**INVENTORY MANAGEMENT SYSTEM FOR RETAILERS**

**CLOUD APP DEVELOPMENT DOMAIN**

**TEAM ID: PNT2022TMID07456**

**A PROJECT REPORT**

***SUBMITTED BY***

|  |
| --- |
| **KARTHIBAN.S.K.** |
| **ARUNACHALAM.R** |
| **DHINESH.M** |
| **GANAPPRIYAN.C** |

**COMPUTER SCIENCE AND ENGINEERING**

**P. A. COLLEGE OF ENGINEERING AND TECHNOLOGY**

**(AUTONOMOUS)**

**POLLACHI, COIMBATORE DT. - 642 002**

**NOVEMBER 2022**

**P. A. COLLEGE OF ENGINEERING AND TECHNOLOGY**

**BONAFIDE CERTIFICATE**

CERTIFIED THAT THIS PROJECT REPORT **“INVENTORY MANAGMENT SYSTEM FOR RETAILERS”** IS THE WORK OF **“KARTHIBAN.S.K. (721719104036), ARUNACHALAM.R (721719104009), DHINESH.M (721719104019), GANAPPRIYAN.C (721719104026)”** WHO CARRIED THE PROJECT WORK UNDER OUR SUPERVISION.

|  |  |  |
| --- | --- | --- |
| **SIGNATURE**  **DR. D. CHITRA**  PROFESSOR  **HEAD OF THE DEPARTMENT**  COMPUTER SCIENCE AND ENGINEERING  P. A. COLLEGE OF ENGINEERING AND TECHNOLOGY |  | **SIGNATURE**  **FACULTY MENTOR**  DR. M.UMASELVI  ASSOCIATE PROFESSOR  COMPUTER SCIENCE AND ENGINEERING  P. A. COLLEGE OF ENGINEERING AND TECHNOLOGY  **SIGNATURE**  **FACULTY EVALUATOR**  Mrs. P. SANGEETHA  ASSISTANT PROFESSOR  COMPUTER SCIENCE AND ENGINEERING  P. A. COLLEGE OF ENGINEERING AND TECHNOLOGY |

SUBMITTED TO THE VIVA- VOCE EXAMINATION HELD ON ----------------------------------

**INTERNAL EXAMINER** **EXTERNAL EXAMINER**

**PROJECT REPORT:**

1. INTRODUCTION

1.1 PROJECT OVERVIEW

1.2 PURPOSE

2. LITERATURE SURVEY

2.1 EXISTING PROBLEM

2.2 REFERENCES

2.3 PROBLEM STATEMENT DEFINITION

3. IDEATION & PROPOSED SOLUTION

3.1 EMPATHY MAP CANVAS

3.2 IDEATION & BRAINSTORMING

3.3 PROPOSED SOLUTION

3.4 PROBLEM SOLUTION FIT

4. REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENT

4.2 NON-FUNCTIONAL REQUIREMENTS

5. PROJECT DESIGN

5.1 DATA FLOW DIAGRAMS

5.2 SOLUTION & TECHNICAL ARCHITECTURE

5.3 USER STORIES

6. PROJECT PLANNING & SCHEDULING

6.1 SPRINT PLANNING & ESTIMATION

6.2 SPRINT DELIVERY SCHEDULE

7. RESULTS

8. ADVANTAGES & DISADVANTAGES

9. CONCLUSION

10. FUTURE SCOPE

11. APPENDIX

SOURCE CODE

GITHUB & PROJECT DEMO LINK

**1. INTRODUCTION**

* 1. **PROJECT OVERVIEW**

Inventory Management System for Retailers is the process of ensuring you carry merchandise that shoppers want, with neither too little nor too much on hand. By managing inventory, retailers meet customer demand without running out of stock or carrying excess supply. In practice, effective retail inventory management results in lower costs and a better understanding of sales patterns. Retail inventory management tools and methods give retailers more information on which to run their businesses. Applications have been developed to help retailers track and manage stocks related to their own products. The System will ask retailers to create their accounts by providing essential details. Retailers can access their accounts by logging into the application. In today’s more turbulent environment there is no longer any possibility of manufacturing and marketing acting independently of each other. It is now generally accepted that the need to understand and meet customer requirements is a prerequisite for survival. At the same time, in the search for improved cost competitiveness, manufacturing management has been the subject of massive renaissance.

**1.2 PURPOSE**

The objective of the project is to create an application that help retailers to track and manage stocks which results in lower costs and a better understanding of sales pattern. Retail inventory management is the process of ensuring retailers meet customer demand without running out of stock or carrying excess supply. By creating an application, retailers can log in to it and can update inventory details, also users will be able to add new stock by submitting essential details related to the stock. Retailers can also view details of the current inventory. The System will automatically send an email alert to the retailers if there is no stock found in their accounts. So that they can order new stock.

**2. LITERATURE SURVEY**

**2.1 EXISTING PROBLEM**

• In existing solutions, we cannot generate detailed, neat, and insightful inventory reports.

• In existing solutions, we cannot monitor your stock at all times with product tracking.

• In existing solutions, we cannot transfer products across your warehouses based on demand.

• The problem retailers face is that there is no system in place to record and maintain their inventory data.

• It is difficult for the owner to record inventory data quickly and securely because they only keep it in a logbook and are not properly organized.

**2.2 REFERENCES**

•Zoho Inventory

Reference Link: Https://Www.Zoho.Com/In/Inventory/

•Gofrugal

Reference Link: Https://Www.Gofrugal.Com/Inventory-Management-Software

•Microbiz Cloud

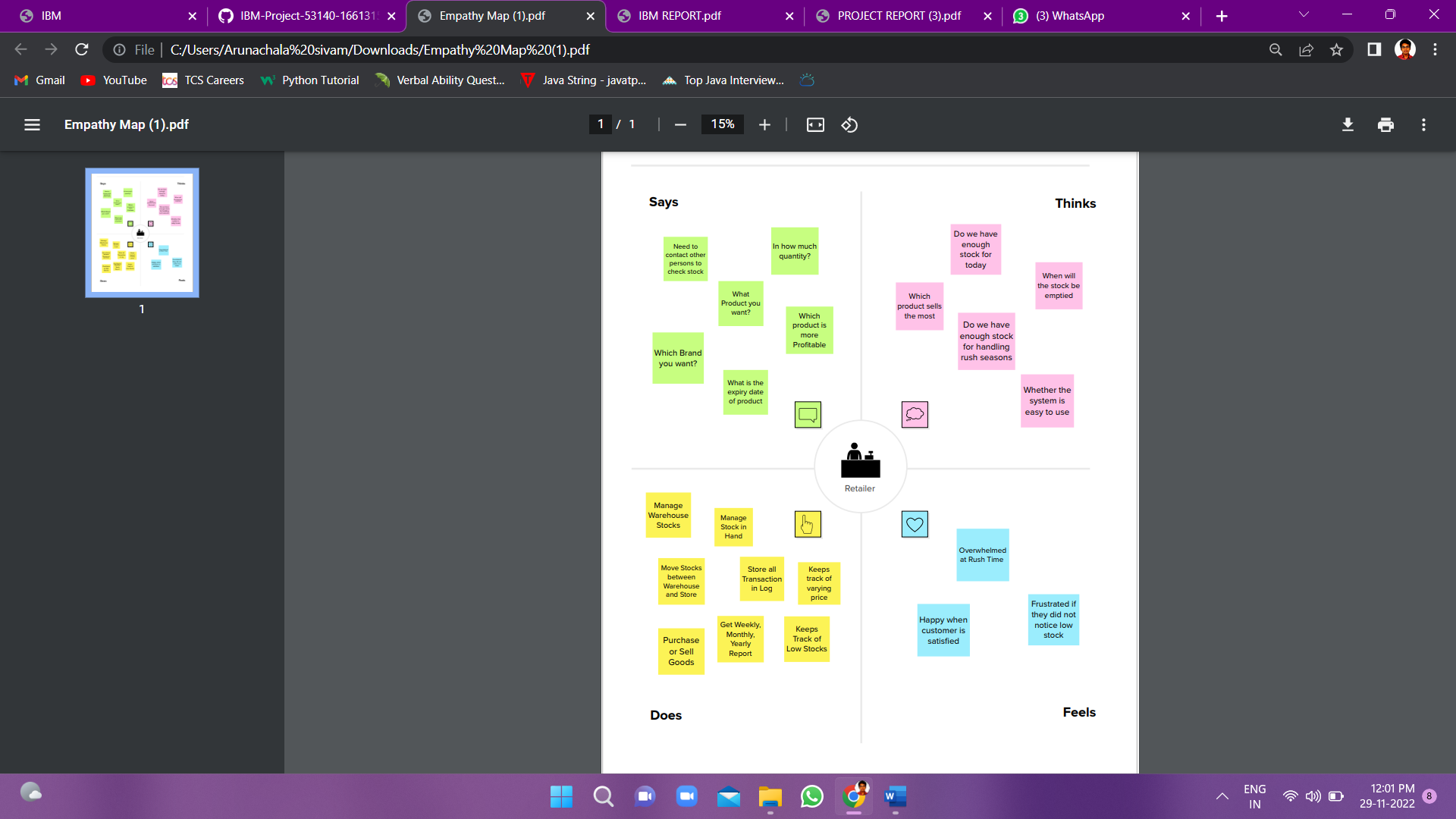
Reference Link: Https://Microbiz.Com/

**2.3 PROBLEM STATEMENT DEFINITION**

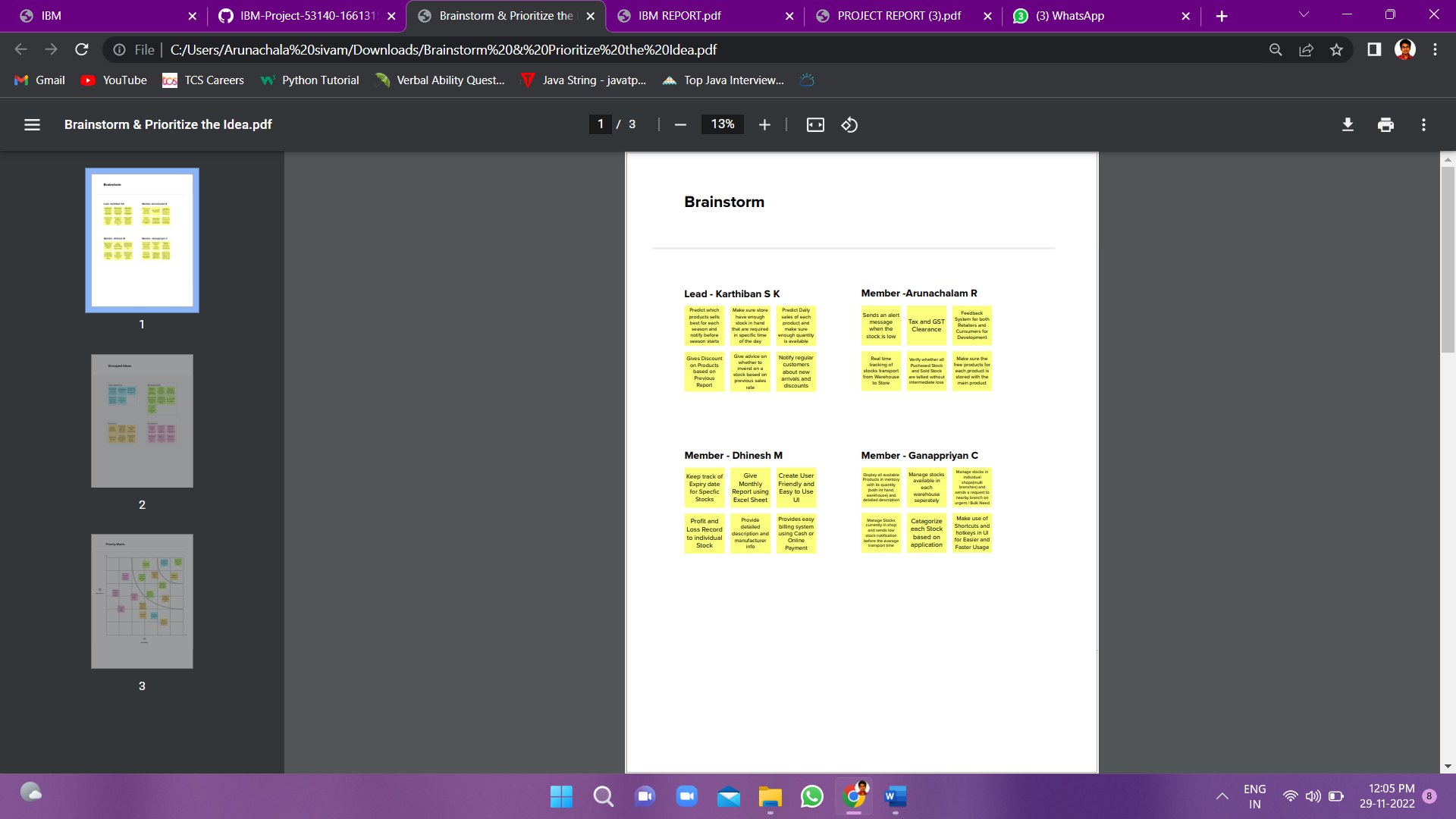
The problem retailers face is that there is no system in place to record and maintain their inventory data. It is difficult for the owner to record inventory data quickly and securely because they only keep it in a logbook and are not properly organized.

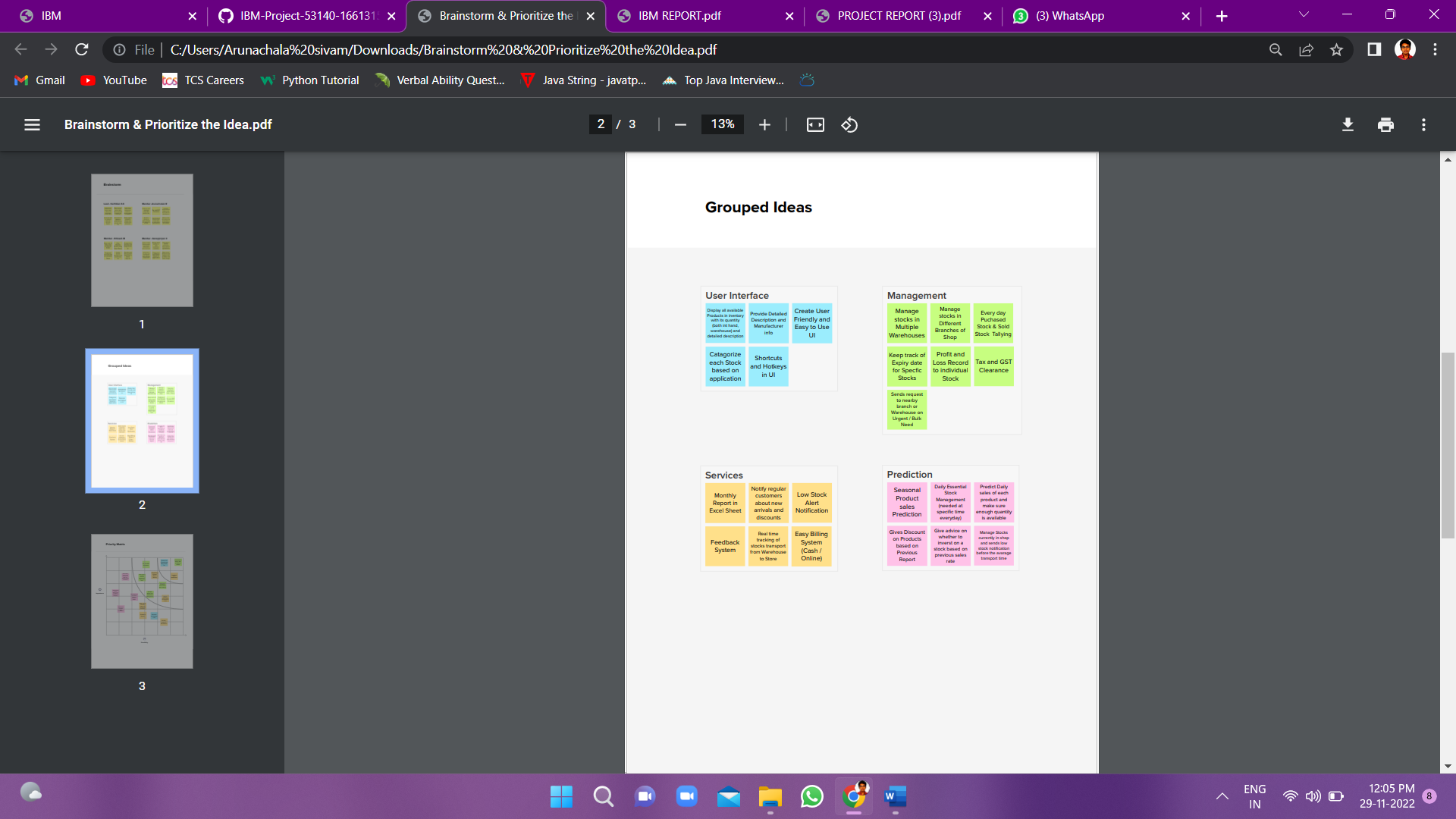
**3. IDEATION & PROPOSED SOLUTION**

3.1 EMPATHY MAP CANVAS

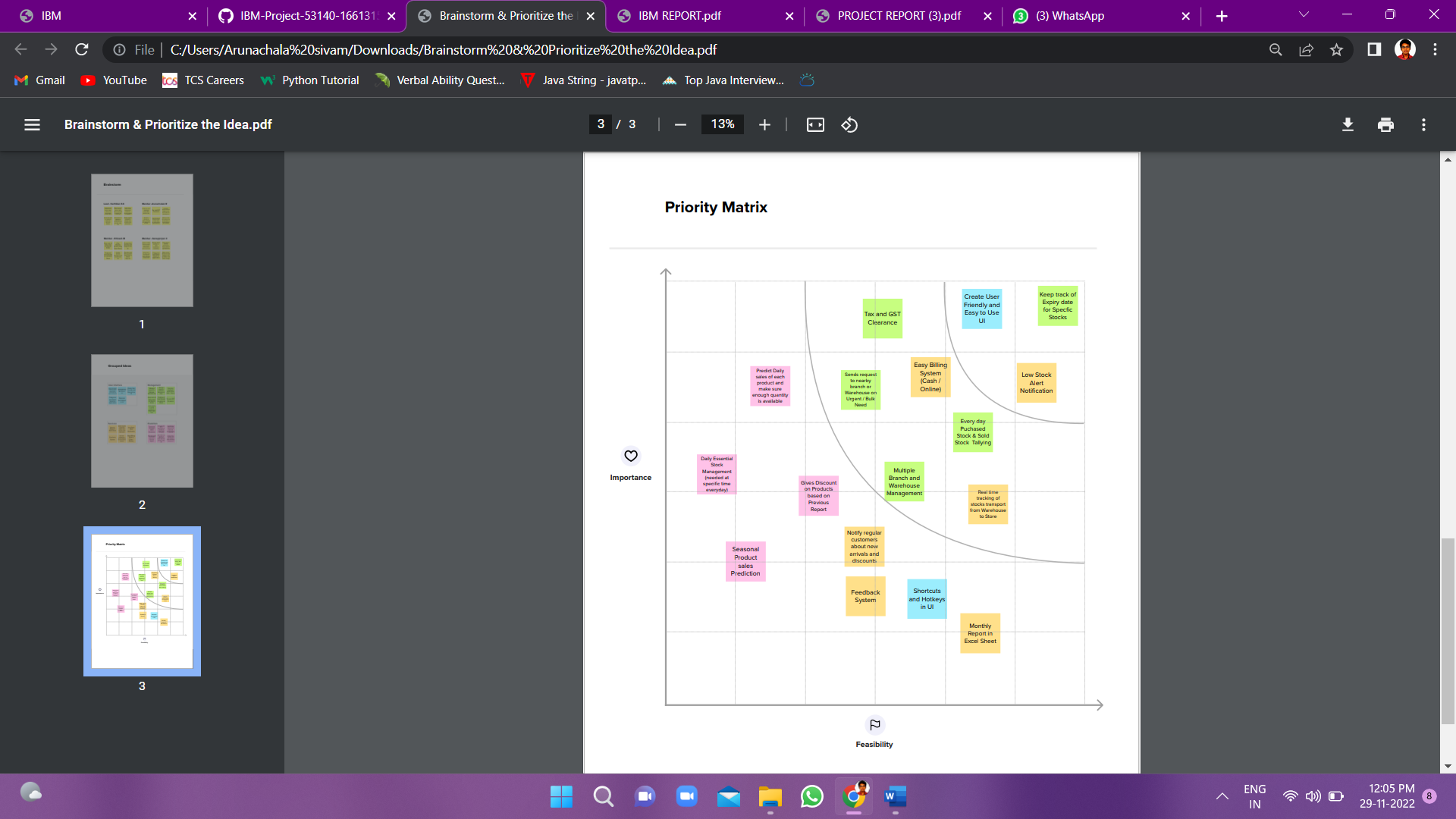


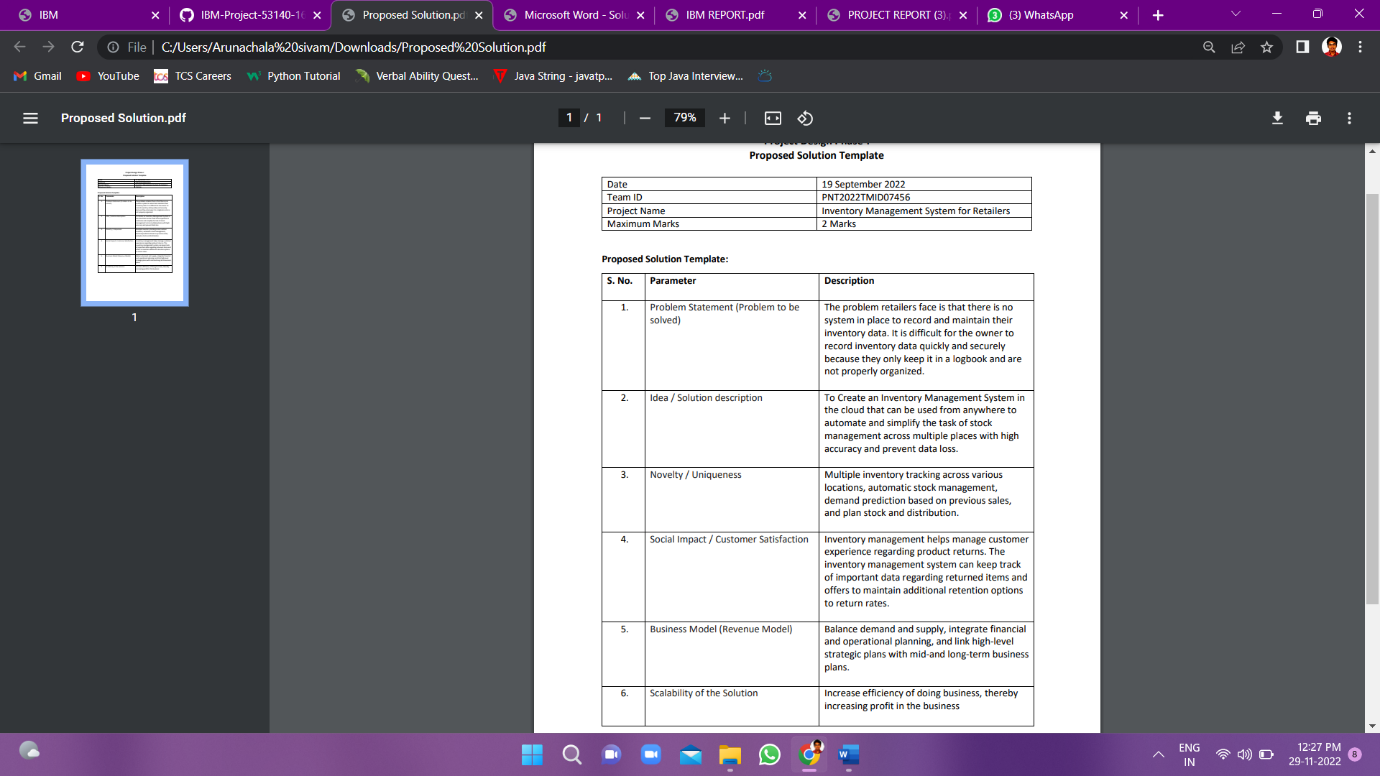
**3.2 IDEATION & BRAINSTORMING**

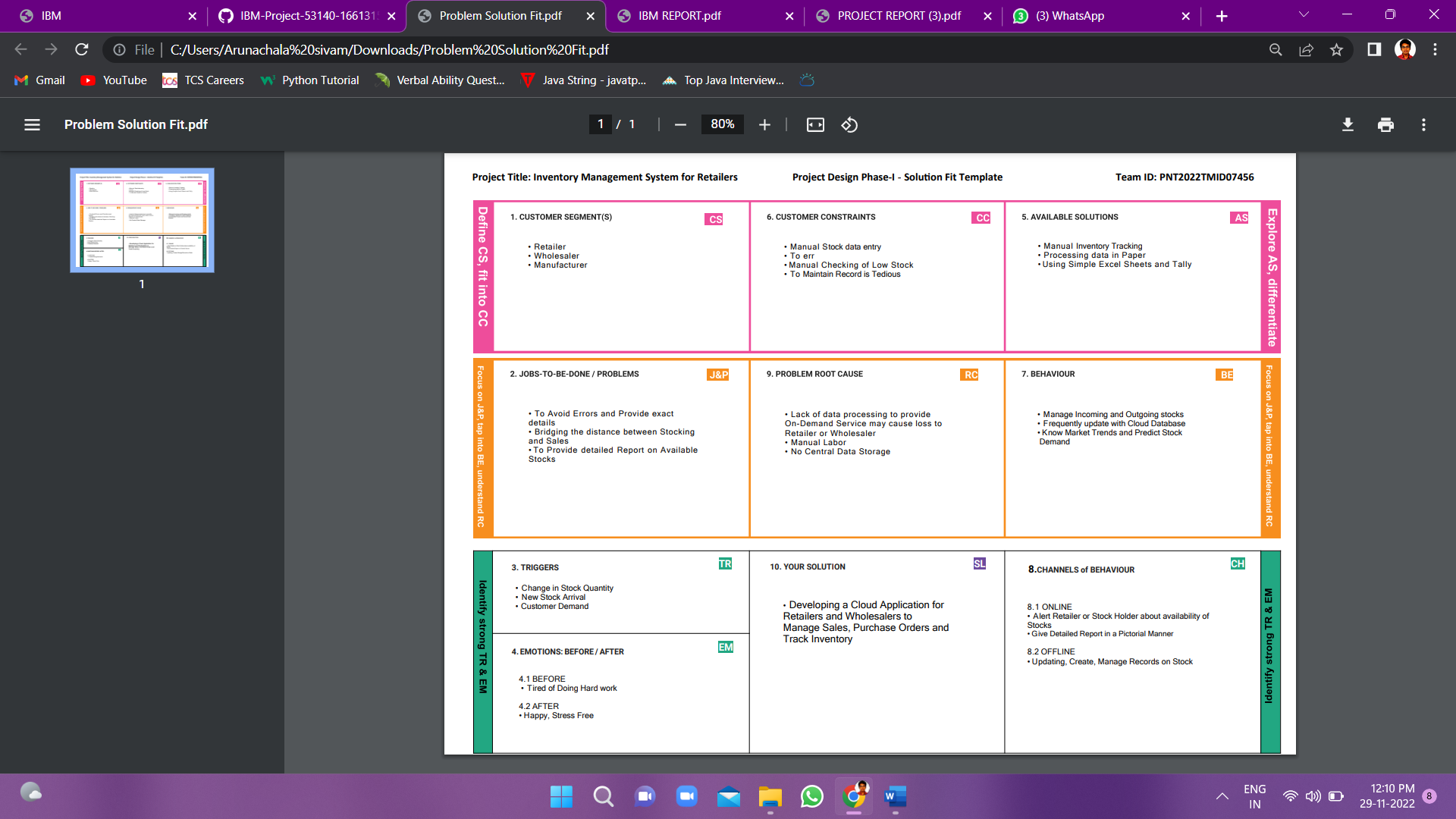
****



3.3 PROPOSED SOLUTION

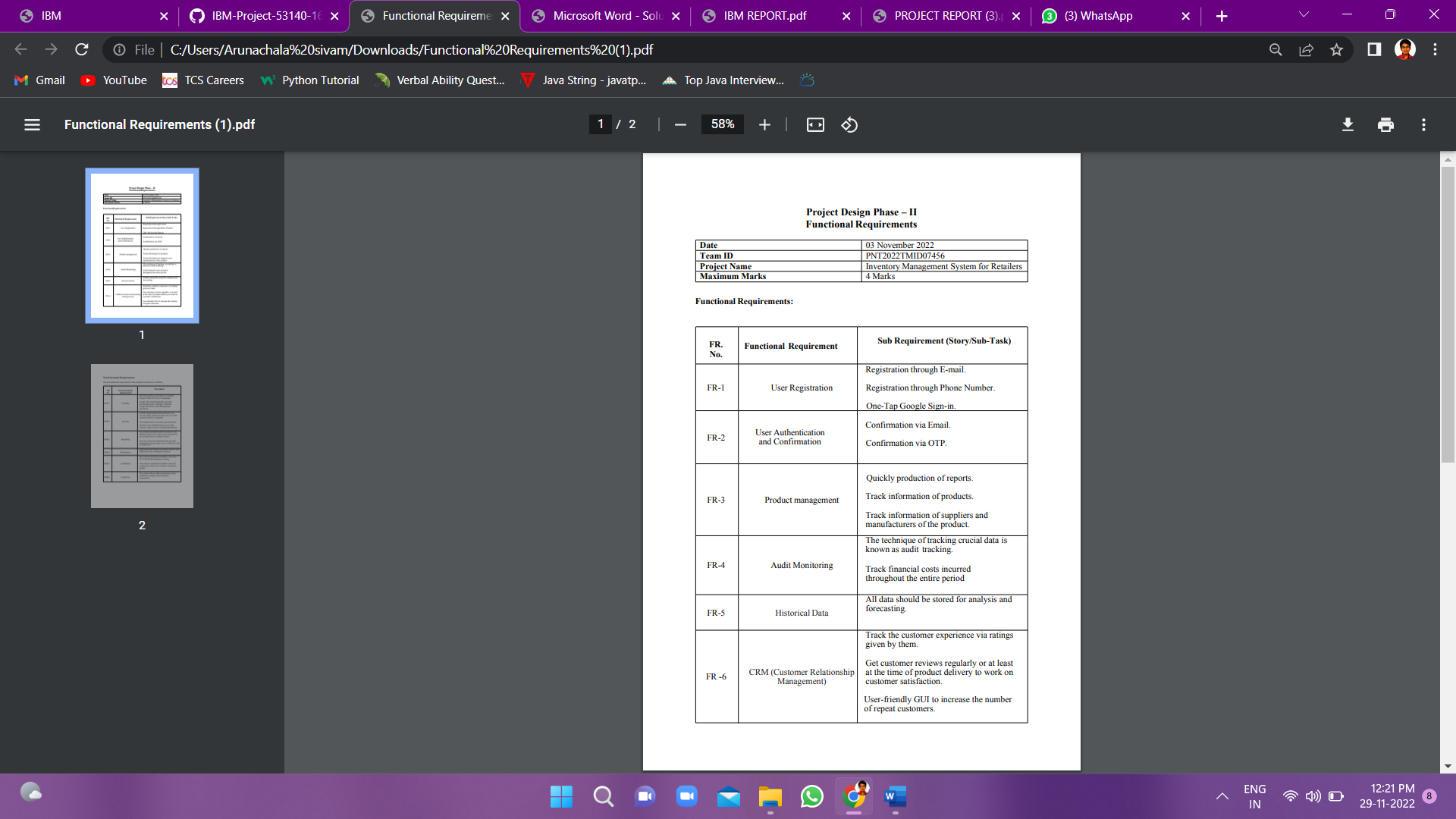


**3.3 Proposed Solution:**

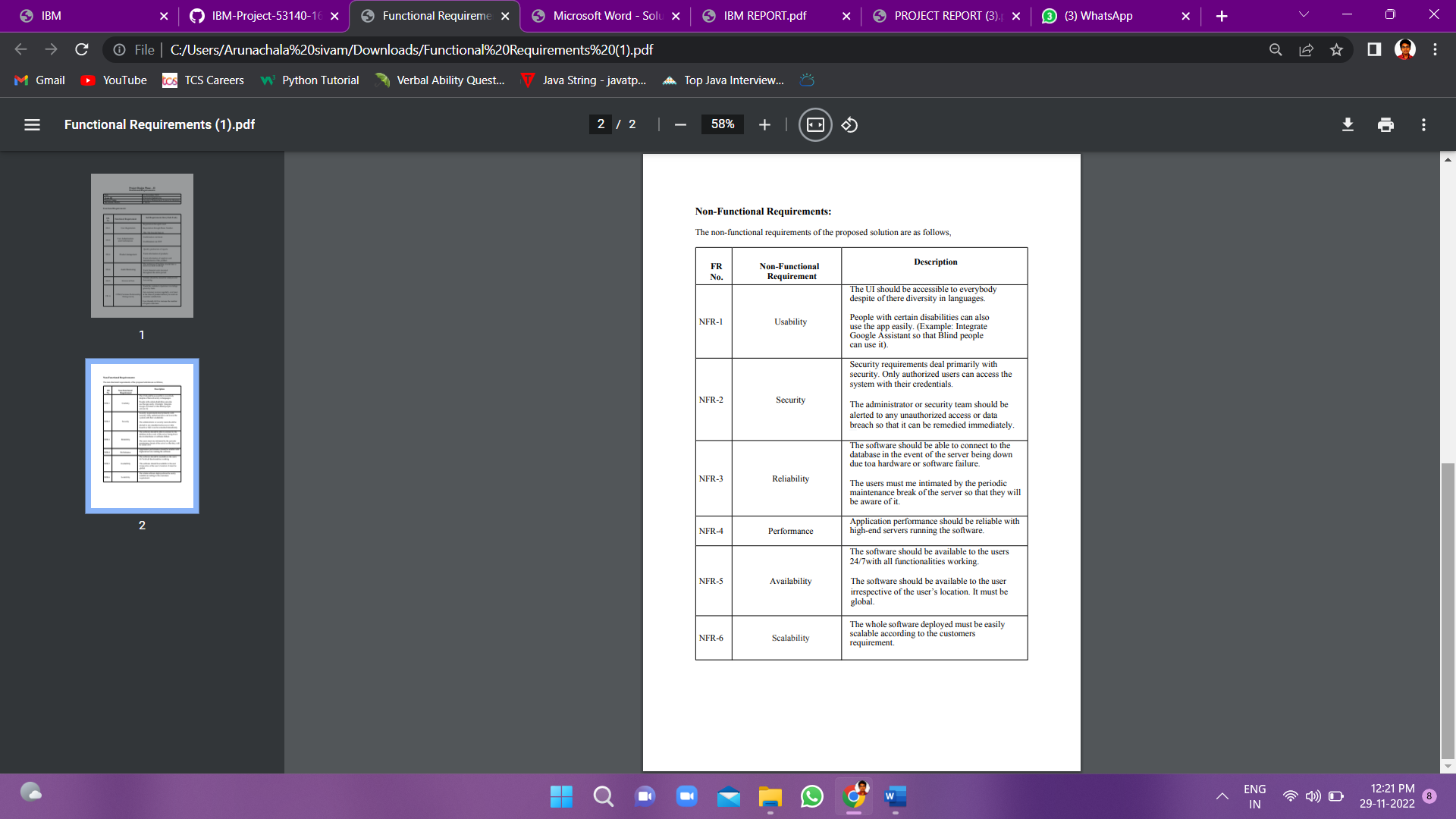
**3.4 PROBLEM SOLUTION FIT**

**4. REQUIREMENT ANALYSIS**

**4.1 FUNCTIONAL REQUIREMENT**

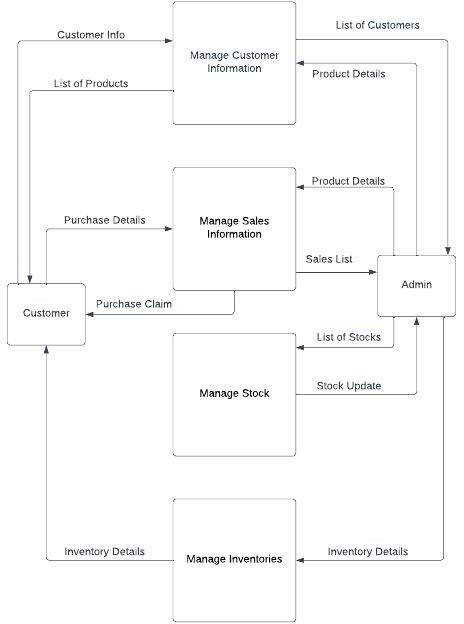
****

**4.2 NON-FUNCTIONAL REQUIREMENTS**



**5. PROJECT DESIGN**

**5.1 DATA FLOW DIAGRAMS**



**5.2 SOLUTION & TECHNICAL ARCHITECTURE**

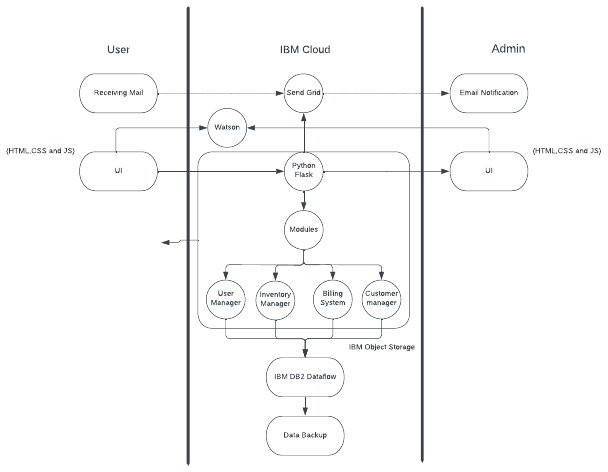


Table-1: Components & Technologies:

|  |  |  |  |
| --- | --- | --- | --- |
| **S.no** | **Component** | **Description** | **Technology** |
| 1. | User Interface | How user interacts with application e.g. Web UI, Mobile App, Chat bot etc. | HTML, CSS, JavaScript etc. |
| 2. | Application Logic | Logic for a process in the application | Python-Flask |
| 3. | Cloud Database | Database Service on Cloud | IBM DB2 |
| 4. | File Storage | File storage requirements | IBM Cloud Object Storage |
| 5. | App Container | Contain the whole application in a single container | Docker Container / IBM Container Registry |
| 6. | Infrastructure  (Server / Cloud) | Application Deployment on Local System / Cloud  Local Server Configuration: Cloud Server Configuration: | Local, Cloud Foundry, Kubernetes, etc. |
| 7. | Send Mails | Sending mails about stocks available in the Inventory to the Retailer | SendGrid |

Table-2: Application Characteristics:

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Characteristics** | **Description** | **Technology** |
| 1. | Open-Source Frameworks | List the open-source frameworks used | React Js / Flask |
| 2. | Security Implementations | List all the security / access controls implemented, use of firewalls etc. | IBM Cloud Security |
| 3. | Availability | Justify the availability of application (e.g., use of load balancers, distributed servers etc.) | Python-Flask |

**5.3 USER STORIES**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **User Type** | **Functional Requirement (Epic)** | | **User Story / Task** | | **Acceptance criteria** | | | **Release** |
| Customer (Mobile user) | Registration | | As a user, I can register for the application by entering my email,  password, and confirming my password. | | I can access my account / dashboard. | | | Sprint-1 |
|  |  | | As a user, I will receive confirmation email once I have registered for the application. | | I can receive confirmation email & click confirm. | | | Sprint-1 |
|  |  | | As a user, I can register for the application through gmail. | | I can register for the application through gmail. | | | Sprint-2 |
|  | Login | | As a user, I can log into the application by entering email & password. | | I can log in by entering gmail password. | | | Sprint-1 |
|  | Dashboard | As a user, I can track data of sales of products and inventory levels. | | I can track data of  sales of products and inventory levels. | |  | Sprint-2 | | |
| Customer (Web user) | Registration | As a user, I can register for the application by entering my email,  password, and confirming my password. | | I can access my account / dashboard. | |  | Sprint-1 | | |
|  |  | As a user, I will receive confirmation email once I have registered for the application. | | I can receive confirmation email. | |  | Sprint-1 | | |
|  |  | As a user, I can register for the application through gmail. | | I can register for the  application through gmail. | |  | Sprint-1 | | |
|  | Login | As a user, I can log into the application by entering email & password | | I can log in by  entering gmail & Password. | |  | Sprint-1 | | |
|  | Dashboard | As a user, I can track data of sales of products and inventory levels. | | I can track data of sales of products and inventory levels. | |  | Sprint-2 | | |
| Customer  Care Executive | Support | As an Executive, I provide answers to questions asked by users. | | I provide the answers for the queries asked by the  users. | |  | Sprint-3 | | |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **User Type** | **Functional Requirement (Epic)** |  | **User Story / Task** | **Acceptance criteria** |  | | **Release** |  |
| Administrator | Manage the Stocks |  | As an administrator, I manage the stocks by adding, shipping and storing the stocks in the storage units. | I manage stock by adding, shipping and storing stock in storage units. |  | | Sprint-1 |  |
|  | Control all the users |  | As an administrator, I can control all the users by performing basic operations. | I can control all the users by performing basic operations. |  | | Sprint-1 |  |
|  | Access the database |  | As an administrator, I can control and access the database. | I can control and access the database. |  | Sprint-1 | | | | Sprint-1 |

**6. PROJECT PLANNING & SCHEDULING**

**6.1 SPRINT PLANNING & ESTIMATION**

|  |  |  |
| --- | --- | --- |
| **TITLE** | **DESCRIPTION** | **DATE** |
| **Literature Survey & Information Gathering** | Literature survey on selected project and gathering information by referring the project’s related technical papers, research publications, etc. | 28 SEPTEMBER 2022 |
| **Prepare Empathy Map** | Prepare empathy map canvas to capture the user’s pains & gains and prepare the list of problem statements. | 03 OCTOBER 2022 |
| **Ideation** | To list by the organizing brainstorm sessions and prioritize the top three ideas based on the feasibility and importance. | 04 OCTOBER 2022 |
| **Proposed Solution** | To prepare the proposed solution documents, which includes the novelty, feasibility of ideas, business model, social impact, scalability of the solution, etc. | 07 OCTOBER 2022 |
| **Problem Solution Fit** | Preparing the problem solution fit document. | 10 OCTOBER 2022 |

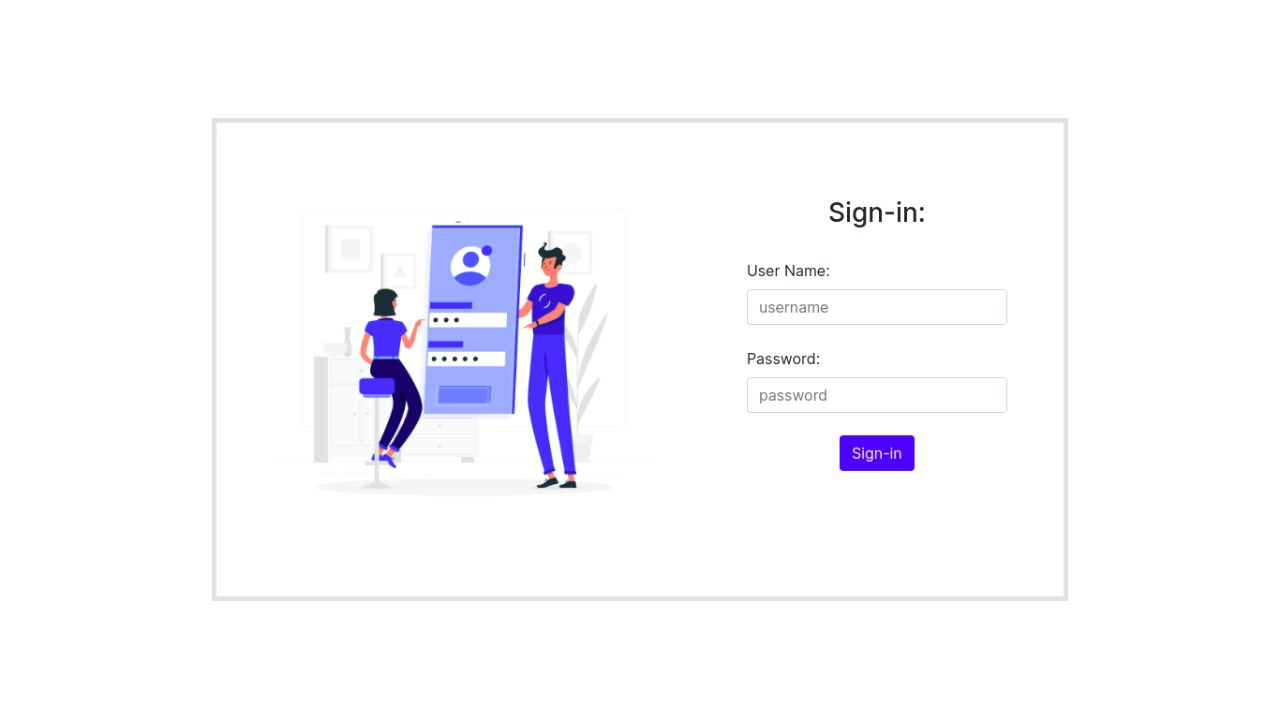
|  |  |  |
| --- | --- | --- |
| **Solution Architecture** | To prepare the solution architecture document | 15 OCTOBER 2022 |
| **Customer Journey** | Prepare the customers journey map help the customers understand the user interaction and experiences with the application from the  beginning to the end. | 20 OCTOBER 2022 |
| **Functional Requirement** | Prepare the functional requirement document. | 28 OCTOBER 2022 |
| **Data Flow Diagrams** | Draw the data flow diagrams and submit for the review. | 03 NOVEMBER 2022 |
| **Technology Architecture** | Prepare technical architecture diagram. | 10 NOVEMBER 2022 |
| **Prepare Milestone & Activity List** | Prepare the milestones and activity of the project. | 10 NOVEMBER 2022 |
| **Project Development –**  **Delivery of Sprint-1, 2, 3 &4** | Develop and submit the developed code by testing it and having no errors. | 20 NOVEMBER 2022 |

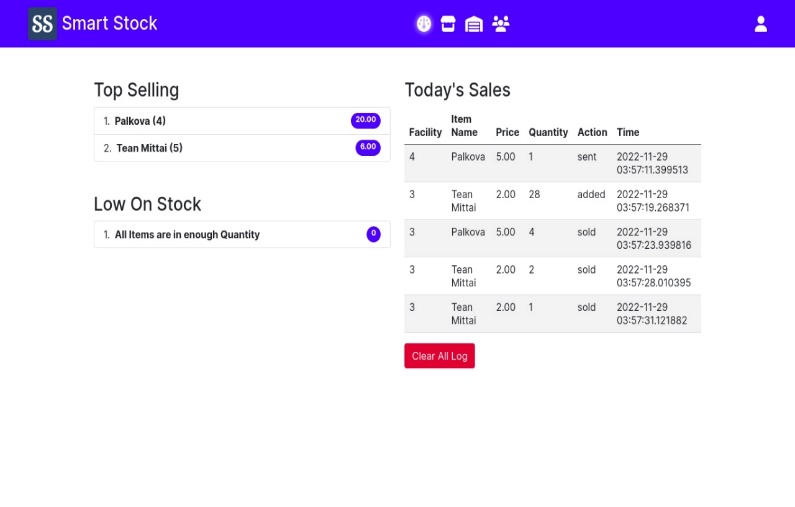
**6.2 SPRINT DELIVERY SCHEDULE**

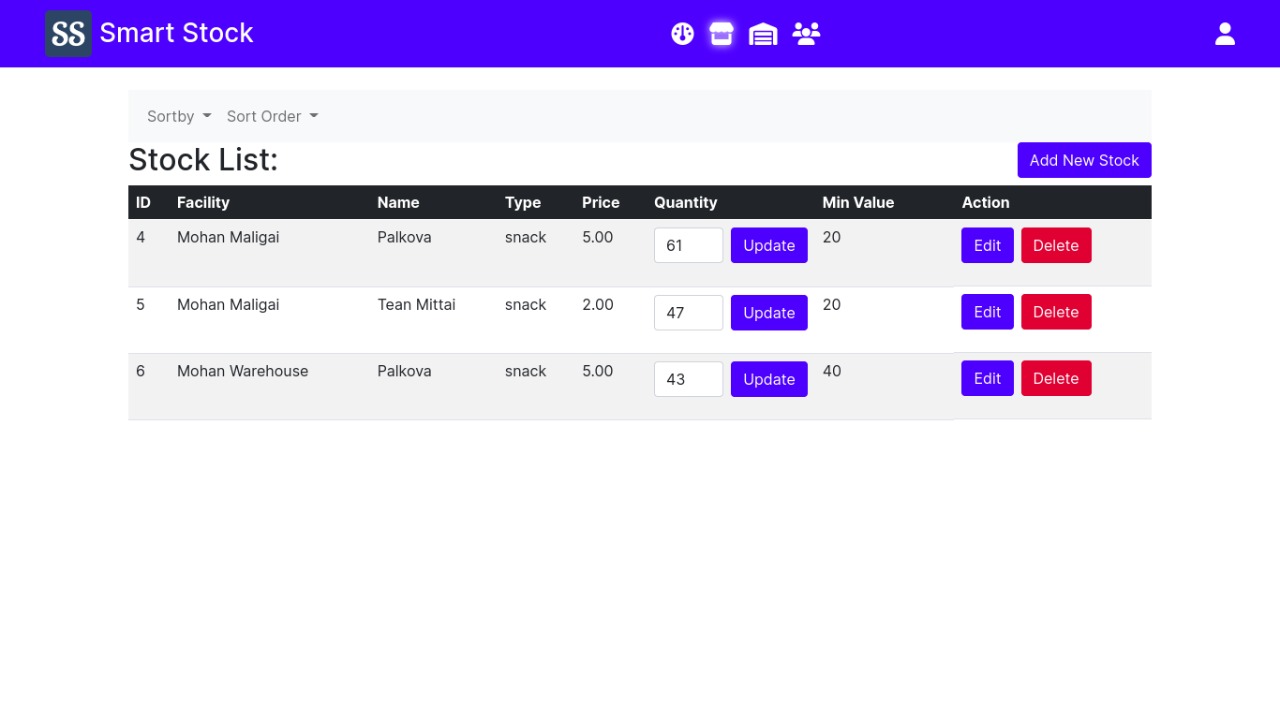
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sprint** | **Functional Requirement (Epic)** | **User Story Number** | **User Story / Task** | **Story Points** | **Priority** | **Team** **Members** |
| Sprint 1 | Registration | USN-1 | As a user, I can register for the application by entering my email, password, and confirming my password. | 5 | High | Karthiban S K  Arunachalam R  Ganappriyan C  Dhinesh M | |
| Sprint 1 |  | USN-2 | As a user, I will receive confirmation email once I have registered for the application | 4 | High | Karthiban S K  Arunachalam R  Ganappriyan C  Dhinesh M | |
| Sprint 1 |  | USN-3 | As a user, I can register for the application through Gmail | 3 | Medium | Karthiban S K  Arunachalam R  Ganappriyan C  Dhinesh M | |
| Sprint 1 | Login | USN-4 | As a user, I can log into the application by entering email & password | 4 | High | Karthiban S K  Arunachalam R  Ganappriyan C  Dhinesh M | |
| Sprint 1 | Dashboard | USN-5 | As a user, I can see the stock in hand and how much stock will be received and check other details. | 4 | High | Karthiban S K  Arunachalam R  Ganappriyan C  Dhinesh M | |
| Sprint 2 | Customer details | USN-6 | As a user, I can see the customer details like name, company, location, and so on. | 3 | Low | Karthiban S K  Arunachalam R  Ganappriyan C  Dhinesh M | |

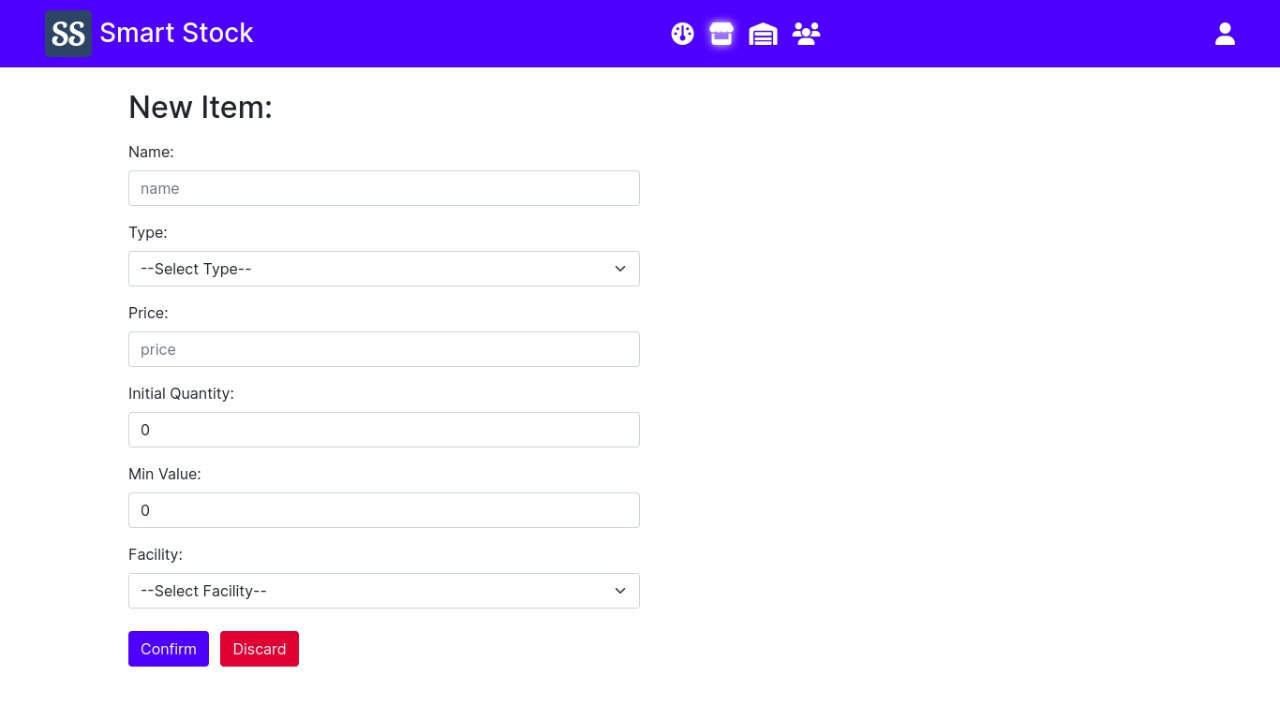
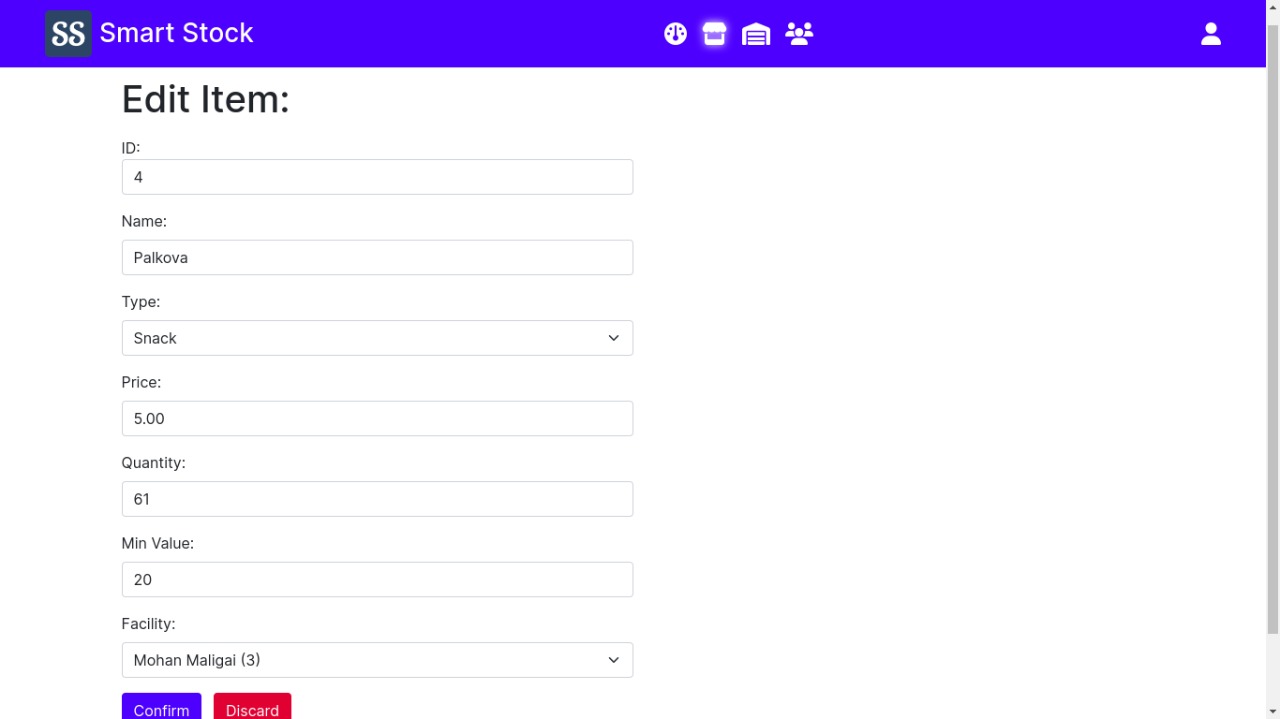
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Sprint 2 | Invoice management | USN-7 | As a user, I can see, manage, and update or modify the invoice of my shop | 1 | Low | Karthiban S K  Arunachalam R  Ganappriyan C  Dhinesh M |
| Sprint 2 | Sale and order manage ment | USN-8 | As a user, I can see, manage, and update the sale and order | 5 | Medium | Karthiban S K  Arunachalam R  Ganappriyan C  Dhinesh M |
| Sprint 2 | Return management | USN-9 | As a user, I can manage the returned items and check for damaged or defective items. | 5 | Medium | Karthiban S K  Arunachalam R  Ganappriyan C  Dhinesh M |
| Sprint 2 | Purchase order managem ent | USN-10 | As a user, I can enter the newly purchased stock and add or remove the stocks. And  upload the purchased details as well. | 5 | Medium | Karthiban S K  Arunachalam R  Ganappriyan C  Dhinesh M |
| Sprint 3 | Stocks | USN-11 | As a user, I can see the stock level, fast-moving, and death stocks. | 4 | High | Karthiban S K  Arunachalam R  Ganappriyan C  Dhinesh M |
| Sprint 3 | Report | USN-12 | As a user, I can see the report of the stock | 1 | Low | Karthiban S K  Arunachalam R  Ganappriyan C  Dhinesh M |
| Sprint 3 | Notification | USN-13 | As a user, it is good if I get a notification for low stock. | 2 | Medium | Karthiban S K  Arunachalam R  Ganappriyan C  Dhinesh M |
| Sprint 3 | Supplier | USN-14 | As a user, I can see the supplier details for a better understanding. | 3 | Low | Karthiban S K  Arunachalam R  Ganappriyan C  Dhinesh M |
| Sprint 3 | Profile | USN-15 | As a user, I can see my profile and give my details after registering as well. | 1 | Low | Karthiban S K  Arunachalam R  Ganappriyan C  Dhinesh M |
| Sprint 4 | Bill | USN-16 | As a user, I like to print the product that is sold now and maintain it. | 4 | Medium | Karthiban S K  Arunachalam R  Ganappriyan C  Dhinesh M |

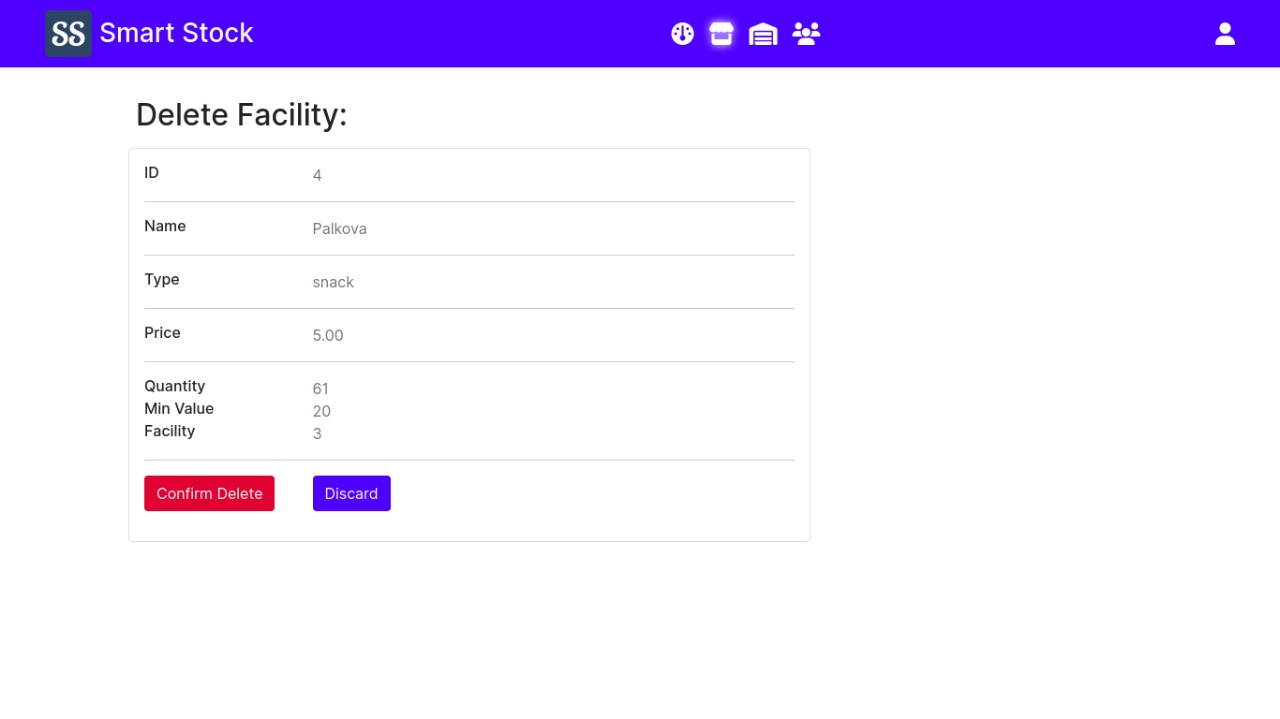
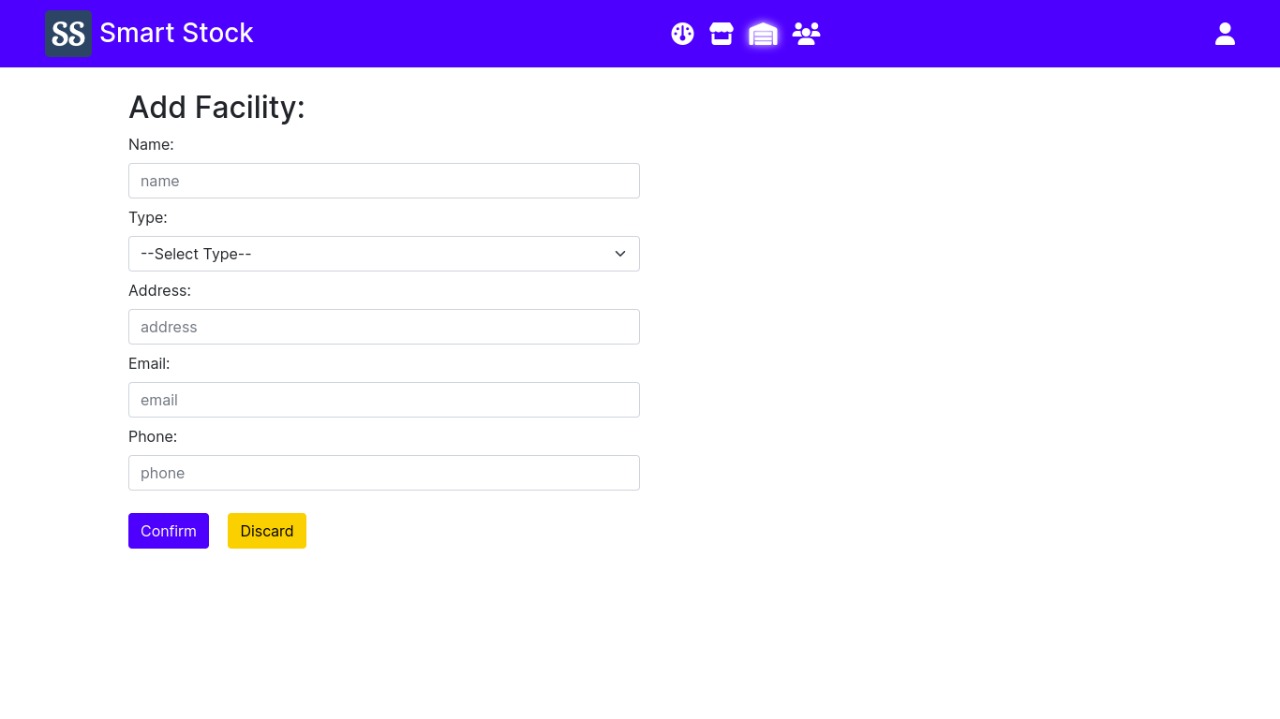
**7. RESULTS**

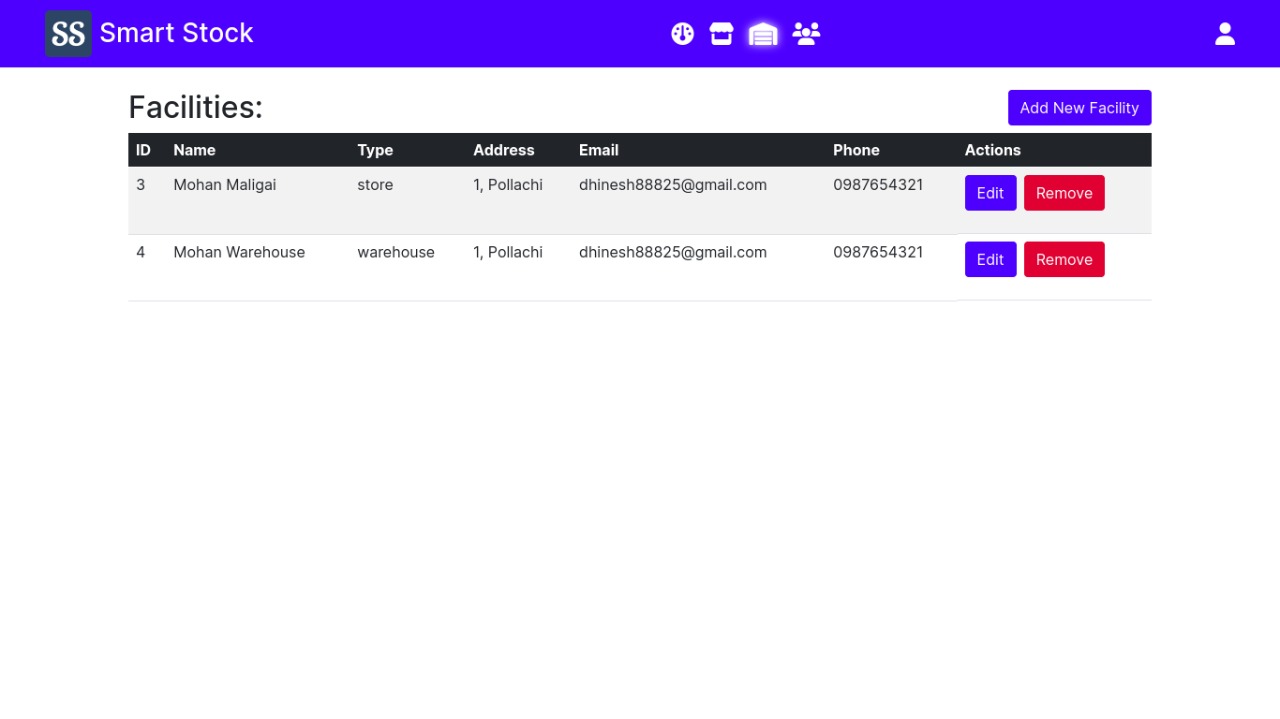
****

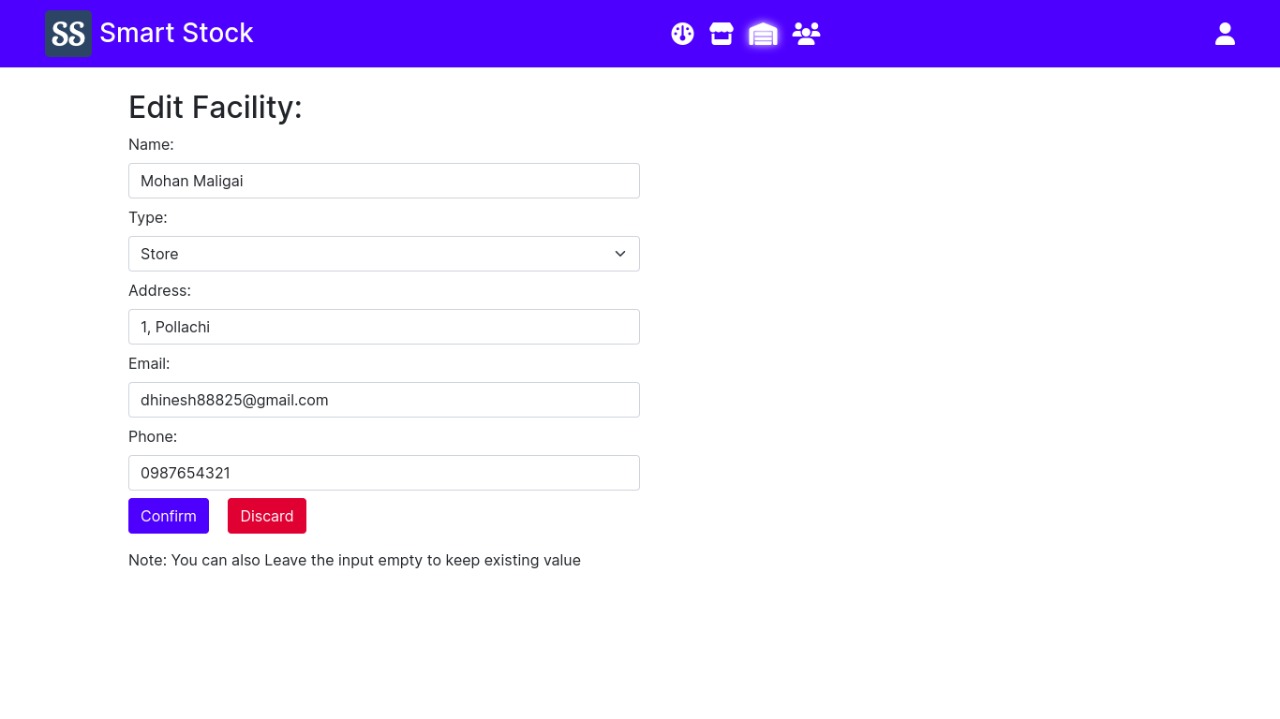
****

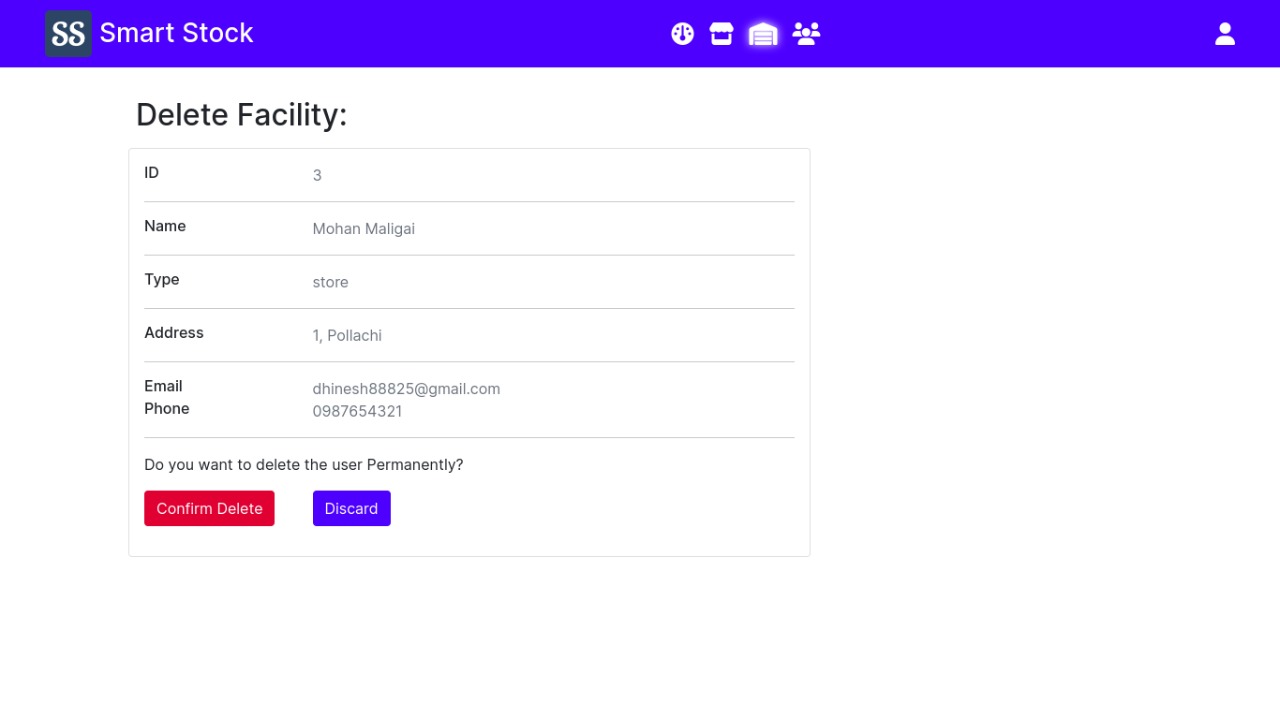
****

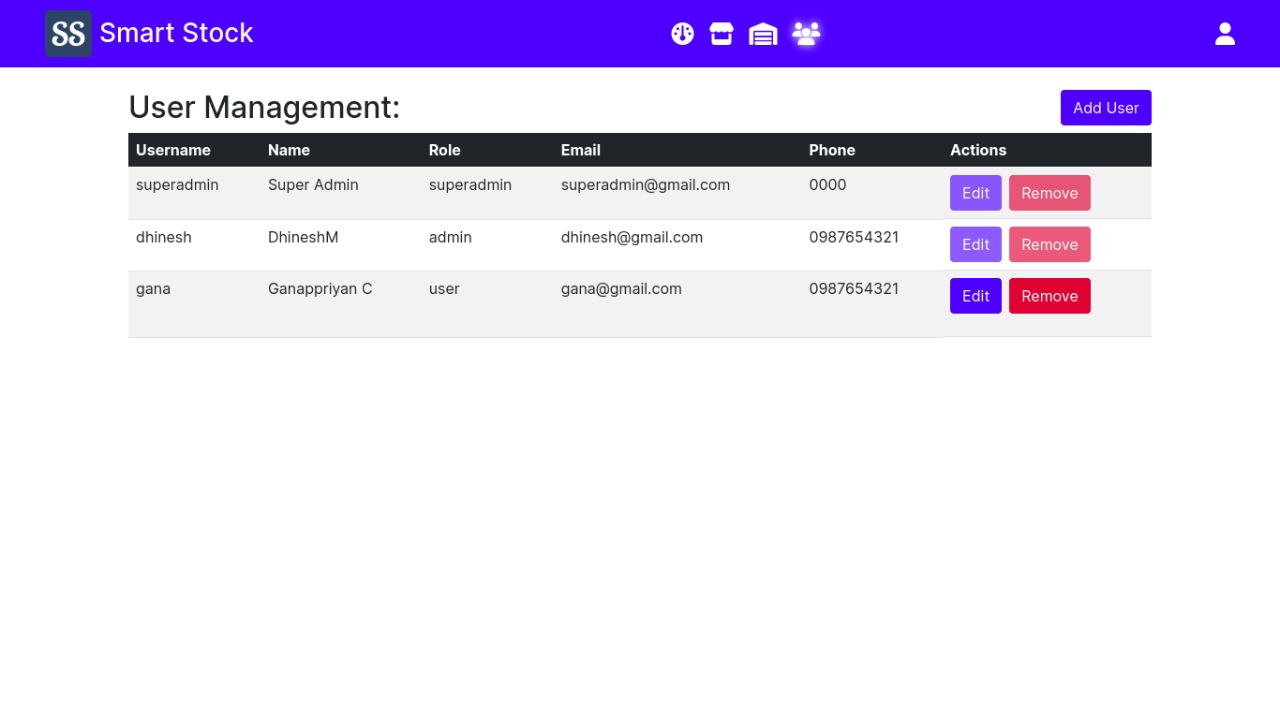
****

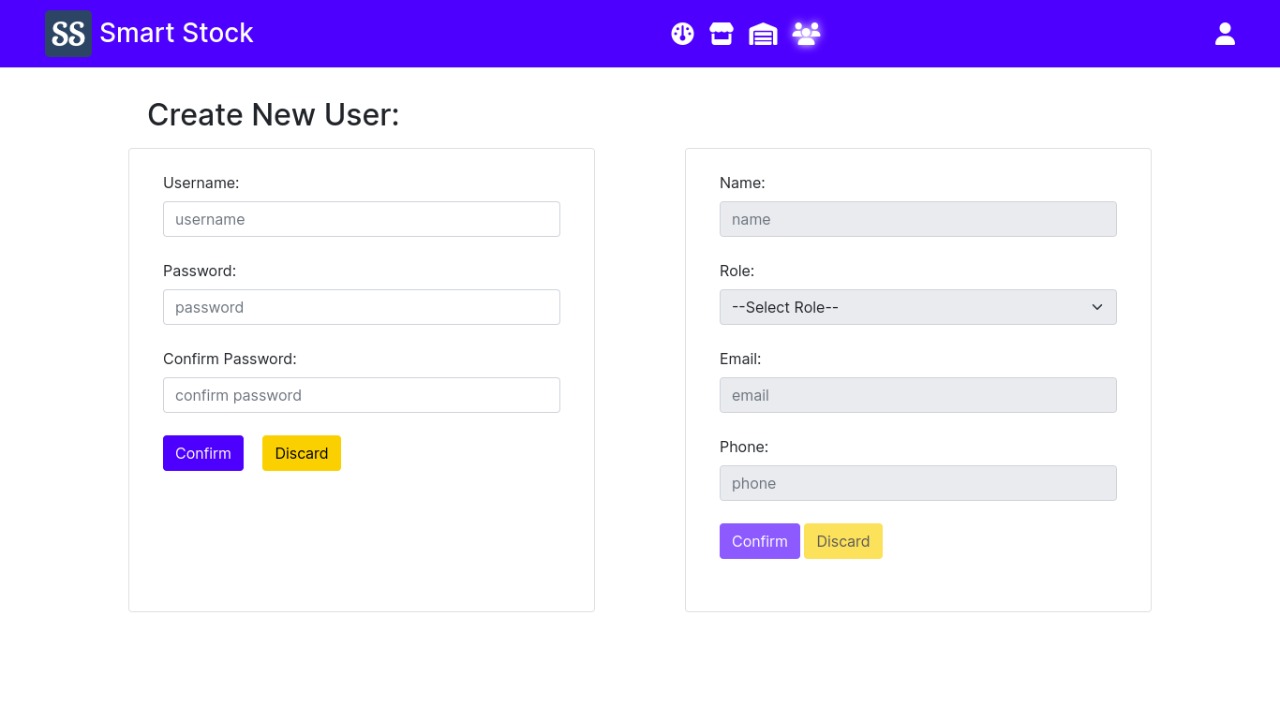
****

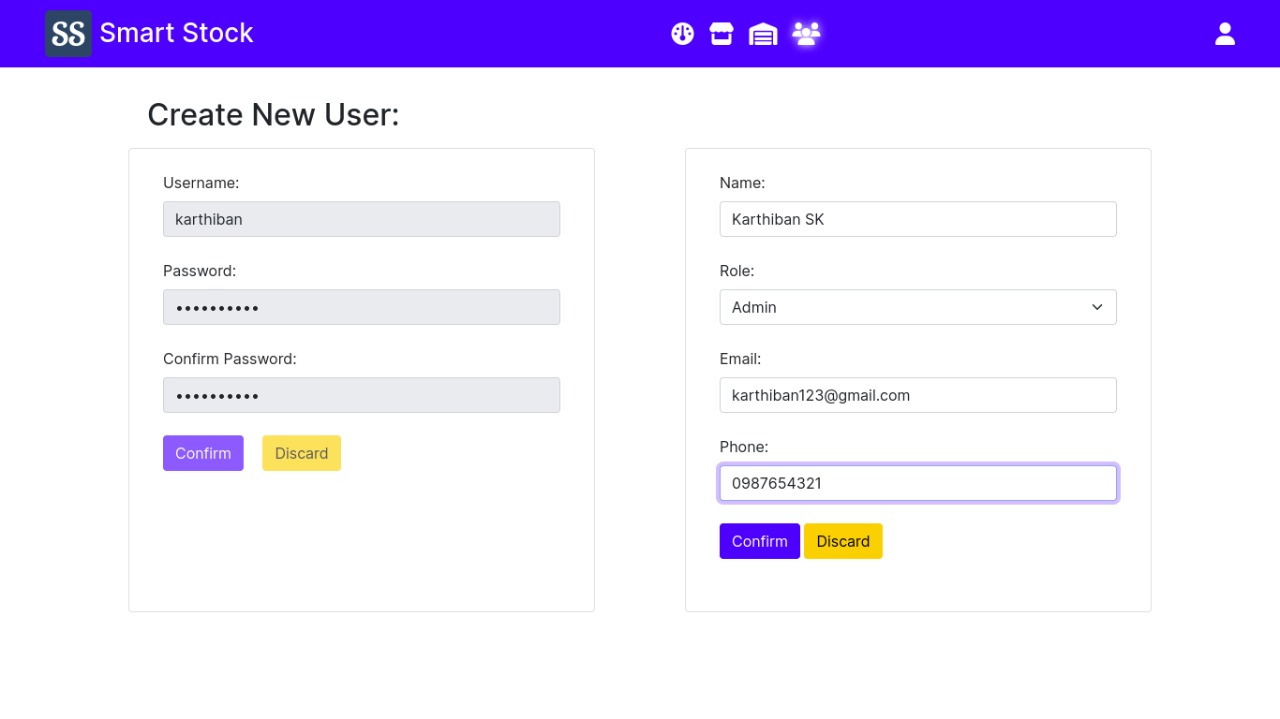
****

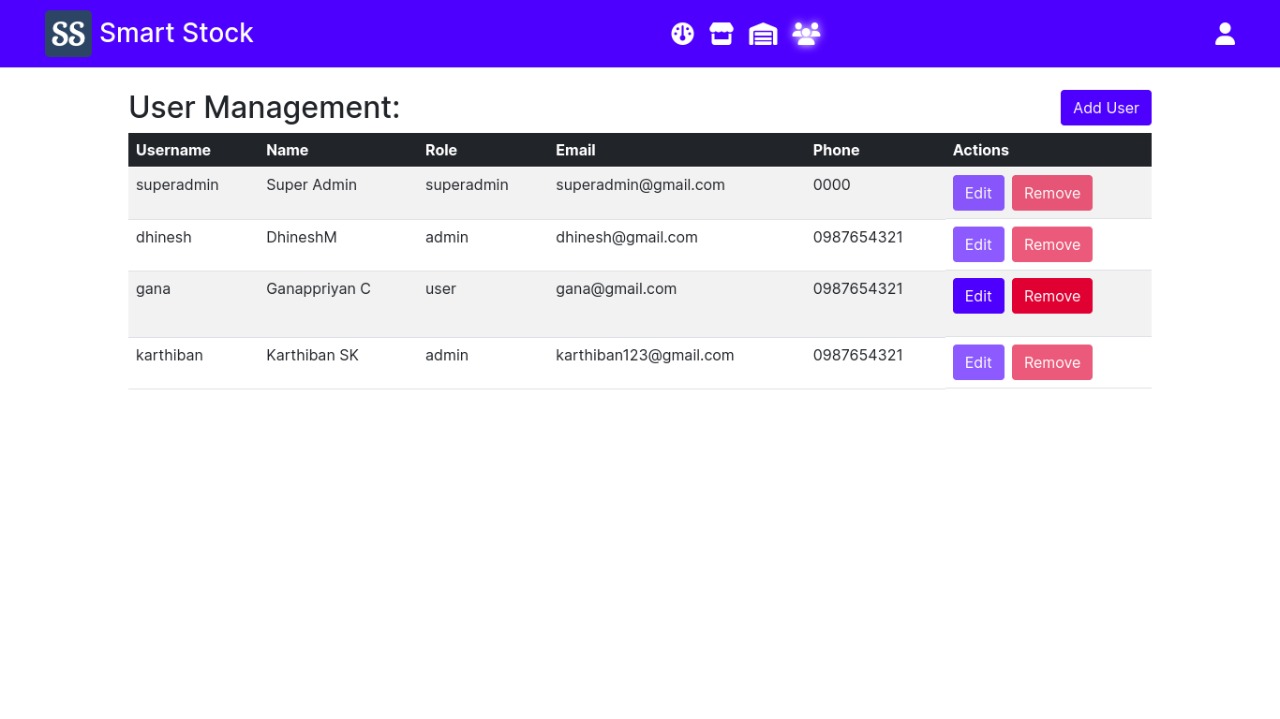
****

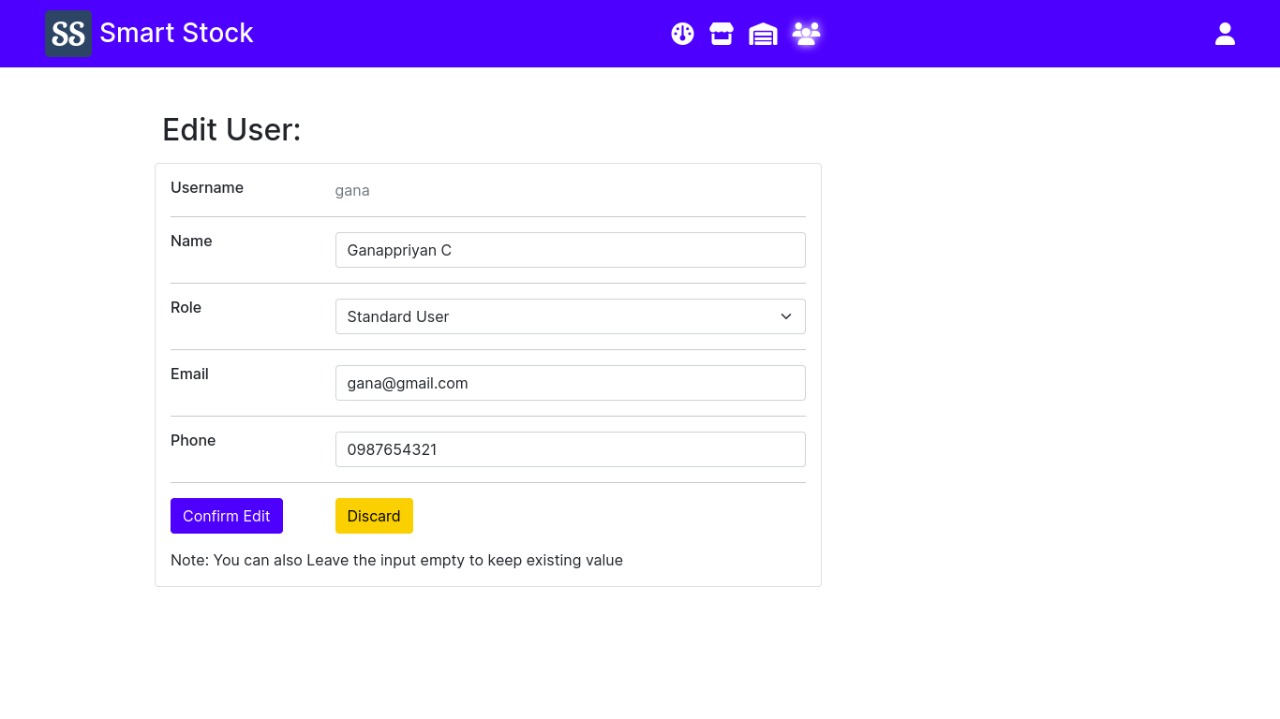
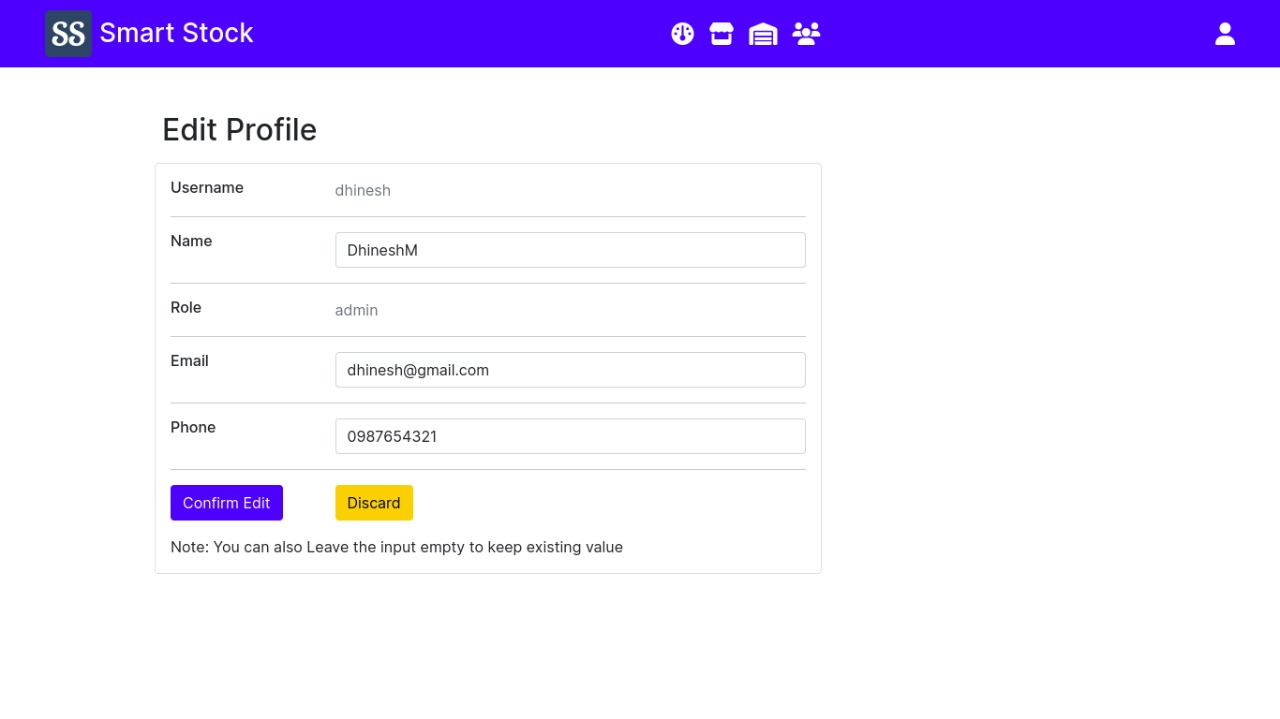
****

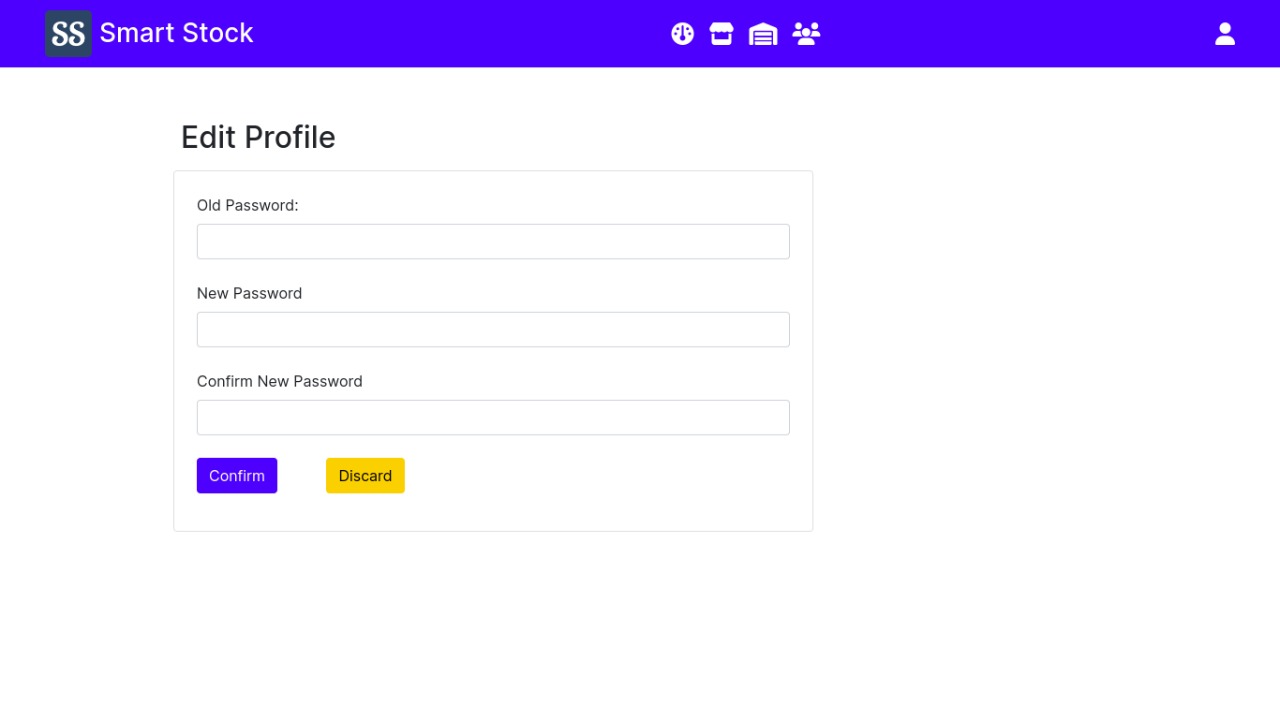
****

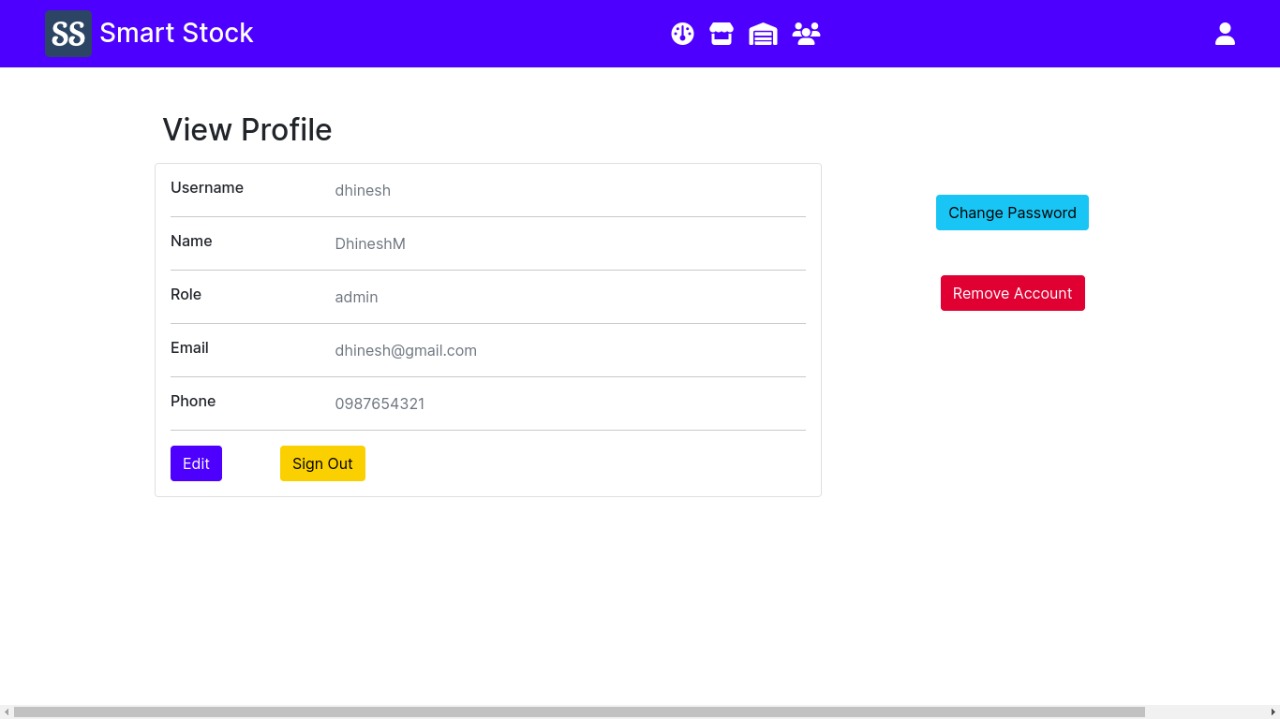
****

****

****

****

****

****

**8. ADVANTAGES & DISADVANTAGES**

**Advantages:**

The inventory management system for retailers is a web-based application. A dashboard is given to retailers where they can able to update, manage and create a product and assign the number of quantities for the products. Whenever a product goes out of stock the inventory management system for retailers alerts the retailers through email stating product goes out of stocks. Thus, inventory management system helps retailers to increase the profit and reduce the risk of holding large quantity of a particular stock. The inventory management system helps the retailers to efficiently utilizes the inventory area i.e., where the store all the products

**10.2 Disadvantages:**

The inventory management system for retailers helps them in many ways, but it needs a manual way of updating the quantity of the product. The retailers need to create and update the quantity of stock in the web-based application. The inventory management system necessitates manual updating of stock quantities, which adds to their workload.

**9. CONCLUSION**

A web-based application is created to manage inventory stocks. Retailers can able to update, create, and manage products in this web application. The inventory management system gives a mail alert, if a product goes out of stock, stating that a product has gone out of stock. This allows retailers to increase their profit, reduce the risk of having too much stock, and make better use of their inventory space.

**10. FUTURE SCOPE**

The future of inventory is going to be flooded with updates in technology. From virtual reality, to artificial intelligence, to digital signage, and even inventory-less stores, there are constant iterations made in this industry to accelerate business and attract customers. And it’s only growing from here. Inventory management systems have become more real-time, giving retailers more data about demographics, spending habits, shopping preferences, and more. With this constant increase in inventory visibility, retailers aim to better their accuracy with their inventory, and continue to appeal to their consumers. Take Nordstrom, for example. Nordstrom has recently introduced a new inventory strategy – only with no inventory. They’ve introduced a store concept called Nordstrom Local, a tiny space where instead of focusing on the inventory, they focus on the customers. Customers crave fast and convenient methods of shopping, and Nordstrom took that and ran with it. The store has alteration services, personal styling, nail salons, beverages, online order pick ups, just no inventory. By introducing this store concept, they are proving their loyalty to their customers and giving them what they really want.

**11. APPENDIX**

**SOURCE CODE**

**app.py:**

from flask import Flask, render\_template, request, session, redirect, url\_for

from modules.DBManager import DBManager

from modules.UserManagement import UserManagement

from modules.FacilityManagement import FacilityManagement

from modules.StockManagement import StockManagement

from modules.Constants import Constants

from modules.SendgridAPI import SendgridAPI

from modules.LogManagement import LogManagement

app = Flask(\_\_name\_\_)

DB = DBManager()

UM = UserManagement(DB)

FM = FacilityManagement(DB)

CON = Constants()

SG = SendgridAPI()

SM = StockManagement(DB, UM, FM, SG)

LM = LogManagement(DB, SM)

# Session Variables:

# page - The page the user is currently on

# message - The message to display to the user

# username - The username of the user

# sort - The column to sort the table by

# filter - The filter to apply to the table

# Root URL

@app.route('/')

def index():

if 'username' in session:

return redirect(url\_for('dashboard'))

session['page'] = 'dashboard'

return render\_template('home.html')

# Dashboard - Page

@app.route('/dashboard')

def dashboard():

session['page'] = 'dashboard'

return render\_template('dashboard.html')

@app.route('/dashboard/clearlog')

def dashboard\_clearlog():

try:

LM.clear\_all\_logs()

session['message'] = 'All logs have been cleared'

except Exception as e:

session['message'] = 'Error: ' + str(e)

return redirect(url\_for('dashboard'))

# Stock - Page

# stock - id, name, type, price, quantity, facility

@app.route('/stock')

def stock():

session['page'] = 'stock'

if 'sort' not in session:

session['sort'] = 'id'

if 'sortorder' not in session:

session['sortorder'] = 'asc'

return render\_template('stock.html', mode='view')

@app.route('/stock/<a>', methods=['GET', 'POST'])

def stock\_mode(a):

if a == 'view':

return render\_template('stock.html', mode='view')

elif a == 'add':

return render\_template('stock.html', mode='add')

elif a == 'update':

id = request.form['id']

quantity = request.form['quantity']

try:

SM.edit\_quantity(id, quantity)

return render\_template('stock.html', mode='view')

except Exception as e:

session['message'] = 'Error: ' + str(e)

return render\_template('stock.html', mode='view')

elif a == 'edit':

id = request.form['id']

return render\_template('stock.html', mode='edit', id=id)

elif a == 'delete':

id = request.form['id']

return render\_template('stock.html', mode='delete', id=id)

else:

return render\_template('404.html')

@app.route('/stock/<a>/<b>', methods=['GET', 'POST'])

def stock\_action(a, b):

if a == 'view':

if b in CON.SORTBY:

session['sort'] = b

elif b in ('asc', 'desc'):

session['sortorder'] = b

else:

return render\_template('404.html')

return redirect(url\_for('stock'))

if b == 'discard':

return redirect(url\_for('stock'))

elif b == 'save':

if a == 'add':

try:

name = request.form['name']

type = request.form['type']

price = request.form['price']

quantity = request.form['quantity']

minvalue = request.form['minvalue']

facility = request.form['facility']

SM.add\_item(name, type, price, quantity, minvalue, facility)

return redirect(url\_for('stock'))

except Exception as e:

session['message'] = 'Error: ' + str(e)

return render\_template('stock.html', mode='add')

elif a == 'edit':

try:

id = request.form['id']

name = request.form['name']

type = request.form['type']

price = request.form['price']

quantity = request.form['quantity']

minvalue = request.form['minvalue']

facility = request.form['facility']

SM.edit\_item(id, name, type, price,

quantity, minvalue, facility)

return redirect(url\_for('stock'))

except Exception as e:

session['message'] = 'Error: ' + str(e)

return render\_template('stock.html', mode='edit', id=id)

elif a == 'delete':

try:

id = request.form['id']

SM.remove\_item(id)

return redirect(url\_for('stock'))

except Exception as e:

session['message'] = 'Error: ' + str(e)

return render\_template('stock.html', mode='delete', id=id)

else:

return render\_template('404.html')

else:

return render\_template('404.html')

# Facilities - Page

# facility - id(auto), name(50), type(20), address(255), email(320), phone(15)

@app.route('/facilities')

def facilities():

session['page'] = 'facilities'

return render\_template('facilities.html', mode='view')

# Modes: view, add, edit, delete

@app.route('/facilities/<a>', methods=['GET', 'POST'])

def facilities\_mode(a):

if a == 'add':

return render\_template('facilities.html', mode='add')

elif a == 'edit':

id = request.form['id']

return render\_template('facilities.html', mode='edit', id=id)

elif a == 'delete':

id = request.form['id']

return render\_template('facilities.html', mode='delete', id=id)

else:

return render\_template('404.html')

@app.route('/facilities/<a>/<b>', methods=['GET', 'POST'])

def facilities\_action(a, b):

if b == 'discard':

return redirect(url\_for('facilities'))

elif b == 'save':

if a == 'add':

try:

name = request.form['name']

type = request.form['type']

address = request.form['address']

email = request.form['email']

phone = request.form['phone']

FM.add\_facility(name, type, address, email, phone)

return redirect(url\_for('facilities'))

except Exception as e:

session['message'] = 'Error: ' + str(e)

return render\_template('facilities.html', mode='add')

elif a == 'edit':

try:

id = request.form['id']

name = request.form['name']

type = request.form['type']

address = request.form['address']

email = request.form['email']

phone = request.form['phone']

FM.edit\_facility(id, name, type, address, email, phone)

return redirect(url\_for('facilities'))

except Exception as e:

session['message'] = 'Error: ' + str(e)

return render\_template('facilities.html', mode='edit', id=id)

elif a == 'delete':

try:

id = request.form['id']

FM.remove\_facility(id)

return redirect(url\_for('facilities'))

except Exception as e:

session['message'] = 'Error: ' + str(e)

return render\_template('facilities.html', mode='delete', id=id)

# User Management - Page

# Userdata - username(50), password(100), name(50), role(20), email(320), phone(15)

@app.route('/usermanagement')

def usermanagement():

session['page'] = 'usermanagement'

return render\_template('usermanagement.html', mode='view')

# Modes - view, add, add2, edit, delete

@app.route('/usermanagement/<a>', methods=['GET', 'POST'])

def usermanagement\_mode(a):

if a == 'add':

return render\_template('usermanagement.html', mode='add', username='', password='')

elif a == 'edit':

username = request.form['username']

return render\_template('usermanagement.html', mode='edit', username=username)

elif a == 'delete':

username = request.form['username']

return render\_template('usermanagement.html', mode='delete', username=username)

else:

return render\_template('404.html')

@app.route('/usermanagement/<a>/<b>', methods=['GET', 'POST'])

def usermanagement\_action(a, b):

if b == 'discard':

if a == 'add2':

username = request.form['username']

password = request.form['password']

return render\_template('usermanagement.html', mode='add', username=username, password=password)

return redirect(url\_for('usermanagement'))

elif b == 'save':

if a == 'add':

username = request.form['username']

password = request.form['password']

return render\_template('usermanagement.html', mode='add2', username=username, password=password)

elif a == 'add2':

try:

username = request.form['username']

password = request.form['password']

name = request.form['name']

role = request.form['role']

email = request.form['email']

phone = request.form['phone']

UM.add\_user(username, password, name, role, email, phone)

return redirect(url\_for('usermanagement'))

except Exception as e:

session['message'] = 'Error: ' + str(e)

return render\_template('usermanagement.html', mode='add2', username=username, password=password)

elif a == 'edit':

try:

username = request.form['username']

name = request.form['name']

role = request.form['role']

email = request.form['email']

phone = request.form['phone']

UM.edit\_user(username, name, role, email, phone)

return redirect(url\_for('usermanagement'))

except Exception as e:

session['message'] = 'Error: ' + str(e)

return render\_template('usermanagement.html', mode='edit', username=username)

elif a == 'delete':

try:

username = request.form['username']

UM.remove\_user(username)

return redirect(url\_for('usermanagement'))

except Exception as e:

session['message'] = 'Error: ' + str(e)

return render\_template('usermanagement.html', mode='delete', username=username)

else:

return render\_template('404.html')

# Profile - Page

@app.route('/profile')

def profile():

session['page'] = 'profile'

return render\_template('profile.html', mode='view')

# Modes: view, edit, changepassword

@app.route('/profile/<a>', methods=['POST', 'GET'])

def profile\_mode(a):

if a == 'edit':

return render\_template('profile.html', mode='edit')

elif a == 'changepassword':

return render\_template('profile.html', mode='changepassword')

elif a == 'remove':

try:

password = request.form['password']

if not UM.check\_user(session['username'], password):

session['message'] = 'Password Incorrect'

return redirect(url\_for('profile'))

UM.remove\_user(session['username'])

session.pop('username', None)

session['message'] = 'User Removed Successfully'

return redirect(url\_for('signin'))

except Exception as e:

session['message'] = str(e)

return redirect(url\_for('profile'))

elif a == 'signout':

try:

session.pop('username', None)

session['message'] = 'Signed Out Successfully'

except Exception as e:

session['message'] = str(e)

return redirect(url\_for('signin'))

else:

return render\_template('404.html')

@app.route('/profile/<a>/<b>', methods=['POST', 'GET'])

def profile\_action(a, b):

if b == 'discard':

return redirect(url\_for('profile'))

elif b == 'save':

if a == 'edit':

try:

name, email, phone = request.form['name'], request.form['email'], request.form['phone']

UM.edit\_user(

username=session['username'], newname=name, newemail=email, newphone=phone)

return redirect(url\_for('profile'))

except Exception as e:

session['message'] = str(e)

return render\_template('profile.html', mode='edit')

elif a == 'changepassword':

try:

password, newpassword = request.form['password'], request.form['newpassword']

if not UM.check\_user(session['username'], password):

session['message'] = 'Password Incorrect'

return render\_template('profile.html', mode='changepassword')

UM.change\_password(session['username'], newpassword)

session['message'] = 'Password Changed Successfully'

except Exception as e:

session['message'] = str(e)

return redirect(url\_for('profile'))

else:

return render\_template('404.html')

else:

return render\_template('404.html')

# Sign In - Page

@app.route('/signin')

def signin():

if 'user' in session:

session['message'] = 'You are already Signed in'

return redirect(url\_for(session['page']))

return render\_template('signin.html')

@app.route('/signin/handle\_form', methods=['POST', 'GET'])

def signin\_action():

try:

username, password = request.form['username'], request.form['password']

if UM.check\_user(username, password):

session['username'] = username

return redirect(url\_for(session['page']))

else:

session['message'] = 'Invalid Username or Password'

return redirect('/')

except Exception as e:

session['message'] = str(e)

return redirect('/signin')

@app.errorhandler(404)

def page\_not\_found(e):

return render\_template('404.html'), 404

@app.context\_processor

def inject\_data():

return dict(UM=UM, FM=FM, SM=SM, CON=CON, LM=LM)

if \_\_name\_\_ == '\_\_main\_\_':

app.config['SECRET\_KEY'] = '123456789'

app.run(debug=True, port=5050)

**home.html:**

<!-- Home Page -->

<html>

<head>

{% include "head.html" %}

</head>

<body>

{% if 'username' in session %}

<div class="container d-flex flex-column justify-content-center">

<h1 class="text-center">You are already Signed in</h1>

<div class="text-center">

<a class="btn btn-primary" href="/dashboard">Dashboard</a>

</div>

</div>

{% else %}

<main class="container d-flex justify-content-center align-items-center" style="width: 100vw; height: 100vh;">

<div class="border border-5">

<div class="container d-flex" style="width: fit-content;">

<img src="{{ url\_for('static', filename='img/login.jpg') }}" class="m-5" style="width: 30vw"

alt="Login Illustration">

<form class="m-5" action="/signin/handle\_form" method="POST">

<div class="form-outline m-4">

<div class="col p-2 h3 text-center">Sign-in:</div>

</div>

<!-- Username Input -->

<div class="form-outline mb-4">

<label class="form-label" for="username">User Name:</label>

<input class="form-control" type="text" name="username" id="username" placeholder="username" minlength="3"

required>

</div>

<!-- Passoword Input -->

<div class="form-outline mb-4">

<label class="form-label" for="password">Password:</label>

<input class="form-control" type="password" name="password" id="password" placeholder="password" minlength="4"

required>

</div>

<!-- Sign-in Button -->

<div class="form-outline mb-4 text-center">

<button class="btn btn-primary btn-block mb-4" type="submit">Sign-in</button>

</div>

</form>

</div>

</div>

</main>

{% endif %}

{% include "scripts.html" %}

</body>

</html>

**profile.html:**

<!-- Profile Page -->

<html>

<head>{% include "head.html" %}</head>

<body>

{% include "header.html" %}

<main class="container p-4">

{% if 'username' not in session %}

<div class="container d-flex flex-column justify-content-center">

<h1 class="text-center">You need to sign in to view this page</h1>

<div class="text-center">

<a class="btn btn-primary" href="/signin">Signin</a>

</div>

</div>

{% else %}

{% set user = UM.get\_user(session['username']) %}

{% if mode=='view' %}

<div class="container d-flex m-3">

<div class="col-md-8">

<div class="col p-2 h2">View Profile</div>

<div class="card mb-3">

<div class="card-body">

<div class="row">

<div class="col-sm-3">

<h6 class="mb-0">Username</h6>

</div>

<div class="col-sm-9 text-secondary">

{{user.USERNAME}}

</div>

</div>

<hr>

<div class="row">

<div class="col-sm-3">

<h6 class="mb-0">Name</h6>

</div>

<div class="col-sm-9 text-secondary">

{{user.name}}

</div>

</div>

<hr>

<div class="row">

<div class="col-sm-3">

<h6 class="mb-0">Role</h6>

</div>

<div class="col-sm-9 text-secondary">

{{user.role}}

</div>

</div>

<hr>

<div class="row">

<div class="col-sm-3">

<h6 class="mb-0">Email</h6>

</div>

<div class="col-sm-9 text-secondary">

{{user.email}}

</div>

</div>

<hr>

<div class="row">

<div class="col-sm-3">

<h6 class="mb-0">Phone</h6>

</div>

<div class="col-sm-9 text-secondary">

{{user.phone}}

</div>

</div>

<hr>

<div class="row">

<div class="col-sm-2">

<a class="btn btn-primary" href="/profile/edit">Edit</a>

</div>

<div class="col-sm-2">

<a class="btn btn-warning" href="/profile/signout">Sign Out</a>

</div>

</div>

</div>

</div>

</div>

<div class="col-sm-7">

<div class="row mt-5 mb-5"></div>

<div class="row mb-5">

<div class="col-sm-8 text-center">

<a class="btn btn-info" href="/profile/changepassword">Change Password</a>

</div>

</div>

<div class="row text-center">

{% if user.role != 'superadmin' %}

<form class="col-sm-8" action="/profile/remove" method="post" id="removeuser">

<button class="btn btn-danger btn-block mb-4" onclick="removeacc()" type="button"

id="removepasswordbtn">Remove Account</button>

<button class="btn btn-danger btn-block mb-4" type="submit" style="display: none;"

id="removepasswordbtnsub">Confirm Remove Account</button>

<input class="form-control ms-5 me-5" type="password" name="password" id="removepassword"

placeholder="Password" minlength="4" style="display: none;" required>

</form>

<script>

function removeacc() {

if (document.getElementById('removepassword').style.display == 'none') {

document.getElementById('removepassword').style.display = 'inline';

document.getElementById('removepasswordbtn').style.display = 'none';

document.getElementById('removepasswordbtnsub').style.display = 'inline';

}

}

</script>

{% else %}

<div class="col-sm-8">

<button class="btn btn-danger btn-block mb-4" disabled>Remove Account</button>

</div>

{% endif %}

</div>

</div>

</div>

{% elif mode=='edit' %}

<form class="container d-flex m-3" action="/profile/edit/save" method="post">

<div class="col-md-8">

<div class="col p-2 h2">Edit Profile</div>

<div class="card mb-3">

<div class="card-body">

<div class="row">

<div class="col-sm-3">

<h6 class="mb-0">Username</h6>

</div>

<div class="col-sm-9 text-secondary">

{{user.USERNAME}}

</div>

</div>

<hr>

<div class="row">

<div class="col-sm-3">

<h6 class="mb-0">Name</h6>

</div>

<div class="col-sm-9 text-secondary">

<input class="form-control me-5" type="text" name="name" placeholder="Name" value="{{ user.name }}">

</div>

</div>

<hr>

<div class="row">

<div class="col-sm-3">

<h6 class="mb-0">Role</h6>

</div>

<div class="col-sm-9 text-secondary">

{{user.role}}

</div>

</div>

<hr>

<div class="row">

<div class="col-sm-3">

<h6 class="mb-0">Email</h6>

</div>

<div class="col-sm-9 text-secondary">

<input class="form-control me-5" type="email" name="email" placeholder="Email" value="{{ user.email }}">

</div>

</div>

<hr>

<div class="row">

<div class="col-sm-3">

<h6 class="mb-0">Phone</h6>

</div>

<div class="col-sm-9 text-secondary">

<input class="form-control me-5" type="tel" name="phone" placeholder="Phone" value="{{ user.phone }}">

</div>

</div>

<hr>

<div class="row">

<div class="col-sm-3">

<button class="btn btn-primary" type="submit">Confirm Edit</button>

</div>

<div class="col-sm-2">

<a class="btn btn-warning" href="/profile/edit/discard">Discard</a>

</div>

</div>

<div class="mt-3">Note: You can also Leave the input empty to keep existing value</div>

</div>

</div>

</div>

</form>

{% elif mode=='changepassword' %}

<div class="m-4 ms-5">

<div class="p-2 h2">Edit Profile</div>

<div class="card" style="width: 50vw;">

<form action="/profile/changepassword/save" method="post" id="changepassword">

<div class="form-outline m-4">

<label class="form-label" for="password">Old Password: </label>

<input class="form-control" type="password" id="password" name="password" minlength="4" required>

</div>

<div class="form-outline m-4">

<label class="form-label" for="newpassword">New Password</label>

<input class="form-control" type="password" id="newpassword" name="newpassword" minlength="4"

onchange="onChangePassword()" required>

</div>

<div class="form-outline m-4">

<label class="form-label" for="confirmnewpassword">Confirm New Password</label>

<input class="form-control" type="password" id="confirmnewpassword" name="confirmnewpassword"

minlength="4" onchange="onChangePassword()" required>

</div>

<div class="form-outline m-4">

<button class="btn btn-primary btn-block me-5" type="submit">Confirm</button>

<a class="btn btn-warning btn-block" href="/profile/changepassword/discard">Discard</a>

</div>

</form>

</div>

</div>

<script>

function onChangePassword() {

const password = document.querySelector('input[name=newpassword]');

const confirm = document.querySelector('input[name=confirmnewpassword]');

if (confirm.value === password.value)

confirm.setCustomValidity('');

else

confirm.setCustomValidity('Passwords do not match');

}

</script>

{% endif %}

{% endif %}

</main>

{% include "scripts.html" %}

</body>

</html>

**signin.html**

<!-- Signin Page -->

<html>

<head>{% include "head.html" %}</head>

<body>

{% include "header.html" %}

<main class="container p-4">

{% if 'username' in session %}

<div class="container d-flex flex-column justify-content-center">

<h1 class="text-center">You are already Signed in</h1>

<div class="text-center">

<a class="btn btn-primary" href="/dashboard">Dashboard</a>

</div>

</div>

{% else %}

<div class="container d-flex" style="width: fit-content;">

<img src="{{ url\_for('static', filename='img/login.jpg') }}" class="m-5" style="width: 30vw"

alt="Login Illustration">

<form class="m-5" action="/signin/handle\_form" method="POST">

<div class="form-outline m-4">

<div class="col p-2 h3 text-center">Sign-in:</div>

</div>

<!-- Username Input -->

<div class="form-outline mb-4">

<label class="form-label" for="username">User Name:</label>

<input class="form-control" type="text" name="username" id="username" placeholder="username" minlength="3"

required>

</div>

<!-- Passoword Input -->

<div class="form-outline mb-4">

<label class="form-label" for="password">Password:</label>

<input class="form-control" type="password" name="password" id="password" placeholder="password" minlength="4"

required>

</div>

<!-- Sign-in Button -->

<div class="form-outline mb-4 text-center">

<button class="btn btn-primary btn-block mb-4" type="submit">Sign-in</button>

</div>

</form>

</div>

{% endif %}

</main>

{% include "scripts.html" %}

</body>

</html>

**stock.html**

<!-- Stock Page -->

<html>

<head>{% include "head.html" %}</head>

<body>

{% include "header.html" %}

<main class="container p-4">

<div>

{% if 'username' not in session %}

<div class="container d-flex flex-column justify-content-center">

<h1 class="text-center">You need to sign in to view this page</h1>

<div class="text-center">

<a class="btn btn-primary" href="/signin">Signin</a>

</div>

</div>

{% else %}<!-- View Stock -->

{% if mode == 'view' %}

<nav class="navbar navbar-expand-lg navbar-light bg-light">

<div class="container-fluid">

<ul class="navbar-nav">

<li class="nav-item dropdown">

<a class="nav-link dropdown-toggle" href="#" id="navbarDropdown" role="button" data-bs-toggle="dropdown" aria-expanded="false">

Sortby

</a>

<ul class="dropdown-menu" aria-labelledby="navbarDropdown">

<li><a class="dropdown-item" href="/stock/view/id">ID</a></li>

<li><a class="dropdown-item" href="/stock/view/name">Name</a></li>

<li><a class="dropdown-item" href="/stock/view/type">Type</a></li>

<li><a class="dropdown-item" href="/stock/view/price">Price</a></li>

<li><a class="dropdown-item" href="/stock/view/quantity">Quantity</a></li>

<li><a class="dropdown-item" href="/stock/view/facility">Facility</a></li>

</ul>

</li>

<li class="nav-item dropdown">

<a class="nav-link dropdown-toggle" href="#" id="navbarDropdown" role="button" data-bs-toggle="dropdown" aria-expanded="false">

Sort Order

</a>

<ul class="dropdown-menu" aria-labelledby="navbarDropdown">

<li><a class="dropdown-item" href="/stock/view/asc">Ascending</a></li>

<li><a class="dropdown-item" href="/stock/view/desc">Descending</a></li>

</ul>

</li>

</ul>

</div>

</nav>

<div class="d-flex justify-content-between">

<h2>Stock List:</h2>

<a href="/stock/add"><button type="button" class="btn btn-primary">Add New Stock</button></a>

</div>

<table class="table table-striped">

<thead class="table-dark">

<th>ID</th>

<th>Facility</th>

<th>Name</th>

<th>Type</th>

<th>Price</th>

<th>Quantity</th>

<th>Min Value</th>

<th></th>

<th>Action</th>

</thead>

{% set items = SM.get\_items(session['sort'], session['sortorder']) %}

{% for i in items %}

<tr>

<td>{{ i.id }}</td>

<td>{{ FM.get\_facility(i.facility).name }}</td>

<td>{{ i.name }}</td>

<td>{{ i.type }}</td>

<td>{{ i.price }}</td>

<td>

<form class="d-flex" action="/stock/update" method="post" style="max-width: fit-content;">

<input type="hidden" name="id" value="{{ i.id }}">

<input class="form-control me-2" type="number" name="quantity" value="{{ i.quantity }}">

<button class="btn btn-primary" type="submit">Update</button>

</form>

</td>

<td>{{ i.minvalue }}</td>

<td></td>

<td class="d-flex">

<form action="/stock/edit" method="post">

<input type="hidden" name="id" value="{{ i.id }}">

<button class="btn btn-primary me-2" type="submit">Edit</button>

</form>

<form action="/stock/delete" method="post">

<input type="hidden" name="id" value="{{ i.id }}">

<button class="btn btn-danger" type="submit">Delete</button>

</form>

</td>

</tr>

{% endfor %}

</table>

<!-- New Item -->

{% elif mode == 'add' %}

<div style="width: 40vw;">

<h2>New Item:</h2>

<form action="/stock/add/save" method="post">

<div class="form-outline mt-3">

<label class="form-label" for="name">Name: </label>

<input class="form-control" type="text" id="name" name="name" placeholder="name" required>

</div>

<div class="form-outline mt-3">

<label class="form-label" for="type">Type: </label>

<select class="form-select" name="type" id="type" required>

<option value="">--Select Type--</option>

{% for i in CON.CATEGORIES %}

<option value="{{i}}">{{i.capitalize()}}</option>

{% endfor %}

</select>

</div>

<div class="form-outline mt-3">

<label class="form-label" for="price">Price: </label>

<input class="form-control" type="text" id="price" name="price" placeholder="price" required>

</div>

<div class="form-outline mt-3">

<label class="form-label" for="quantity">Initial Quantity: </label>

<input class="form-control" type="number" id="quantity" name="quantity" value="0" placeholder="quantity" required>

</div>

<div class="form-outline mt-3">

<label class="form-label" for="minvalue">Min Value: </label>

<input class="form-control" type="number" id="minvalue" name="minvalue" value="0" placeholder="min value" required>

</div>

<div class="form-outline mt-3">

<label class="form-label" for="facility">Facility: </label>

<select class="form-select" name="facility" id="facility" required>

<option value="">--Select Facility--</option>

{% for i in FM.get\_facilities() %}

<option value="{{ i.id }}">{{ i.name }} ({{ i.id }})</option>

{% endfor %}

</select>

</div>

<div class="form-outline mt-4">

<button class="btn btn-primary me-2" type="submit">Confirm</button>

<a href="/stock/add/discard"><button class="btn btn-danger" type="button">Discard</button></a>

</div>

</form>

</div>

<!-- Edit Item -->

{% elif mode == 'edit' %}

{% set i = SM.get\_item(id) %}

<div style="width: 40vw;">

<h1>Edit Item:</h1>

<form action="/stock/edit/save" method="post">

<div class="form-outline mt-3">

<span class="form-label">ID:</span>

<span class="form-control">{{ i.id }}</span>

<input type="hidden" name="id" value="{{ i.id }}">

</div>

<div class="form-outline mt-3">

<label class="form-label" for="name">Name:</label>

<input class="form-control" type="text" id="name" name="name" placeholder="name" value="{{ i.name }}">

</div>

<div class="form-outline mt-3">

<label class="form-label" for="type">Type:</label>

<select class="form-select" name="type" id="type">

{% for j in CON.CATEGORIES %}

{% if j == i.type %}

<option value="{{j}}">{{j.capitalize()}}</option>

{% endif %}

{% endfor %}

{% for j in CON.CATEGORIES %}

{% if j != i.type %}

<option value="{{j}}">{{j.capitalize()}}</option>

{% endif %}

{% endfor %}

</select>

</div>

<div class="form-outline mt-3">

<label class="form-label" for="price">Price: </label>

<input class="form-control" type="text" id="price" name="price" placeholder="price" value="{{ i.price }}">

</div>

<div class="form-outline mt-3">

<label class="form-label" for="quantity">Quantity: </label>

<input class="form-control" type="number" id="quantity" name="quantity" value="{{ i.quantity }}" placeholder="quantity">

</div>

<div class="form-outline mt-3">

<label class="form-label" for="minvalue">Min Value: </label>

<input class="form-control" type="number" id="minvalue" name="minvalue" value="{{ i.minvalue }}" placeholder="min value">

</div>

<div class="form-outline mt-3">

<label class="form-label" for="facility">Facility: </label>

<select class="form-select" name="facility" id="facility">

{% set l = FM.get\_facilities() %}

{% for j in l %}

{% if i.facility == j.id %}

<option value="{{j.name}}">{{ j.name }} ({{ j.id }})</option>

{% endif %}

{% endfor %}

{% for j in l %}

{% if i.facility != j.id %}

<option value="{{j.name}}">{{ j.name }} ({{ j.id }})</option>

{% endif %}

{% endfor %}

</select>

</div>

<div class="form-outline mt-3">

<button class="btn btn-primary me-2" type="submit">Confirm</button>

<a href="/stock/edit/discard"><button class="btn btn-danger" type="button">Discard</button></a>

</div>

<div class="mt-3">Note: You can also Leave the input empty to keep existing value</div>

</form>

</div>

<!-- Delete Item -->

{% elif mode == 'delete' %}

{% set i = SM.get\_item(id) %}

<div class="col-md-8">

<div class="col p-2 h2">Delete Facility:</div>

<div class="card mb-3">

<div class="card-body">

<div class="row">

<div class="col-sm-3">

<h6 class="mb-0">ID</h6>

</div>

<div class="col-sm-9 text-secondary">

{{ i.id }}

</div>

</div>

<hr>

<div class="row">

<div class="col-sm-3">

<h6 class="mb-0">Name</h6>

</div>

<div class="col-sm-9 text-secondary">

{{ i.name }}

</div>

</div>

<hr>

<div class="row">

<div class="col-sm-3">

<h6 class="mb-0">Type</h6>

</div>

<div class="col-sm-9 text-secondary">

{{ i.type }}

</div>

</div>

<hr>

<div class="row">

<div class="col-sm-3">

<h6 class="mb-0">Price</h6>

</div>

<div class="col-sm-9 text-secondary">

{{ i.price }}

</div>

</div>

<hr>

<div class="row">

<div class="col-sm-3">

<h6 class="mb-0">Quantity</h6>

</div>

<div class="col-sm-9 text-secondary">

{{ i.quantity }}

</div>

</div>

<div class="row">

<div class="col-sm-3">

<h6 class="mb-0">Min Value</h6>

</div>

<div class="col-sm-9 text-secondary">

{{ i.minvalue }}

</div>

</div>

<div class="row">

<div class="col-sm-3">

<h6 class="mb-0">Facility</h6>

</div>

<div class="col-sm-9 text-secondary">

{{ i.facility }}

</div>

</div>

<hr>

<div class="row">

<div class="col-sm-3">

<form action="/stock/delete/save" method="post">

<input type="hidden" name="id" value="{{ i.id }}">

<button class="btn btn-danger" type="submit">Confirm Delete</button>

</form>

</div>

<div class="col-sm-2">

<a class="btn btn-primary" href="/stock/delete/discard">Discard</a>

</div>

</div>

</div>

</div>

</div>

<!-- Invalid URL -->

{% else %}

<h1>Invalid Mode</h1>

{% endif %}

{% endif %}

</main>

{% include "scripts.html" %}

</body>

</html>

**dashboard.html**

<!-- Dashboard Page -->

<html>

<head>{% include "head.html" %}</head>

<body>

{% include "header.html" %}

<main class="container p-5">

{% if 'username' not in session %}

<div class="container d-flex flex-column justify-content-center">

<h1 class="text-center">You need to sign in to view this page</h1>

<div class="text-center">

<a class="btn btn-primary" href="/signin">Signin</a>

</div>

</div>

{% else %}

<div class="row">

<div class="col">

<div class="row">

<h3>Top Selling</h3>

<div class="col">

<ol class="list-group list-group-numbered">

{% set soldlist = LM.get\_top\_sold(5) %}

{% if soldlist %}

{% for i in soldlist %}

<li class="list-group-item d-flex justify-content-between">

<span class="fw-bold ms-2 me-auto">{{i.name}} ({{i.item}})</span>

<span class="badge bg-primary rounded-pill">{{i['revenue']}}</span>

</li>

{% endfor %}

{% else %}

<li class="list-group-item d-flex justify-content-between">

<span class="fw-bold ms-2 me-auto">Not Items Sold Today</span>

<span class="badge bg-primary rounded-pill">0</span>

</li>

{% endif %}

</ol>

</div>

</div>

<div class="row mt-5">

<h3>Low On Stock</h3>

<div class="col">

<ol class="list-group list-group-numbered">

{% set minlist = SM.get\_minlist() %}

{% if minlist %}

{% for i in minlist %}

<li class="list-group-item d-flex justify-content-between">

<span class="fw-bold ms-2 me-auto">{{i.name}} ({{i.id}})</span>

<span class="badge bg-primary rounded-pill">{{i.quantity}}</span>

</li>

{% endfor %}

{% else %}

<li class="list-group-item d-flex justify-content-between">

<span class="fw-bold ms-2 me-auto">All Items are in enough Quantity</span>

<span class="badge bg-primary rounded-pill">0</span>

</li>

{% endif %}

</ol>

</div>

</div>

</div>

<div class="col">

<h3>Today's Sales</h3>

<div class="col">

<table class="table table-striped">

{% set soldlist = LM.get\_today\_logs() %}

{% if soldlist %}

<thead>

<th>Facility</th>

<th>Item Name</th>

<th>Price</th>

<th>Quantity</th>

<th>Action</th>

<th>Time</th>

</thead>

{% for i in soldlist %}

<tr>

<td>{{ i['facility'] }}</td>

<td>{{ i['name'] }}</td>

<td>{{ i['price'] }}</td>

<td>{{ i['quantity'] }}</td>

<td>{{ i['action'] }}</td>

<td>{{ i['timestamp'] }}</td>

</tr>

{% endfor %}

{% else %}

<span class="fw-bold ms-2 me-auto">Not Items Sold Today</span>

{% endif %}

</table>

</div>

<a class="btn btn-danger" href="/dashboard/clearlog">Clear All Log</a>

</div>

</div>

{% endif %}

</main>

{% include "scripts.html" %}

</body>

</html>

**GITHUB & PROJECT DEMO LINK**

https://drive.google.com/file/d/1-BuGSWjGPssoM8nYV-v47qu9QQUzm-Pb/view?usp=drivesdk