no subscription)
- Has Machine Learning to Predict Order Priority

Disadvantage:

- Uses Cognos, hence not Mobile Responsive.

11. CONCLUSION

The project Global Sales Data Analytics has been successfully performed using Agile Methodologies. During the project, the DataSet was Cleaned, Prepared, to use for Analytics. Dashboards, Reports and Stories were prepared. Mysql was used as a database for Storing and maintaining user Tokens and Sessions. Programmer Joke API was used. Flask for hosting the webapp, and Machine learning models to predict Order Priority.

12. FUTURE SCOPE

- We can build a much better machine learning model which can predict Order Priority more accurately.
- Create a more Mobile Friendly UI

13. APPENDIX

Dashboard: A graphical representation of analyses performed by algorithms.

Cloud: A broad term that refers to any internet-based application or service that is hosted remotely.

Cloud Computing: A distributed computing system hosted and running on remote servers and accessible from anywhere on the internet.

Big Data: Big data refers to datasets whose size is beyond the ability of typical database software tools to capture, store, manage and analyze.

Business Intelligence: The general term used for the identification, extraction and analysis of data.

API (Application Program Interface): A set of programming standards and instructions for accessing or building web-based software applications.

Application: Software that enables a computer to perform a certain task.

Classification Analysis: A systematic process for obtaining important and relevant information about data (metadata) and assigning data to a particular group or class.

Data Cleansing: The process of reviewing and revising data to delete duplicate entries, correct misspelling and other errors, add missing data and provide consistency.

Data Warehouse: A repository for enterprise-wide data but in a structured format after cleaning and integrating with other sources. Data warehouses are typically used for conventional data (but not exclusively).

Data Set: A collection of data, very often in tabular form.

Database: A digital collection of data and the structure around which the data is organized. The data is typically entered into and accessed via a database management system.

Database Management System (DBMS): Software that collects and provides access to data in a structured format.

Machine Learning: A method of designing systems that can learn, adjust and improve based on the data fed to them. Using predictive and statistical algorithms that are fed to these machines, they learn and continually zero in on “correct” behavior and insights and they keep improving as more data flows through the system.

Mean: The weighted average of data. The population mean is denoted by ? (Greek letter mu) and the sample mean is denoted by x?.

Median: The middle value of a data set when arranged in order of magnitude.

Mode: The measurement that occurs most often in a data set.

SQL (Structured Query Language): A programming language for retrieving data from a relational database.

Source Code - https://github.com/SU-Ki-MA/globalsalesdata

Video Link - https://youtu.be/OqS90_7rWzg