INVENTORY MANAGEMENT SYSTEM FOR RETAILERS

18PF15- PROFESSIONAL READINESS FOR INNOVATION, EMPLOYABILITY AND ENTREPRENEURSHIP

REPORT

Submitted by

Maalavika S	717819p219
Nivethida S	717819p224
Poornaa Y	717819p226
Ramuni Nithin Kumar	717819p228
Sivasakthivel S	717819p242

in partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

KARPAGAM
COLLEGE OF ENGINEERING
Rediscover | Refine | Redefine
Accredited by NAAC with 'A' grade
Autonomous | Affiliated to Anna University
(An ISO 9001:2015 and ISO 14001:2015 Certified Institution)

ANNA UNIVERSITY: CHENNAI - 600 025

NOVEMBER 2022

CONTENTS:

CHAPTER 1

INTRODUCTION

1.1) Project Overview:

Retail inventory management 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. Once retailers successfully log in to the application they can update their inventory details, also users will be able to add new stock by submitting essential details related to the stock. They can 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.

1.2)Purpose

Saves Time

Paper-based retail inventory management can take a lot of time and effort. The retail inventory management software can cut short your in-store inventory process cycles through automation. Automation would give you time to focus on other productive business tasks.

Eliminates Errors

Traditional retail inventory processes can be vulnerable to errors. Inventory process errors in retail would not only increase your expenses but would also impact your business reputation. The retail inventory software would make sure to minimize human intervention in the process. Thus, it would reduce errors considerably.

Improves Transparency

In the retail industry, the visibility of the real-time status of the various items in the inventory is very critical. It would impact many other retail processes and important business decisions. It is challenging to keep track of multiple items in the inventory round the clock through a paper-based process. A retail inventory management system can give you 360-degree item information anytime.

Efficient Stock Counting

If done manually, stock counting is a tedious and error-prone process. The retail inventory management software can automatically count the items in your warehouse with better accuracy. Hence, it can provide you with updated inventory reports.

The Anywhere for Retail Advantage

Retail companies deal with the incredible volume of inventory on an everyday basis. The retail inventory management solution can be an asset to any retail company.

CHAPTER 2

LITERATURE SURVEY

2.2) References:

Paper 1: A Review of Inventory Management System (Varalakshmi G S,Shivaleela S, June 2021)

This inventory system speeds up the process while minimizing manual labour, humanerror, and manual delays. Sales data can be tracked by this inventory management system inaddition to inventories. A web application for Windows that focuses on inventory and sales clearance is the inventory management system. For Windows operating systems, it was developed. There are numerous aspects in the inventory management system. This web programme offers logical capabilities for automatically choosing the best replenishment tactics and assessing ideal inventory levels. Additionally, it is able to compute reorder points automatically, highlight potential stock-outs, and determine stock levels. By reducing delays,this method avoids the possibility of stock-outs of commodities with high demand.

Paper 2: Forecasting intermittent demand for inventory management byretailers (Xin Tian a b , Haoqing Wang a b , Erjiang E c, September 2021)

In order to estimate intermittent demand, this study suggests a Markov-combined method (MCM) that takes into account product history and inventory levels. The prediction procedure is split into two steps by them. The transition probabilities of the four fundamentaldemand and inventory states are computed in the first stage. The second stage involves choosing the appropriate and relevant prediction method based on the anticipated situation. Additionally, they validate our findings and demonstrate that the MCM forecasts more precisely than the Single Exponential Smoothing (SES), Syntetos-Boylan Approximation (SBA), and Croston (CR) approaches using two sizable datasets from the two largest e-commerce enterprises in China. Due to its ease of computation and generally higher accuracy, the MCM can be used as a substitute approach for anticipating intermittent demand.

Paper 3: Inventory management, managerial competence and financialperformance of small businesses. (LauraA. Orobia, Joweria Nakibuuka, Juma Bananuka, Richard Akisimire. 29 May2020)

Establishing the links between inventory management, managerial skill, and financialperformance as well as determining whether inventory management acts as a mediator in the relationship between managerial skill and financial performance are the two main goalsof this study. Cross-sectional and correlational study designs were used. 304 Ugandan smallcompanies participated in a questionnaire survey. Analysis of Moments Structures (AMOS) software was used to evaluate hypotheses using a bootstrap analysis technique. According to the findings, managerial skill and inventory management are highly related to the financial health of small firms. The relationship between managerial skill and financial performance is also partially mediated by inventory management. Additionally, separately testing the indirecteffects of inventory management is done rather than just concentrating on the direct effects of managerial competency and inventory control.

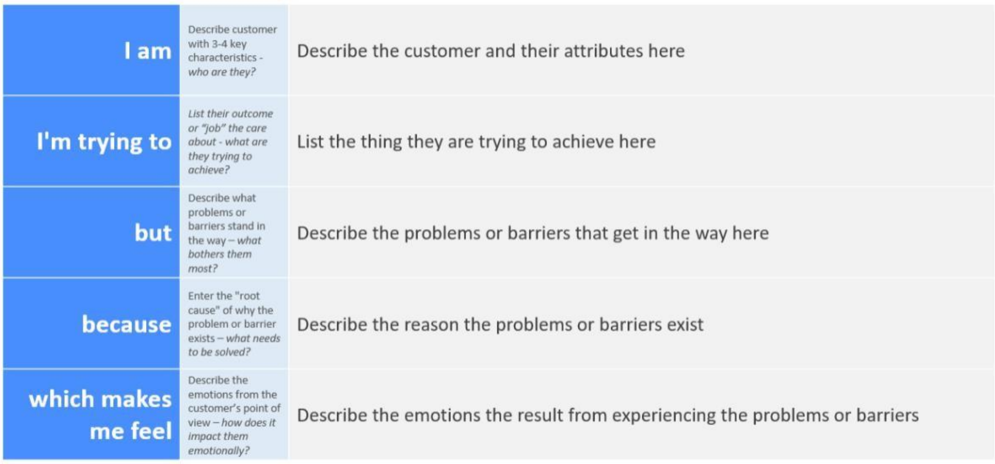
Paper 4: Optimal inventory control of obsolete products with price-dependentdemand. (Hassan Zamani ,Mohammad Reza Gholamian, November 2020)

The purpose of this study is to create an inventory policy for shops selling products with abrupt obsolescence that maximises profits while taking into account the type of obsolescence's exponential length. The study was conducted using a real-world case study of a tablet PC, where demand is believed to decrease as the price at which it is sold by the store increases. However, when obsolescence occurs, demand drops abruptly to zero. The mathematical model was created using the concepts of inventory management, taking into account the decision factors of order amount and retailer selling price. Sensitivity analyses on crucial model parameters were carried out using real-worlddata as a numerical example.

Paper 5: Coordinated inventory control and pricing policies for online retailerswith perishable products in the presence of social learning

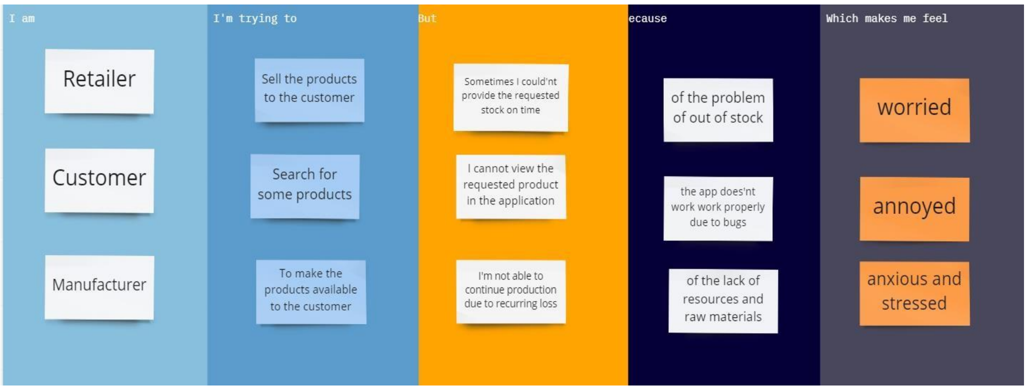
This study intends to investigate how social learning affects the coordinated dynamic pricing and inventory control problem for a perishable good. Through social learning, it is envisioned that online merchants using the Expiration Date-Based Pricing (EDBP) policy to sell perishable goods will beable to offset the practice's implied low quality. A mathematical model is created to frame the issue, and its structural characteristics are examined for a two-period lifetime product. In order to gain somemanagerial insights, numerical analysis is also carried out in a real-world case study. The findings gained demonstrate that the online shop can advertise the EDBP by using a system of user-generated online reviews. Additionally, the online retailer should modify the product pricing and inventory control regulations in accordance with the development of the system in order to effectively leverage it. Finally, by taking into account customers' social learning behaviour in the price and inventory rules, the company's profit and waste avoidance are increased.

2.3) Problem statement definition:



I am	Describe customer with 3-4 key characteristics - *who are they?*	Describe the customer and their attributes here
I'm trying to	*List their outcome or "job" the care about - what are they trying to achieve?*	List the thing they are trying to achieve here
but	Describe what problems or barriers stand in the way – *what bothers them most?*	Describe the problems or barriers that get in the way here
because	Enter the "root cause" of why the problem or barrier exists – *what needs to be solved?*	Describe the reason the problems or barriers exist
which makes me feel	Describe the emotions from the customer's point of view – *how does it impact them emotionally?*	Describe the emotions the result from experiencing the problems or barriers

Example:



I am	I'm trying to	But	ecause	Which makes me feel
Retailer	Sell the products to the customer	Sometimes I could'nt provide the requested stock on time	of the problem of out of stock	worried
Customer	Search for some products	I cannot view the requested product in the application	the app does'nt work work properly due to bugs	annoyed
Manufacturer	To make the products available to the customer	I'm not able to continue production due to recurring loss	of the lack of resources and raw materials	anxious and stressed

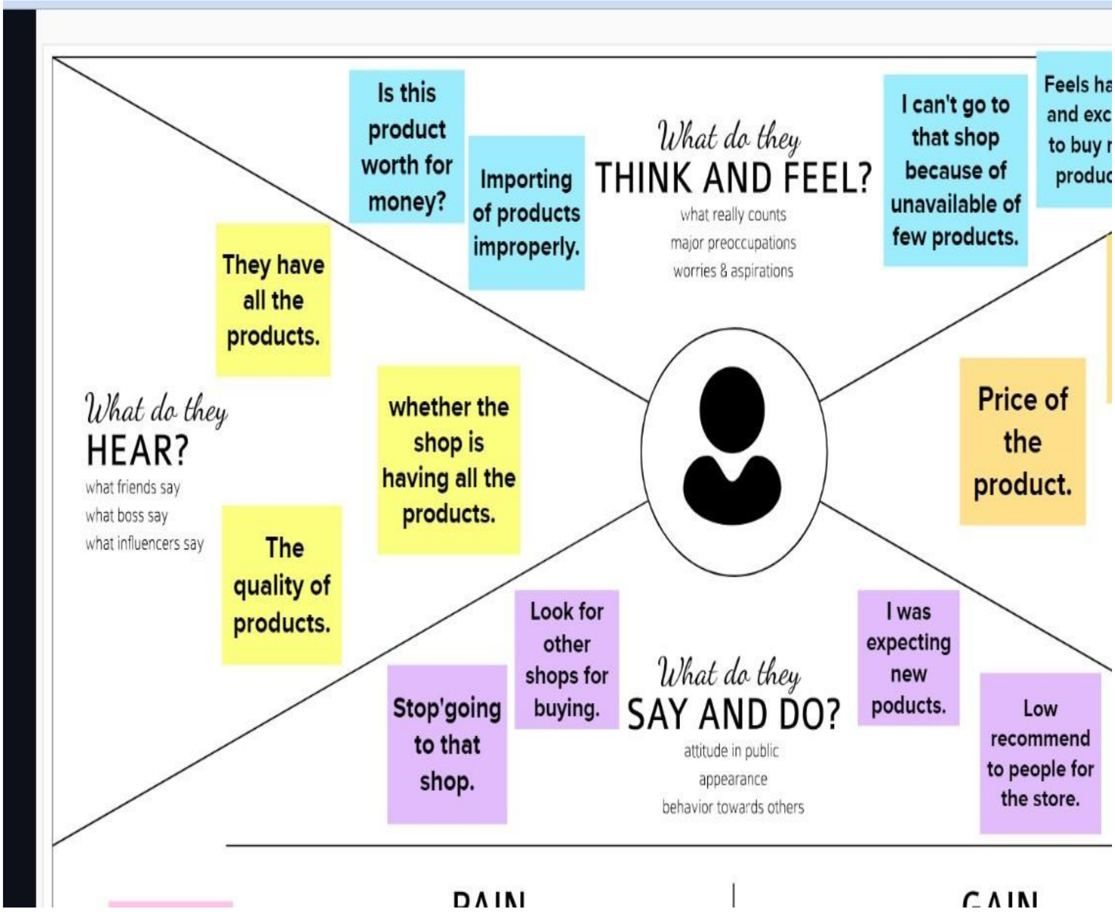
Problem Statement (PS)	**I am (Customer)**	**I'm trying to**	**But**	**Because**	**Which makes me feel**
PS-1	Retailer	Sell the products to customer	Could not provide stock on time	Of out of stock	worried
PS-2	Customer	Search for needed products	Cannot view the product	Of bugs	annoyed

PS-3	Manufacturer	Make products available to customer	Could not manufacture due to loss	Of the lack of resourcesand capital	Anxious and stressed

CHAPTER 3

IDEATION & PROPOSED SOLUTION

3.1)Empathy Map Canvas



Is this product worth for money?

Importing of products improperly.

I can't go to that shop because of unavailable of few products.

Feels ha and exc to buy r produc

They have all the products.

whether the shop is having all the products.

The quality of products.

Price of the product.

Look for other shops for buying.

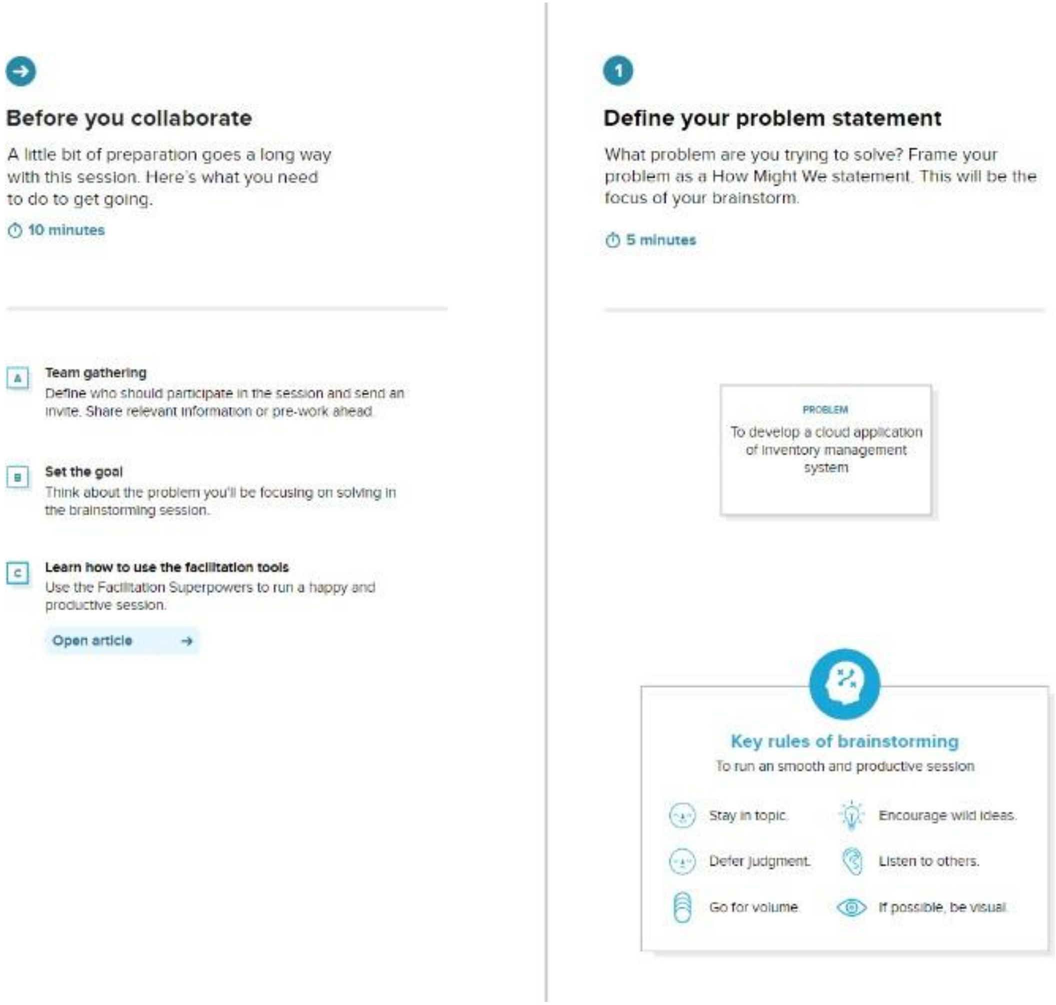
Stop'going to that shop.

I was expecting new poducts.

Low recommend to people for the store.

3.2 Ideation & Brainstorming

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



Before you collaborate

A little bit of preparation goes a long way with this session. Here's what you need to do to get going.

10 minutes

A **Team gathering**
Define who should participate in the session and send an invite. Share relevant information or pre-work ahead.

B **Set the goal**
Think about the problem you'll be focusing on solving in the brainstorming session.

C **Learn how to use the facilitation tools**
Use the Facilitation Superpowers to run a happy and productive session.

Open article →

1 **Define your problem statement**

What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.

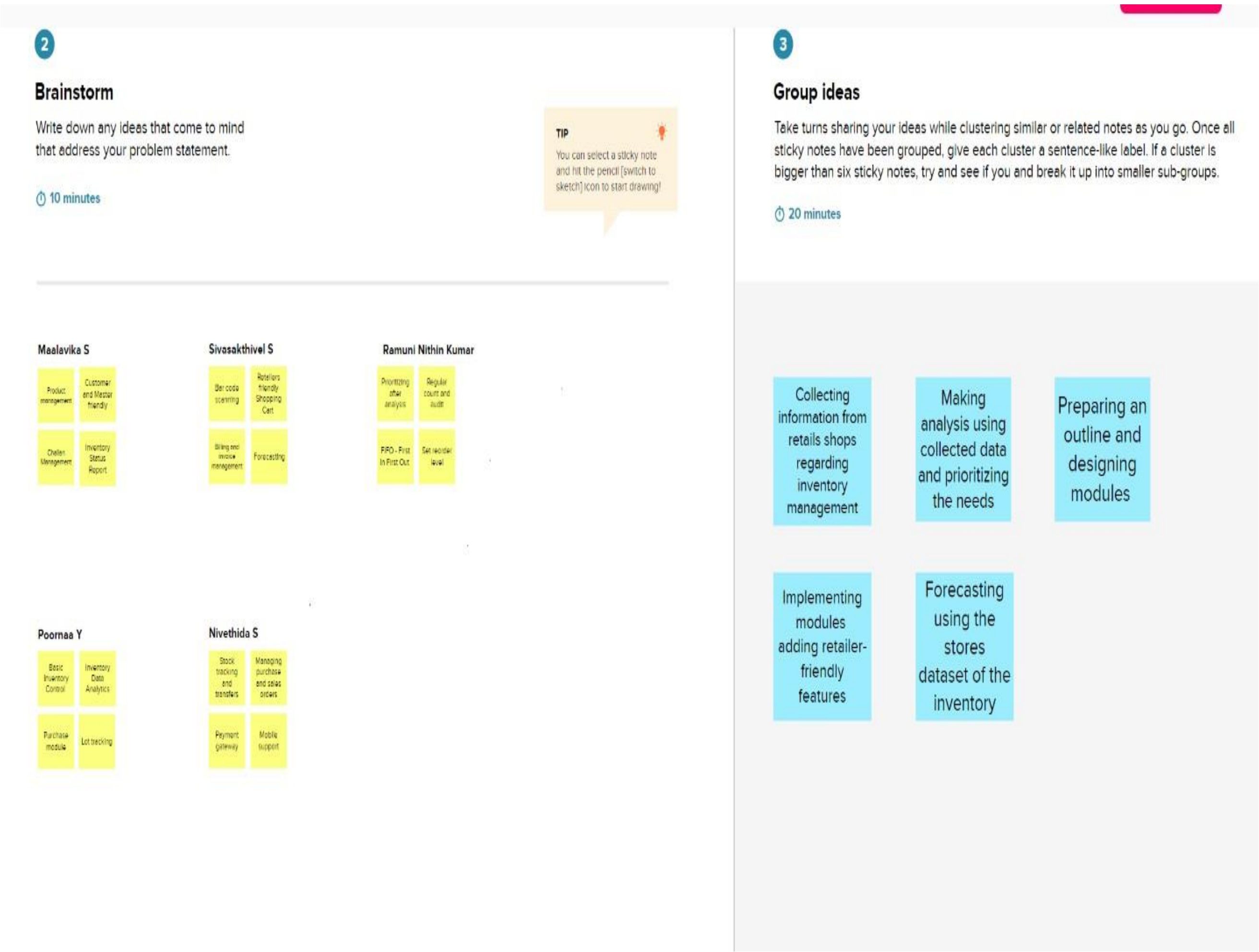
5 minutes

PROBLEM
To develop a cloud application of inventory management system

Key rules of brainstorming
To run an smooth and productive session

- Stay in topic.
- Encourage wild ideas.
- Defer judgment.
- Listen to others.
- Go for volume.
- If possible, be visual.

Step-2: Brainstorm, Idea Listing and Grouping



2

Brainstorm

Write down any ideas that come to mind that address your problem statement.

10 minutes

TIP

You can select a sticky note and hit the pencil [switch to sketch] icon to start drawing!

Maalavika S

- Product management
- Customer and Master friendly
- Challen Management
- Inventory Status Report

Sivasakthivel S

- Bar code scanning
- Retailers friendly Shopping Cart
- Billing and invoice management
- Forecasting

Ramuni Nithin Kumar

- Prioritizing after analysis
- Regular count and audit
- FIFO - First In First Out
- Set reorder level

Poornaa Y

- Basic Inventory Control
- Inventory Data Analytics
- Purchase module
- Lot tracking

Nivethida S

- Stock tracking and transfers
- Managing purchase and sales orders
- Payment gateway
- Mobile support

3

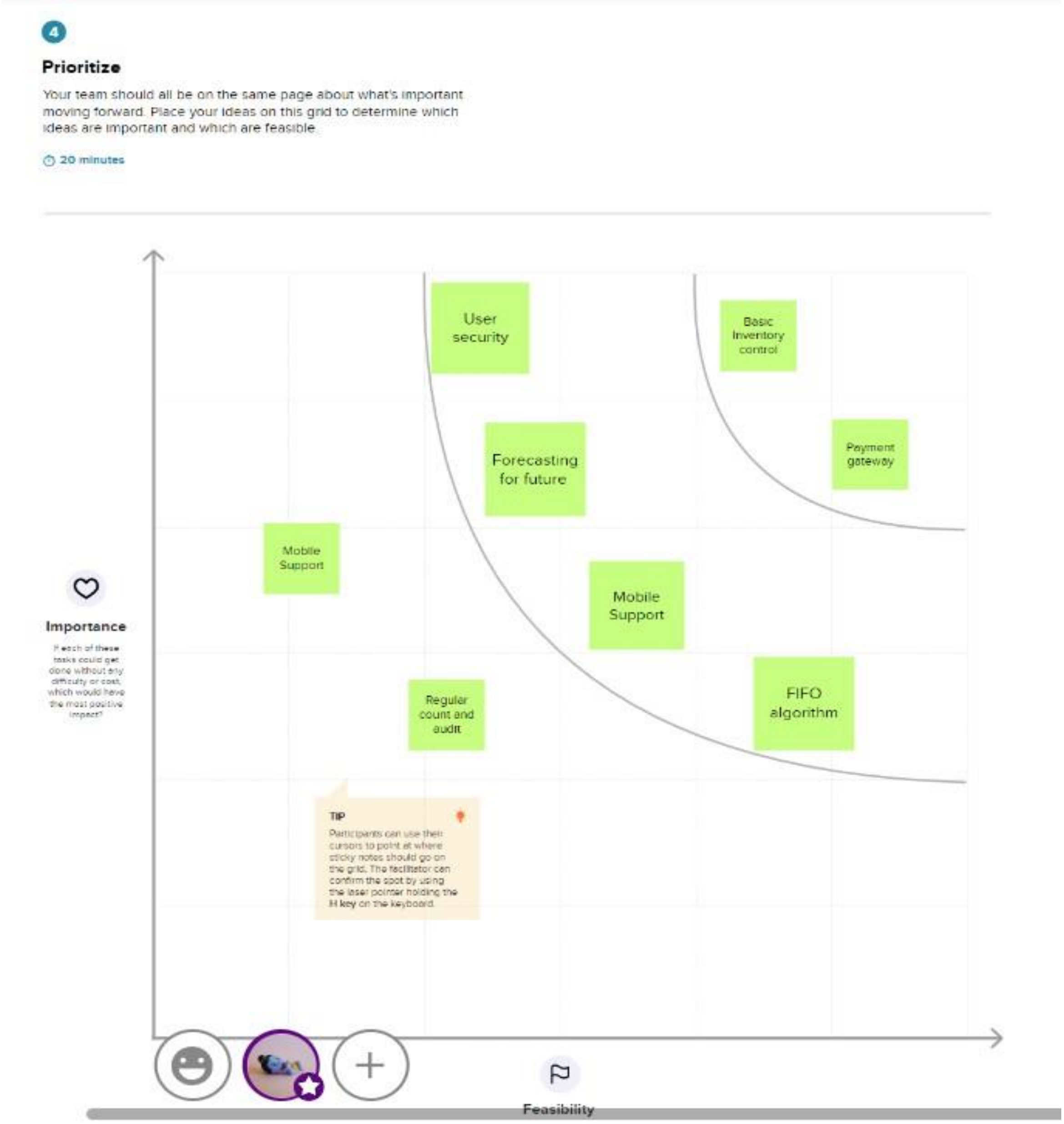
Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. Once all sticky notes have been grouped, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you and break it up into smaller sub-groups.

20 minutes

- Collecting information from retails shops regarding inventory management
- Making analysis using collected data and prioritizing the needs
- Preparing an outline and designing modules
- Implementing modules adding retailer-friendly features
- Forecasting using the stores dataset of the inventory

Step-3: Idea Prioritization



4

Prioritize

Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

20 minutes

Importance

If each of these tasks could get done without any difficulty or cost, which would have the most positive impact?

TIP

Participants can use their cursors to point at where sticky notes should go on the grid. The facilitator can confirm the spot by using the laser pointer holding the **H key** on the keyboard.

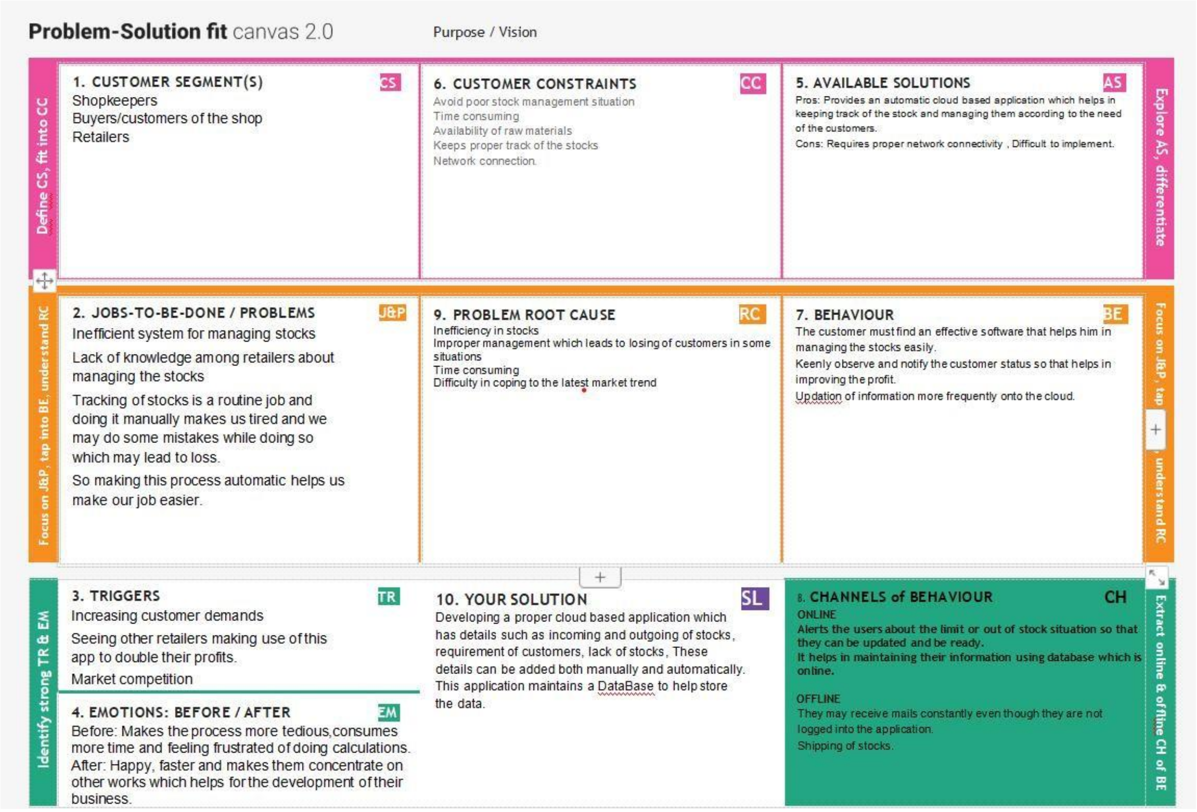
3.3 Proposed Solution

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Retailers don't have a formalized plan for managing their inventory data and hence they struggle. The admin finds it difficult to capture the data quickly and securely because they only save the inventory data in the logbook and are poorly organized. The retailer requires a method to efficiently analyze situations, prevent out-of- stock issues, prevent overstocking, and keep customers coming back in order to manage the inventory and run their business.
2.	Idea / Solution description	With the right platform, processes can be automated, inventory management techniques can be improved, and customer experiences can be enhanced. developing software that keeps track of the stock on hand, notifies the merchant when there is a shortage of merchandise in advance, frequently checks the stock count, and delivers the goods promptly.
3.	Novelty / Uniqueness	● Track inventory across numerous sites. ● Planning of production and distribution. ● Manage orders both online and offline. ● Increase scalability and flexibility with achoice of add-ons; ● Charge simply and affordably. ● End-to-end tracking increases customer satisfaction. ● It also handles reorder points automatically. ● Forecast demand ● Boosted sales

4.	Social Impact / Customer Satisfaction	Less time will be lost searching for anonexistent product, which will satisfy customers. If the method is automated each and every day and each time a purchase is made, the workload for retailers will be reduced to a minimal. Customer satisfaction will rise if they receive timely and pertinent responses from the stores.
5.	Business Model (Revenue Model)	Supply and demand must be balanced, financial and operational planning must be integrated, and high-level strategic plans must be linked to medium- and long-term business plans. Businesses may decide which goods to order, when, and in what amounts with the help of inventory management.
6.	Scalability of the Solution	With an automated inventory management system, a business will have the structure and real-time metrics it needs to maintain its competitiveness and achieve its growth goals. The profitability and efficiency of thecorporation will increase. Putting in place a system that anybody, anywhere may use to buy things can be beneficial. To prevent inventory shrinkage, the stock can be updated every day and after each purchase.

‘

3.4) Problem Solution fit



Problem-Solution fit canvas 2.0

Purpose / Vision

Define CS, fit into CC

1. CUSTOMER SEGMENT(S) CS

Shopkeepers
Buyers/customers of the shop
Retailers

6. CUSTOMER CONSTRAINTS CC

Avoid poor stock management situation
Time consuming
Availability of raw materials
Keeps proper track of the stocks
Network connection.

5. AVAILABLE SOLUTIONS AS

Pros: Provides an automatic cloud based application which helps in keeping track of the stock and managing them according to the need of the customers.

Cons: Requires proper network connectivity , Difficult to implement.

Explore AS, differentiate

Focus on J&P, tap into BE, understand RC

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

Inefficient system for managing stocks

Lack of knowledge among retailers about managing the stocks

Tracking of stocks is a routine job and doing it manually makes us tired and we may do some mistakes while doing so which may lead to loss.

So making this process automatic helps us make our job easier.

9. PROBLEM ROOT CAUSE RC

Inefficiency in stocks
Improper management which leads to losing of customers in some situations
Time consuming
Difficulty in coping to the latest market trend

7. BEHAVIOUR BE

The customer must find an effective software that helps him in managing the stocks easily.

Keenly observe and notify the customer status so that helps in improving the profit.

Updation of information more frequently onto the cloud.

Focus on J&P, tap , understand RC

Identify strong TR & EM

3. TRIGGERS TR

Increasing customer demands

Seeing other retailers making use of this app to double their profits.

Market competition

4. EMOTIONS: BEFORE / AFTER EM

Before: Makes the process more tedious,consumes more time and feeling frustrated of doing calculations.
After: Happy, faster and makes them concentrate on other works which helps for the development of their business.

10. YOUR SOLUTION SL

Developing a proper cloud based application which has details such as incoming and outgoing of stocks, requirement of customers, lack of stocks, These details can be added both manually and automatically. This application maintains a DataBase to help store the data.

8. **CHANNELS of BEHAVIOUR** CH

ONLINE
Alerts the users about the limit or out of stock situation so that they can be updated and be ready.
It helps in maintaining their information using database which is online.

OFFLINE
They may receive mails constantly even though they are not logged into the application.

Shipping of stocks.

Extract online & offline CH of BE

CHAPTER 4

REQUIREMENT ANALYSIS

4.1 Functional requirement

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form Registration through linkedIn Registration through Gmail
FR-2	User Confirmation	Confirmation via EmailConfirmation via OTP
FR-3	User Login	Login with username and password Login with mail idand password Login with phone number
FR-4	Dashboard	View product availability, name of the product, stock keep unit, brand, retail price, product category, lot number, expiration date, vendor information, wholesale cost,etc.,
FR-5	Login Details	Login details along with time through email Login details along with time through phone number
FR-6	Unavailability alert	Alert message through send-grid
FR-7	Monitoring of stock	Audit monitoring through incoming and outgoing stocks
FR-8	Updation	Update through user account
FR-9	Database	Usage of IBM cloud storage for storing the data.

4.2 Non-Functional requirement

Following are the non-functional requirements of the proposed solution.

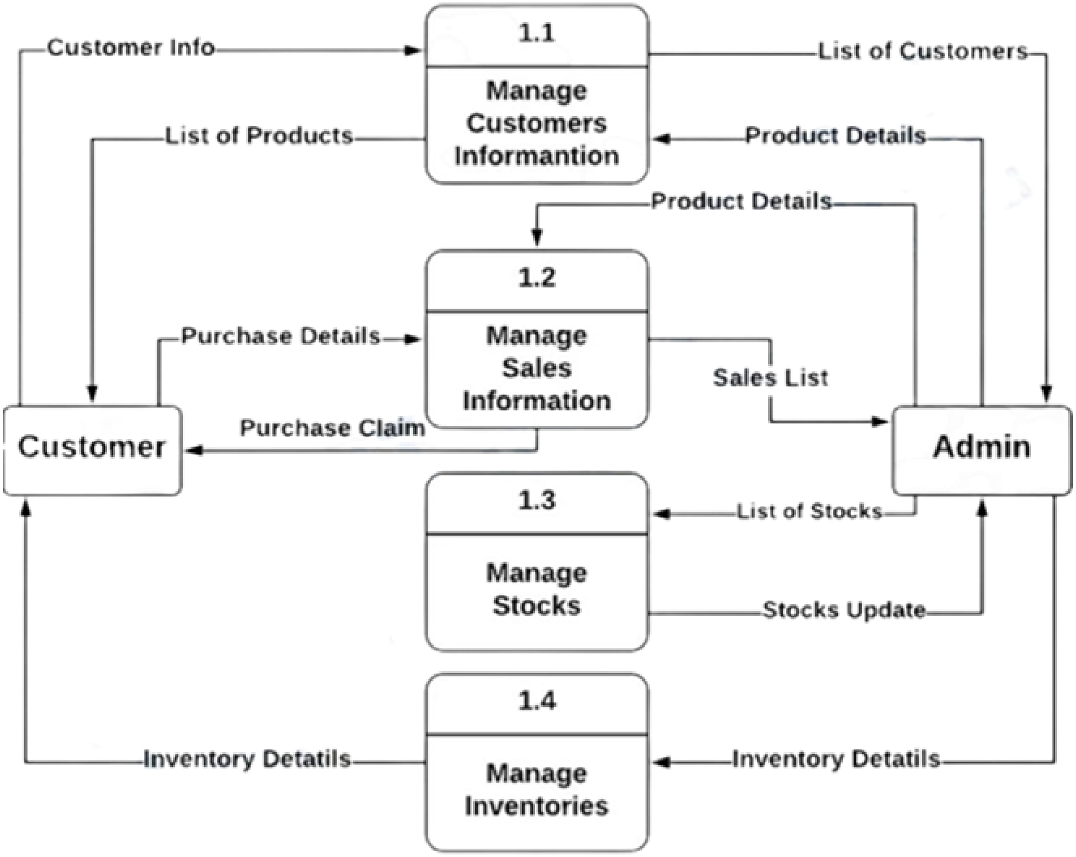
FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	Usability is crucial to the success of using aninventory tracking system in your business. The system's interface must be simple to use and clearly convey all relevant information and connections, and its menus must have buttons that are easy to understand. If your workforce needs to spend hours on training, the software is not worthwhile. Choosea solution that makes inventory management straightforward. Desktop browsers are compatible with this version.
NFR-2	**Security**	It is a technique for ensuring that things are kept safely and with the best possible management oversight. It is essential for efficient warehouse management since the productivity and security of a warehouse affect how well a company works. In this instance, the system can only be accessed byauthorised users who have their username and password.
NFR-3	**Reliability**	The system must continuously provide the user with accurate inventory status. Any mistakes are corrected by routinely comparing the actual levels to the levels displayed by the system.
NFR-4	**Performance**	The system must successfully perform operations including updating the database's stock information, adding new stocks, and deleting existing stocks each time a user requests a process. All of the system's functionality must be available to the user every time it is powered on. The system's calculations must adhere to the criteria that the customer has defined and shouldn't change unless the customer expressly asks it.

NFR-5	**Availability**	The programme will only be accessible to the organization's administrator, and it is he who will record client and product information. Whether an item is available for customer orders depends on its inventory status. Additionally, the administrator has the ability to add, remove, or change the stock and stock data.
NFR-6	**Scalability**	The business will become considerably more scalable with the use of an automated inventory management system for inventory tracking, enabling it to capitalize on rising sales and maintain steady growth.

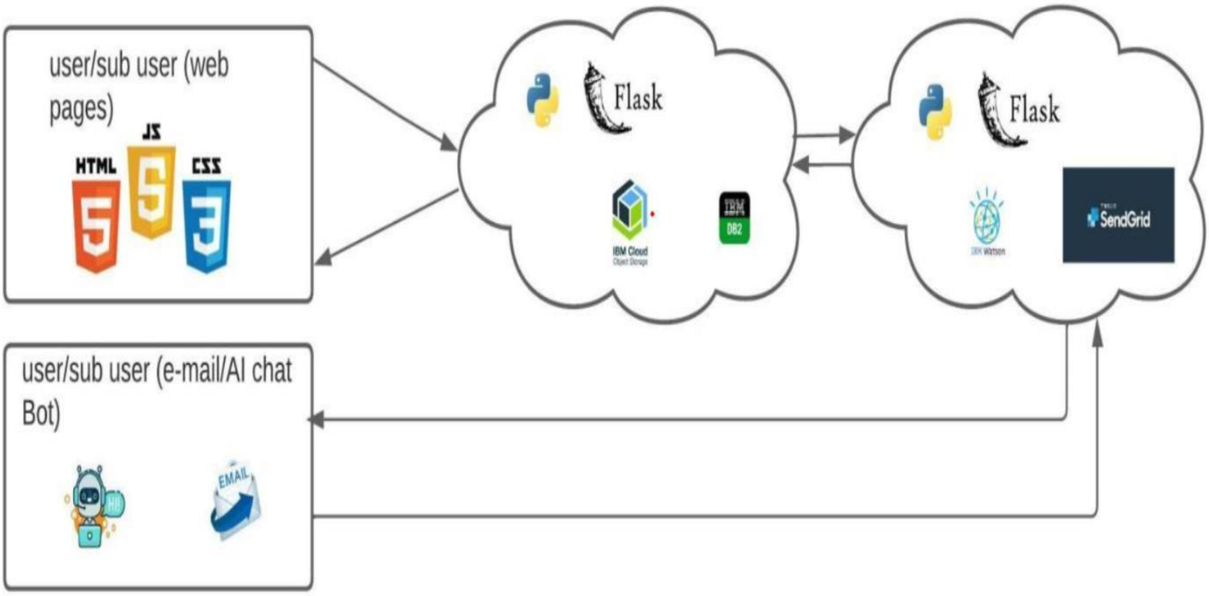
CHAPTER 5

PRODUCT DESIGN

5.1 Data Flow Diagram:



5.2) Solution and Technical Architecture:



S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	Styling our page,Python flask, sending email alert through sendgrid and storing DB using IBMDB2	flask,sendgrid,Kubernetes,IBMDB2
2.	Security Implementations	User can only login using their credentials, their password will hashe making It secure to use	IBM container registry
3.	Scalable Architecture	Large data can be stored easily using kubernetes	Web server - HTML, CSS, Javascript Application server - Python Flask, Docker, Container Registry Database server - IBM DB2

Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	How user interacts	flask,sendgrid,Kubernetes,IBMDB2

		with application e.g. Web UI, Mobile App, Chatbot etc.	
2.	Application Logic-1	Login/Registeration page- In this, the user can create an account for their inventory system	IBM container registry
3.	Application Logic-2	Contains dashboard which displays their activity, stock details and also customer details.	Web server - HTML, CSS, Javascript Application server - Python Flask, Docker, Container Registry Database server - IBM DB2
4.	Application Logic-3	Notification/alert about the stock status	HTML, CSS, JavaScript React Js
5.	Database	Stores user and sub users details using database.	Python flask
6.	Cloud Database	Stores details about stock and updated automatically through cloud services	Python flask
7.	File Storage	File storage requirements	Python flask
8.	External API-1	Purpose of sendgrid is to send an alert email to the user regarding stocks	MySQL, NoSQL
9.	External API-2	Enables us to store and retrieve dicker images throughout the container	IBM DB2, IBM Cloudant
10.	Infrastructure (Server / Cloud)	Application Deployment on Local System / Cloud Local Server Configuration:flask Cloud Server Configuration : Kubernetes	IBM Block Storage
			Sendgrid

			IBM container registry
			Local, Cloud Foundry, Kubernetes

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
4.	Availability	System will be always available and handy to use	IBM loadbalancer
5.	Performance	Fast and efficient. DB is maintained so data can be accessed easily.	IBMDB2 ,flask,kubernetes,Docker

5.3)User stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	High	Sprint-1
		USN-3	As a user, I can register for the application through Facebook	Low	Sprint-2

		USN-4	As a user, I can register for the application through Gmail	Medium	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering email & password	High	Sprint-1
	Dashboard	USN-6	As a user, I can view the available product list and the inventory data.	High	Sprint-2
		USN-7	As a user, I can view the order and track the shipping status	Medium	Sprint-3
Customer (Web user)	Registration	USN-8	As a user, I can register for the application by entering my email, password, and confirming my password.	High	Sprint-1
		USN-9	As a user, I will receive confirmation email once I have registered for the application	High	Sprint-1
		USN-10	As a user, I can register for the application through Facebook	Low	Sprint-2
		USN-11	As a user, I can register for the application through Gmail	Medium	Sprint-1
	Login	USN-12	As a user, I can log into the application by entering email & password	High	Sprint-1

User Type	**Functional Requirement (Epic)**	**User Story Number**	**User Story / Task**	**Priority**	**Release**
	Dashboard	USN-13	As a user, I can view the available	High	Sprint-2

			product list and the inventory data.		
		USN-14	As a user, I can view the order and track the shipping status	Medium	Sprint-3
Customer Care Executive	Chat bot	USN-15	As a customer care executive,I can view the complaints on chat bot and assist the users	Medium	Sprint-4
Administrator	Alerts	USN-16	As an administrator, I would handle user registrations and maintenance of accounts	High	Sprint-3
		USN-17	As an administrator, I can refill the stock on receiving the alerts	High	Sprint-3

CHAPTER-6

PROJECT PLANNING

& SCHEDULING

6.1) Sprint planning and Estimation:

Milestones:

Tasks	Assigned to	Start Date	End Date	Status
User Registration	Poornaa Y	24-oct-2022	29-oct-2022	In progress
Confirmation	Sivasakthivel S, Poornaa Y	24-oct-2022	29-oct-2022	In progress
Login	Nivethida S, Maalavika S	24-oct-2022	29-oct-2022	In progress
Dashboard	Ramuni nithin kumar,Maalavika S	31-oct-2022	05-nov-2022	Pending
Add items to stock	Sivasakthivel S	31-oct-2022	10-nov-2022	Pending
Stock Update	Ramuni nithin kumar, Nivethida S	07-nov-2022	14-nov-2022	Pending
Customer care	Maalavika S, Poornaa Y	14-nov-2022	19-nov-2022	Pending
Alert messaging	Maalavika S	14-nov-2022	19-nov-2022	Pending

Activity List:

Activity Number	Activity Name	Detailed Activity Description	Assigned To	Duration (Start to End Date)	Status
1	Create Flask Project	An application Framework written in Python	Poornaa Y Sivasakthivel s Nivethida s Maalavika s Ramuni nithin kumar	-	Completed
2	Create IBM Cloud	Create and log into IBM Cloud	Poornaa Y Sivasakthivel s Nivethida s Maalavika s Ramuni nithin kumar	-	Completed

3	Install IBM Cloud CLI	General-Purpose developer tool that provides access to your IBM Cloud Account	Poornaa Y Sivasakthivel s Nivethida s Maalavika s Ramuni nithin kumar	-	Completed
4	Docker CLI	Use Docker CLI configuration to customize settings	Poornaa Y Sivasakthivel s Nivethida s Maalavika s Ramuni nithin kumar	24 Oct 2022 to 29 Oct 2022	In Progress
5	Create Account in Send grid	Create account in SendGrid to send mails	Poornaa Y Sivasakthivel s Nivethida s Maalavika s Ramuni nithin kumar	31 Oct 2022 to 5 Nov 2022	In Progress

IMPLEMENTING WEB APPLICATION					
6	Create UI to Interact with Application	Pages such as Registration, Login page, Displaying items etc.	Poornaa Y Sivasakthivel s Nivethida s Maalavika s Ramuni nithinkumar	07 Nov 2022 to 12 Nov 2022	In Progress

6.2) Sprint delivery Schedule:

Product Backlog, Sprint Schedule, and Estimation (4 Marks)

Use the below template to create product backlog and sprint schedule

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN -1	User can create an account by providing business mail id and password	5	High	5
Sprint-2	Registration /Login	USN -2	Two step authentication using one time password to provide mail id or phone number	10	High	5
Sprint-1	Login	USN -3	Using registered mail Id	5	High	5
Sprint-1	dashboard	USN -4	User need to complete account settings like giving the details about their inventory and their branches	10	High	5
Sprint-2	Add item to stock	USN -5	User can able to add the item to stock	10	High	5
Sprint-3	Stock update	USN -6	To update the stock for check availability	10	High	5
Sprint-3	Request to customer care	USN -7	Customer care	10		
Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members

					High	5
Sprint-4	Alert Messaging	USN -8	User and mangers can get the details of the stock moment via mail or chat bot .	20	Medium	5

Project Tracker, Velocity & Burn down Chart: (4 Marks)

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	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	14 Nov 2022	20	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

$$AV = \frac{sprint\ duration}{velocity} = \frac{20}{10} = 2$$

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)

CHAPTER-7

CODING AND SOLUTIONING

7.1) Feature 1:

Sign-in.html

```
<html>
  <head>
    <title>
      Sign In
    </title>

    <link rel="stylesheet" href="signInStyle.css">
  </head>
  <body>
    <div class="signIn-box">
      <div class="form-box">
        <div class="button-box">
          <div id="btn"></div>
          <button type="button" class="toggle-
btn"
onclick="User()">&ensp;&ensp;<b>User</b></but
ton>
          <button type="button" class="toggle-
btn" onclick="Admin()">&emsp; &emsp;
<b>Admin</b></button>
        </div>
        <div class="social-icons">
         <a href="https://twitter.com/login"><img
src="twitter.jpg"></a>
```

```
                <a
href="http://www.linkedin.com"><img
src="linkedIn.jpg"></a>
               <a
href="https://accounts.google.com"><img
src="google.png"></a>
           </div>
           <form id="User" class="input-group">
               <input type="text" class="input-field"
placeholder="User Id" required>
               <input type="text" class="input-field"
placeholder="Password" required>
               <input type="checkbox" class="check-
box"><span>Remember Password</span>
               <div class="submit">
               <button type="submit" class="submit-
btn">Sign In</button><br>
               <button type="submit" class="submit-
btn"><a href="home.html">Back</a></button>
               <br><h5 style="color:#777">&ensp;
&emsp; No account?<a href="signUp.html"
style="color:azure">&ensp; Sign Up</a></h5>
               </div>
           </form>
           <form id="Admin" class="input-group">
               <input type="text" class="input-field"
placeholder="Admin Id" required>
               <input type="text" class="input-field"
```

```
placeholder="Password" required>
                <input type="checkbox" class="check-
box"><span>Remember Password</span>
                <div class="submit">
                    <button type="submit" class="submit-
btn">Sign In</button><br>
                    <button type="submit" class="submit-
btn"><a href="home.html">Back</a></button>
                    <br><h5 style="color:#777">&ensp;
&emsp; No account?<a href="signUp.html"
style="color:azure">&ensp; Sign Up</a></h5>
                    </div>
            </form>
            </div>

        </div>
        <script>
            var x=document.getElementById("User");
            var y=document.getElementById("Admin");
            var z=document.getElementById("btn");

            function Admin(){
                x.style.left="-400px";
                y.style.left="50px";
                z.style.left="110px";
            }
            function User(){
                x.style.left="50px";
                y.style.left="450px";
                z.style.left="0px";
            }
        </script>
    </body>
</html>
```

Signin.css:

```
  *{
        margin:0;
        padding:0;
        font-family: sans-serif;
  }
  .signIn-box{
        height:100%;
        width:100%;
        background-image: linear-
  gradient(rgba(0,0,0,0.4),rgba(0,0,0,0.4)),url("cover
  5.jpg");
        background-position:center;
        background-size:cover;
        position:absolute;
  }
  .submit a{
        text-decoration: none;
        color: black;
        height:100%;
        width:100%;
```

```
}
.form-box{
	width:380px;
	height:480px;
	position:relative;
	margin:6% auto;
	background:transparent;;
	padding: 5px;
	overflow:hidden;
}
.button-box{
	width:220px;
	margin:35px auto;
	position:relative;
	box-shadow:0 0 20px 9px cadetblue;
	border-radius:30px;
}
.toggle-btn{
	padding:10px 20px;
	cursor:pointer;
	background:transparent;
	color: white;
	border:0;
	outline:none;
	position:relative;
}
#btn{
	top:0;
	left:0;
	position:absolute;
	width:110px;
	height:100%;
	background:linear-gradient(to right,rgb(127,
193, 255), aquamarine);
	border-radius:30px;
	transition:.5s;
}
.social-icons{
	margin:30px auto;
	text-align:center;
}
.social-icons img{
	width:30px;
	margin:0 12px;
	box-shadow:0 0 20px 0 cadetblue;
	cursor:pointer;
	border-radius:50%;
}
.input-group{
	top:180px;
	position:absolute;
	width:200px;
	transition:.5s;
}
.input-field{
	width:100%;
	padding:10px 0;
	margin:5px 0;
	color:beige;
	border-left:0;
	border-top:0;
	border-right:0;
	border-bottom:1px solid aquamarine;
	outline:none;
	background:transparent;
}
.submit-btn{
	width:85%;
	padding:10px 30px;
	cursor:pointer;
	display:block;
	margin:auto;
	background:linear-gradient(to right,rgb(127,
193, 255),aquamarine);
	border:0;
	outline:none;
	border-radius:30px;
}
.check-box{
	margin: 30px 10px 30px 0;
}
span{
	color:#777;
	font-size:12px;
```

```
        font-family:Arial, Helvetica, sans-serif;
        bottom: 155px;
        position:absolute;
  }
  #User{
        left:50px;

  }
  #Admin{
        left:450px;
}
```

Signup.html:

```
  <html>
     <head>
         <title>
             Sign Up
         </title>

         <link rel="stylesheet"
  href="signUpStyle.css">
     </head>
     <body>
         <div class="signUp-box">
             <div class="form-box">
                 <div class="button-box">
                     <div id="btn"></div>
                     <button type="button" class="toggle-
  btn">&ensp;&ensp;<b>&emsp;&emsp;&emsp;&e
```

```
msp;Sign Up</b></button>
                    </div>
            <form id="Sign-Up" class="input-group">
                <input type="text" class="input-field"
placeholder="Username" required>
                <input type="text" class="input-field"
placeholder="Password" required>
                <input type="text" class="input-field"
placeholder="Confirm Password" required>
                <div class="drop-down">

<select><option>User</option><option>Admin</o
ption></select>
                </div>
                <input type="text" class="input-field"
placeholder="Company Name">
                <input type="text" class="input-field"
placeholder="Country" required>
                <input type="email" class="input-field"
placeholder="E mail" required>


                <input type="checkbox" class="check-
box"><span>Remember Password</span>
                <div class="submit">
                <button type="submit" class="submit-
btn">Sign Up</button><br>
                <button type="submit" class="submit-
btn"><a href="home.html">Back</a></button>
```

```
                </div>
             </form>
             </div>

          </div>
       </body>
</html>
```

Signupstyle.css

```
{
       margin:0;
       padding:0;
       font-family: sans-serif;
}
.signUp-box{
       height:150%;
       width:100%;
       background-image: linear-
gradient(rgba(0,0,0,0.4),rgba(0,0,0,0.4)),url(
"cover5.jpg");
       background-position:center;
       background-size:cover;
       position:absolute;
}
.submit a{
       text-decoration: none;
       color: black;
       height:100%;
```

```
	width:100%;
}
.form-box{
	width:400px;
	height:700px;
	position:relative;
	margin: 2.8% auto;
	background:transparent;;
	padding: 5px;
	overflow:hidden;
}
.button-box{
	width:220px;
	margin:35px auto;
	position:relative;
	box-shadow:0 0 20px 9px cadetblue;
	border-radius:30px;
}
.toggle-btn{
	padding:10px 20px;
	cursor:pointer;
	background:transparent;
	color: black;
	border:0;
	outline:none;
	position:relative;
}
#btn{
	top:0;
	left:0;
	position:absolute;
	width:220px;
	height:100%;
	background:linear-gradient(to
right,rgb(127, 193, 255), aquamarine);
	border-radius:30px;
	transition:.5s;
}
.input-group{
	top:140px;
	position:absolute;
	width:200px;
	transition:.5s;
}
.input-field{
	width:100%;
	padding:10px 0;
	margin:5px 0;
	color:beige;
	border-left:0;
	border-top:0;
	border-right:0;
	border-bottom:1px solid aquamarine;
	outline:none;
	background:transparent;
}
```

```
.drop-down select{
   width:100%;
      padding:10px 0;
      margin:5px 0;
      color:white;
      border-left:0;
      border-top:0;
      border-right:0;
      border-bottom:1px solid aquamarine;
      outline:none;
      background:transparent;
}
.drop-down option{
      color: white;
      background:transparent;
   display:block;
   border-left:none;
   background-color: rgb(11, 11, 58);

}
.submit-btn{
      width:85%;
      padding:10px 30px;
      cursor:pointer;
      display:block;
      margin:auto;
      background:linear-gradient(to
right,rgb(127, 193, 255),aquamarine);
```

```
        border:0;
        outline:none;
        border-radius:30px;
    }
    .check-box{
        margin: 30px 10px 30px 0;
    }
    span{
        color:#777;
        font-size:12px;
        font-family:Arial, Helvetica, sans-serif;
        bottom: 120px;
        position:absolute;
    }
    #Sign-Up{
        left:50px;

    }
```

Home.html

```
<!DOCTYPE html>
<html>
<head>
<title>Simply Shopping</title>
<link rel="stylesheet" href="homestyle.css">
</head>
<body>
<header>
<div class="wrapper">
```

```
     <div class = "logo">
          <img src="SS1.jpg" alt="logo"><br><br>SIMPLY SHOPPING
     </div>
   <ul class = "nav-area">
       <li><a href = "home.html">Home</a></li>
       <li><a href = "SignUp.html">Sign Up</a></li>
       <li><a href = "signIn.html">Sign In</a></li>
       <li><a href = "products.html">Products</a></li>
       <li><a href = "AboutUs.html">About Us</a></li>
    </ul>
</div>
<div class= "welcome-txt">
   <h1>THE SUPER INVENTORY PLATFORM</h1>
   <a href ="https://twitter.com/login">Contact us</a>
</div>
</header>
</body>
</html>
```

Homestyle.css

```
@charset "ISO-8859-1";
*{
          margin:0;
          padding:0;
}
.wrapper{
          width:1170px;
          margin:auto;
  color:#fff;
  font-size: 20px;
```

```
}
.wrapper:hover{
  color:aqua;
}
header{
        background: linear-
gradient(rgba(0,0,0,0.6),rgba(0,0,0,0.6)),url("cover5.jpg");
        height:100vh;
        -webkit-background-size:cover;
        background-size:cover;
        background-position:center center;
        position:relative;
}
.nav-area{
  float:right;
  list-style:none;
  margin-top: 30px;
}
.nav-area li{
  display:inline-block;
}
.nav-area li a{
  color: white;
  text-decoration:none;
  padding: 5px 20px;
  font-family: poppins;
  font-size:14px;
}
.nav-area li a:hover{
  background :cadetblue;
  color:#333;
}
.logo img{
  width:100px;
  float:left;
  height:auto;
}
.welcome-txt{
  position:absolute;
  width:700px;
  height:300px;
  margin:14% 28%;
  text-align: center;
  color:aquamarine;
  font-size: 25px;
  font-family:'Lucida Sans', 'Lucida Sans Regular', 'Lucida Grande', 'Lucida Sans
Unicode', Geneva, Verdana, sans-serif;
}
.welcome-txt a{
   color:aliceblue;
   border: 1px solid #fff;
   padding: 10px 25px;
   text-decoration:none;
   text-transform:uppercase;
   font: size 14px;
   margin-top: 20px;
   display: inline-block;
}
```

```
.welcome-txt a:hover{
  background: #000;
  color:darkseagreen;
}
```

Products.html

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>All Products</title>
        <!--star rating--><link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.2.0/css/all.min.css"
integrity="sha512-
xh6O/CkQoPOWDdYTDqeRdPCVd1SpvCA9XXcUnZS2FmJNp1coAFzvtCN
9BmamE+4aHK8yyUHUSCcJHgXloTyT2A==" crossorigin="anonymous"
referrerpolicy="no-referrer" />
        <link rel="stylesheet" href="productsStyle.css">
        <link rel="preconnect" href="https://fonts.googleapis.com">
<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
<link
href="https://fonts.googleapis.com/css2?family=Roboto:ital,wght@0,700;1,100
;1,900&display=swap" rel="stylesheet">
<link href="{{url_for('static', filename='productsStyle.css')}}"
rel="stylesheet">
    </head>
    <body >
        <header>
        <div class="container">
            <div class="navbar">
```

```
                <div class="logo">
                <img src="SS1.jpg"width="65px">
            </div>
            <nav>
                <ul>
                    <a href="Addproducts.html"class="btn">Update Products </a>
&ensp;
                    <li><a href="home.html">Home</a></li>
                    <li><a href="">Products</a></li>
                    <li><a href="AboutUs.html">About </a></li>
                    <li><a href="https://twitter.com/login">Contact</a></li>
                    <img src="https://s3.jp-tok.cloud-object-
storage.appdomain.cloud/kce7116/inventory.jpeg" width="30px"
height="30px">
                    <br>
                    <br>
                    <br>


                </div>


                </ul>
            </nav>

        </div>




<!-- -----featured catagories-------- -->
```

```
<!-- -----featured catagories-------- -->
<div class="smallcontainer">

    <div class="row">
        <div class="pro">
        <div class="col4">
            <img  src="pens.jpeg" >
            <h4>Pens bunch</h4><br>
            <div class="rating">
            <i class="fa fa-star"></i>
            <i class="fa fa-star"></i>
            <i class="fa fa-star"></i>
            <i class="fa fa-star"></i>
            <i class="fa fa-star-half-stroke"></i></div><br>
            <p>Rs.200.00</p></div>
            <button type="submit" class="submit-btn">Add to cart</button>
        </div> <div class="col4"><div class="pro">
            <img src="book1.jpeg" >
            <h4>Children Storybooks</h4><br>
            <div class="rating">
            <i class="fa fa-star"></i>
            <i class="fa fa-star"></i>
            <i class="fa fa-star"></i>
           <i class="fa fa-star-half-stroke"></i>
           <i class="fa-regular fa-star"></i></div><br>
            <p>Rs.1800.00</p></div>


            <br>
            <button type="submit" class="submit-btn">Add to cart</button><br>
        </div> <div class="col4"><div class="pro">
            <img src="paints.jpeg" >
            <h4>Paint box</h4><br>
            <div class="rating">
            <i class="fa fa-star"></i>
            <i class="fa fa-star"></i>
            <i class="fa fa-star"></i>
            <i class="fa fa-star"></i>
            <i class="fa-regular fa-star"></i></div><br>
            <p>Rs.150.00</p> </div>
            <br>
            <button type="submit" class="submit-btn">Add to cart</button>


        </div>



    </div></div>


            </div>





            </header>


        </body></html>
```

Homestyle.css

```
*{
}
    margin:0;
    padding:0;
    box-sizing:border-box;
}
body{

    font-family:poppins;
}
header{
        background: linear-
gradient(rgba(0,0,0,0.6),rgba(0,0,0,0.6)),url("cover5.jpg");
        height:100vh;
        -webkit-background-size:cover;
        background-size:cover;
        background-position:center center;
        position:relative;
}
 .navbar li a:hover{
    height: auto;
    width: 100%;
    padding: 5px 20px;
    background :cadetblue;
    color:#333;
}
 nav{
    flex :1;
    text-align: right ;
}
nav ul{
    display: inline-block;
    list-style-type:  none;
}
nav ul li{
    display: inline-block;
    margin-right: 20px;
}

 a{
    text-decoration: none;
    color:white;
}
p{
    color:white;
}
.container{
    max-width:1300px ;
    margin: auto ;
    padding-left: 25px;
    padding-right: 25px;
}
.row{
display: flex;
align-self: center;
flex-wrap: wrap;
justify-content: space-around;
}
.col2{
    flex-basis: 50%;
```

```
   min-width: 300px;
}
.col2 img{
   max-width: 80%;
   margin: 10px;
   padding: 50px 0px;
}
.col2 h1{
   line-height: 60 px;
   margin: 25 px 0px;
}
.btn{
   display: inline-block;
   background: aquamarine;
   color: #ffff;
   border-radius: 10px;

   padding: 5px;
   margin: 30 px 0;
   border-radius: 30 px;
   transition:0.5s;
}
.submit-btn{
        width:85%;
        padding:10px 30px;
        cursor:pointer;
        display:block;
        margin:auto;
        background:linear-gradient(to right,rgb(127, 193, 255),aquamarine);
```

```
            border:0;
            outline:none;
            border-radius:30px;
    }
    .btn:hover{
       background:aqua;
    }
    .header{
       background: linear-gradient(rgb(61, 28, 126),#911313),rgb(28, 212, 59);
    }
    .header .row{
       margin-top: 70px;
    }
    .catagories{
       margin: 70px 0;
    }
    .col3{
       flex: 25%;
       min-width: 250px;
       margin-bottom: 30px;
       margin: 10px;
    }
    .col3 img {
       width: 100%;

    }
    .smallcontainer{
       max-width: 1080px;
       margin: auto;
    padding-left: 25px;
    padding-right: 25px;
}
.col4{
    flex-basis: 20%;
    padding: 10px;
    min-width: 150px;
    margin-bottom: 50px;
    transition: transform 0.5s;
}
.col4 img {
    width: 100%;
}
.title{
    text-align: center;
    margin: 0 auto 80px;
    position: relative;
    line-height: 60 px;
    color:white;
}
.title::after{
    content :'';
    background:white;
width: 80px;
height: 5px;
border-radius: 5px;
position:absolute;
bottom: -5px;
left:50%;n
```

```
transform: translatex(-50%);}
h4{
    color:white;
    font-weight: normal;
}
.col-4 p{
font-size: 14px;
}
.rating .fa{
    color:white;
}
.pro:hover{
     transform:translateY(-10px);

    }
#col-i{
    height: 150px;
}
```

Addproducts.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">
    <title>Add Products</title>

<link rel="stylesheet" href="AddproductsStyle.css">
</head>
```

```
<body>
    <header>
    <link
href="//maxcdn.bootstrapcdn.com/bootstrap/3.3.0/css/bootstrap.min.css"
rel="stylesheet" id="bootstrap-css">
<script
src="//maxcdn.bootstrapcdn.com/bootstrap/3.3.0/js/bootstrap.min.js"></script>
<script src="//code.jquery.com/jquery-1.11.1.min.js"></script>
<script>
$(document).ready(function(){

    var quantitiy=0;
       $('.quantity-right-plus').click(function(e){

            // Stop acting like a button
            e.preventDefault();
            // Get the field name
            var quantity = parseInt($('#quantity').val());

            // If is not undefined

                $('#quantity').val(quantity + 1);



                // Increment

        });

         $('.quantity-left-minus').click(function(e){
            // Stop acting like a button
            e.preventDefault();
            // Get the field name
            var quantity = parseInt($('#quantity').val());

            // If is not undefined

                // Increment
                if(quantity>0){
                $('#quantity').val(quantity - 1);
                }
        });

    });
    </script>
    <div class="home-content">
        <div class="overview-boxes">



        <section class="catogories" >
          <div class="container-fluid">
              <div class="row">
                  <div class="col-lg-6">

                      <div class="row">

                          <div class="col-lg-6 col-md-6 col-12 p-1" style="margin-top:
25px;">
                              <div class="catagories_item">
          </div>
        </div>
        <div class="col-lg-6 p-0" style=" margin-top: 25px;">
          <div class="catagories_item">
          <div class="catagories_item catagories_large_item" >




      </div></div>
        </div>
        <div class="col-lg-6 col-md-6 col-12 p-1">
          <div class="catagories_item">

          </div>
        </div>
        <div class="col-lg-6 col-md-6 col-12 p-1">
          <div class="catagories_item">

          </div>
        </div>

      </div>
    </div>

     <div class="container register-form">

       <div class="form">
         <div class="d-flex align-items-center mb-3 pb-1"style=" margin-
```

```
top: 50px; align:center">

             <span class="h1 fw-bold mb-0">ADD A PRODUCT</span>
             <br>
             <br>
            </div>

             <div class="form-content">
              <form class="mx-1 mx-md-4" action = "{{
url_for('addproduct') }}" method = "POST">
               <div class="row">
                 <div class="container">
                  <div class="row">
                     <div class="col-md-6">
                       <h2>Product Name</h2>
                         <input type="text" name="pname" value=""
placeholder="Product Name" class="form-control" />
                     </div>
                   </div>
                 </div>
                 <div class="col-md-6">
                     <div class="flex-row align-items-center mb-4">
                       <h2>Quantity</h2>
                       <div class="input-group">
                         <span class="input-group-btn">
                           <button type="button" class="quantity-left-
minus btn btn-danger btn-number"  data-type="minus" data-field="">
                             <span class="glyphicon glyphicon-
minus"></span>
```

```
                           </button>
                         </span>
                         <input type="text" id="quantity"
name="quantity" class="form-control input-number" value="10" min="1"
max="100">
                           <span class="input-group-btn">
                             <button type="button" class="quantity-
right-plus btn btn-success btn-number" data-type="plus" data-field="">
                               <span class="glyphicon glyphicon-
plus"></span>
                             </button>
                           </span>
                       </div>
                     </div>
                 </div>
               </div>
              <div class="container">
                <div class="row">
                   <div class="col-md-6">
                     <h2>Submitter Name</h2>
                       <input type="text" name="name" value=""
placeholder="Product Name" class="form-control" />
                       <br><br>



                   </div>


                 </div>
                 <button type="submit" class="submit-
```

```
btn">Submit</button><br>
                        </div>






                        </form>
                      </div>
                  </div>
                  </div>



                      </div>
                  </div>
              </div>


          </div>


    </section>
  </header>
</body>
</html>
```

Addproductsstyle.css

```
@charset "ISO-8859-1";
*{
          margin:0;
          padding:0;
}
header{
        background: linear-
gradient(rgba(0,0,0,0.6),rgba(0,0,0,0.6)),url("cover5.jpg");
        height:100vh;
        -webkit-background-size:cover;
        background-size:cover;
        background-position:center center;
        position:relative;
   font-family:poppins;
   color:white
}
body{

   font-family:poppins;
}
h2{
   font-family:poppins;
   color:white
}
.submit-btn{
   width: 20%;
        padding:10px 30px;
        cursor:pointer;
        background:linear-gradient(to right,rgb(127, 193, 255),aquamarine);
        border:0;
        outline:none;
        border-radius:30px;
}
```

Complaint.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">
   <title>Complaint Form</title>
</head>

<body>
   <header>
   <form id="fs-frm" name="complaint-form" accept-charset="utf-8"
action="/complaintdata" method="post">
      <fieldset id="fs-frm-inputs">
        <label for="full-name">Full Name</label>
        <input type="text" name="name" id="full-name" placeholder="First and
Last" required="">
        <label for="email-address">Email Address</label>
        <input type="email" name="mail" id="email-address"
placeholder="email@domain.tld" required="">
        <label for="complaint">Complaint</label>
        <textarea rows="6" name="complaint" id="complaint"
placeholder="Please Enter your complaint in here. So we can improve
ourselves." required=""></textarea>
        <input type="hidden" name="_subject" id="email-subject"
value="Complaint Form Submission">
      </fieldset>
      <input type="submit" value="File Complaint">
```

```
</form></header><style>/* reset */


#fs-frm input,
#fs-frm select,
#fs-frm textarea,
#fs-frm fieldset,
#fs-frm optgroup,
#fs-frm label,
#fs-frm #card-element:disabled {
  font-family: inherit;
  font-size: 100%;
  background-size:cover;
  color: inherit;
  border: none;
  border-radius: 0;
  display: block;
  width: 100%;
  padding: 0;
  margin: 0;
}
#fs-frm label,
#fs-frm legend,
#fs-frm ::placeholder {
  font-size: .825rem;
  margin-bottom: .5rem;
  padding-top: .2rem;
  display: flex;
```

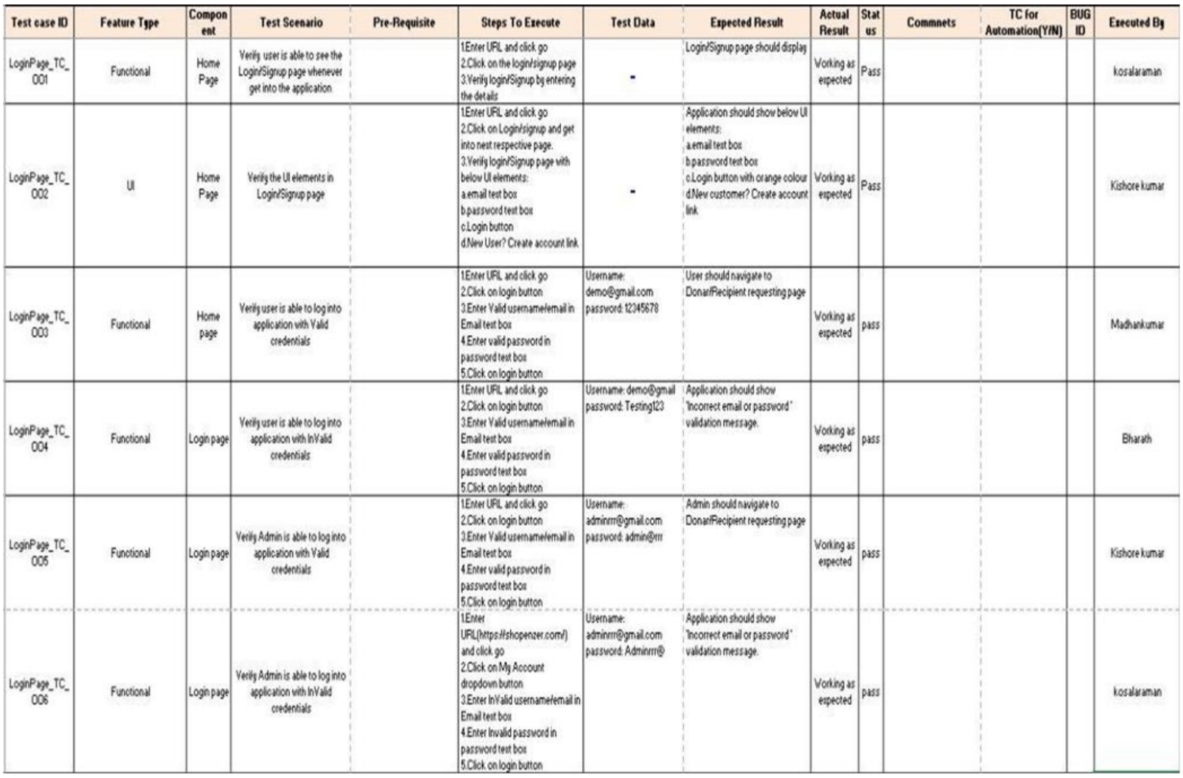
```
        align-items: baseline;
      }
      header{
          background: linear-
gradient(rgba(0,0,0,0.6),rgba(0,0,0,0.6)),url("cover5.jpg");
          height:100vh;
          -webkit-background-size:cover;
          background-size:cover;
          background-position:center center;
          position:relative;
}

      /* border, padding, margin, width */
      #fs-frm input,
      #fs-frm select,
      #fs-frm textarea,
      #fs-frm #card-element {
        border: 1px solid rgba(0,0,0,0.2);
        background-color: rgb(177, 215, 215);
        padding: .75em 1rem;
        margin-bottom: 1.5rem;
      }
      #fs-frm input:focus,
      #fs-frm select:focus,
      #fs-frm textarea:focus {
        background-color: white;
        outline-style: solid;
        outline-width: thin;
        outline-color: gray;
```

```
  outline-offset: -1px;
 }
 #fs-frm [type="text"],
 #fs-frm [type="email"] {
  width: 100%;
 }
 #fs-frm [type="button"],
 #fs-frm [type="submit"],
 #fs-frm [type="reset"] {
  width: auto;
  cursor: pointer;
  -webkit-appearance: button;
  -moz-appearance: button;
  appearance: button;
 }
 #fs-frm [type="button"]:focus,
 #fs-frm [type="submit"]:focus,
 #fs-frm [type="reset"]:focus {
  outline: none;
 }
 #fs-frm [type="submit"],
 #fs-frm [type="reset"] {
  margin-bottom: 0;
 }
 #fs-frm select {
  text-transform: none;
}
 #fs-frm [type="checkbox"] {
  -webkit-appearance: checkbox;
  -moz-appearance: checkbox;
  appearance: checkbox;
  display: inline-block;
  width: auto;
  margin: 0 .5em 0 0 !important;
}
#fs-frm [type="radio"] {
  -webkit-appearance: radio;
  -moz-appearance: radio;
  appearance: radio;
}
/* address, locale */
#fs-frm fieldset.locale input[name="city"],
#fs-frm fieldset.locale select[name="state"],
#fs-frm fieldset.locale input[name="postal-code"] {
  display: inline;
}
#fs-frm fieldset.locale input[name="city"] {
  width: 52%;
}
#fs-frm fieldset.locale select[name="state"],
#fs-frm fieldset.locale input[name="postal-code"] {
  width: 20%;
}
#fs-frm fieldset.locale input[name="city"],
#fs-frm fieldset.locale select[name="state"] {
  margin-right: 3%;
}
</style> </body> </html>
```

CHAPTER-8

TESTING

8.1)Test case:



Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Commnets	TC for Automation(Y/N)	BUG ID	Executed By
LoginPage_TC_001	Functional	Home Page	Verify user is able to see the Login/Signup page whenever get into the application		1.Enter URL and click go 2.Click on the login/signup page 3.Verify login/Signup by entering the details	-	Login/Signup page should display	Working as expected	Pass				kosalaraman
LoginPage_TC_002	UI	Home Page	Verify the UI elements in Login/Signup page		1.Enter URL and click go 2.Click on Login/signup and get into nest respective page. 3.Verify login/Signup page with below UI elements: a.email test box b.password test box c.Login button d.New User? Create account link	-	Application should show below UI elements: a.email test box b.password test box c.Login button with orange colour d.New customer? Create account link	Working as expected	Pass				Kishore kumar
LoginPage_TC_003	Functional	Home page	Verify user is able to log into application with Valid credentials		1.Enter URL and click go 2.Click on login button 3.Enter Valid username/email in Email test box 4.Enter valid password in password test box 5.Click on login button	Username: demo@gmail.com password: 12345678	User should navigate to Donar/Recipient requesting page	Working as expected	pass				Madhankumar
LoginPage_TC_004	Functional	Login page	Verify user is able to log into application with InValid credentials		1.Enter URL and click go 2.Click on login button 3.Enter Valid username/email in Email test box 4.Enter valid password in password test box 5.Click on login button	Username: demo@gmail password: Testing123	Application should show "Incorrect email or password" validation message.	Working as expected	pass				Bharath
LoginPage_TC_005	Functional	Login page	Verify Admin is able to log into application with Valid credentials		1.Enter URL and click go 2.Click on login button 3.Enter Valid username/email in Email test box 4.Enter valid password in password test box 5.Click on login button	Username: adminrrr@gmail.com password: admin@rrr	Admin should navigate to Donar/Recipient requesting page	Working as expected	pass				Kishore kumar
LoginPage_TC_006	Functional	Login page	Verify Admin is able to log into application with InValid credentials		1.Enter URL(https://shopenzer.com/) and click go 2.Click on My Account dropdown button 3.Enter InValid username/email in Email test box 4.Enter Invalid password in password test box 5.Click on login button	Username: adminrrr@gmail.com password: Adminrrr@	Application should show "Incorrect email or password" validation message.	Working as expected	pass				kosalaraman

8.2) USER ACCEPTANCE TEST

1. Purpose of Document

The purpose of this document is to briefly explain the test coverage and open issues of the [ProductName] project at the time of the release to User Acceptance Testing (UAT).

2. Defect Analysis

This report shows the number of resolved or closed bugs at each severity level, and how they were resolved

Resolution	Severity1	Severity2	Severity3	Severity4	Subtotal
By Design	10	4	2	4	20
Duplicate	1	0	1	0	2
External	2	2	1	1	6
Fixed	4	1	1	10	16
Not Reproduced	0	0	0	0	0
Skipped	1	1	0	1	3
Won't Fix	0	2	2	0	4
Totals	18	10	7	16	51

3. Test Case Analysis

This report shows the number of test cases that have passed, failed, and untested.

Section	Total Cases	Not Tested	Fail	Pass
Print Engine	9	0	0	9
Client Application	10	0	0	10
Security	1	0	0	1

Outsource Shipping	0	0	0	0
Exception Reporting	9	0	0	9
Final Report Output	9	0	0	9
Version Control	1	0	0	1

CHAPTER-9

RESULT

9.1) PERFORMANCE METRICS

- **Formal code metrics -** Such as Lines of Code (LOC), code complexity, Instruction Path Length, etc. In modern development environments, these are considered less useful.

- **Developer productivity metrics**—Such as active days, assignment scope, efficiency and code churn. These metrics can help you understand how much time and work developers are investing in a software project.

- **Agile process metrics**—Such as lead time, cycle time and velocity. They measure the progress of a dev team in producing working, shipping-quality software features.

- **Operational metrics**—Such as Mean Time Between Failures (MTBF) and Mean Time to Recover (MTTR). This checks how software is running in production and how effective operations staff are at maintaining it.

- **Test metrics**—Such as code coverage, percent of automated tests, and defects in production. This measures how comprehensively a system is tested, which should be correlated with software quality.

- **Customer satisfaction**—Such as Net Promoter Score (NPS), Customer Effort Score (CES) and Customer Satisfaction Score (CSAT). The ultimate measurement of how customers experience the software and their interaction with the software vendor.

CHAPTER-10

ADVANTAGES AND DISADVANTAGES

Advantages

- **Speed**: This website is fast and offers great accuracy as compared to manual registered keeping.
- **Maintenance**: Less maintenance is required
- **User Friendly**: It is very easy to use and understand. It is easily workable and accessible for everyone.
- **Fast Results**: It would help you to provide plasma donors easily depending upon the availability of it.

Disadvantages

- **Internet**: It would require an internet connection for the working of the website.
- **Auto-Verification**: It cannot automatically verify the genuine users.

CHAPTER-11
CONCLUSION

Inventory management is a very complex but essential part of the supply chain. **An effective inventory management system helps to reduce stock-related costs such as warehousing, carrying, and ordering costs**. **Inventory management** has to do with keeping accurate records of goods that are ready for shipment. In practice, effective **retail inventory management** results in lower costs and a better understanding of sales patterns. The goal of **inventory management** is to understand stock levels and stock's location in warehouses. **Inventory management software helps in save time and cost and helps in managing our daily activities in neat and error free manner.**

CHAPTER-12

FURTURE SCOPE

Upgrading the UI that is more user-friendly which will help many users to access the website and also ensures that many plasma donors can be added into the community.

Using elastic load balancer, it helps to handle multiple requests at the same time which will maintain the uptime of the website with negligible downtime.

CHAPTER-13

APPENDIX

SOURCE CODE:

App.py

```
import datetime
from datetime import datetime
import email
from pickletools import read_unicodestring1
from turtle import st, update
import ibm_db
from flask import Flask, flash, render_template, request, session, url_for, redirect
from flask_mail import Mail, Message
from flask import *

# This is to get the database access from connect.py code
import connect

app = Flask(__name__)
app.secret_key = 'your secret key'
mail = Mail(app)

app.config['MAIL_SERVER']='smtp.gmail.com'
app.config['MAIL_PORT'] = 465
app.config['MAIL_USERNAME'] = 'team06inventory@gmail.com'
app.config['MAIL_PASSWORD'] = 'pjqnmjtrdlqkjqfj'
app.config['MAIL_USE_TLS'] = False
app.config['MAIL_USE_SSL'] = True
mail = Mail(app)

#
itemData={"id":pid,"name":pname,"quantity":quantity,"price":price,"minquanti
ty":minquan}

@app.route("/")
def homepage():
    return render_template("home.html")

@app.route("/login.html")
def loginpage():
    return render_template("login.html")

@app.route("/adminlogin.html", methods = ['GET','POST'])
def adminlogin():
    return render_template("adminlogin.html")

@app.route('/admindata', methods=['POST', 'GET'])
def admin():
    # userdatabase = []
    if request.method == 'POST':
        email = request.form.get('adminemail')
        password = request.form.get('adminpassword')

        sql = "SELECT * FROM ADMIN WHERE EMAIL = ? AND PASSWORD
= ?"
        stmt = ibm_db.prepare(connect.conn,sql)
        ibm_db.bind_param(stmt,1,email)
        ibm_db.bind_param(stmt,2,password)
        ibm_db.execute(stmt)
        account = ibm_db.fetch_assoc(stmt)
        print(account)
        if account:
            return render_template("admin/index.html")
    return render_template("adminlogin.html")

@app.route("/logindata", methods=['GET','POST'])
def login():
    if request.method == 'POST':
        email = request.form.get('email')
        password = request.form.get('password')

        sql = "SELECT * FROM SHOP WHERE EMAIL = ? AND PASSWORD
= ?"
        stmt = ibm_db.prepare(connect.conn,sql)

        ibm_db.bind_param(stmt,1,email)
        ibm_db.bind_param(stmt,2,password)
        ibm_db.execute(stmt)
        account = ibm_db.fetch_assoc(stmt)
        print(account)
        if account:
            return render_template("index.html")
        else:
            return ("Invalid username or password")
    return render_template("login.html")

@app.route("/register.html")
def register():
    return render_template("register.html")

@app.route("/registerdata", methods=['GET','POST'])
def registernew():
    if request.method == 'POST':
        name = request.form['name']
        email = request.form['email']
        password = request.form['pwd']
        mobile = request.form['ph']

        sql = "SELECT * FROM SHOP WHERE EMAIL = ?"
        stmt = ibm_db.prepare(connect.conn,sql)
        ibm_db.bind_param(stmt,1,email)
        ibm_db.execute(stmt)
        account = ibm_db.fetch_assoc(stmt)
        print(account)
        if account:
            msg = "Already existed account! Kindly Login"
            return render_template("login.html")
        else:
            sql = "INSERT INTO SHOP
(NAME,EMAIL,PASSWORD,MOBILENUMBER)
VALUES('{0}','{1}','{2}','{3}')"
            res =
ibm_db.exec_immediate(connect.conn,sql.format(name,email,password,mobile)
)
          mesg = Message(
             'Hello',
             sender ='team06inventory@gmail.com',
             recipients = [email]
            )
       mesg.body = 'Welcome to Shopzy. Thank you for registering with
us.\nHappy Shop(zy)ing!!!.\nLogin id:\n email:'+email+'\nPassword:'+password
       mail.send(mesg)
       msg = "Your account has been registered successfully!l"
       if res:
          return render_template("login.html",msg=msg)

@app.route('/index.html')
def front():
   return render_template("index.html")

@app.route("/products.html")
def dashboard():
   return render_template("products.html")

@app.route("/addproducts.html")
def addprod():
   return render_template("addproducts.html")



@app.route("/addproducts.html",methods = ['POST', 'GET'])
def addproduct():
   if request.method == 'POST':
       pname = request.form['pname']
       quantity = request.form['quantity']
       the_time = datetime.now()
       the_time = the_time.replace(second=0, microsecond=0)
       name =  request.form['name']

       sql = "SELECT * FROM LIST WHERE PRODUCTNAME =?"
       prep_stmt = ibm_db.prepare(connect.conn, sql)
       ibm_db.bind_param(prep_stmt,1,pname)
       ibm_db.execute(prep_stmt)
       product = ibm_db.fetch_assoc(prep_stmt)
       if product:

         if product['PRODUCTNAME']==pname:

          return render_template('addproducts.html', msg="Product already added!
Add a new product.")
       #   else:
       #   sql ="INSERT INTO LIST
(PRODUCTNAME,QUANTITY,DATE,HOLDERNAME) VALUES (?,?,?,?);"
       #     prep_stmt = ibm_db.prepare(connect.conn, sql)
       #     ibm_db.bind_param(prep_stmt, 1, pname)
       #     ibm_db.bind_param(prep_stmt, 2, quantity)
       #     ibm_db.bind_param(prep_stmt, 3, str(the_time))
       #     ibm_db.bind_param(prep_stmt, 4, name)
       #     ibm_db.execute(prep_stmt)
       #     return render_template('addproducts.html', msg="Product added")
       else:
          sql ="INSERT INTO LIST
```

```
(PRODUCTNAME,QUANTITY,DATE,HOLDERNAME) VALUES (?,?,?,?);"
            prep_stmt = ibm_db.prepare(connect.conn, sql)
            ibm_db.bind_param(prep_stmt, 1, pname)
            ibm_db.bind_param(prep_stmt, 2, quantity)
            ibm_db.bind_param(prep_stmt, 3, str(the_time))
            ibm_db.bind_param(prep_stmt, 4, name)
            ibm_db.execute(prep_stmt)
            return render_template('addproducts.html', msg="Product added")
    return render_template("addproducts.html")


# @app.route('/productlist')
# def productlist():
#   if request.method == 'POST':
#         pname = request.form['pname']
#   if session['loggedin'] == True:
#     products = []
#     sql = "SELECT * FROM LIST WHERE PRODUCTNAME = ?"
#     prep_stmt = ibm_db.prepare(connect.conn, sql)
#     ibm_db.bind_param(prep_stmt, 1,pname)
#     ibm_db.execute(prep_stmt)
#     dictionary = ibm_db.fetch_assoc(prep_stmt)
#     while dictionary != False:
#       # print ("The Name is : ",  dictionary)
#       products.append(dictionary)
#       dictionary = ibm_db.fetch_both(prep_stmt)

#     if products:
#       return render_template("list.html", products = products , session = session)
#     else:
#       return render_template("list.html")
#   else:
#     return redirect(url_for('home'))




@app.route('/contact.html')
def contact():
    return render_template("contact.html")

@app.route('/complaint.html')
def compalint():
    return render_template("complaint.html")

@app.route("/complaintdata", methods=['POST', 'GET'])
def complaintdata():
    if request.method == 'POST':
        name = request.form['name']
        mail = request.form['mail']
        complaint =  request.form['complaint']
        sql = "INSERT INTO COMPLAINT (NAME,MAIL,COMPLAINT) VALUES (?,?,?);"
        prep_stmt = ibm_db.prepare(connect.conn, sql)
        ibm_db.bind_param(prep_stmt, 1, name)
        ibm_db.bind_param(prep_stmt, 2, mail)
        ibm_db.bind_param(prep_stmt, 3, complaint)
        ibm_db.execute(prep_stmt)
        # flash("Complaint Sent", "Thank you for contacting us.")
        return render_template('complaint.html', msg = "Complaint Sent. Thank you
for contacting us.")
    return render_template("complaint.html")



# For Admin
@app.route("/updateproducts.html")
def updateprod():
    return render_template("admin/updateproducts.html")



@app.route('/list.html')
def list():
    return render_template("list.html")





@app.route('/contactsupport')
def contactsupport():
  if session['loggedin'] == True:
    return render_template('dashboard/contactsupport.html')
  else:
    return redirect(url_for('home'))

@app.route("/updateproducts",methods = ['POST', 'GET'])
def updateproducts():
    if request.method == 'POST':
        pid = request.form['pid']
        pname = request.form['pname']
        quantity = request.form['quantity']
        minquan = request.form['minquan']
        price =  request.form['price']


        sql = "SELECT * FROM INVENTORY WHERE NAME =?"
        prep_stmt = ibm_db.prepare(connect.conn, sql)
        ibm_db.bind_param(prep_stmt,1,pname)
        ibm_db.execute(prep_stmt)
        product = ibm_db.fetch_assoc(prep_stmt)

itemData={"id":pid,"name":pname,"quantity":quantity,"price":price,"minquanti
ty":minquan}
        if product:

          if product['NAME']==pname:

            return render_template('admin/updateproducts.html', msg="Product
already existed! Add a new product.")



        else:
            sql ="INSERT INTO INVENTORY
(ID,NAME,QUANTITY,MINQUANTITY,PRICE) VALUES (?,?,?,?,?);"
            prep_stmt = ibm_db.prepare(connect.conn, sql)
            ibm_db.bind_param(prep_stmt,1,itemData["id"])
            ibm_db.bind_param(prep_stmt,2,itemData["name"])
            ibm_db.bind_param(prep_stmt,3,itemData["quantity"])
```

```
            ibm_db.bind_param(prep_stmt,5,itemData["minquantity"])
            ibm_db.bind_param(prep_stmt,4,itemData["price"])

            ibm_db.execute(prep_stmt)
            return render_template('admin/updateproducts.html', msg="Product
added")
    sql = "SELECT * FROM INVENTORY WHERE MINQUANTITY <=
QUANTITY"


    stmt = ibm_db.prepare(connect.conn,sql)
    ibm_db.execute(stmt)
    data = ibm_db.fetch_assoc(stmt)

    alertMsg='Following products are to be placed \n'
    if itemData["minquantity"]<=itemData["quantity"]:
        mesg = Message(
                'Hello',
                sender ='team06inventory@gmail.com',
                recipients = [email]
               )
        mesg.body = data
        mail.send(mesg)
        msg = "The following items need to be purchaswed for next day!!\nl"
    return render_template("admin/updateproducts.html")

# scheduler = BlockingScheduler()
# @scheduler.scheduled_job(IntervalTrigger(hours=3))
# def train_model():
```

```
# scheduler.start()



@app.route('/logout')
def logout():
    session.pop('loggedin', None)
    session.pop('id', None)
    session.pop('email', None)
    session.pop('name', None)
    return render_template("home.html")




if __name__ == '__main__':
    app.run(host='0.0.0.0', port=5000, debug=True)
```

Connect.py

```
import ibm_db

def list_all():
    sql = "SELECT * from SHOPZY"
    stmt = ibm_db.exec_immediate(conn, sql)
#     dictionary = ibm_db.fetch_both(stmt)
#     while dictionary !=False:

#         print ("The Name is: ", dictionary["NAME"])
#         print ("The Email is: ", dictionary["EMAIL"])
#         print ("The Password is: \n", dictionary["PASSWORD"])
#         print ("The Mobile no is: ", dictionary["MOBILE NUMBER"])

#         dictionary = ibm_db.fetch_both(stmt)

# def insert_values(name, email, password, mobilenumber ):
#      sql = "INSERT INTO userlogin VALUES('{}','{}','{}','{}')".format(name, email, password, mobilenumber )
#     stmt = ibm_db.exec_immediate(conn,sql)
#     print ("Number of affected rows: ", ibm_db.num_rows(stmt))

try:
    conn = ibm_db.connect("DATABASE=bludb;HOSTNAME=1bbf73c5-d84a-4bb0-85b9-ab1a4348f4a4.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud;PORT=32286;SECURITY=SSL;SSLServerCertificate:DigiCertGlobalRootCA;PROTOCOL=TCPIP;UID=kkm30366;PWD=Fm6dKUmIMCpzpeM0", ", ")
    print("DB is success")


except:
    print("Connection failed")
```

13.2)Github Repository Link: https://github.com/IBM-EPBL/IBM-Project-9745-1659071983