NALAYATHIRAN PROJECT UNDER NAAN MUDHALVAN SCHEME

BY THE INITIATIVE OF THE

GOVERNMENT OF TAMILNADU

SUBMITTED BY

COLLEGE NAME : JEPPIAAR ENGINEERING COLLEGE

TEAM ID : PNT2022TMID26994

PROJECT DOMAIN : CLOUD APPLICATION DEVELOPMENT

PROJECT NAME : INVENTORY MANAGMENT SYSTEM FOR RETAILER

TEAM LEADER :SIVA.P (310819104083)

TEAM MEMBERS :LOGITHASAN R (310819104048)

:PRAVEEN RAJ.D(310819104060)

:SANJAY KUMAR.L(310819104074)

Under The Guidance of

SPOC : Mr. GOUDHAMAN MARIMUTHU

Industrial Mentor : Mr/Mrs. Vasudeva Hanush

Faculty Mentor : Mrs. ANITHA GNANASELVI J

Faculty Evaluator : Mrs. TAMIL ROJA

TABLE OF CONTENT

1. INTRODUCTION

Inventory management information system is high performance software, which speed up the business operation of the organization . Every organization , which deals with the raw materials, put its great effort in the efficient utilization of its raw, material according to its need and requirement . The organization has to perform number of tasks and operations in order to run its business in manual system .

For example From NaavebUROM

- Estimation of new raw material required.
- Preparation of purchase order.

Preparation of inward sale invoice This Software “Inventory Management System” , is used for recording the information about the day to day transaction of stock of an organization. It stores purchase information of the products with credit/debit information form the supplier. Similarly, it stores sales information with credit/debit about the customer. If a product is purchased, then the related information is stored in stocks , that is , stocks are up to date. Another part I it prepare sales report after product it sold. in the sales information, the information about who sold the product is also kept, so there is no problem for misunderstandings in future.

1.1. PROJECT OVERVIEW

Inventory management system is an application which is helpful for business operate. Inventory management is a challenging problem area in supply chain management. Companies need to have inventories in warehouses in order to fulfil customer demand, meanwhile these inventories have holding costs and this is frozen fund that can be lost. Therefore, the task of inventory management is to find the quantity of inventories that will fulfil the demand, avoiding overstocks. This paper presents a case study for the assembling company on inventory management. It is proposed to use inventory management in order to decrease stock levels and to apply an agent system for automation of inventory management. processess Inventory management system (IMS) use for a departmental store.

1.2. PURPOSE:

A case study at ‘Guckenheimer’ (an on-site corporate restaurant management and catering company) cited issues regarding a basic resources requirement list that has to be maintained manually by the staff. To keep track of their inventory levels they have to calculate a list of the groceries utilized during a course of time, calculate and analyze the requirements for the future, and place their next order to the vendors if needed. This process takes up a lot of time and human effort, and is also prone to human error. This poses a problem of a situation that the staff at ‘Guckenheimer,’ as well as many other restaurants faces. It takes up a lot of time to manually keep track of sales and place correct orders to vendors, wasting useful labor in trivial works. A product which would assist in tackling the above mentioned problems would prove to be fruitful to clients such as ‘Guckenheimer’ and similar enterprises as this product would help convert the unproductive time to something more useful, by removing the unnecessary error prone complications and efforts.

2. LITERATURE SURVEY

2.1. EXISTING PROBLEM

Products are considered as the business resources for the organization. This includes managing the product with appropriate way to review any time as per the requirement. Therefore it is important to have a computer based IMS which has the ability to generate reports, maintain the balance of the stock, details about the purchase and sales in the organization. Before developing this application we came up with 2Inventory Management System existing in the market, which helps to give the knowledge for the development of our project. These application software are only used by the large organization but so we came up with the application which can be used by the small company for the management of their stock in the production houses. After analyzing the other inventory management system we decided to include some of common and key features that should be included in every inventory management system. So we decided to include those things that help the small organization in away or other.

2.2. REFERENCES

We have referred several documentations for the purpose of development phases.

[1]https://www.camcode.com/asset-tags/what-is-an-inventory-management- system/

[2]Jimmy Wales, online encyclopedia WiKipedia , http://www.wikipedia.org

[3]James Gosling. Java (Programming Language) , http://www.java.com

[4]Names Allaire, Netbeans-Fully-featured Java IDE, http://www.netbeans.org

[5]James Gosling , Welcome to java world.com: how-to feature and columns by Java expert; news; Java applets;

sample code ; tips , http://www.javaworld.com [6]

Pressman, Roger S.

"Software Engineering A Practitioner" Approch [7] John

Osborn , JavaBeans:

Developing Component Software in Java [8] Doug Lea

Concurrent Programming in Java:

Design Principles and Pattern, Addison-Wesley , November,1996 [9] Design Report, submitted 9th November 2012.

https://skydrive.live.com/redir?resid=2CEDE9F7F5F99604!196&authkey=

!AO5I ghTCML6xAk8

[10] Testing Document, submitted 26th November 2012. https://www.camcode.com/asset-tags/what-is-an-inventory-

management-

2.3. PROBLEM STATEMENT DEFINITION:

The problem statement aims to make desktop application for retailers and to track all areas of IMS like purchase details, sales details, stock management. The application helps the retailer to have complete insights about the products stored in the inventory and can manage them flexibly.

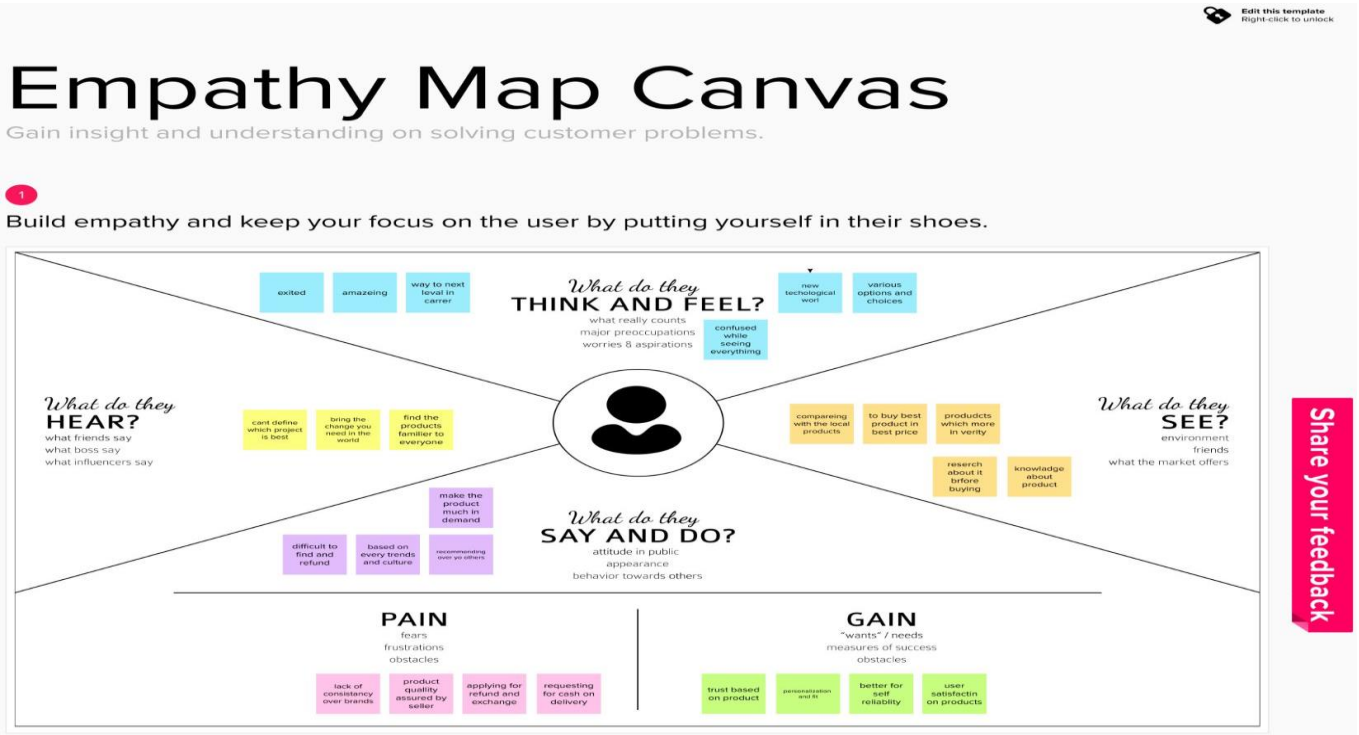
Problem statement(PS)	I am (customer)	I'm trying to	But	Because	Which makes me feel
PS-1	Retailer	Find the product counts in the stock	It's hard	It takes more time	Tired
PS-2	Retailer	Calculate the bill for transportation purpose	It's hard	It take long time	Uninterested
PS-3	Retailer	Find the customer's 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. IDEATION & PROPOSED SOLUTION

We have analyzed different systems and proposed an ideation phase of our developed management system.

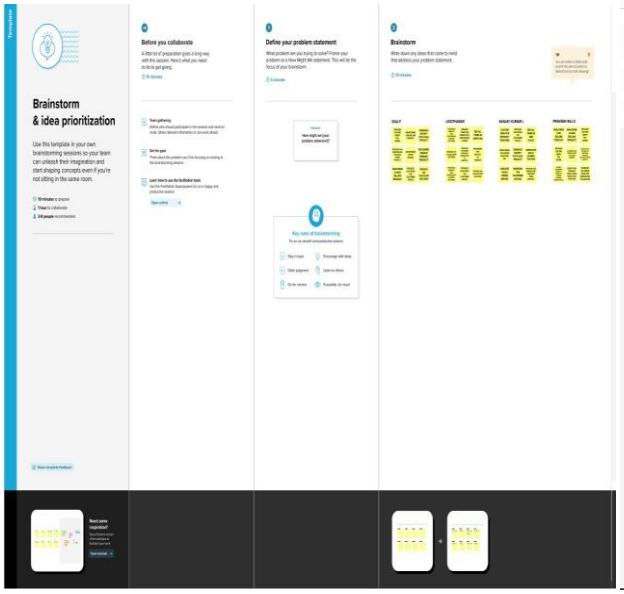
3.1. EMPATHY MAP CANVAS:

An empathy map canvas helps brands provide a better experience for users by helping teams understand the perspectives and mindset of their customers. Using a template to create an empathy map canvas reduces the preparation time and standardizes the process so you create empathy map canvases of similar quality.



3.2. IDEATION & BRAINSTORMING

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



3.3. PROPOSED SOLUTION

PREPROCESSING PHASES:

Home:

This first module manages Home Screen Which is Provide A Home Page of my Software. After clicking home butten . butten will provide Welcome Screen of the Software etc.

Sales:

This is Provide Sales information And Sales Page it is contain sales_id, Product_code , Product_name , Quantity, Revenue, Sold by etc

Suppliers:

Suppliers page contain suppliers details and its hold basic value with attributes it is provide a suppliers code, full name ,location ,phone etc Products: It is hold the details of product with product code, product name, cost price selling price brand etc.

Purchase:

This is contain detail about purchase . It will provide purchase screen which is hold some value like purchase id ,product code ,product name ,quantity ,total cost etc And Each page has refresh facilities And search facilities and Direct input value interface etc .

Edit:

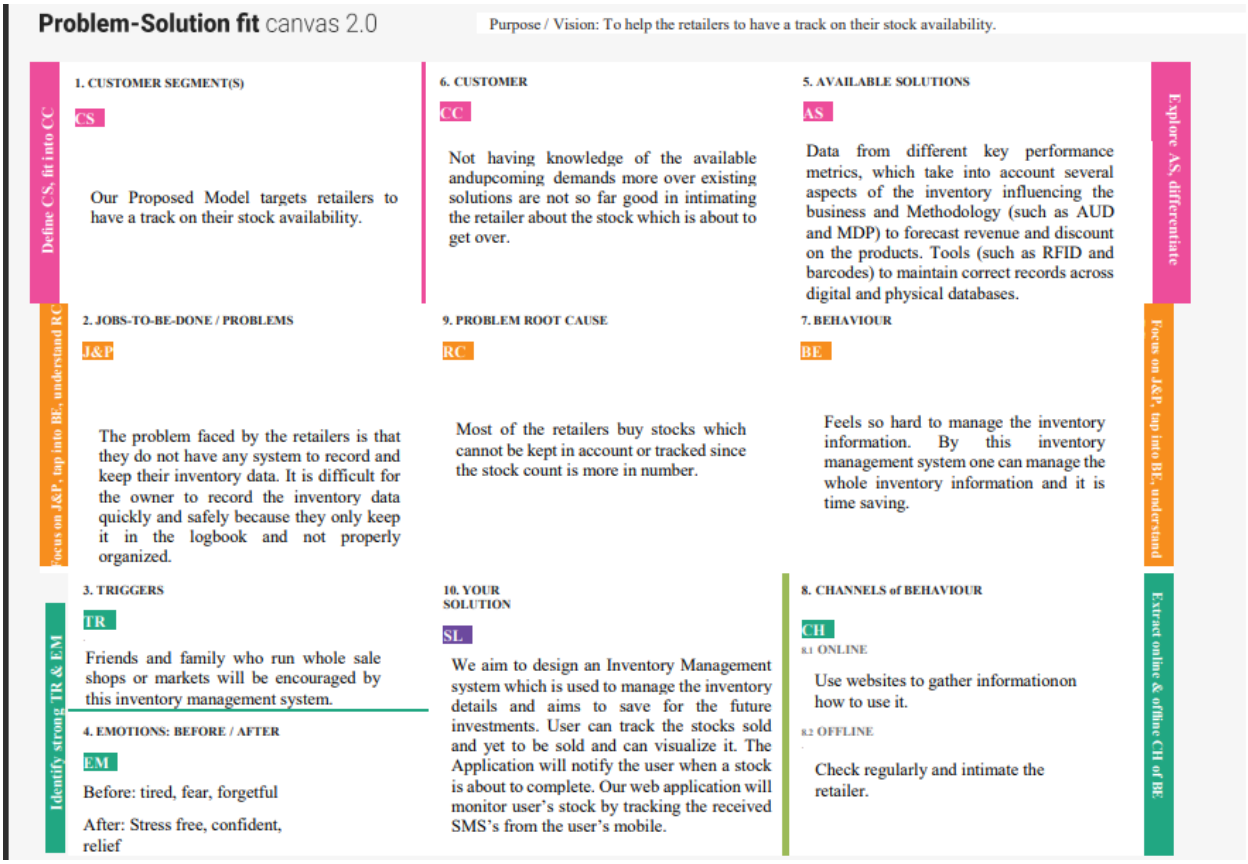
Many Module conatins Edit facilities Which has control of editing value from data base diretly and insert new value etc.

Clear and Delete:

Clear and delete is provide advance facilities of this software Because it is Provide a deletion and clear data process etc

In the proposed system, all the business operations will be automated. Some of the features which the new system will provide are Auto generation of Daily Demand report, Auto generation of Purchase Order of various raw materials. As everything is auto generated, the production delays are avoided. It makes the system more secure as only authenticated users can access the system. Also, there are privileges in which we can authorize a particular user for accessing system or particular modules of the application.

3.4. PROBLEM SOLUTION FIT



4. REQUIREMENT ANALYSIS

4.1. FUNCTIONAL REQUIREMENTS

Functional Requirements:

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 Gmail Registration through LinkedIN
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	Business regulations	Many needs may fit under this category
FR-4	Product management	Easily track product information Quickly produce reports for single or multiple sold products
FR-5	Audit Monitoring	The technique of tracking crucial data is known as audit tracking
FR-6	Historical Data	Specify the amount of storage you need to handle this expansion

4.2 NON-FUNCTIONAL REQUIREMENTS

Non-functional Requirements:

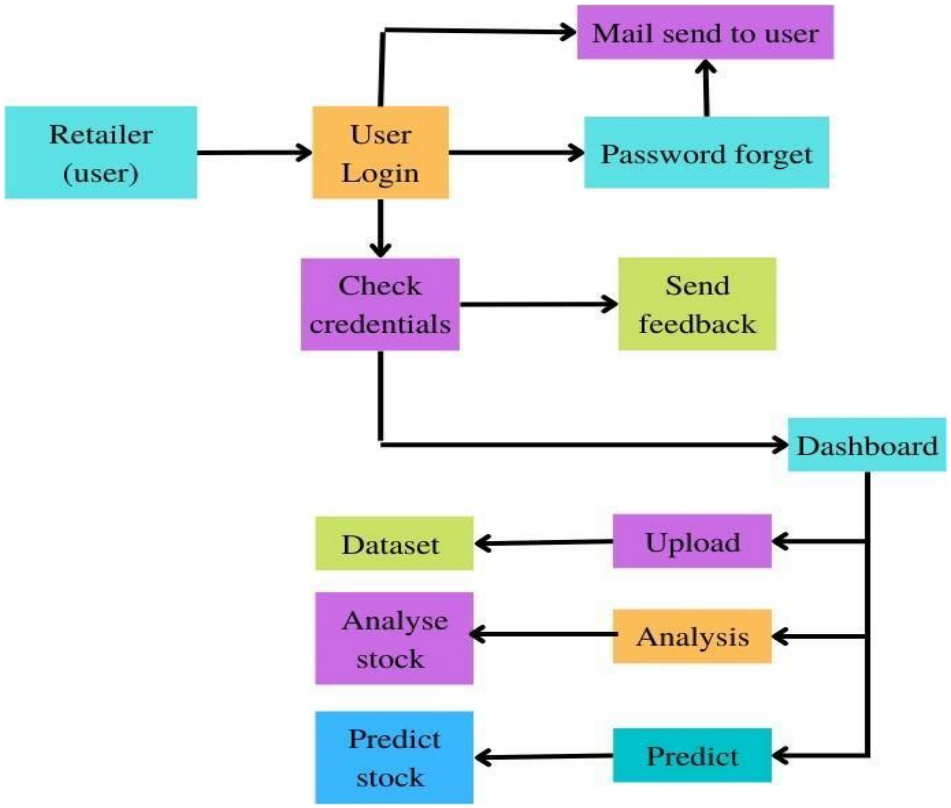
Following are the non-functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	Usability	Backups for database are available
NFR-2	Security	The security requirements deal with the primary security.only authorized users can access the system with user name and password of administrator
NFR-3	Reliability	The software will not be able to connect to the database in the event of the server being down due to a hardware or software failure
NFR-4	Performance	Easy tracking of records and updating can be done
NFR-5	Availability	The software will be available only to administrator of the organization and the product as well as customer details will be recorded by him.He can add customers,Update and delete them as well as add new products and manage them
NFR-6	Scalability	The ability of a system to handle a growing amount of work

5. PROJECT DESIGN

5.1. DATAFLOWDIAGRAMS

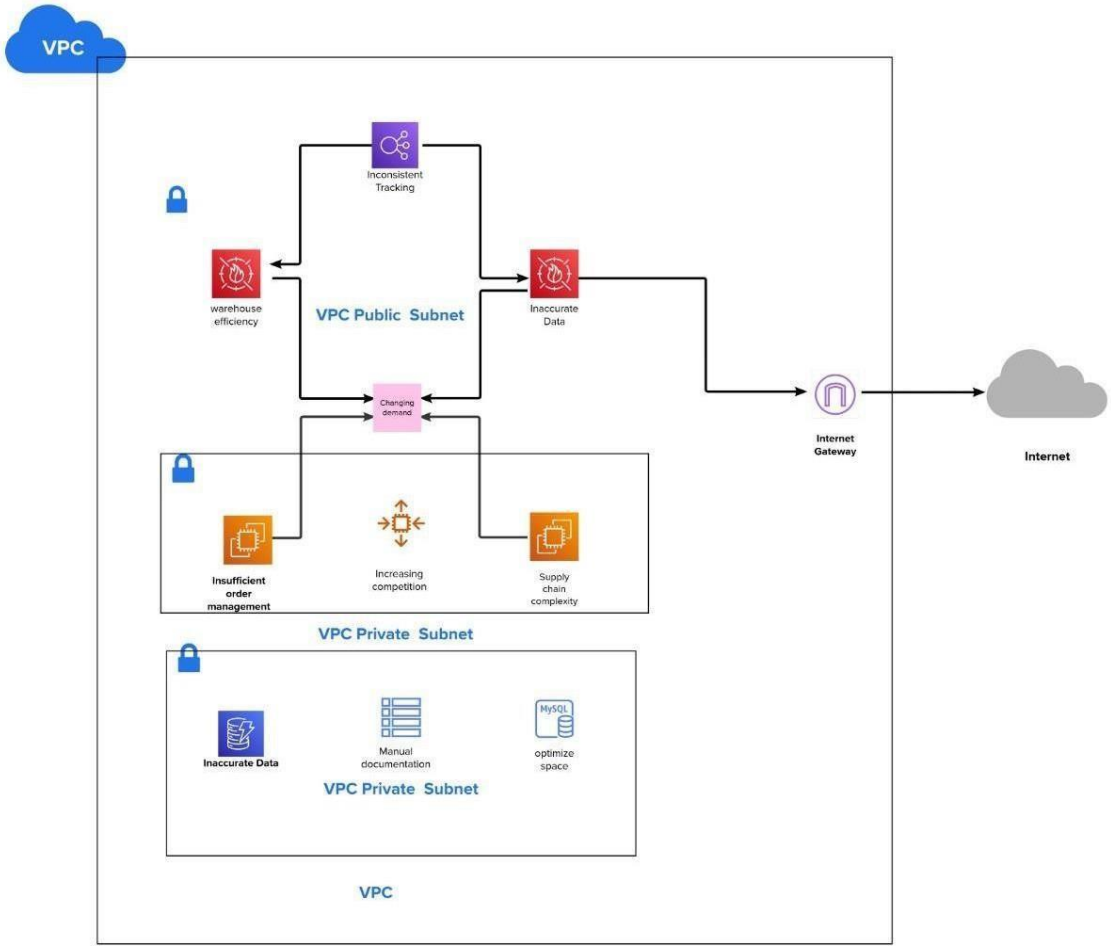
A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.



5.2. SOLUTION & TECHNICAL ARCHITECTURE

- There was no efficient solution available in many companies these days.
- Every process was based on paper work.
- Human fault rate were high.
- Tracing the inventory losses were not possible
- There was no efficient login system.
- After the computer age, every process is started to be integrated into computer environment.

•Now, we have qualified technology to implement new solution to these problems.



5.3. USER STORIES

A user story is the smallest unit of work in an agile framework. It's an end goal, not a feature, expressed from the software user's perspective.

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (web user)	Registration	USN-1	As a user, I can register for the web application by entering my email, password, and confirming my password.	I can access my account / dashboard.	High	Sprint-1
		USN-2	As a user, after completing the registration I will receive confirmation email once I have registered for the web application.	I can receive confirmation email, click confirm.	High	Sprint-1
		USN-3	As a user, I can register for the web application through facebook	I can register, access the dashboard with facebook login.	Medium	Sprint-2
		USN-4	As a user, I can register for the web application through google account.	I can register, access the dashboard with gmail login.	High	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering email & password after installing the web application.	I can access the dashboard by login into the application.	High	Sprint-2
	Dashboard	USN-6	As a user, I can view the year wise price representation of the dataset.	I can analyse the stocks using line graph in my retail store.	Medium	Sprint-2
		USN-7	As a user, I can view the year wise stock representation of the dataset.	I can analyse the stocks using line graph.	Medium	Sprint-3

6. PROJECT PLANNING & SCHEDULING

6.1. SPRINT PLANNING & ESTIMATION

Sprint 1:

1. We created a Flask Project.
2. Added all the routes needed for our project
3. Created Tables in IBM Cloud.

Sprint 2:

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

2.We styled 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. Integration of Send grid into our application

Sprint 4:

1. Deploying the application using Docker and Kubernetes

6.2. SPRINT DELIVERY SCHEDULE

Project Planning Phase

Project Planning Template (Product Backlog, Sprint Planning, Stories, Story points)

Date	22 October 2022
Team ID	PNT2022TMID26994
Project Name	Inventory Management System for Retailers
Maximum Marks	8 Marks

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	As a user, I can register for the application by using my email & password and confirming my login credentials.	3	High	Siva, Logithasan Sanjaikumar Praveen Raj,
Sprint-1		USN-2	As a user, I can login through my E-mail.	3	Medium	Siva, Logithasan Sanjaikumar, Praveen Raj,
Sprint-1	Confirmation	USN-3	As a user, I can receive my confirmation email once I have registered for the application.	2	High	LogithasanYaseen, Logithasan Sanjaikumar, Praveen Raj,
Sprint-1	Login	USN-4	As a user, I can log in to the authorized account by entering the registered email and password.	3	Medium	Siva, Logithasan Sanjaikumar, Praveen Raj,
Sprint-2	Dashboard	USN-5	As a user, I can view the products that are available currently.	4	High	Siva, Logithasan Sanjaikumar, Praveen Raj,
Sprint-2	Stocks update	USN-6	As a user, I can add products which are not available in the inventory and restock the products.	3	Medium	Siva, Logithasan Sanjaikumar, Praveen Raj,
Sprint-3	Sales prediction	USN-7	As a user, I can get access to sales prediction tool which can help me to predict better restock management of product.	6	Medium	Siva, Logithasan Sanjaikumar, Praveen Raj,
Sprint-4	Request for customer care	USN-8	As a user, I am able to request customer care to get in touch with the administrators and enquire the doubts and problems.	4	Medium	Siva, Logithasan Sanjaikumar, Praveen Raj,
Sprint-4	Giving feedback	USN-9	As a user, I am able to send feedback forms reporting any ideas for improving or resolving any issues I am facing to get it resolved.	3	Medium	Siva, Logithasan Sanjaikumar, Praveen Raj,

Project Tracker, Velocity & Burndown 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	11	6 Days	24 Oct 2022	29 Oct 2022	11	29 Oct 2022
Sprint-2	7	6 Days	31 Oct 2022	05 Nov 2022	7	05 Nov 2022
Sprint-3	6	6 Days	07 Nov 2022	12 Nov 2022	6	12 Nov 2022
Sprint-4	7	6 Days	14 Nov 2022	19 Nov 2022	7	19 Nov 2022

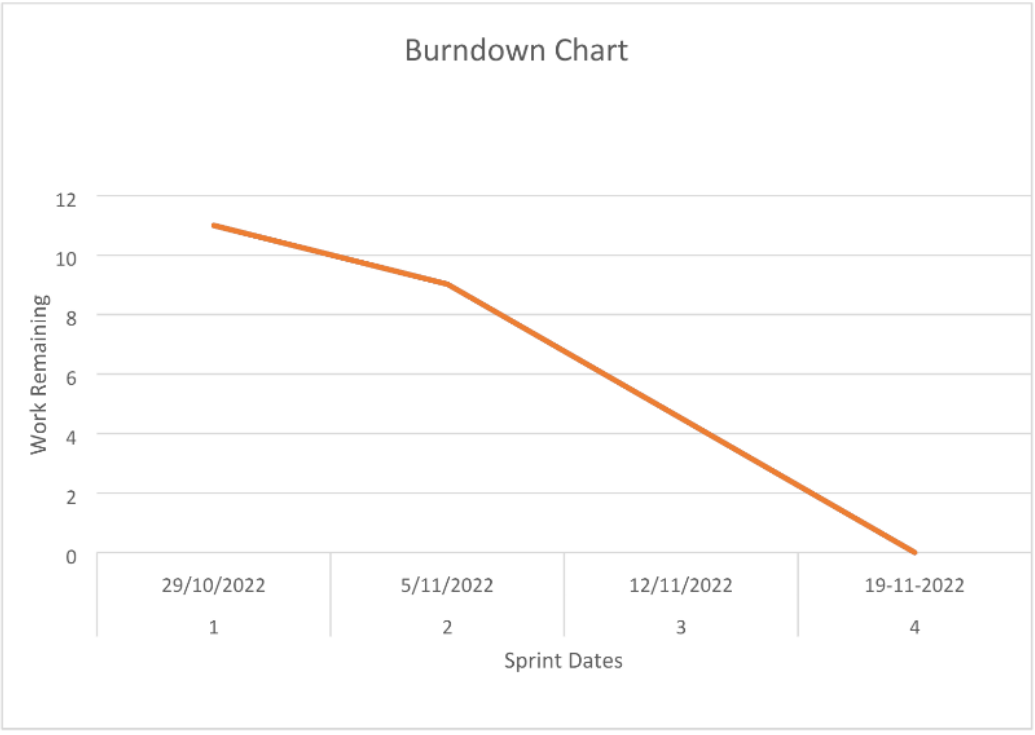
Velocity:

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

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

Our velocity should be:

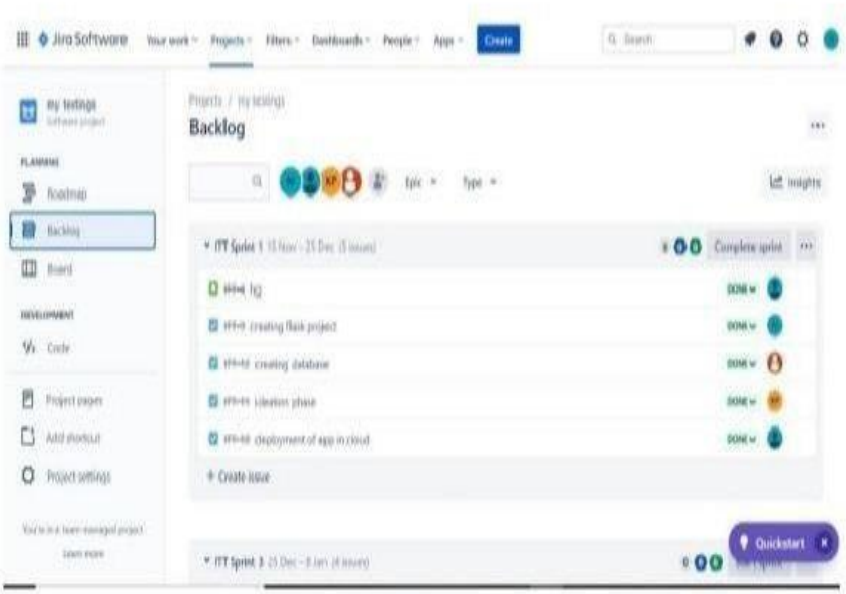
$$AV = \frac{(11+7+6+7)}{24} = \frac{31}{24} = 1.29$$



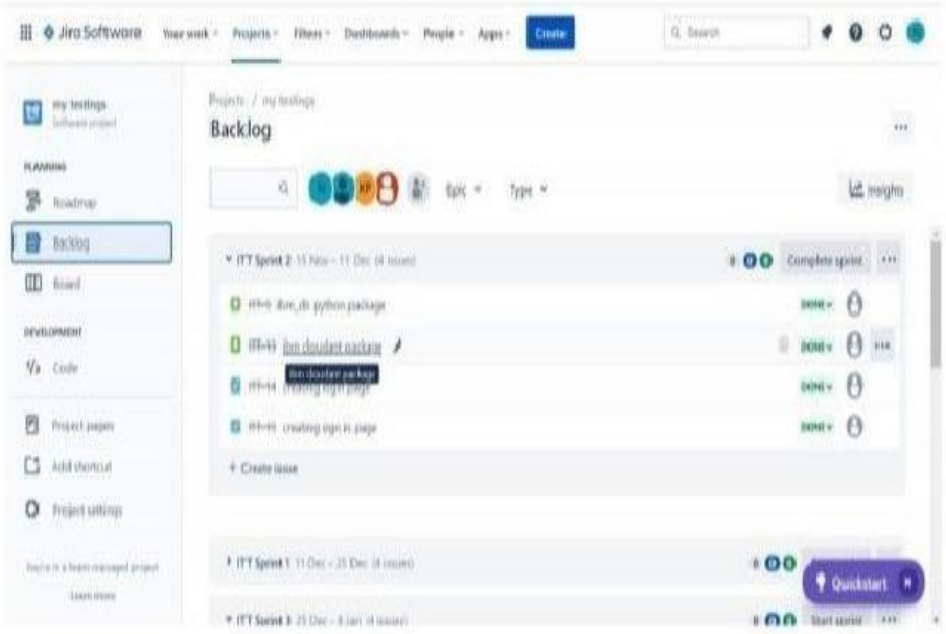
6.3. REPORTS FROM JIIRA

IT organizations have the challenge 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 operations. According to Gartner, mastering the discipline of effective asset management is a huge cost savings for companies.

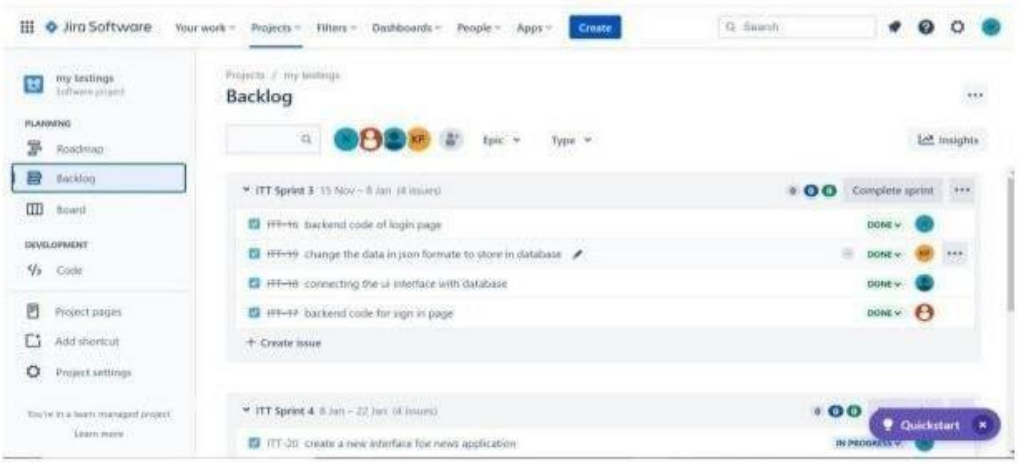
Sprint-1



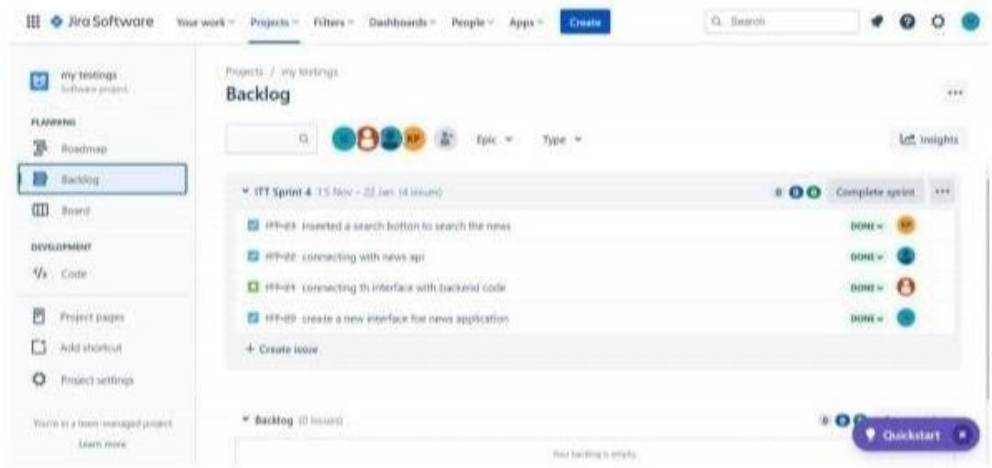
Sprint-2



Sprint-3

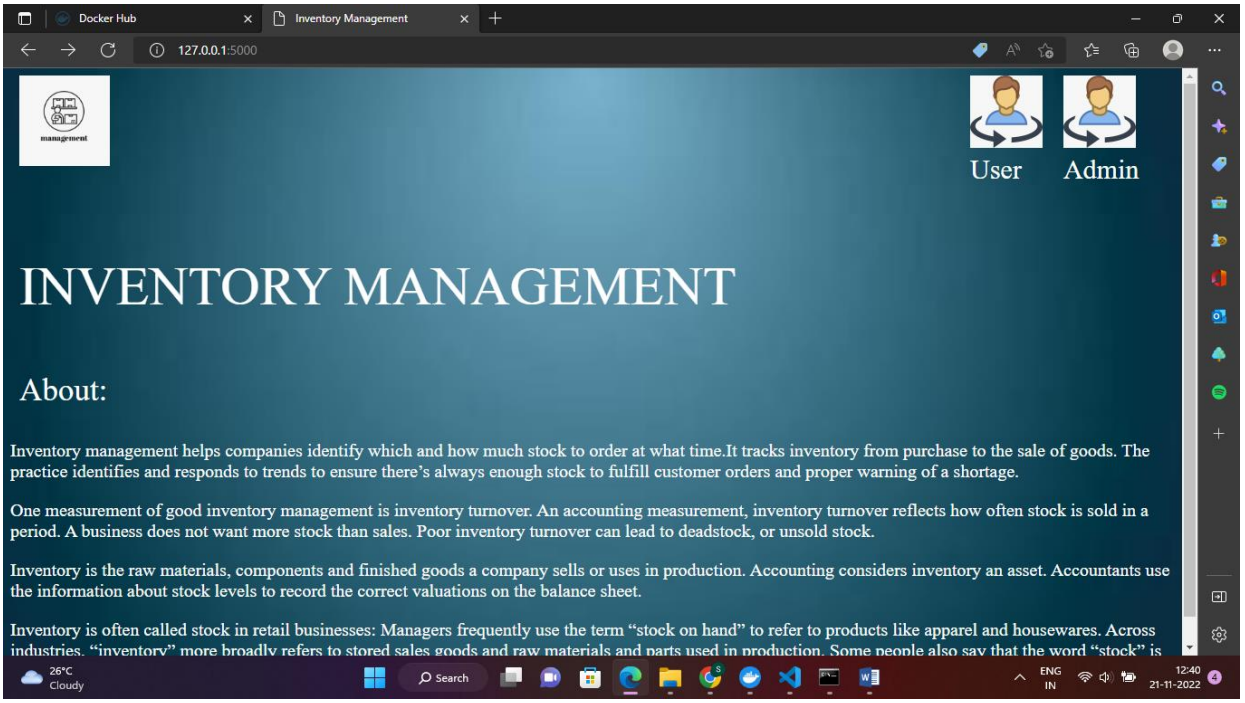


Sprint-4

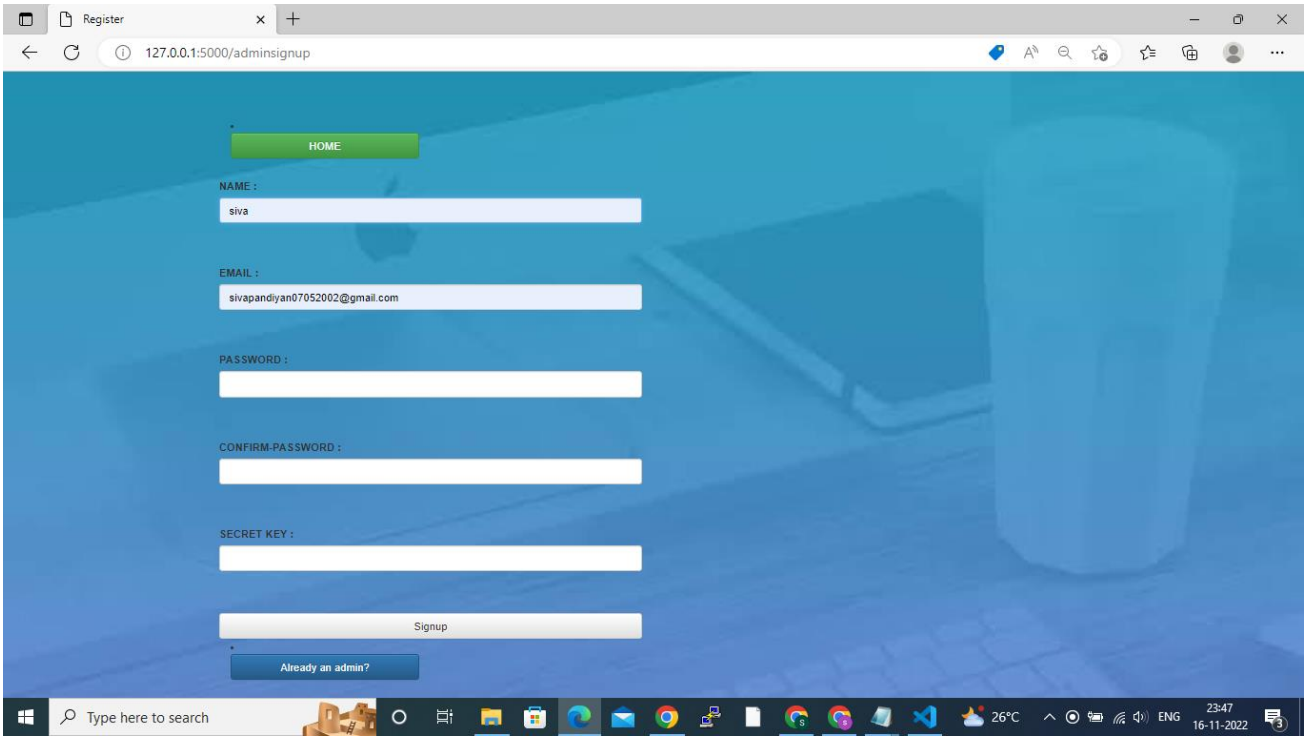


7.CODING & SOLUTIONING

7.1. FEATURE1



7.2. FEATURE 2



7.3. DATABASE SCHEMA

PRODUCT_ID	PRODUCT_NAME	PRODUCT_PRICE	PRODUCT_STOCK
1	Mouse	100	10
2	Pen	50	25
3	Box	150	25
4	Hard Disk	4800	10
5	CPU	10000	5
6	Key Board	1500	20
7	Mobile	15000	7
8	Projector	22000	3

8. TESTING

8.1. TEST CASE

Features to be tested	Test Description
Login to the system	This tests the login interface of the system.
Adding a Recipe to database	This test is conducted to verify if a recipe is successfully added to the database. This will check if the recipe is added to its header table and also check if the recipe details are added to the recipe details table.
Adding an Ingredient to database	This tests checks if new ingredient is added correctly to the database with the specified details.
Adding a Vendor to the database	This test checks if the newly added vendor is correctly added to the database with the specified details.
Checking the threshold levels	This test is conducted to verify if the ingredients that are below the threshold levels are listed by the function when called by the user. The verification is done by referring to the database.
Updating the sales for the day	This test is conducted to test the sales update in the database. The test checks if the database is updated with the correct ingredient values based on the sales data input to the system.
Updating the order reception to database	This test is conducted to test the correct updating of the database after receiving the order from the vendor.
Create Orders	This test is conducted to check the order creation capability of the system. The list of ingredients that is generated for order must comply with the set conditions of threshold levels

8.2. USER ACCEPTANCE TESTING

Test case : Testing the Add Recipe Interface and its functioning

Case 1: Testing the Quantity input field.

Case 2: Testing the Recipe Name field.

Case 3: Testing the Ingredients in recipe list and Quantity of ingredient list. Case 4: Testing the available ingredients list.

Case 5: Testing the all the above cases together and checking if the entries are updated to the tables in database.

Test Case : Check Threshold Interface

Case 1: Check if the Ingredients under the threshold values are shown in the Ingredients below threshold list.

Case 2: Check if the Create order button asks the user to enter values for all the ingredients listed under the ingredients below threshold list.

Case 3: Check if pressing the Process Order button creates a file with the order details in it.

Test Case : Testing the Update after sales interface

Case 1: Test the Recipe list box.

Case 2: Test the quantity text field..

Case 3: Test the recipe sold list box quantity sold list box.

Case 4: Test if the details are updated to the database when requested.

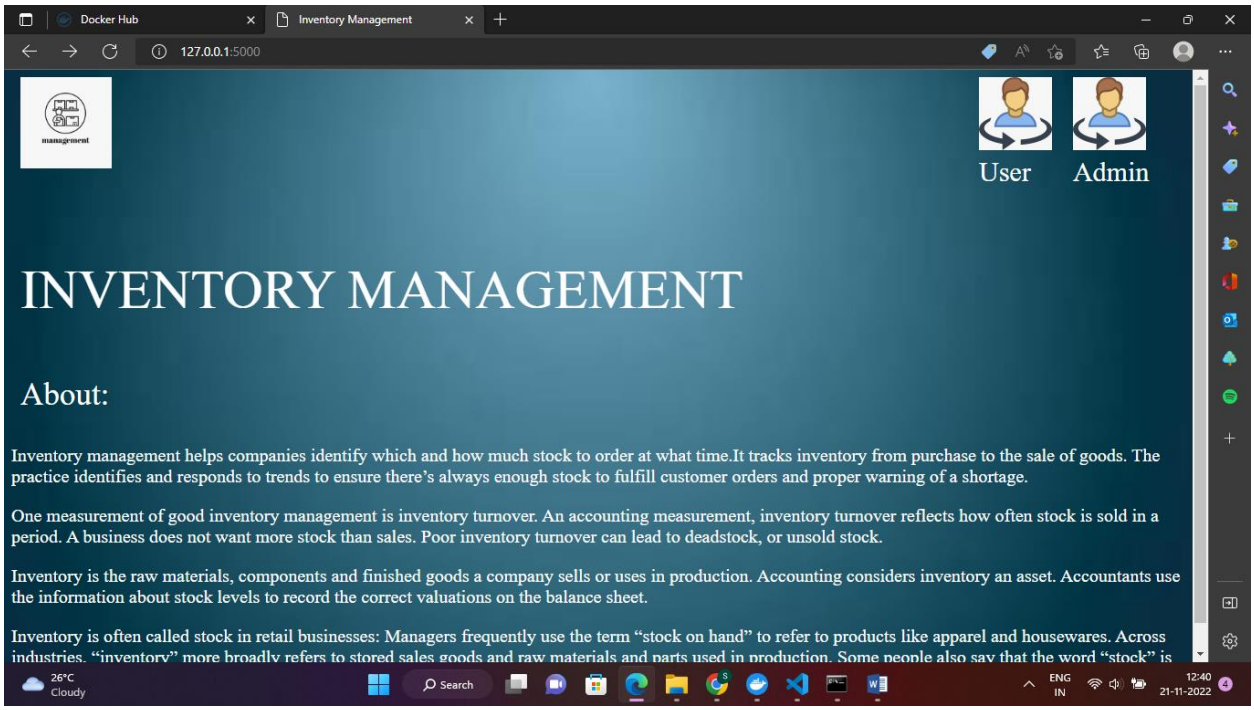
9. RESULTS

9.1. PERFORMANCE METRICS

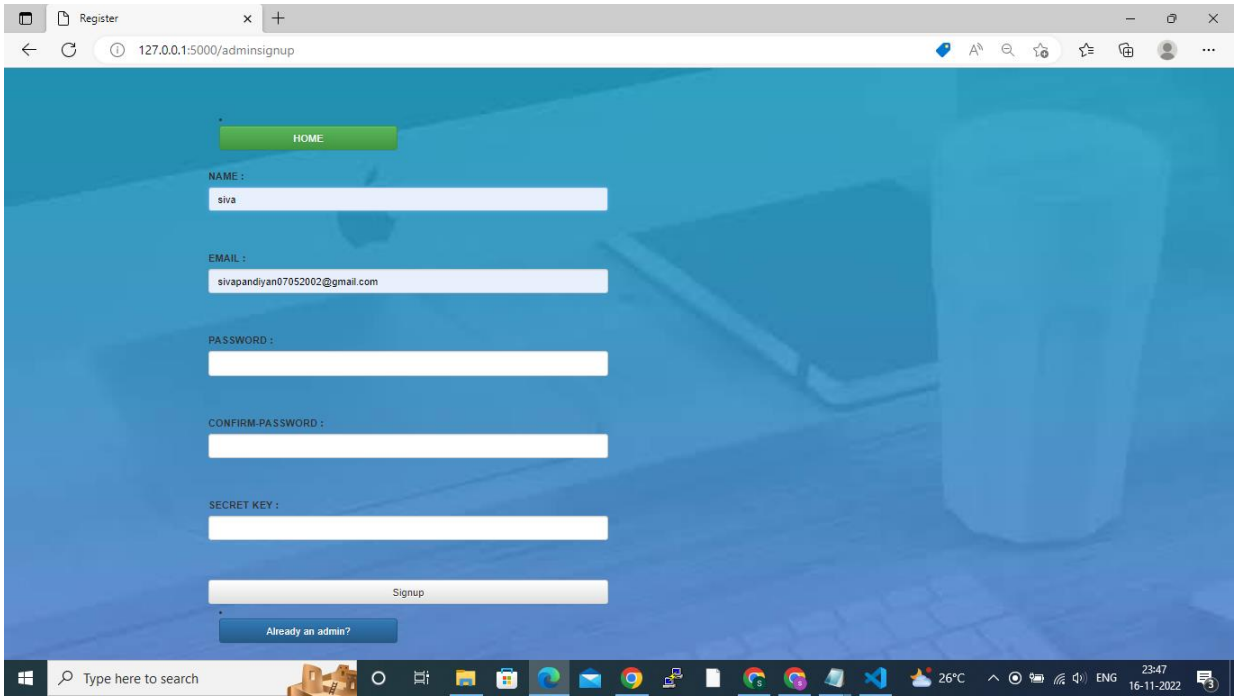
Inventory Performance is a measure of how effectively and efficiently inventory is used and replenished. The goal of inventory performance metrics is to compare actual on-hand dollars versus forecasted cost of goods sold. Many Lean practitioners claim that inventory performance is the single best indicator of the overall operational performance of a facility. Inventory performance looks at and is measured using either Inventory Days OnHand (DOH) or Inventory Turns.

- **Inventory Days On-Hand:** The number of days it would take to consume current on-hand inventory. Always measure multiple inventory item numbers in terms of currency (i.e. COGS).
- **Inventory Turns**: The number of times inventory is replaced in a year.

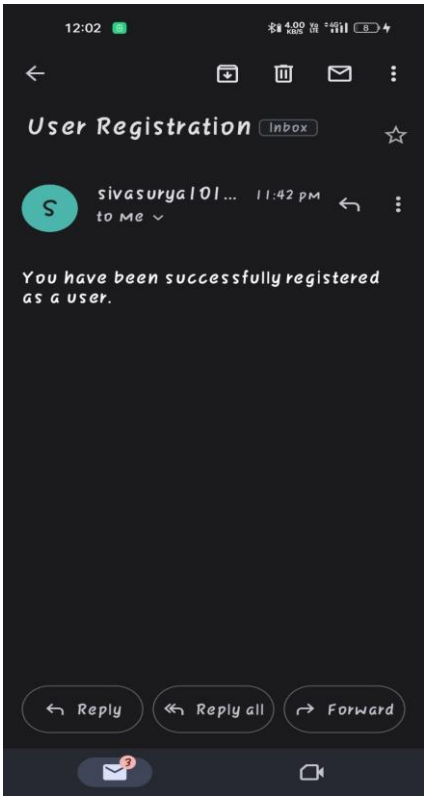
HOME PAGE:



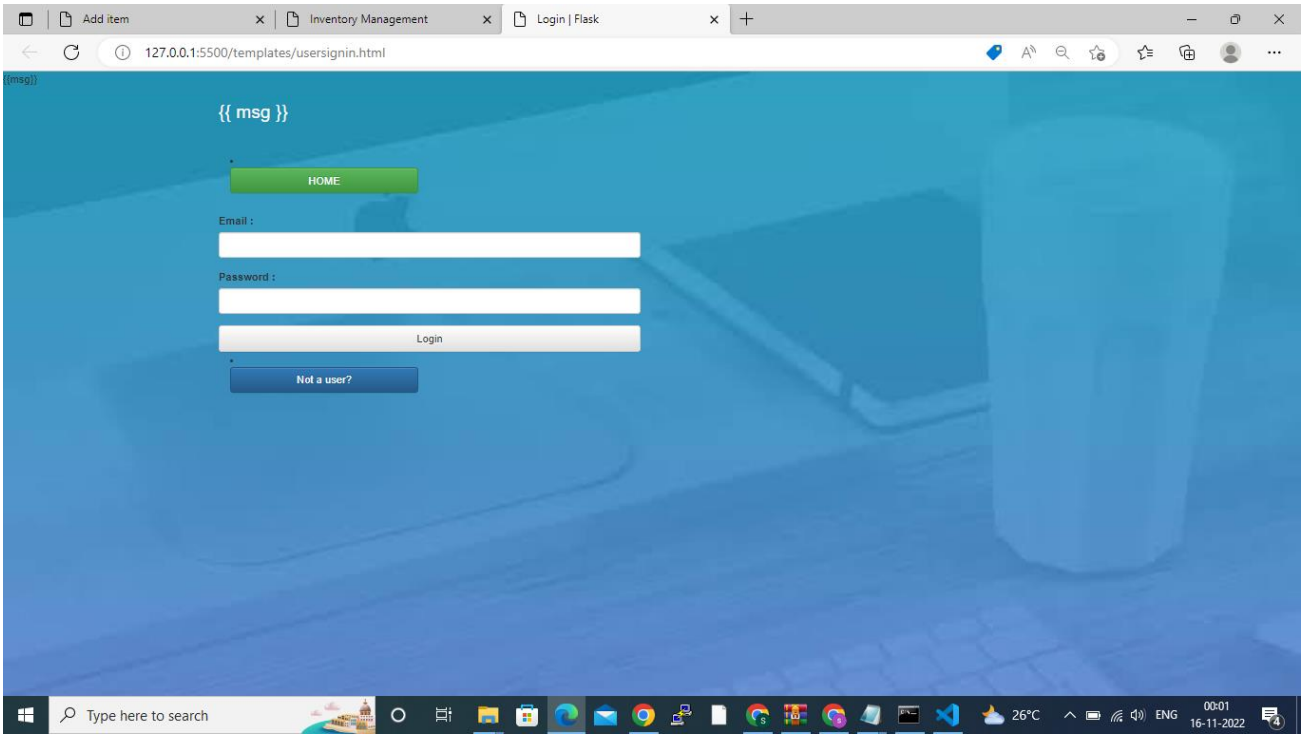
REGISTRATION FORM



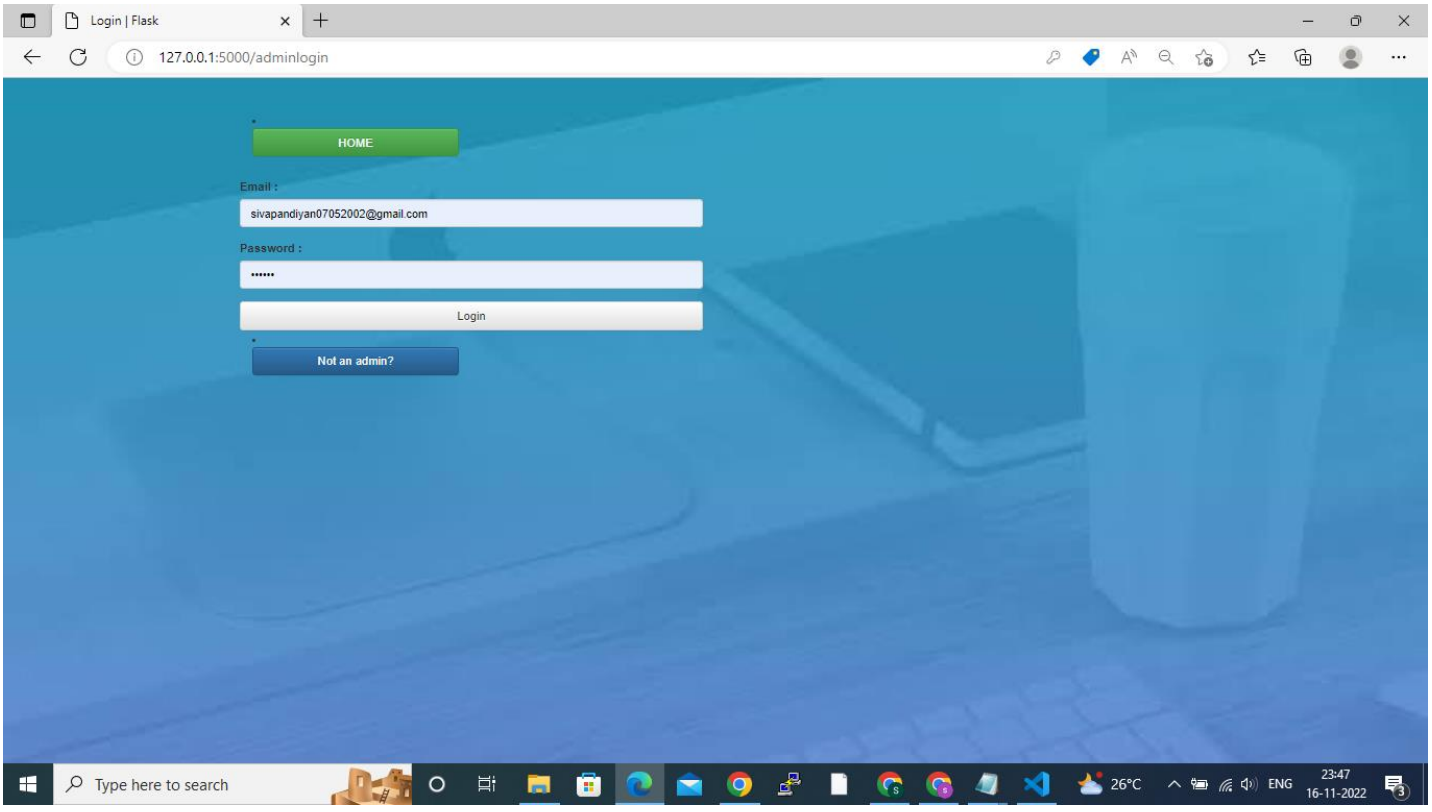
REGISTRATION SUCCESSFULLY :



USER LOGIN FORM:

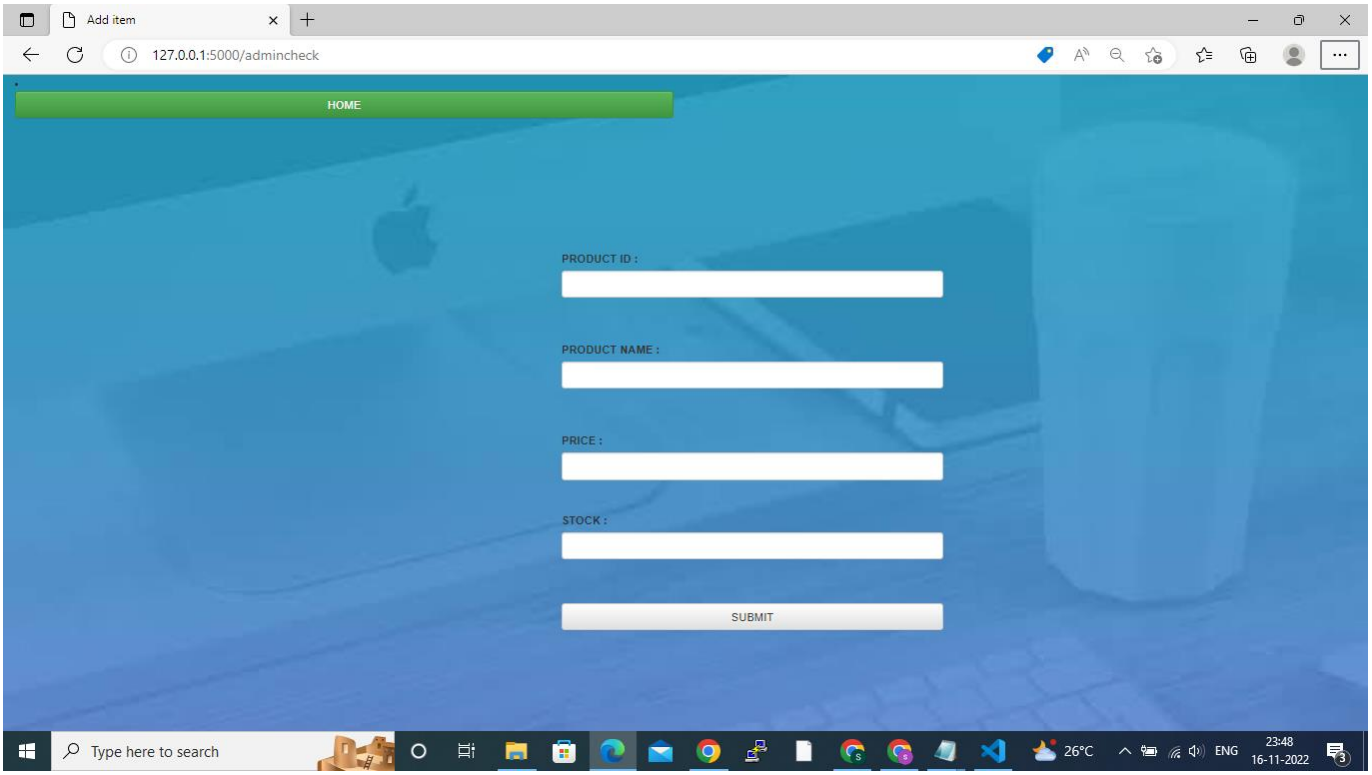


ADMIN LOGIN:



DASHBOARD:

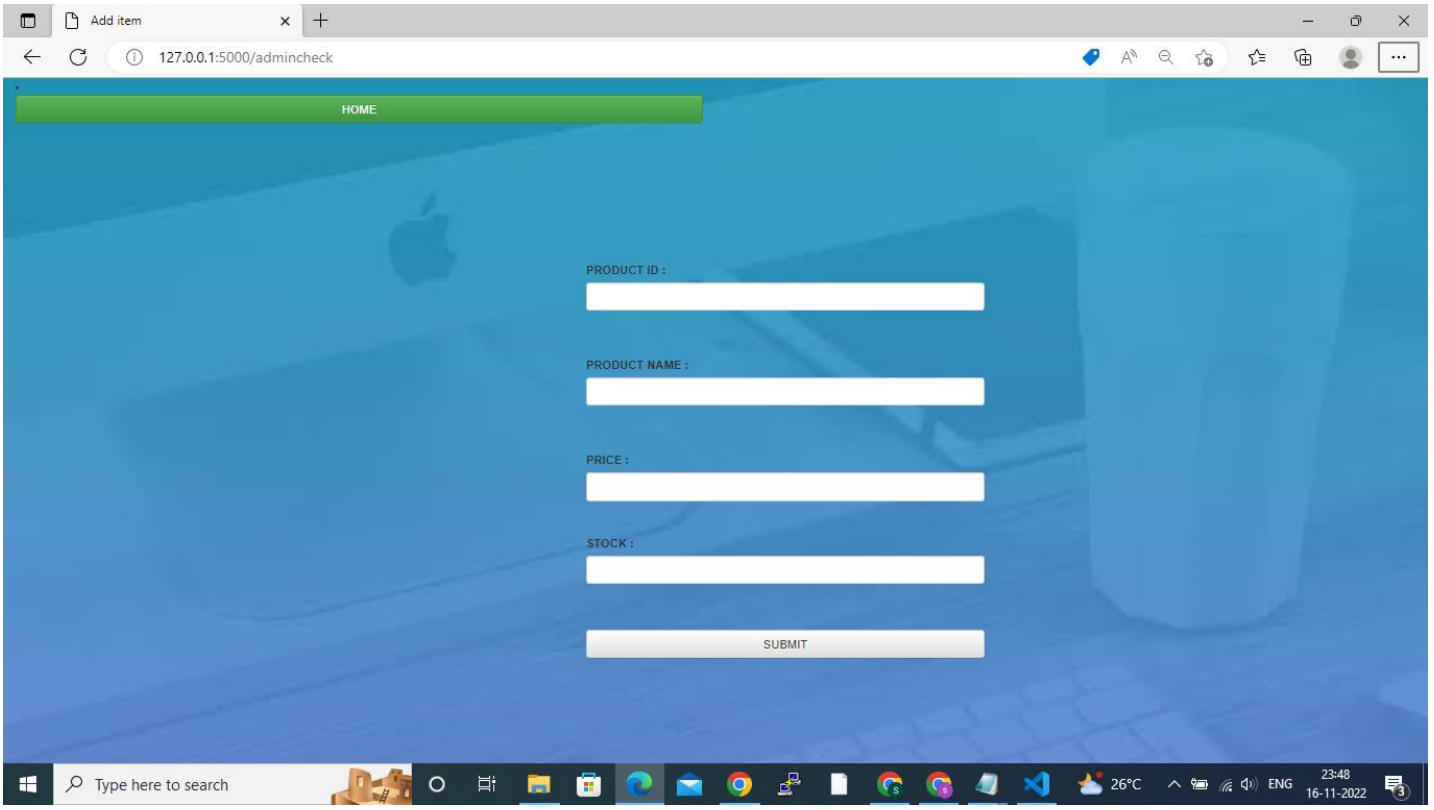
ADDSTOCK:



VIEW STOCK:

PRODUCT_ID	PRODUCT_NAME	PRODUCT_PRICE	PRODUCT_STOCK
1	Mouse	100	10
2	Pen	50	25
3	Box	150	25
4	Hard Disk	4000	10
5	CPU	10000	5
6	Key Board	1500	20
7	Mobile	15000	7
8	Projector	22000	3

UPDATE STOCK



10. ADVANTAGES & DISADVANTAGES

ADVANTAGES

1. **It helps to maintain the right amount of stocks:** contrary to the belief that is held by some people, inventory management does not seek to reduce the amount of inventory that you have in stock, however, it seeks to maintain an equilibrium point where your inventory is working at a maximum efficiency and you do not have to have many stocks or too few stocks at hand at any particular point in time. The goal is to find that zone where you are never losing money in your inventory in either direction. With the aid of an efficient inventory management strategy, it is easy to improve the accuracy of inventory order.
2. **It leads to a more organized warehouse:** with the aid of a good inventory management system, you can easily organize your warehouse. If your warehouse is not organized, you will find it very difficult to manage your inventory. A lot of businesses choose to optimize their warehouse by putting the items that have the highest sales together in a place that is easy to access in the warehouse. This ultimately helps to speed up order fulfilment and keeps clients happy.

DISADVANTAGES

1. **Bureaucracy:** even though inventory management allows employees at every level of the company to read and manipulate company stock and product inventory, the infrastructure required to build such a system adds a layer of bureaucracy to the whole process and the business in general. In instances where inventory control is in-house, this includes the number of new hires that are not present to regulate the warehouse and facilitate transactions. In instances where the inventory management is in the hands of a third party, the cost is a subscription price and a dependence on another separate company to manage its infrastructure. No matter the choice you go for, it translates to a higher overhead cost and more layers of management between the owner and the customer. From the view point of the customer, a problem that requires senior management to handle will take a longer period of time before it will be trashed out.

2. Impersonal touch: another disadvantage of inventory management is a lack of personal touch. Large supply chain management systems make products more accessible across the globe and most provide customer service support in case of difficulty, but the increase in infrastructure can often mean a decrease in the personal touch that helps a company to stand out above the rest. For instance, the sales manager of a small manufacturing company that sells plumbing supplies to local plumbers can throw in an extra box of washers or elbows at no charge to the customer without raising any alarms. This is done for the sake of customer relations and often makes the customer feel like he is special. While free materials can also be provided under inventory management, processing time and paper work make obtaining the material feel more like a chore for the customer or even an entitlement.

11. CONCLUSION

The project “Inventory Management System for Retailers” mainly as the name suggests deals with the calculation of the available and processed resources for an accurate inventory control and process management for a domain specific client who are related to the subject of food chains/outlets. This enables the inventory to be applied at every level in the hierarchy of the products and its complex combinations of recipes. A system that accurately calculates the atomic ingredients used for making a recipe then automatically performs the back end operation pertaining to a database of many relational tables onto which the changes are being made with each and every operation performed on the front end and which also shows up if at the time of retrieval. The most important part of Inventory controlling is its ability to check for threshold levels and alert the manager to replenish the stock before it reaches a danger zone. So as when an ingredient level goes below the threshold level then it routes an alert to the manager. Then if needed accordingly an automated order form is produced so as to each specific vendor along with the quantities needed for replenishment. As a part of the standard maintaining a drill of risk management is done in order to sustain during the days of special occasion or holidays when the demand reaches to rather more different scale as compared to other days. These occasions call on for special inclusions into the menu which reflects on the recipes and in turn reflects the ingredients being used up eventually. Thus was provided the liberty of adding special recipe to the menu for some special occasion and is regarded as a key feature.

.

12. FUTURE SCOPE

- The Fourth Industrial Revolution will continue to drive technological change that will impact the way that we manage inventories.
- Successful companies will view inventory as a strategic asset, rather than an aggravating expense or an evil to be tolerated.
- Collaboration with supply chain partners, coupled with a holistic approach to supply chain management, will be key to effective inventory management.

13.APPENDIX

App.py

```
from turtle import st
from flask import Flask,render_template,request,redirect,url_for,session
from markupsafe import escape

import os
from sendgrid import SendGridAPIClient
from sendgrid.helpers.mail import Mail

import ibm_db

conn = ibm_db.connect("DATABASE=bludb;HOSTNAME=125f9f61-9715-46f9-9399-
c8177b21803b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=30426;SECURITY=
SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=lhh64039;PWD=1WGH7WcVEb
etE7Bf","")
print ("Database connection established", conn)
app = Flask(_name_)
@app.route('/')
def home():
  return render_template('index.html')
@app.route('/usersignup')
def usersignup():
  return render_template('usersignup.html')

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

@app.route('/userlogin')
def userlogin():
  return render_template('usersignin.html')

@app.route('/adminlogin')
def adminlogin():
  return render_template('adminsignin.html')
```

```
@app.route('/adminnotify')
def adminnotify():
    message =
Mail(from_email="sivasurya1018@gmail.com",to_emails="sivapandiyan07052002@gmail.com",subject="Out of stock",html_content="<p>Some products are out of stock please check it out</p>")

    try:
     sg =
SendGridAPIClient("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")
     response = sg.send(message)
     return render_template('userpanel.html',msg="Admin have been notified through mail")
    except Exception as e:
     print(e)
     return render_template('userpanel.html',msg="Error")
@app.route('/userregisteration',methods = ['POST', 'GET'])
def userregistration():
  if request.method == 'POST':

    name = request.form['name']
    email = request.form['email']
    password = request.form['password']
    confirmpassword = request.form['confirmpassword']
    sql = "SELECT * FROM USER_DATA WHERE EMAIL=? "

    stmt = ibm_db.prepare(conn, sql)
    ibm_db.bind_param(stmt,1,email)

    ibm_db.execute(stmt)
    account = ibm_db.fetch_assoc(stmt)

    if account:
      return render_template('index.html', msg="You are already a user, please login using your details")
    else:
      insert_sql = "INSERT INTO USER_DATA VALUES (?,?,?,?)"
      prep_stmt = ibm_db.prepare(conn, insert_sql)
      ibm_db.bind_param(prep_stmt, 1, name)
```

```
    ibm_db.bind_param(prep_stmt, 2, email)
    ibm_db.bind_param(prep_stmt, 3, password)
    ibm_db.bind_param(prep_stmt, 4, confirmpassword)

    ibm_db.execute(prep_stmt)
    message = Mail(from_email="sivasurya1018@gmail.com",to_emails=email,subject="User Registration",html_content="<p>You have been successfully registered as a user.</p>")

  try:
   sg = SendGridAPIClient("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")
   response = sg.send(message)
   return render_template('usersignin.html',msg="Registration successfull. Please login using your credentials")
  except Exception as e:
   print(e)
   return render_template('usersignin.html',msg="unable to send message to your mail" )
@app.route('/usercheck',methods = ['POST', 'GET'])
def usercheck():

 if request.method == 'POST':

  email = request.form['email']
  password = request.form['password']

  sql = "SELECT * FROM USER_DATA WHERE EMAIL=? and PASSWORD= ?"
  stmt = ibm_db.prepare(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)

  if account:
   return render_template('userpanel.html')
  else:
```

```
    return render_template('usersignin.html')
    """
    sql = "SELECT USER_NAME FROM USER_DATA WHERE EMAIL=?"
    stmt = ibm_db.prepare(conn, sql)
    ibm_db.bind_param(stmt,1,email)
    ibm_db.execute(stmt)
    account = ibm_db.fetch_assoc(stmt)
    message = Mail(from_email="sivasurya1018@gmail.com",to_emails=email,subject="User registration",html_content="<p>You have been successfully registered as a user.</p>")

    try:
     sg = SendGridAPIClient("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")
     response = sg.send(message)
     return render_template('userpanel.html',msg=account)
    except Exception as e:
     print(e)
  else:
    return render_template('index.html', msg="Your login credentials are not mached with our records.Please login correctly or signup")
"""
@app.route('/adminregistration',methods = ['POST', 'GET'])
def adminregistration():
 if request.method == 'POST':

  name = request.form['name']
  email = request.form['email']
  password = request.form['password']
  confirmpassword = request.form['confirmpassword']
  ece = request.form['secretkey']

  if ece:


    sql = "SELECT * FROM ADMIN_DATA WHERE EMAIL=? "

    stmt = ibm_db.prepare(conn, sql)
```

```
    ibm_db.bind_param(stmt,1,email)
    ibm_db.execute(stmt)
    account = ibm_db.fetch_assoc(stmt)

    if account:
      return render_template('adminsignin.html', msg="You are already an admin, please login using your details")
    else:
     insert_sql = "INSERT INTO ADMIN_DATA VALUES (?,?,?,?)"
     prep_stmt = ibm_db.prepare(conn, insert_sql)
     ibm_db.bind_param(prep_stmt, 1, name)
     ibm_db.bind_param(prep_stmt, 2, email)
     ibm_db.bind_param(prep_stmt, 3, password)
     ibm_db.bind_param(prep_stmt, 4, confirmpassword)

     ibm_db.execute(prep_stmt)

     return render_template('adminsignin.html',msg="Admin registration successfull.Please login with your credentials." )
  else:
    return render_template('adminsignin.html',msg="Your secret key is invalid provide it correctly" )
@app.route('/admincheck',methods = ['POST', 'GET'])
def admincheck():

 if request.method == 'POST':

  email = request.form['email']
  password = request.form['password']

  sql = "SELECT * FROM ADMIN_DATA WHERE EMAIL=? and PASSWORD= ?"
  stmt = ibm_db.prepare(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)
  if account:
```

```
    return render_template('adminpanel.html')
  else:
    return render_template('adminsignin.html')
    """
    sql = "SELECT ADMIN_NAME FROM ADMIN_DATA WHERE EMAIL=?"
    stmt = ibm_db.prepare(conn, sql)
    ibm_db.bind_param(stmt,1,email)
    ibm_db.execute(stmt)
    account = ibm_db.fetch_assoc(stmt)

    return render_template('adminpanel.html',msg=account)



  else:
    return render_template('adminsignin.html',msg="Your login credentials are not matched with our records.Please login correctly or signup" )
"""
@app.route('/adminpanel',methods = ['POST', 'GET'])
def adminpanel():
 if request.method == 'POST':

  id = request.form['id']
  name = request.form['name']
  price = request.form['price']
  stock = request.form['stock']



  sql = "SELECT * FROM PRODUCT_DATA WHERE NAME=? and ID=? "

  stmt = ibm_db.prepare(conn, sql)
  ibm_db.bind_param(stmt,1,name)
  ibm_db.bind_param(stmt,2,id)
  ibm_db.execute(stmt)

  account = ibm_db.fetch_assoc(stmt)

  if account:
```

```
    return render_template('adminpanel.html', msg="Product already added")
  else:
    insert_sql = "INSERT INTO PRODUCT_DATA VALUES (?,?,?,?)"
    prep_stmt = ibm_db.prepare(conn, insert_sql)
    ibm_db.bind_param(prep_stmt, 1, id)
    ibm_db.bind_param(prep_stmt, 2, name)
    ibm_db.bind_param(prep_stmt, 3, price)
    ibm_db.bind_param(prep_stmt, 4, stock)

    ibm_db.execute(prep_stmt)
  return render_template('adminpanel.html',msg="Product successfully added" )
```

GITHUB:

GITHUB LINK

DEMO LINK:

Demo vedio link

Final application link:

Final link