IBM-NALAIYATHIRAN PROJECT

HX8001/ PROFESSIONAL READLINESS FOR INNOVATION, EMPLOYABILITY AN ENTERPRENEURSHIP

INDUSTRY MENTOR	**:**	**VASUDEVA HANUSH**
FACULTY MENTOR	**:**	**VINOTHINI MARY W**

TEAM ID	**:**	**PNT2022TMID50251**
TEAM LEAD	**:**	**RAJESHKUMAR.**
TEAM MEMBER	**:**	**MUTHURAJ.T**
TEAM MEMBER	**:**	**JEYAPAUL.A**
TEAM MEMBER	**:**	**SUBBAIAH.R**

Project Report Format

1. INTRODUCTION

1.1 Project Overview

This project is aimed at developing a web based application named Inventory Management System for retailers to manage the inventory system of any organization. The Inventory Management System (IMS) for retailers refers to the system and processes to manage the stock of organization with the involvement of Technology system. This system can be used to store the details of the inventory, stock maintenance, update the inventory details. Without proper inventory control, a large retail store may run out of stock on an important item.

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

A good inventory management system will alert the wholesaler when it is time to record. Inventory Management System is also on important means of automatically tracking large shipment. An automated Inventory Management System helps to minimize the errors while recording the stock

2. LITERATURE SURVEY

2.1 Existing problem

We started research by identifying the need of IMS in the organization. Initially we bounded our research to find the general reasons that emerged the needs of Inventory Management System. Basically the following factors forced us to develop IMS application:

- Cost and affordability Lack of stock management.
- Effective flow of stock transfer and management.
- Difficulty in monitoring the stock management.

2.2 References

TITLE & AUTHOR	**YEAR**	**TECHNIQUE'S**	**FINDING/PROBLEM**
Inventory management practices among Malaysian micro retailing enterprises. Kamilah ahmad,shafie Mohamad zabri.	October, 2016	Inventory management ,retailing sector,Malaysia, microenterprises, SMEs.	The current state of inventory management practices and factors thatinfluence their usein micro retailing enterprises. A questionnaire survey was employed to gather data from the target respondents.

Retailer and salvage retailer relationship when demand deends on productprice, freshness and displayed inventory level. Prayoga dharma, shi-woei lin.	February -17, yi2017	Retailer, salvage retailer, multivariate demand function, centralized.	The aim of inventory management is to maintain and keep an optimum size of inventory for efficient and smooth production and sales operation. There are improve sales forcasting,managing costomer service,working relation withsuppliers.
Inventory management in supply chain. Anju ajay, Dr.siniV pillai.	January, 2018	Inventory management, supplychain, RFID, inventory turnover.	Using numerical experiments , a comparative analysis of the two alternative is conducted to determine suitable for improving supply chainperformance.
Effects of yield and lead-time uncertainity on retailer-managed and vendor-managed inventory management. Soonkyolee,young joo kim,taesu cheong,seung ho yoo.	December - 19, 2019	lead-time,vendor and retailer managedinventory,decentralized supply chain,optimal production and order quantity,single periodinventory.	This paper aims to model the possible relationship(.,decentralized andcentralized) between retailer and salvage retailer.Zero ending inventory is also boost the sale and profit basedon the demand formulation.

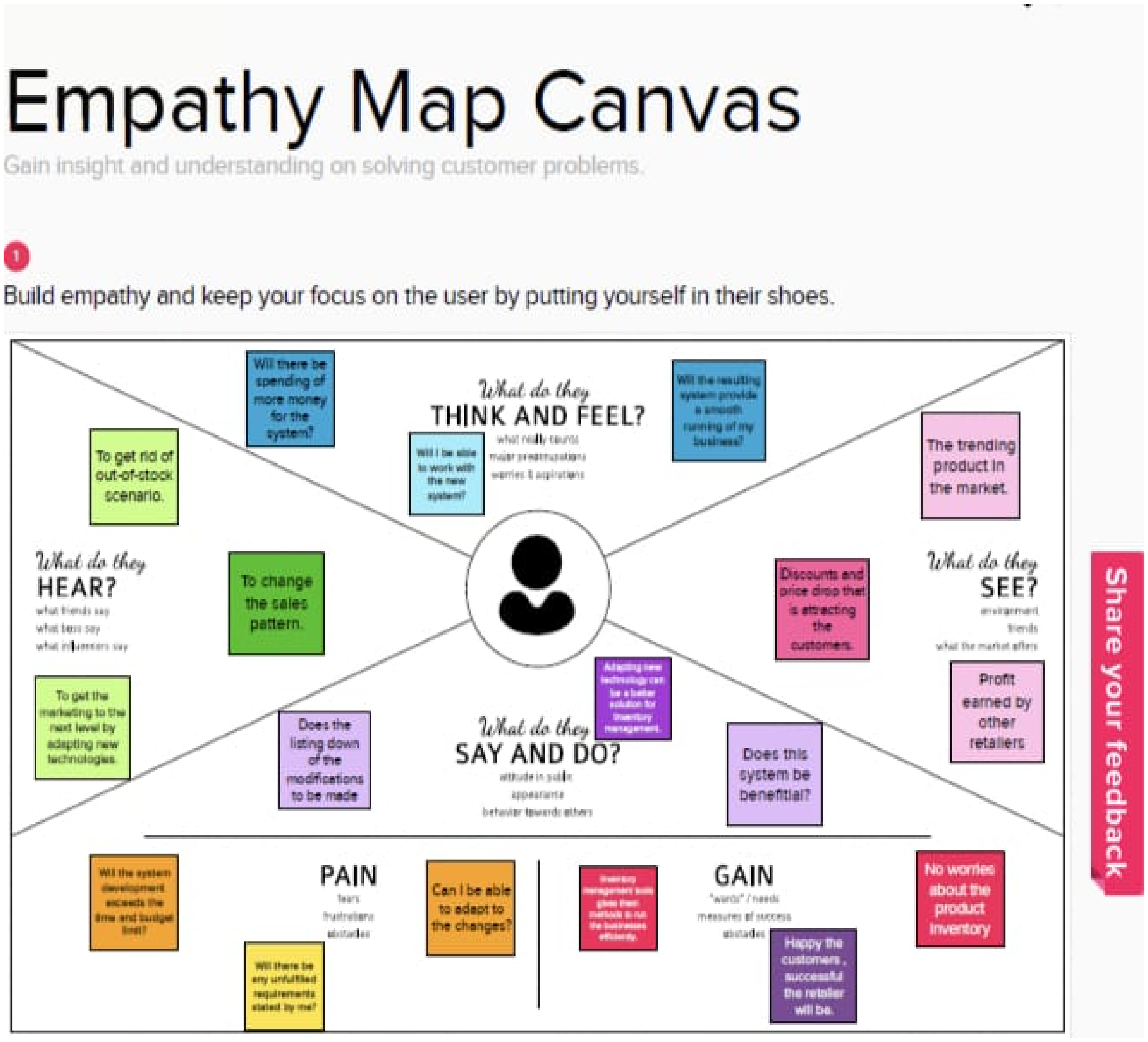
2.3 Problem Statement Definition

After analyzing many existing IMS we have now the obvious vision of the project to be developed. Before we started to build the web application team had many challenges. We defined our problem statement as:

- To make web based application of IMS for small organization.
- To make the system easily managed and can be secured.
- To alert the low stock details and send mail.

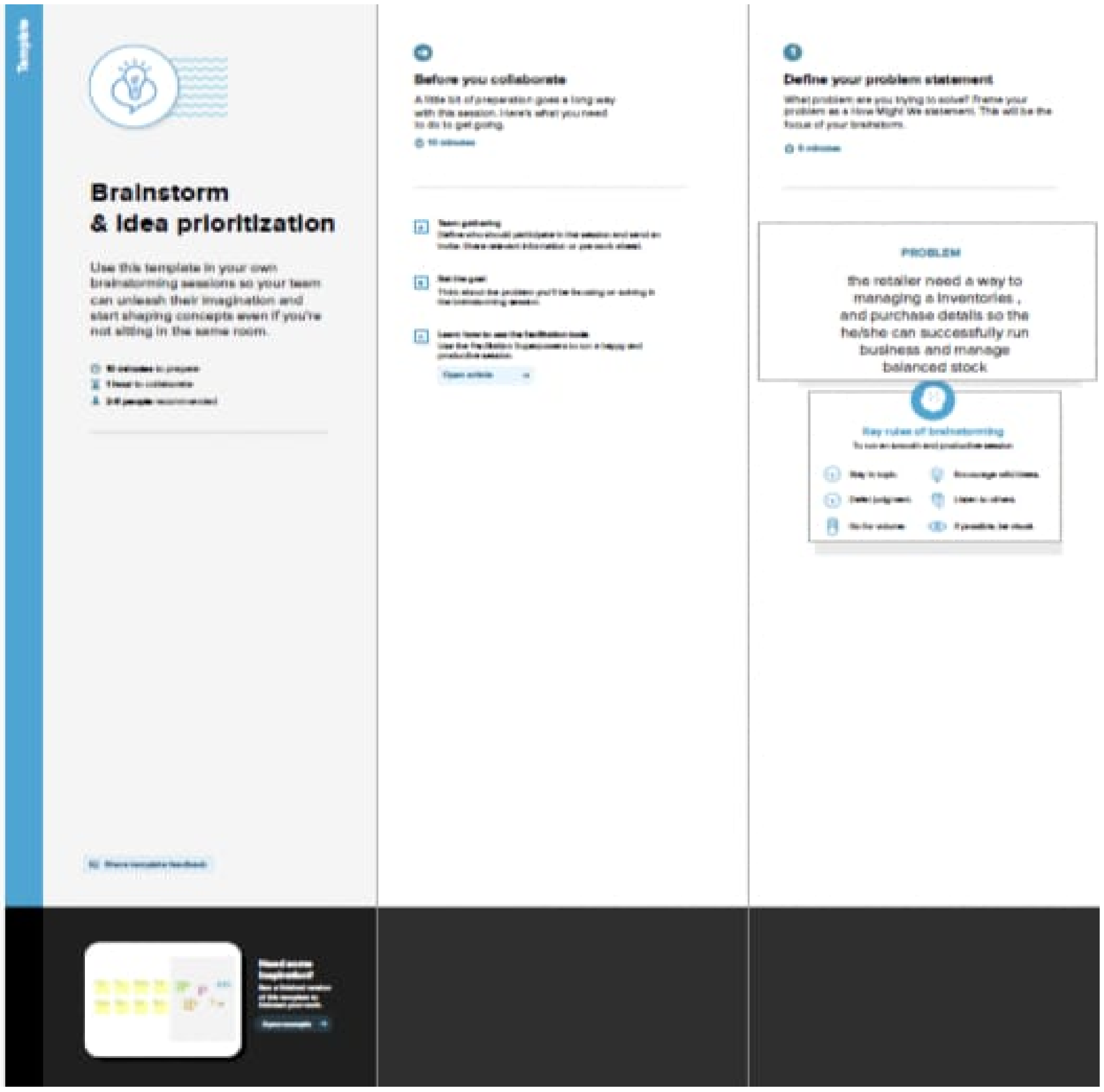
3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas



Empathy Map Canvas
Gain insight and understanding on solving customer problems.
1
Build empathy and keep your focus on the user by putting yourself in their shoes.
What do they THINK AND FEEL?
what really counts
major preoccupations
worries & aspirations
Will there be spending of more money for the system?
Will I be able to work with the new system?
Will the resulting system provide a smooth running of my business?
What do they HEAR?
what friends say
what boss say
what influencers say
To get rid of out-of-stock scenario.
To change the sales pattern.
To get the marketing to the next level by adapting new technologies.
What do they SEE?
environment
friends
what the market offers
The trending product in the market.
Discounts and price drop that is attracting the customers.
Profit earned by other retailers
What do they SAY AND DO?
attitude in public
appearance
behavior towards others
Does the listing down of the modifications to be made
Adapting new technology can be a better solution for inventory management.
Does this system be benefitial?
PAIN
fears
frustrations
obstacles
Will the system development exceeds the time and budget limit?
Will there be any unfulfilled requirements stated by me?
Can I be able to adapt to the changes?
GAIN
"wants" / needs
measures of success
obstacles
Inventory management tools gives them methods to run the businesses efficiently.
Happy the customers , successful the retailer will be.
No worries about the product inventory
Share your feedback

3.2 Ideation & Brainstorming



Template

Brainstorm & idea prioritization

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

10 minutes to prepare
1 hour to collaborate
2-8 people recommended

Share template feedback

Need some inspiration?
See a finished version of this template to kickstart your work.
Open example

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

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

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

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

Open article

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.

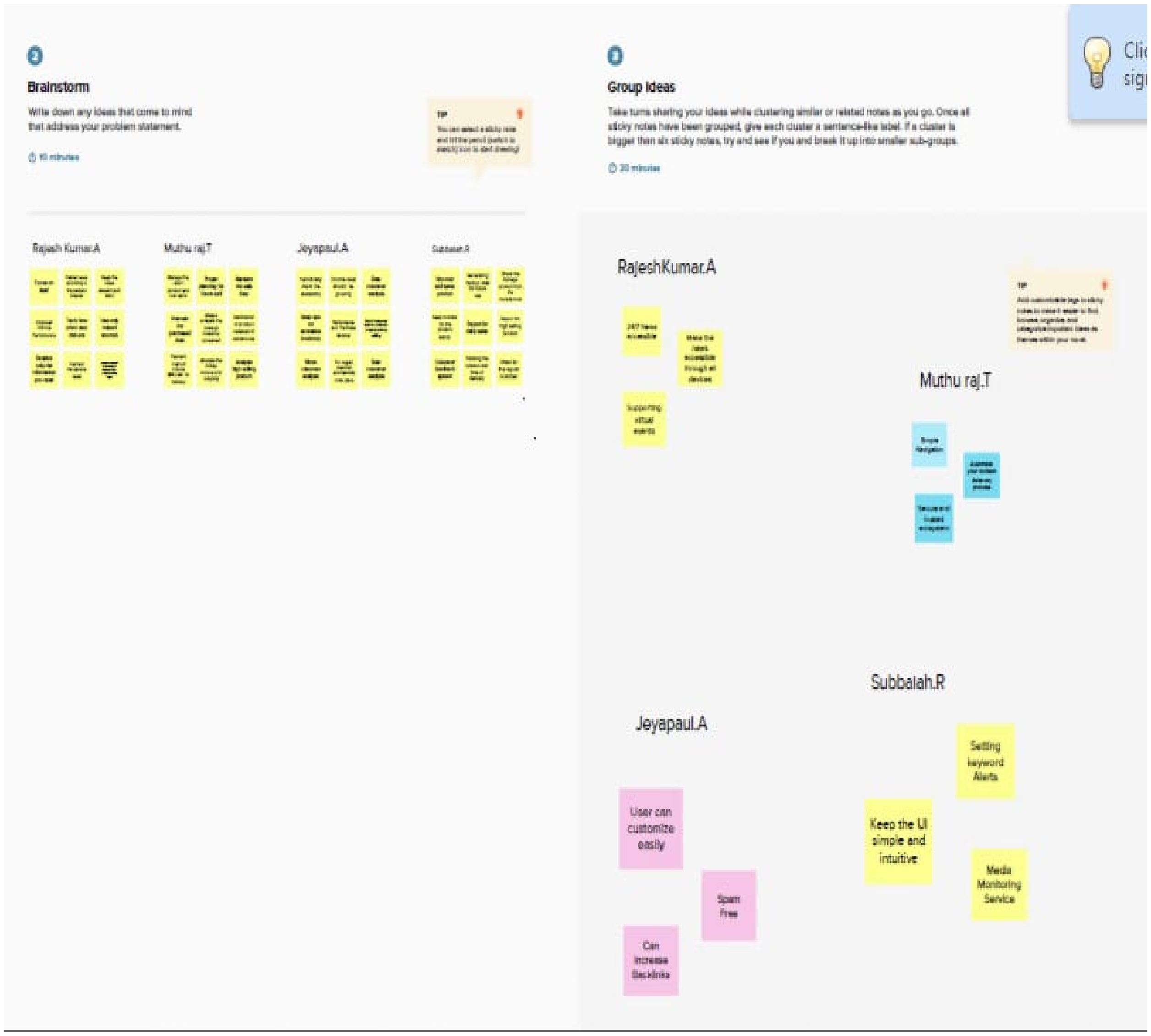
5 minutes

PROBLEM

the retailer need a way to managing a inventories , and purchase details so the he/she can successfully run business and manage balanced stock

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.



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!

Rajesh Kumar.A

Muthu raj.T

Jeyapaul.A

Subbaiah.R

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

RajeshKumar.A

24/7 News accessible

Make the news accessible through all devices

Supporting virtual events

TIP

Add customizable tags to sticky notes to make it easier to find, browse, organize, and categorize important ideas as themes within your mural.

Muthu raj.T

Simple Navigation

Automate your content delivery process

Secure and trusted ecosystem

Jeyapaul.A

User can customize easily

Spam Free

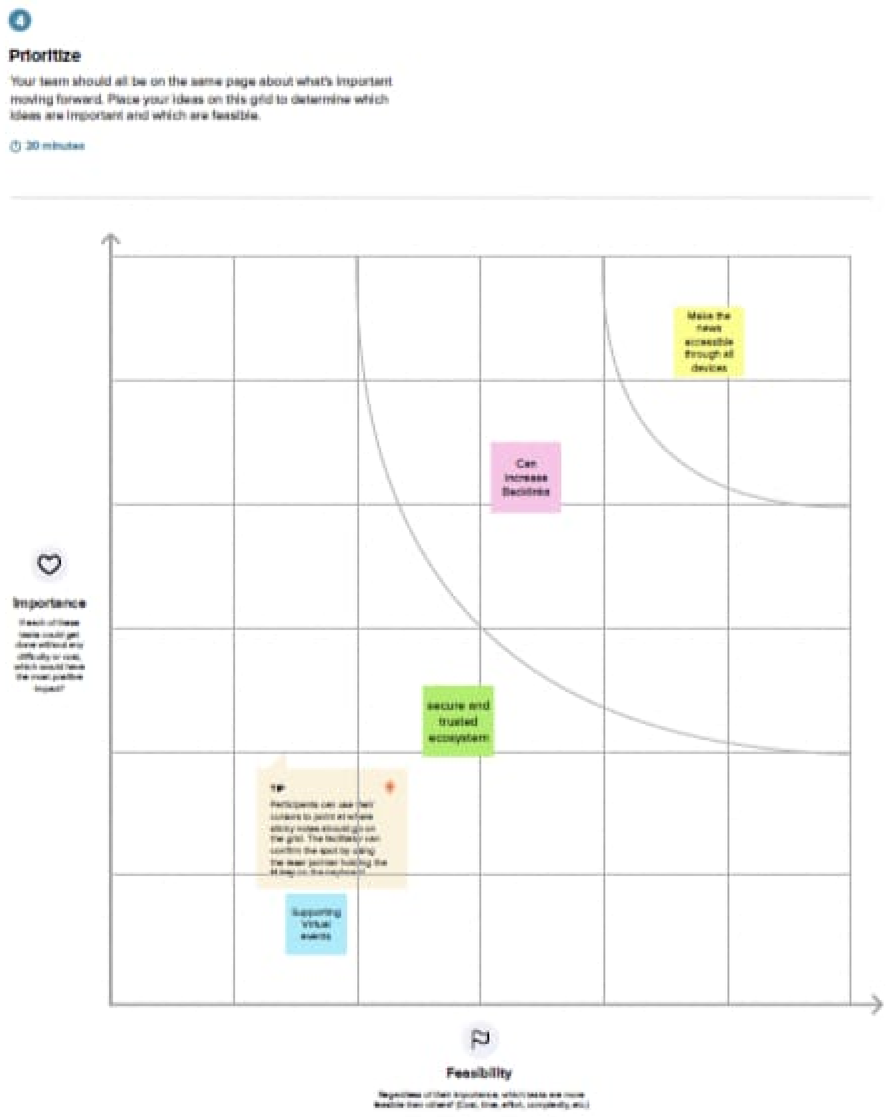
Can increase Backlinks

Subbaiah.R

Setting keyword Alerts

Keep the UI simple and intuitive

Media Monitoring Service



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

Make the news accessible through all devices
Can increase Backlinks
secure and trusted ecosystem
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.
Supporting Virtual events
Importance
If each of these tasks could get done without any difficulty or cost, which would have the most positive impact?
Feasibility
Regardless of their importance, which tasks are more feasible than others? (Cost, time, effort, complexity, etc.)

3.3 Define the problemstatement

Date	8 september 2022
Team ID	PNT2022TMID50251
Project name	Inventory management system for retailers
Maximum mark	2 Marks

I am	I’m trying to	But	Because	Which makes me feel
A retailer	**Find the productdetails**	**It’s hard anddifficult**	**It takes longtime**	**Feel irritated and frustrated on continuous validation**

Problem statement(PS)	**I am (customer)**	**I’m tryingto**	**But**	**Because**	**Which makes me feel**
PS-1	Retailer	Find theproduct counts in the stock	It’s hard	It takes more time	Tired
PS-2	Retailer	Calculate the bill for transportation purpose	It’s hard	It take long time	Uninterested
PS-3	Retailer	Find the customer ’s Review	It’s hard to gathering information	I don’t have enough contacts	Disappointed
PS -4	Retailer	Maintain the Ledger	It’s difficult to secure	It may be lost	Afraid
PS-5	Retailer	Find the high Demand	It’s difficult to calculate	It takes more	Challenging

3.4 Proposed Solution

Date	22 September 2022
Team ID	PNT2022TMID50251
Project Name	Inventory Management System for Retailers
Maximum Marks	2 Marks

Proposed Solution Template:

S.No.	Parameter	Descrption
1	Problem Statement (Problem to be solved)	Retailers who run their business with large scale or small scale stocks.Itis crucial for an organization today to understand its inventory toachieve both efficient and fast operations, that too, at an affordable cost.Lack of the right inventory at the right time can mean back orders, excess inventory, etc. These drive up costs. Late delivery due tostock-outs is bound to give you a bad reputation. Inaccurate calculations of stock and price. Late deliveries are due to lateplanning. Poor tracking may lead to back orders.Overstocking of discounted products and neglecting the trends of seasonal sales may result in excess inventory.Therefore considering the economic crisis of theretailers and to reduce the manpower efficiently while handling data, it is very important to have a best inventorymanagement system for retailers.

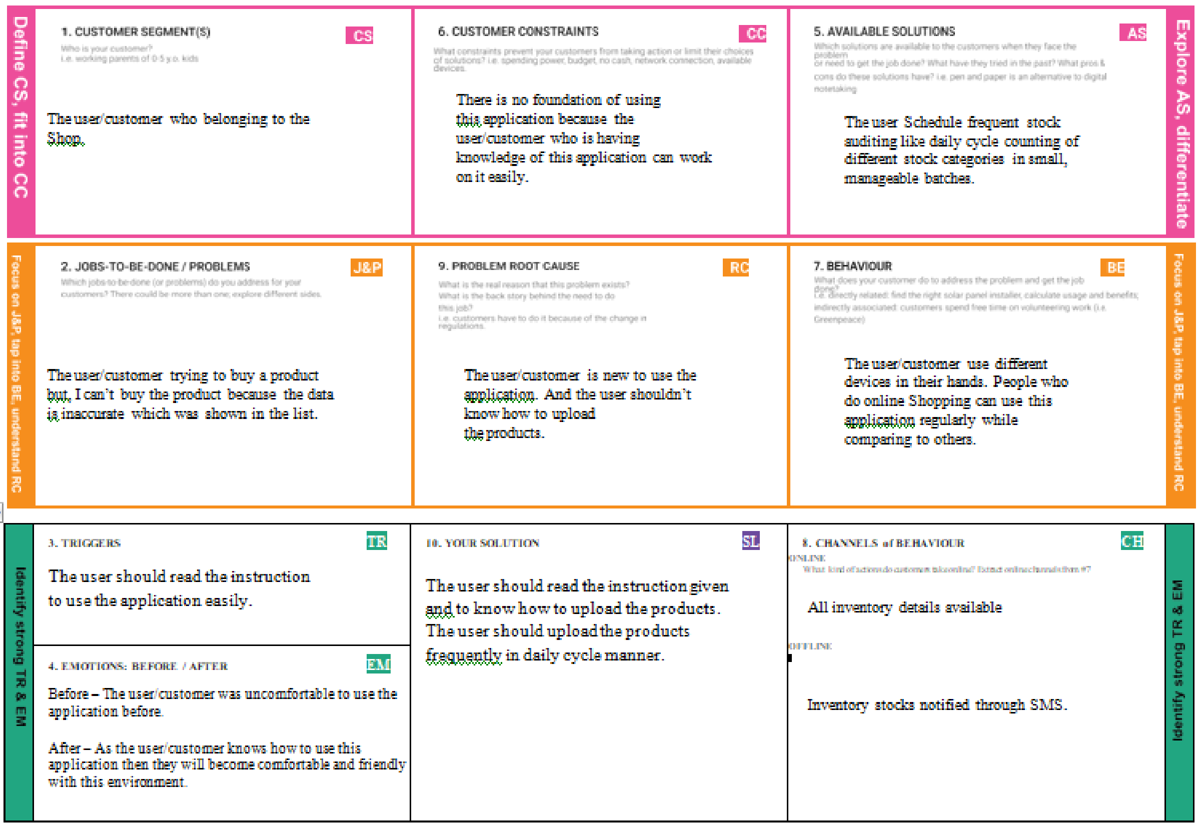
2	Idea / Solution Description	Applications have been developed to help retailers track and manage stocks related to their own products. The System will ask retailers to create their accounts by providing essential details. Retailers can access their accounts by logging into theapplication. Once retailers 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 retailersif there is no stock found in their accounts. So that they canorder new stock.
3	Novelty / Uniqueness	User can track the record of goods available using the application. Inventory tracking helps to improve inventory management and ensures.
4	Social Impact / Customer Satisfaction	Customer satisfaction is the key for success of a business.The availability of product is just one way in whichan inventory management system creates customer satisfaction. Inventory management systems are designed to monitor product availability,determine purchasing schedules for better customer interaction.
5	Business Modern evenue Model)	the inventory management system has seperate on two types. there are, meets consumer demands and increases sale.it will maintain on management for inventory and tracking the inventory.
6	Scalability of the Solution	Scalability is an aspect or rather a functional quality of a system, software or solution.This proposed system for inventory management system can accommodate expansionwithout restricting the existing workflow and ensure an increase in the output or efficiency of the process.

3.5 Problem Solution fit:

Project Title: Inventory Management System for Retailers

Project Design Phase-I –Problem- Solution Fit Template

Team ID: PNT2022TMID50120



Define CS, fit into CC

1. CUSTOMER SEGMENT(S) CS

Who is your customer?
i.e. working parents of 0-5 y.o. kids

The user/customer who belonging to the Shop.

6. CUSTOMER CONSTRAINTS CC

What constraints prevent your customers from taking action or limit their choices of solutions? i.e. spending power, budget, no cash, network connection, available devices.

There is no foundation of using this application because the user/customer who is having knowledge of this application can work on it easily.

5. AVAILABLE SOLUTIONS AS

Which solutions are available to the customers when they face the problem
or need to get the job done? What have they tried in the past? What pros & cons do these solutions have? i.e. pen and paper is an alternative to digital notetaking

The user Schedule frequent stock auditing like daily cycle counting of different stock categories in small, manageable batches.

Explore AS, differentiate

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

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

Which jobs-to-be-done (or problems) do you address for your customers? There could be more than one; explore different sides.

The user/customer trying to buy a product but, I can't buy the product because the data is inaccurate which was shown in the list.

9. PROBLEM ROOT CAUSE RC

What is the real reason that this problem exists?
What is the back story behind the need to do this job?
i.e. customers have to do it because of the change in regulations.

The user/customer is new to use the application. And the user shouldn't know how to upload the products.

7. BEHAVIOUR BE

What does your customer do to address the problem and get the job done?
i.e. directly related: find the right solar panel installer, calculate usage and benefits; indirectly associated: customers spend free time on volunteering work (i.e. Greenpeace)

The user/customer use different devices in their hands. People who do online Shopping can use this application regularly while comparing to others.

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

Identify strong TR & EM

3. TRIGGERS TR

The user should read the instruction to use the application easily.

4. EMOTIONS: BEFORE / AFTER EM

Before – The user/customer was uncomfortable to use the application before.

After – As the user/customer knows how to use this application then they will become comfortable and friendly with this environment.

10. YOUR SOLUTION SL

The user should read the instruction given and to know how to upload the products. The user should upload the products frequently in daily cycle manner.

8. CHANNELS of BEHAVIOUR CH

ONLINE

What kind of actions do customers take online? Extract online channels from #7

All inventory details available

OFFLINE

Inventory stocks notified through SMS.

Identify strong TR & EM

4. REQUIREMENT ANALYSIS

4.1 Functional requirement

Create UI to interact with the application

- Registration Page
- Login Page
- Display items in the Dashboard
- Adding items
- Test it

User requirement are categorized by the user type.

Admin

- Able to create new inventory item.
- Able to edit the entry as per entry.
- Able to add, modify and delete the stock entry.

Inventory management

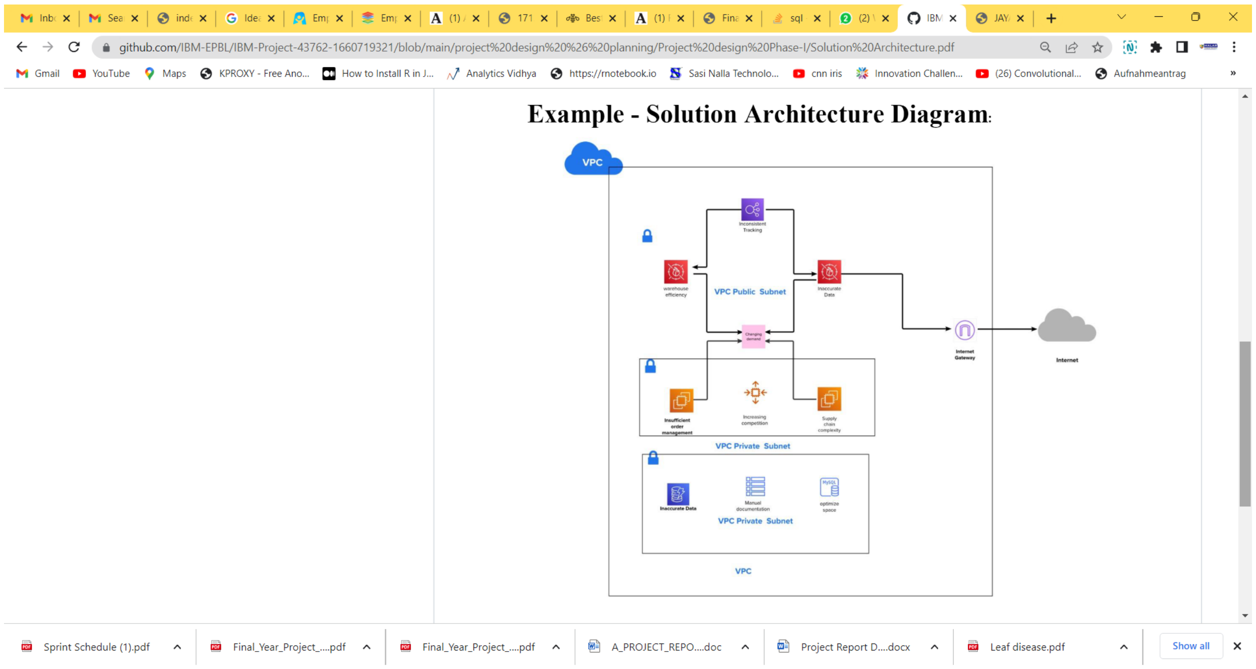
- Able to check the stock available.
- Able to send Email when the quantity of inventory item meets low stock level.

4.2 Non-Functional requirements

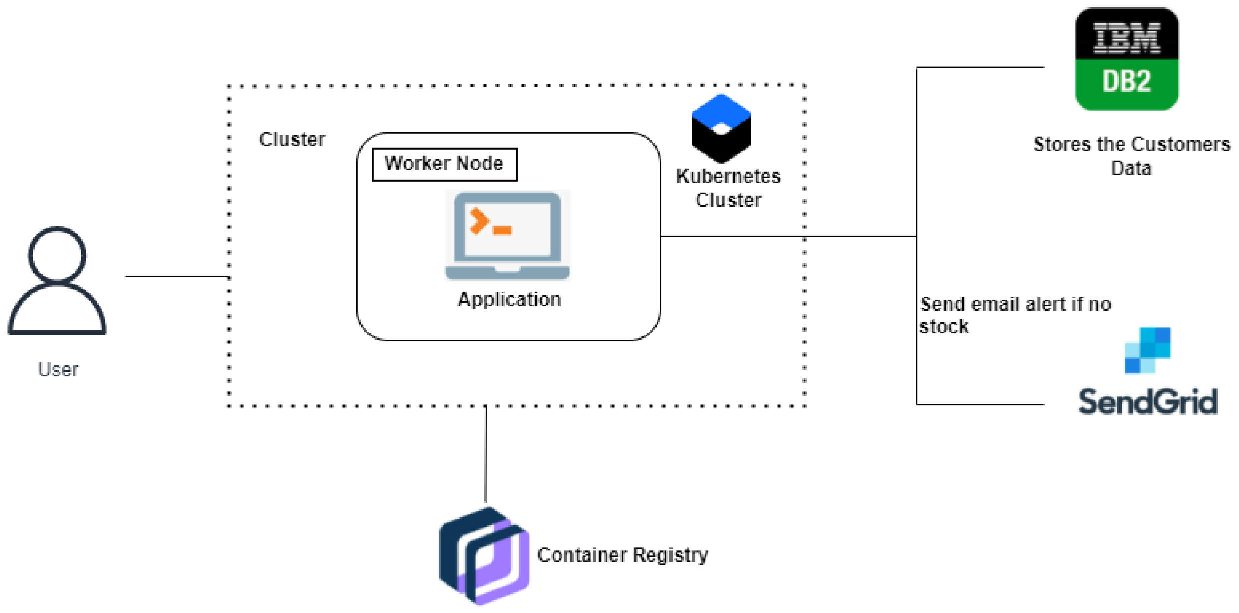
NFR No.	**Non-Functional Requirements**	**Description**
NFR-1	Usability	The system uses a web browser as an interface, which all users are familier about and no specific training is required.
NFR-2	Security	Every data specific to user could be accessed only by the respective user as every login activity is authendicated and authorized.
NFR-3	Reliability	The user should be able to access the correct data at all times.
NFR-4	Performance	The system should not able to take a longer time to send a response to the user that is in need of and the resources should be allocated accordingly for different tasks such as the visualization can take more time but ehere ae registering a sale/updating the inventory system.
NFR-5	Availability	The system should be accessible at all times-24/7 when the users aren't notified about the server maintanence.
NFR-6	Scalability	The system should be able to accept any kind of new changes in the near future such as increase in the user could throughput of data ,extending it to hand-held devices.

5. PROJECT DESIGN

5.1 Data Flow Diagrams



5.2 Solution & Technical Architecture



5.3 User Stories

Sprint	Functional Requirements	User Story Number	User Story/Task	Story Points	Priority	Team Meambers
sprint 1	Retailers	USN-1	The retailers can enter the registration on our store details and owner details enter with login.	20	high	A.rajesh kuamar T.muthu raj R.subbaiah A.jeyapaul
sprint 2	Inventory	USN-2	Once the retailers successfully login in to the application they can update their inventory details also user will be able to add new stock by submitting essential related to the stock.	20	high	A.rajesh kumar t.muthu raj A.jeyapaul
sprint 3	E-mail	USN-3	The system will automatically send an email alert to the retailers if there is no stock found in their account.	20	high	A.rajesh kumar T.muthu raj R.subbaiah A.jeyapaul
sprint 4	Final Delivery	USN-3	Container of the application using docker, kubernetes and deployment of the application create the documentation and final submit the application.	20	high	A.rajesh kumar T.muthu raj R.subbaiah A.jeyapaul

6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

sprint 1:

1.We create a Flask Project.

2.Added all routers needed for our project.

3.Created table in IBM cloud.

sprint 2:

1.We added all the html templates needed for our project

2.We style those pages using CSS and bootstrap.

3.We wrote queries to connect IBM cloud database.

4.Finished all the fetching and posting stuff of IBM cloud database integration.

sprint 3:

1.Inegration of send grid into our application

sprint 4:

1.Deploying the application using Docker and Kubernetes.

6.2 Sprint Delivery Schedule

Sprint	Functional Requirements (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.**	**2**	**high**	**A.rajesh kumar T.muthu raj R.subbaiah A.jeyapaul**
Sprint-1		**USN-2**	**As a user ,I can register for the application through E-mail.**	**1**	**Medium**	**A.rajesh kumar T.muthu raj R.subbaiah A.jeyapaul**
Sprint-1	**Confirmation**	**USN-3**	**As a user ,I will receive confirmation email once I have registered for the application.**	**2**	**Medium**	**A.rajesh kumar T.muthu raj R.subbaiah A.jeyapaul**
Sprint-1	**Login**	**USN-4**	**As a user ,I can login to the application by entering email & password.**	**2**	**High**	**A.rajesh kumar T.muthu raj A.jeyapaul**
Sprint-2	**Dashboard**	**USN-5**	**As a user ,I can view product which are available.**	**4**	**High**	**A.rajesh kumar T.muthu raj R.subbaiah A.jeyapaul**

Sprint-3	**E-mail**	**USN-6**	**As a user, I will check on alert message to the no stock found in their account.**	**5**	**Medium**	**A.rajesh kumar T.muthu raj R.subbaiah A.jeyapaul**
Sprint-3	**Stock Update**	**USN-7**	**As a user,I can which are not available in the dashboard to stock limit.**	**5**	**Medium**	**A.rajesh kumar T.muthu raj R.subbaiah A.jeyapaul**
Sprint-4	**Contact Administrator**	**USN-8**	**I can be able to report any difficulties I experience as a report.**	**5**	**Medium**	**A.rajesh kumar T.muthu raj R.subbaiah A.jeyapaul**

Sprint	**Total Story Points**	**Duration**	**Sprint Start**	**Sprint End Date(Planned)**	**Story Points Completed**	**Sprint Release Date**
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**	**12 nov-2022**	**20**	**12 Nov-2022**
Sprint-4	**20**	**6 days**	**14 Nov-2022**	**19 Nov-2022**	**20**	**19 Nov-2022**

AV = sprint duration/velocity = 20/10 = 2.

6.3 Reports from JIRA

IT Organization have the challenges of ensuring system uptime, supporting users, and managing inventory of both hardware and software. IT teams gain significant efficiencies when one tool can support multiple business operation. According tq gather, mastering scipline of effective asset management is a huge cost savings for companies.

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

7.1 Feature 1

Flask Framework is added.

7.2 Feature 2

Send Mail using SendGrid

We recommend using SendGrid Python, our client library, available on G..

We recommend using SendGrid Python, our client library, available on GitHub, with full documentation...

https://docs.sendgrid.com/for-developers/sending-email/v3-python-code-example

7.3 Database Schema (if Applicable)

DB2 is used as database.

There are various ways of accessing databases such as JDBC, JavaScript..

There are various ways of accessing databases such as JDBC, JavaScript, JSP, Python and many others. Here, we will be specifically talking…..

https://medium.com/mozilla-firefox-club/accessing-ibm-db2-database-using-python-c356a4a76bf3

8. TESTING

8.1 Test Cases

Testing can be verification and validation or reliability estimation. The primary objective if testing includes:

- To identifying defects in the application.
- The most important role of testing is simply to provide information.
- to check the proper working of the application while inserting updating and

- deleting the entry of the products.

8.2 User Acceptance Testing

User Acceptance Testing (UAT) is a type of testing performed by the end user or the client to verify/accept the software system before moving the software application to the production environment. UAT is done in the final phase of testing after functional, integration and system testing is done.

9. RESULTS

9.1 Performance Metrics

Inventory performance is a measure of how effectively and efficiently inventory is used and replenished. The goal of inventory performance metrices is to compare actual on-hand dollars versus forecast cost of good sold. Many learn practitioners claim that inventory performance is the single best indicators of the overall operational performance of a facility.

Inventory performance looks at and is measured using either inventory Days On-Hand (DOH) OR INVENTORY TURNS.

1.Inventory Days On-Hand:

The number of days it would take to consume curent on-hand inventory. Always multiple invenory item numbers in the terms of currency (ie., COGS).

2.Inventory Turns:

The number of times inventory is replaced in a year.

10. ADVANTAGES & DISADVANTAGES

10.1 Advantages

- Used for small organization
- Low stock alert as email

10.2 Disadvantages

- This application is not suitable for those organization where there is large quantity of product and different level of warehouses.
- This software application is able to generate only simple reports.
- Single admin panel is only made.
- It is not suitable for large organization.

11. CONCLUSION

To conclude, Inventory Management System for retailers is a simple web based application basically suitable for small organization. It has every basic items which are used for the small organization. Our team is successful in making the application where we can update, insert and delete the item as per the requirement. This application also sends the email alert when low stock level meets. This application matches for small organization. Through it has some limitations, our team strongly believes that the implementation of this system will surely benefit the organization.

12. FUTURE SCOPE

Since this project was started with very little knowledge about the Inventory Management System, we came to know about the enhancement capability during the process of building it. Some of the scope we can increase for the betterment and effectiveness oar listed below:

- Interactive user interface design.
- Manage Stock Godown wise.
- Lost and breakage

13. APPENDIX

13.1 Source Code

```
# This is a sample Python script.

# Press Shift+F10 to execute it or replace it with your code.

 # Press Double Shift to search everywhere for classes, files, tool windows, actions, and settings.


import ibm_db

from flask import Flask, render_template, request, redirect, url_for, flash, session

from inventory.Vendor import Vendor

from inventory.Inventory import Inventory

import sendgrid

import os

from sendgrid.helpers.mail import *


 app = Flask(__name__)

 app.secret_key = b'_5#y2L"F4Q8z\n\xec]/'



@app.route("/")

def show_login():

    return redirect(url_for('login'))



 @app.route("/vendor/signup", methods=['GET', 'POST'])

def vendor_signup():

    if request.method == 'POST':

        vendor = Vendor()

        vendor.Id = ""

        vendor.Name = request.form['name']

        vendor.Shop_Name = request.form['shop_name']
        vendor.GST = request.form['gst']
        vendor.Mobile = request.form['mobile']
        vendor.Address = request.form['address']
        vendor.Email = request.form['email']
        vendor.Password = request.form['password']
        vendor.save()

        flash(u'Vendor Sign up done, you login now with your username and password.',
'success')

        return redirect(url_for('login'))
    else:
        return render_template('register_vendor.html')


 @app.route("/login", methods=['GET', 'POST'])
def login():
    if request.method == 'POST':
        if request.form['username'] != "" and request.form['password'] != "":
            vendor = Vendor()
            vendor.Email = request.form['username']
            vendor.Password = request.form['password']
            result = vendor.login()

            print(result)
            if len(result) > 0:
                session['name'] = result[0]['NAME']
                session['vendor_id'] = result[0]['ID']
                email_low_stock_alert(session['vendor_id'])
                return redirect(url_for('dashboard'))
            else:
                flash(u'username or password is incorrect.', 'danger')
                return redirect(url_for('login'))
    else:
        return render_template('login.html')


@app.route("/dashboard", methods=['GET'])
def dashboard():
    if session['name'] is None:
        return redirect(url_for('login'))

    inventory = Inventory()
    inventory = inventory.display()
    return render_template('dashboard.html')


@app.route("/inventory", methods=['GET'])
def view_inventory():
    if session['name'] is None:
        return redirect(url_for('login'))

    inventory = Inventory()
    inventory.VendorId = session['vendor_id']
    inventory = inventory.display()

    print(inventory)
    return render_template('view_inventory.html', inventory=inventory)
```

```
 @app.route("/low_inventory", methods=['GET'])
def low_inventory():
    if session['name'] is None:
        return redirect(url_for('login'))

    inventory = Inventory()
    inventory.VendorId = session['vendor_id']
    inventory = inventory.get_low_stock()
    return render_template('low_inventory.html', inventory=inventory)


 @app.route("/inventory/new", methods=['GET'])
def inventory_new():
    if session['name'] is None:
        return redirect(url_for('login'))
    inventory = Inventory()
    return render_template('inventory_item.html', item=inventory)


 @app.route("/inventory/edit/<int:id>", methods=['GET'])
def inventory_edit(id):
    if session['name'] is None:
        return redirect(url_for('login'))

    inventory = Inventory()
    inventory.VendorId = session['vendor_id']
    inventory = inventory.get(id)

    if len(inventory) > 0:
        item = inventory[0]
        return render_template('inventory_item.html', item=item)
    else:
        flash(u'Inventory item is not found with id = ' + str(id), 'danger')
        return render_template('inventory_item.html', item=inventory)


@app.route("/inventory/save", methods=['POST'])
def inventory_save():
    if session['name'] is None:
        return redirect(url_for('login'))

    inventory = Inventory()
    if request.form['id'] != "":
        inventory.Id = request.form['id']
    inventory.VendorId = session['vendor_id']
    inventory.Category = request.form['category']
    inventory.ItemName = request.form['item_name']
    inventory.Wholesaleprice = request.form['wholesale_price']
    inventory.Retailprice = request.form['retail_price']
    inventory.Qty = request.form['qty']
    inventory.Low_Stock_Limit = request.form['low_stock_limit']
    inventory.LotNo = request.form['lot_no']
    inventory.Note = request.form['note']
    inventory.save()

    flash(u'Inventory has been saved successfully.', 'success')
    return redirect(url_for('view_inventory'))


@app.route('/logout')
```

```
def logout():
    session.clear()
    return redirect(url_for('login'))



def email_low_stock_alert(vendor_id):
    if session['name'] is None:
        return redirect(url_for('login'))

    vendor = Vendor()
    vendors = vendor.get(vendor_id)
    vendor = vendors[0]

    inv = Inventory()
    inventory = inv.get_low_stock()

    if len(inventory) > 0:
        sg =
sendgrid.SendGridAPIClient(api_key="SG.PEMDvdpVSeqVl9BCQP5xjw.KSZztqZz5nx291
w0SmyXvug_nrTm5HpelEMCSkFj4Cs")
        from_email = Email("rajesh@malaris.com")
        to_email = To(vendor["EMAIL"])
        subject = "Vendor Low Stock Notification"
        content = Content("text/html", render_template('email_low_stock.html',
inventory=inventory, vendor=vendor))
        mail = Mail(from_email, to_email, subject, content)
        response = sg.client.mail.send.post(request_body=mail.get())
        print(response.status_code)
        print(response.body)
        print(response.headers)
```

```python
if __name__ == "__main__":
    port = int(os.environ.get('PORT', 5000))
    app.run(debug=True, host='0.0.0.0', port=port)
```

login.html

```html
 <html lang = "en">
   <head>
      <meta charset = "utf-8">
      <meta name = "viewport" content = "width = device-width, initial-scale = 1, shrink-to-fit = no">
      <link rel = "stylesheet"
         href = "https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css"
         integrity = "sha384-
MCw98/SFnGE8fJT3GXwEOngsV7Zt27NXFoaoApmYm81iuXoPkFOJwJ8ERdknLPMO"
         crossorigin = "anonymous">
      <title>Login Form </title>

   </head>
   <body>
   <form method="post" action="/login">
<section class="vh-100" style="background-color: #2a9c85;">
  <div class="container h-100">
    <div class="row d-flex justify-content-center align-items-center h-100">
      <div class="col-xl-9">
         <div class="container">

{% with messages = get_flashed_messages(with_categories=true) %}
{% if messages %}
{% for category, message in messages %}
  <div class="flashes alert alert-{{category}}">
    <strong>{{ message }}</strong>
  </div>
{% endfor %}

{% endif %}
{% endwith %}

        <h1 class="text-white mb-4"> Login Form </h1>
        <div class="card" style="border-radius: 15px;">
          <div class="card-body">
            <div class="row align-items-center pt-4 pb-3">
              <div class="col-md-3 ps-5">
                <h6 class="mb-0"> UserName </h6>
              </div>
              <div class="col-md-9 pe-5">
                <input type="email" class="form-control form-control-lg" placeholder="Enter
Email/UserName" name="username" />
              </div>
            </div>
<hr class="mx-n3">
            <div class="row align-items-center py-3">
              <div class="col-md-3 ps-5">
                <h6 class="mb-0"> Password </h6>
              </div>
              <div class="col-md-9 pe-5">
                <input type="password" class="form-control form-control-lg" placeholder="Enter
Password" name="password" />
              </div>
            </div>
<button type="reset" class="btn btn-primary btn-lg"> Cancel</button>


              <button type="submit" class="btn btn-primary btn-lg"> Save </button>
 </div>
            <div class="text-center">
                <a href="/vendor/signup">dont you have account? vendor sign up here?</a><br><br>
            </div>

        </div>
      </div>
    </div>
  </div>
</section>
  </form>
</body>
</html>
```

inventory.html

```
{% extends "base_template.html" %}

{% block title %}Add Inventory{% endblock %}

{% block content %}

<section class="vh-100">

  <div class="container">

      <div class="row">

          <div class="col-md-6">

              <h4 class="pull-left">Add Inventory</h4>

          </div>

          <div class="col-md-6 d-flex flex-row-reverse">

          </div>

      </div>

      <form method="post" action="/inventory/save">

          <input type="hidden" name="id" value="{{item['ID']}}">

      <div class="card" style="border-radius: 15px;">

          <div class="card-body">

            <div class="row align-items-center pt-4 pb-3">

              <div class="col-md-3 ps-5">

                <h6 class="mb-0"> Category </h6>
```

```
                  </div>

                  <div class="col-md-9 pe-5">

                    <input type="text" class="form-control form-control-lg" placeholder="Enter
Category" name="category"  value="{{item['CATEGORY']}}"/>

                  </div>

                </div>

      <hr class="mx-n3">

                <div class="row align-items-center py-3">

                  <div class="col-md-3 ps-5">

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

                  </div>

                  <div class="col-md-9 pe-5">

                    <input type="text" class="form-control form-control-lg" placeholder="Enter  Item
Name" name="item_name"  value="{{item['ITEMNAME']}}"/>

                  </div>

                </div>

      <hr class="mx-n3">



                <div class="row align-items-center py-3">

                  <div class="col-md-3 ps-5">

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

                  <div class="col-md-9 pe-5">

                    <input type="text" class="form-control form-control-lg" placeholder="Enter
Wholesale Price" name="wholesale_price"  value="{{item['WHOLESALEPRICE']}}"/>

                  </div>

                </div>

      <hr class="mx-n3">

                <div class="row align-items-center py-3">

                  <div class="col-md-3 ps-5">

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

                  </div>

                  <div class="col-md-9 pe-5">

                    <input type="text" class="form-control form-control-lg" placeholder="Enter Retail
Price" name="retail_price"  value="{{item['RETAILPRICE']}}"/>

                  </div>

                </div>

      <hr class="mx-n3">

                  <div class="row align-items-center py-3">

                  <div class="col-md-3 ps-5">

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

                  </div>
```

```
          <div class="col-md-9 pe-5">
            <input type="text" class="form-control form-control-lg" placeholder="Enter Qty"
name="qty"  value="{{item['QTY']}}"/>
          </div>
        </div>
        <div class="row align-items-center py-3">
          <div class="col-md-3 ps-5">
            <h6 class="mb-0"> Low Stock Limit </h6>
          </div>
          <div class="col-md-9 pe-5">
            <input type="text" class="form-control form-control-lg" placeholder="Enter Low
Stock Limit" name="low_stock_limit"  value="{{item['LOW_STOCK_LIMIT']}}"/>
          </div>
        </div>
        <hr class="mx-n3">
        <div class="row align-items-center py-3">
          <div class="col-md-3 ps-5">
            <h6 class="mb-0"> Lot No  </h6>
          </div>
          <div class="col-md-9 pe-5">
            <input type="text" class="form-control form-control-lg" placeholder="Lot No"
```

```
name="lot_no"  value="{{item['LOTNO']}}"/>

          </div>

         </div>

     <hr class="mx-n3">

         <div class="row align-items-center py-3">

          <div class="col-md-3 ps-5">

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

          </div>

          <div class="col-md-9 pe-5">

           <input type="text" class="form-control form-control-lg" placeholder="Enter Note"
name="note"  value="{{item['NOTE']}}"/>

          </div>

         </div>




          <button type="reset" class="btn btn-primary btn-lg"> Cancel</button>




          <button type="submit" class="btn btn-primary btn-lg"> Save Inventory </button>
          </div>
        </div>
      </form>
</div>
</section>
{% endblock %}
```

add inventory.html

```
{% extends "base_template.html" %}
{% block title %}Add Inventory{% endblock %}
{% block content %}
<section class="vh-100">
  <div class="container">
      <div class="row">
          <div class="col-md-6">
              <h4 class="pull-left">Add Inventory</h4>
          </div>
          <div class="col-md-6 d-flex flex-row-reverse">
          </div>
      </div>
      <form method="post" action="/inventory/save">
          <input type="hidden" name="id" value="{{item['ID']}}">
      <div class="card" style="border-radius: 15px;">
          <div class="card-body">
            <div class="row align-items-center pt-4 pb-3">
              <div class="col-md-3 ps-5">
                <h6 class="mb-0"> Category </h6>
              </div>
              <div class="col-md-9 pe-5">
                <input type="text" class="form-control form-control-lg" placeholder="Enter
Category" name="category"  value="{{item['CATEGORY']}}"/>
              </div>
            </div>
<hr class="mx-n3">
            <div class="row align-items-center py-3">
              <div class="col-md-3 ps-5">
                <h6 class="mb-0"> Item Name </h6>
              </div>
              <div class="col-md-9 pe-5">
                <input type="text" class="form-control form-control-lg" placeholder="Enter  Item
Name" name="item_name"  value="{{item['ITEMNAME']}}"/>
```

```
          </div>

         </div>

   <hr class="mx-n3">


         <div class="row align-items-center py-3">

          <div class="col-md-3 ps-5">

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

          </div>

          <div class="col-md-9 pe-5">

           <input type="text" class="form-control form-control-lg" placeholder="Enter
Wholesale Price" name="wholesale_price"  value="{{item['WHOLESALEPRICE']}}"/>

          </div>

         </div>

   <hr class="mx-n3">

         <div class="row align-items-center py-3">

          <div class="col-md-3 ps-5">

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

          </div>

          <div class="col-md-9 pe-5">

           <input type="text" class="form-control form-control-lg" placeholder="Enter Retail
Price" name="retail_price"  value="{{item['RETAILPRICE']}}"/>
```

```
             </div>

            </div>

     <hr class="mx-n3">

             <div class="row align-items-center py-3">

             <div class="col-md-3 ps-5">

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

             </div>

             <div class="col-md-9 pe-5">

               <input type="text" class="form-control form-control-lg" placeholder="Enter Qty"
name="qty"  value="{{item['QTY']}}"/>

             </div>

            </div>

            <div class="row align-items-center py-3">

             <div class="col-md-3 ps-5">

               <h6 class="mb-0"> Low Stock Limit </h6>

             </div>

             <div class="col-md-9 pe-5">

               <input type="text" class="form-control form-control-lg" placeholder="Enter Low
Stock Limit" name="low_stock_limit"  value="{{item['LOW_STOCK_LIMIT']}}"/>

             </div>

            </div>
```

```
              <hr class="mx-n3">

              <div class="row align-items-center py-3">

                <div class="col-md-3 ps-5">

                  <h6 class="mb-0"> Lot No  </h6>

                </div>

                <div class="col-md-9 pe-5">

                  <input type="text" class="form-control form-control-lg" placeholder="Lot No"
name="lot_no"  value="{{item['LOTNO']}}"/>

                </div>

              </div>

         <hr class="mx-n3">

              <div class="row align-items-center py-3">

                <div class="col-md-3 ps-5">

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

                </div>

                <div class="col-md-9 pe-5">

                  <input type="text" class>

    </div>

    </body>

    </html>
```

GITHUB LINK:

https://github.com/IBM-EPBL/IBM-Project-43762-1660719321

DEMO LINK:

https://drive.google.com/file/d/11mUh4ZzdF-JBsP3tYeWuot0jHiV5B4db/view?usp=drivesdk