# HX8001- PROFESSIONAL READINESS FOR INNOVATION,EMPLOYABILITY AND ENTREPRENEURSHIP

INVENTORY MANAGEMENT SYSTEM FOR RETAILERS

**TEAM ID** : PNT2022TMID02644

# TEAM MEMBERS:

SHIVANEESH A

SANKARANARAYANAN S

PAVITHRAN VV

SAMYUKTHA GS

# In partial fulfilment for the award of the degree

**of**

# BACHELOR OF ENGINEERING

**In**

# INFORMATION TECHNOLOGY

RAJALAKSHMI ENGINEERING COLLEGE

THANDALAM

ANNA UNIVERSITY : CHENNAI 600 025

NOV-DEC 2022

# TABLE OF CONTENT

|  |  |  |
| --- | --- | --- |
| **CHAPTER** | **CONTENTS** | **PAGE NO** |
| 1 | **INTRODUCTION**   * 1. PROJECT OVERVIEW   2. PURPOSE | 04  05 |
| 2 | **LITERATURE SURVEY**   * 1. EXISTING PROBLEM   2. REFERENCES   3. PROBLEM STATEMENT DEFINITION | 06  07  08 |
| 3 | **IDEATION & PROPOSED SOLUTION**   * 1. EMPATHY MAP CANVAS   2. IDEATION & BRAINSTROMING   3. PROPOSED SOLUTION   4. PROBLEM SOLUTION FIT | 09  10  10  12 |
| 4 | **REQUIREMENT ANALYSIS**   * 1. FUNCTIONAL REQUIREMENT   2. NON-FUNCTIONAL REQUIREMENTS | 13  14 |
| 5 | **PROJECT DESIGN**  5.1 DATA FLOW DIAGRAMS 5.2SOLUTION&TECHNICAL | 15  17 |

|  |  |  |
| --- | --- | --- |
|  | ARCHITECTURE  5.3 USER STORIES | 18 |
| 6 | **PROJECT PLANNING & SCHEDULING**   * 1. SPRINT PLANNING & ESTIMATION   2. SPRINT DELIVERY SCHEDUL   3. REPORTS FROM JIRA |  |
|  | 19  20  24 |
| 7 | **CODING & SOLUTIONING**   * 1. FEATURE1   2. FEATURE2   3. DATABASE SCHEMA |  |
|  | 26  26  27 |
| 8 | **TESTING**   * 1. TEST CASES   2. USER ACCEPTANCE TESTING | 28  30 |
| 9 | **RESULTS**  9.1 PERFORMANCE METRICS |  |
|  | 37 |
| 10 | **ADVANTAGES & DISADVANTAGES** | 38 |
| 11 | **CONCLUSION** | 40 |
| 12 | **FUTURE SCOPE** | 41 |

|  |  |  |
| --- | --- | --- |
| 13 | **APPENDIX**  SOURCE CODE  GITHUB & PROJECT DEMO LINK | 42  53 |

1. **INTRODUCTION**

### PROJECT OVERVIEW:

Inventory management helps companies identify which and how much stock to order at what time. It tracks inventory from purchase to the sale of goods. The practice identifies and responds to trends to ensure there's always enough stock to fulfil customer orders and proper warning of a shortage

### PURPOSE:

The primary purpose of inventory management is to ensure there is enough goods or materials to meet demand without creating overstock, or excess inventory.

# LITERATURE SURVEY

* 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.

# REFERENCES

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

* + 1. [https://www.camcode.com/asset-tags/what-is-an-inventory-management-](https://www.camcode.com/asset-tags/what-is-an-inventory-management-system/) [system/](https://www.camcode.com/asset-tags/what-is-an-inventory-management-system/)
    2. Jimmy Wales, online encyclopedia WiKipedia ,

[http://www.wikipedia.org](http://www.wikipedia.org/)

* + 1. James Gosling. Java (Programming Language) ,

[http://www.java.com](http://www.java.com/)

* + 1. Names Allaire, Netbeans-Fully-featured Java IDE,

[http://www.netbeans.org](http://www.netbeans.org/)

* + 1. 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](http://www.javaworld.com/)

* + 1. Pressman, Roger S.

“Software Engineering A Practitioner” Approch

* + 1. John Osborn , JavaBeans:

Developing Component Software in Java

* + 1. Doug Lea Concurrent Programming in Java:

Design Principles and Pattern, Addison-Wesley , November,1996

* + 1. Design Report, submitted 9th November 2012. [https://skydrive.live.com/redir?resid=2CEDE9F7F5F99604!196&authkey=](https://skydrive.live.com/redir?resid=2CEDE9F7F5F99604!196&authkey=!AO5I%20%20%20%20%20%20ghTCML6xAk8)

[!AO5I ghTCML6xAk8](https://skydrive.live.com/redir?resid=2CEDE9F7F5F99604!196&authkey=!AO5I%20%20%20%20%20%20ghTCML6xAk8)

* + 1. Testing Document, submitted 26th November 2012. [https://www.camcode.com/asset-tags/what-is-an-inventory-](https://www.camcode.com/asset-tags/what-is-an-inventory-management-)

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

# 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.



# IDEATION & PROPOSED SOLUTION

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

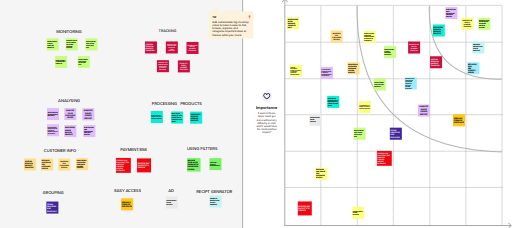
# EMPATHY MAP CANVAS

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.

****

# IDEATION & BRAINSTORMING

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



# 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.

# PROBLEM SOLUTION FIT

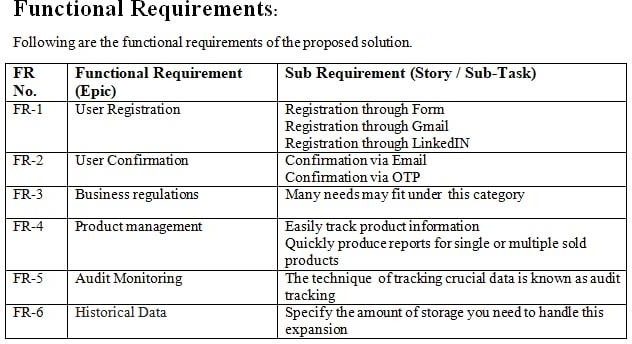
* + - To develop a system that will enhance the monitoring of the sales and inventory
    - To develop a module that can generate monthly sales and inventory

report.

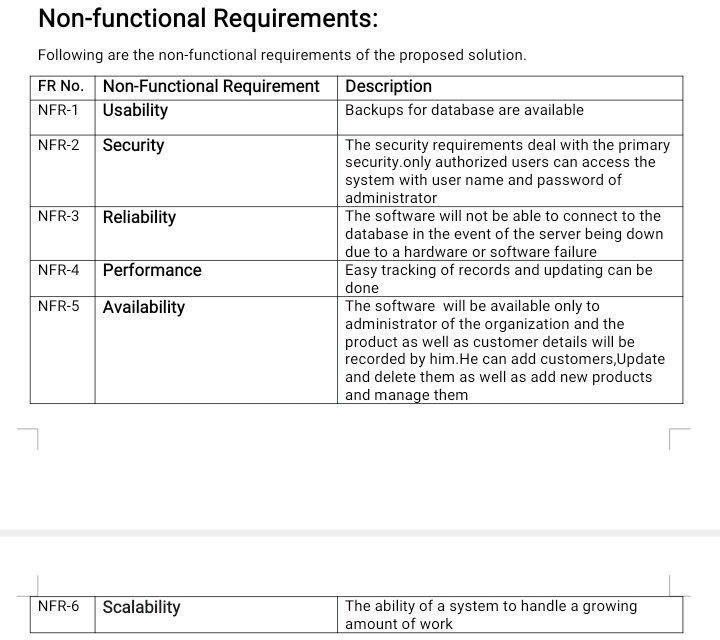
* + - To develop security in terms of keeping the records of the inventory
    - To develop a system that can monitor the stocks inventory in a fast and efficient manner.
    - To accurately record, compute and produce a report of sales.

# 4.REQUIREMENT ANALYSIS

**4.1. FUNCTIONAL REQUIREMENTS**



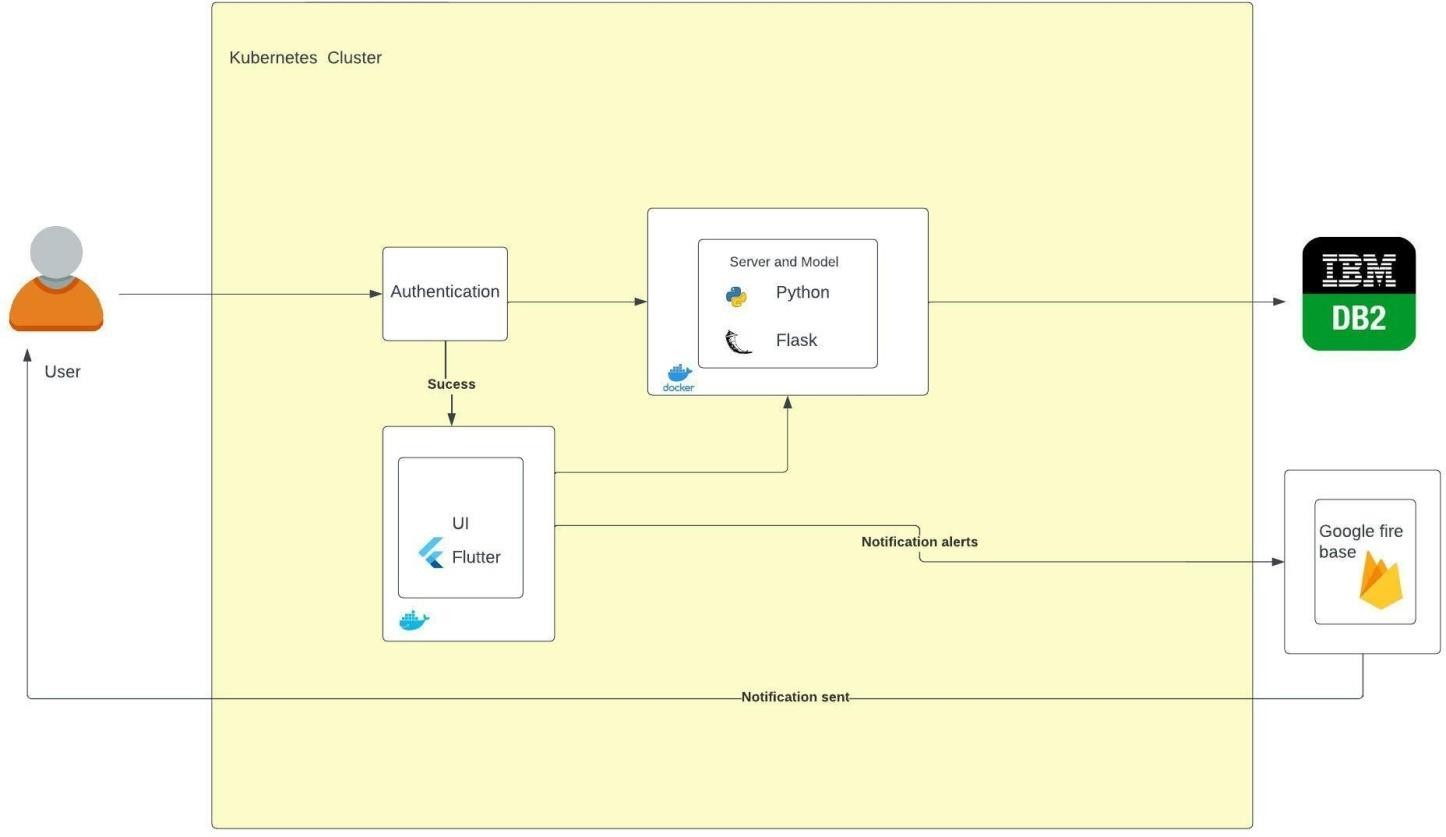
# 4.2 NON-FUNCTIONAL REQUIREMENTS

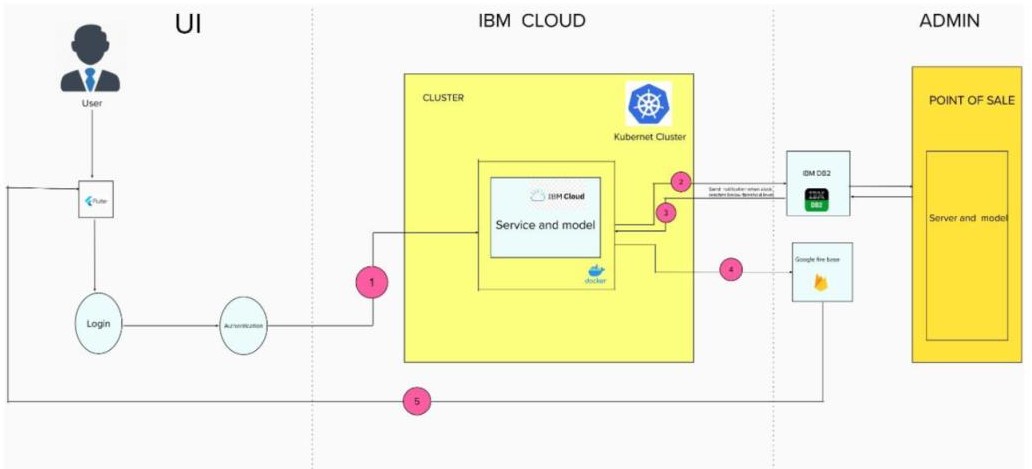


**5. PROJECT DESIGN**

# 5.1.DATAFLOWDIdataflow.pngAGRAM

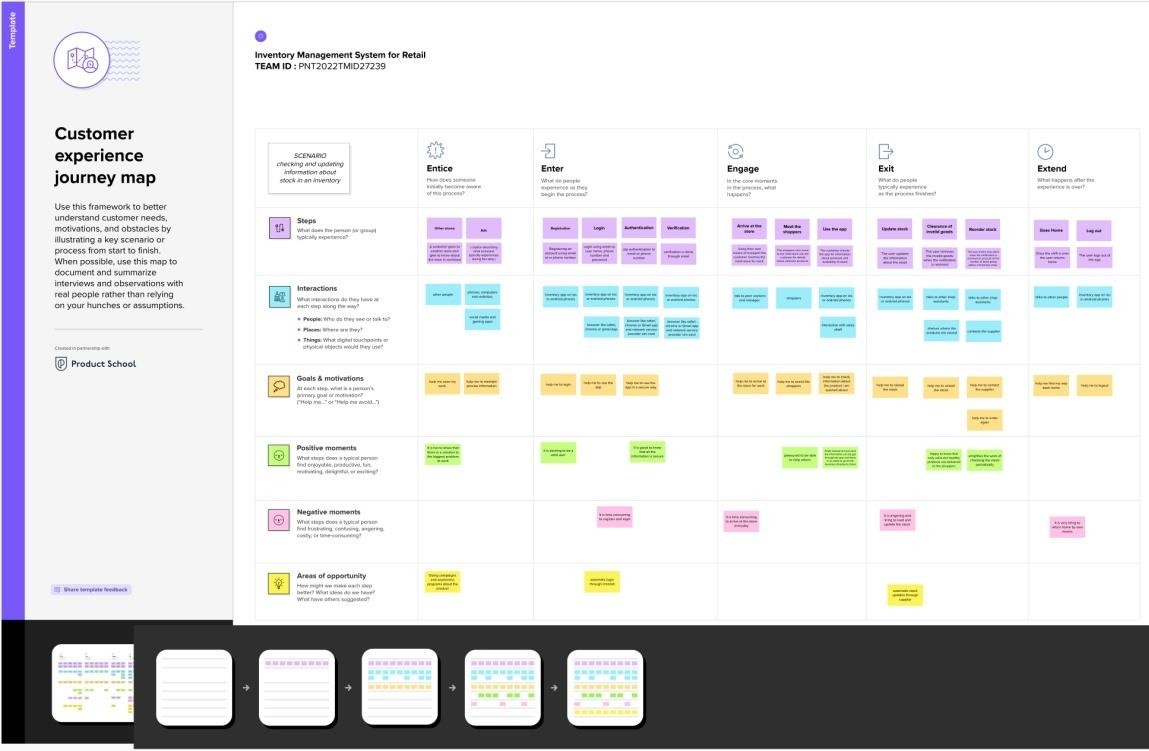
# SOLUTION & TECHNICAL ARCHITECTURE



TECHNICAL ARCHITECTURE:

# 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.



# 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

.

1. 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

.

1. We wrote Queries to connect IBM Cloud Database

.

1. Finished all the Fetching and Posting Stuff of IBM Cloud Database

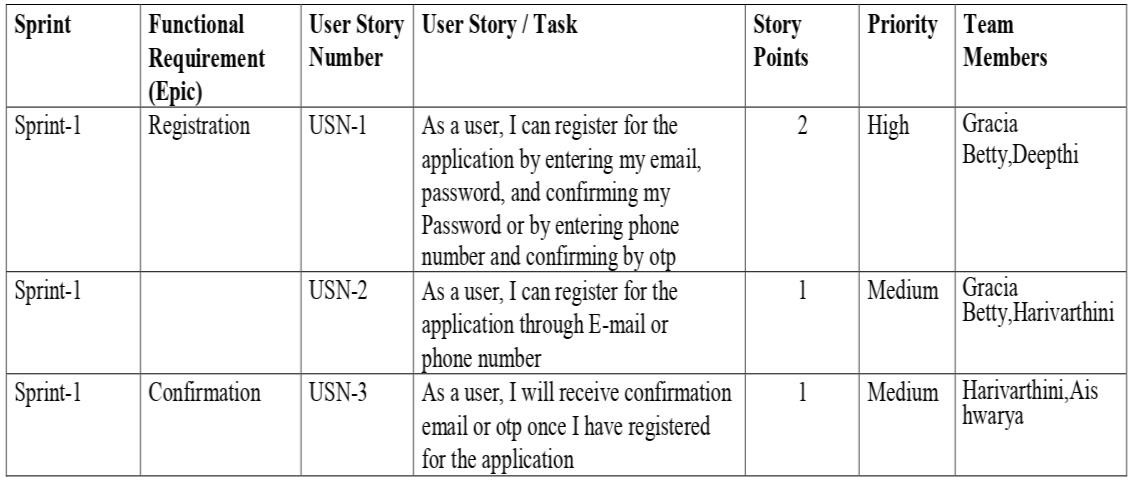
Integration.

# Sprint 3:

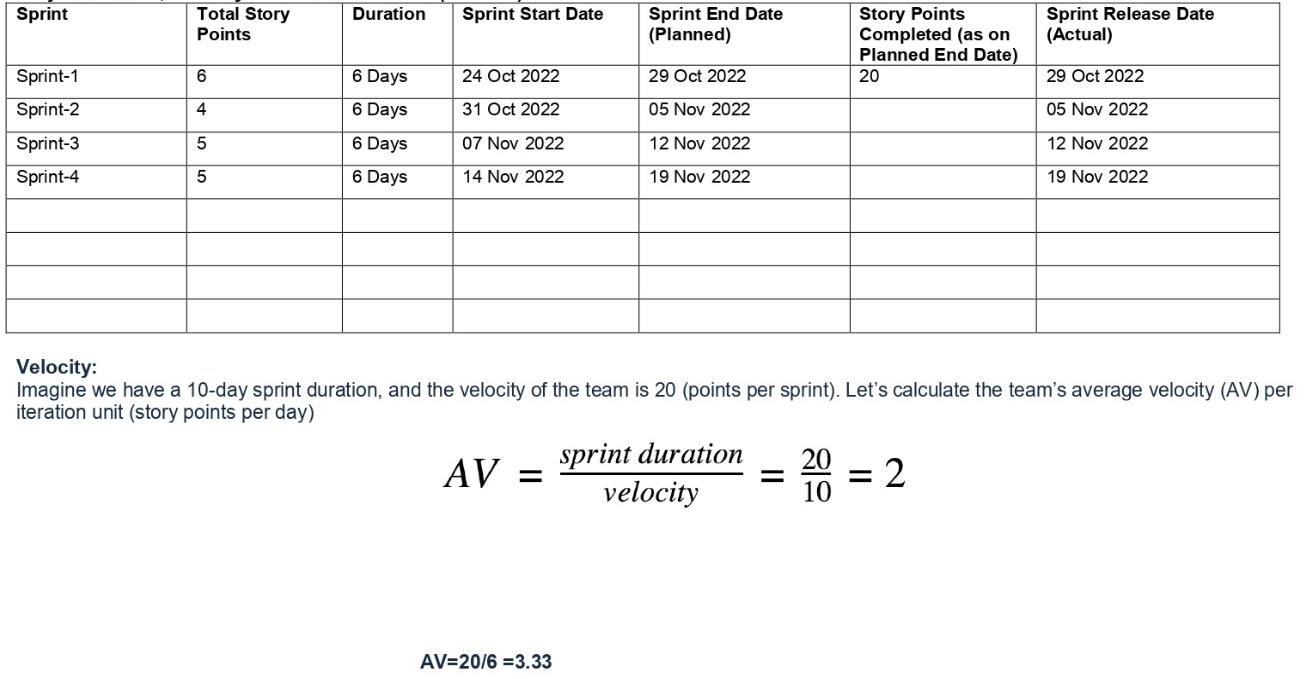
* 1. Integration of Send grid into our application

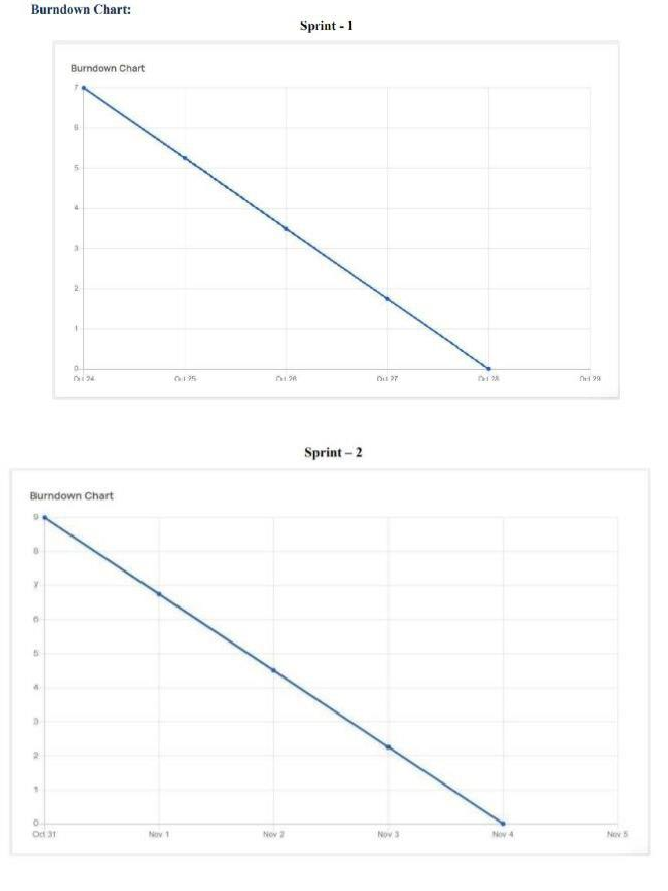
# Sprint 4:

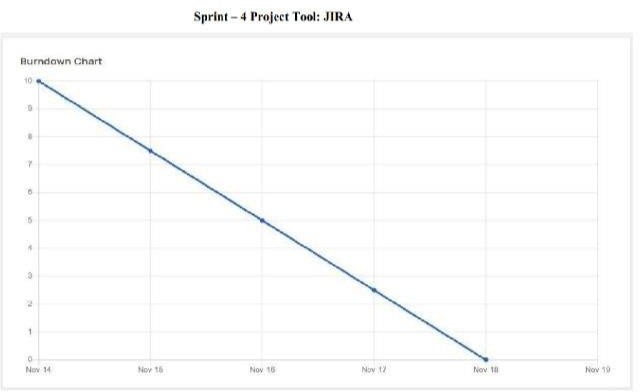
1. Deploying the application using Docker and Kubernetes

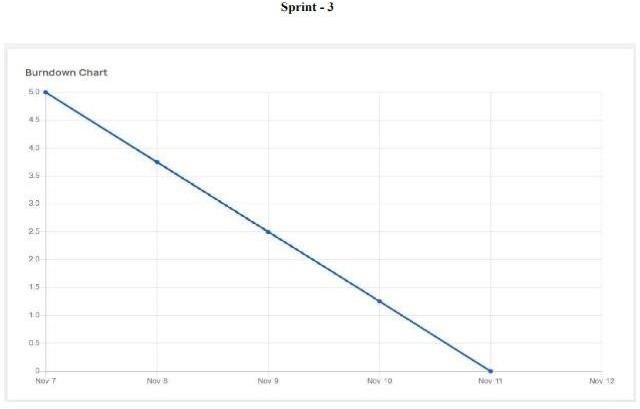


**6.2.Sprint Delivery Schedule:**



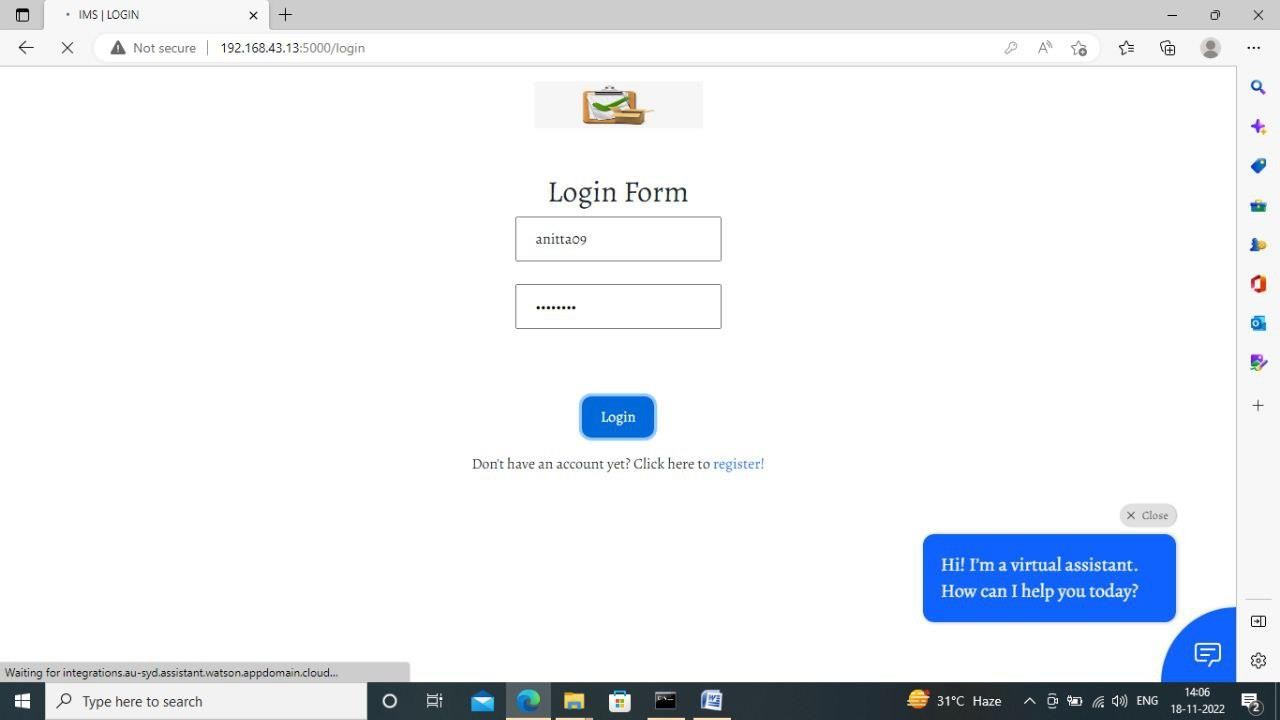




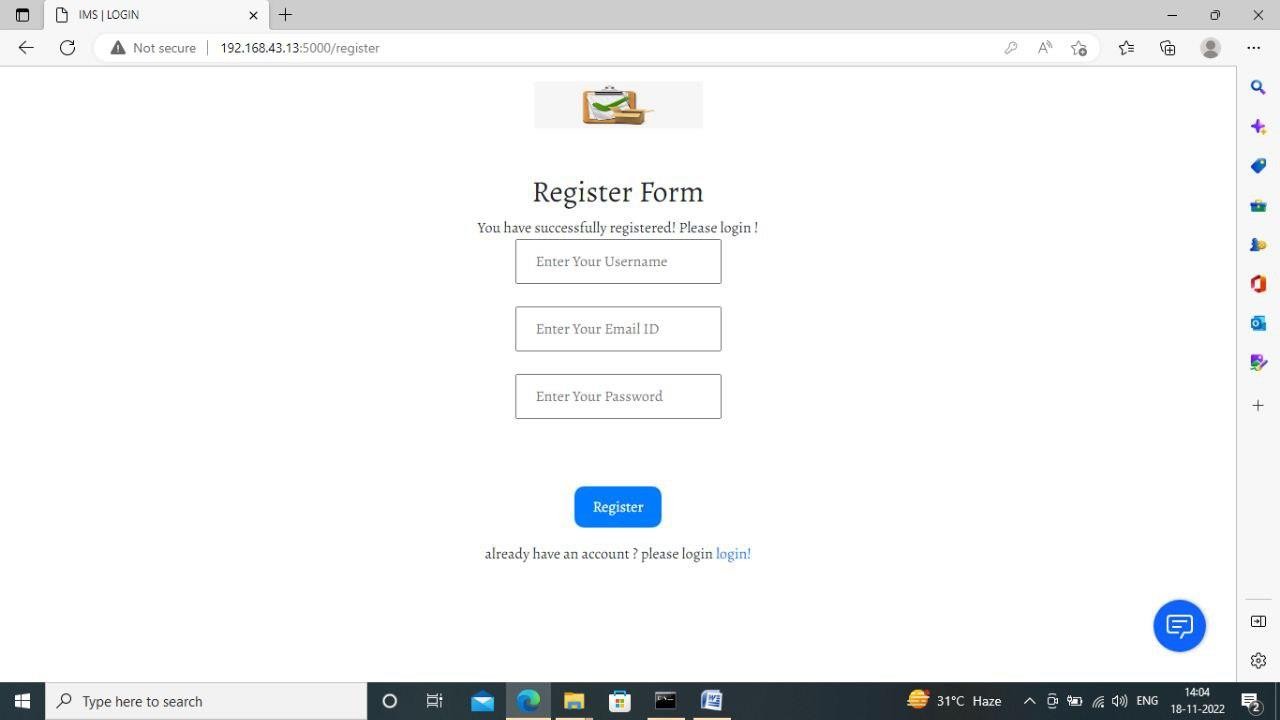


# 7.CODING & SOLUTIONING

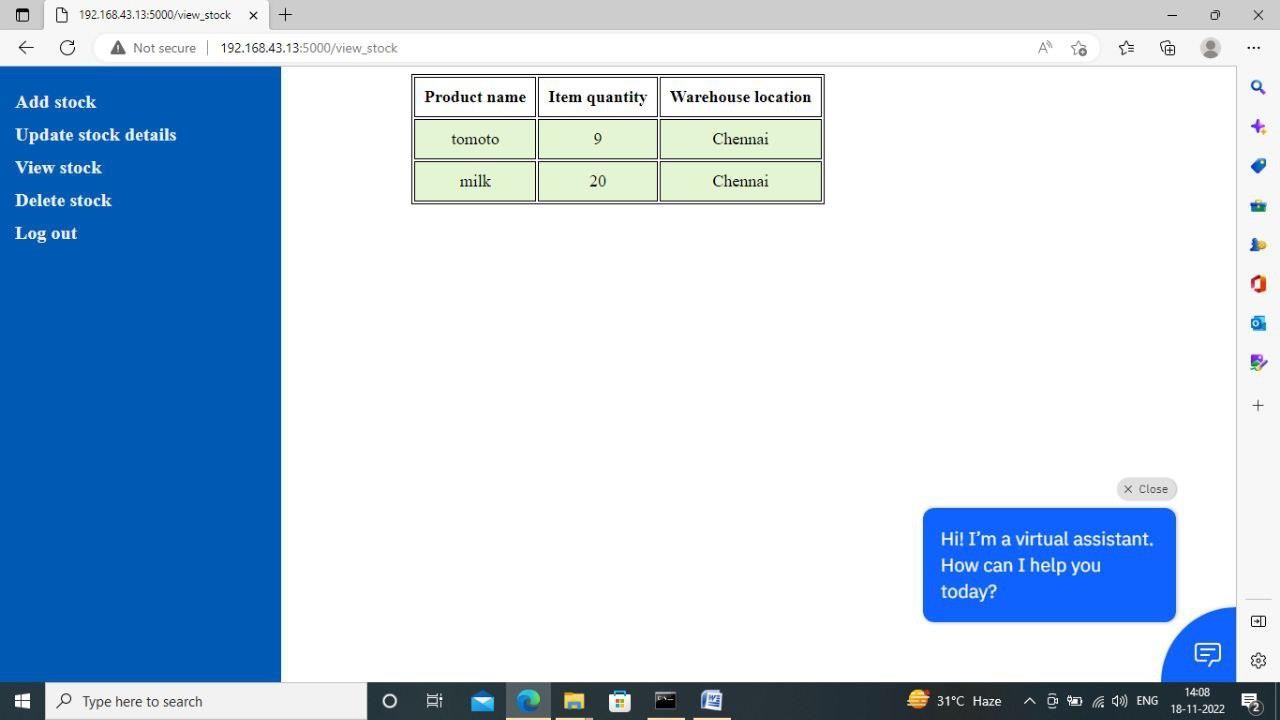
* 1. **FEATURE1**



# FEATURE 2

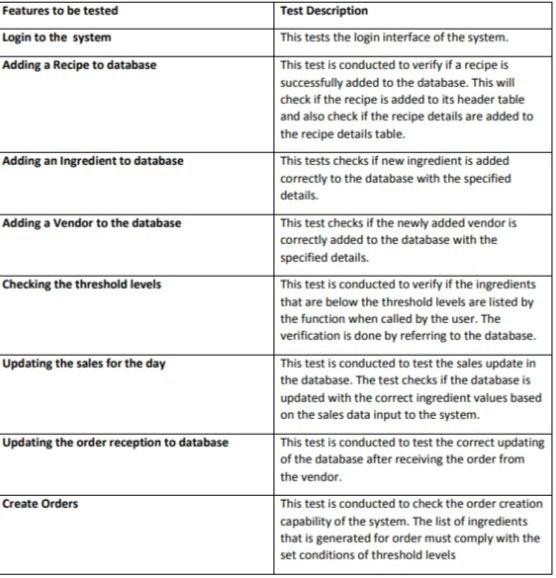


* 1. **DATABASE SCHEMA**



# TESTING

* 1. **TEST CASE**



# 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

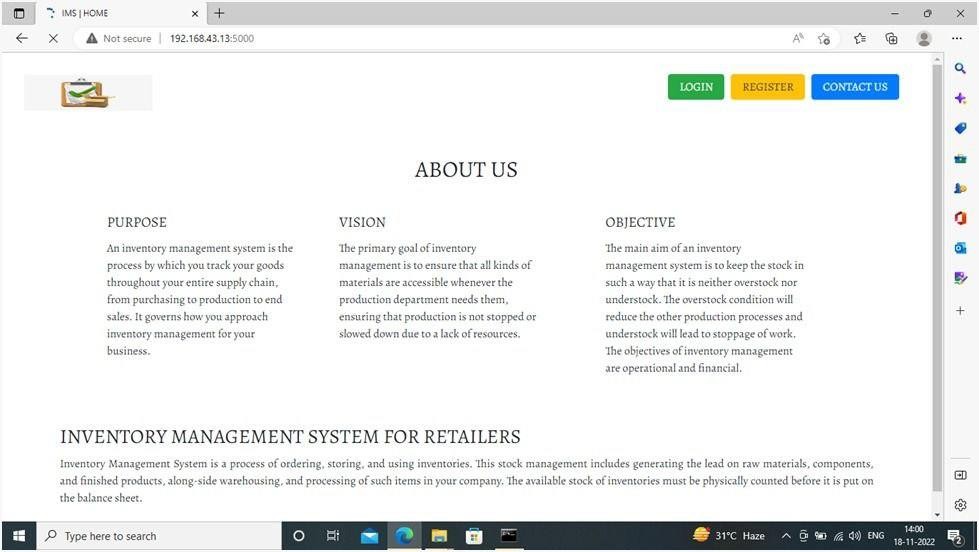
Case 1: Test the Recipe list box. Case 2: Test the quantity text field..

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

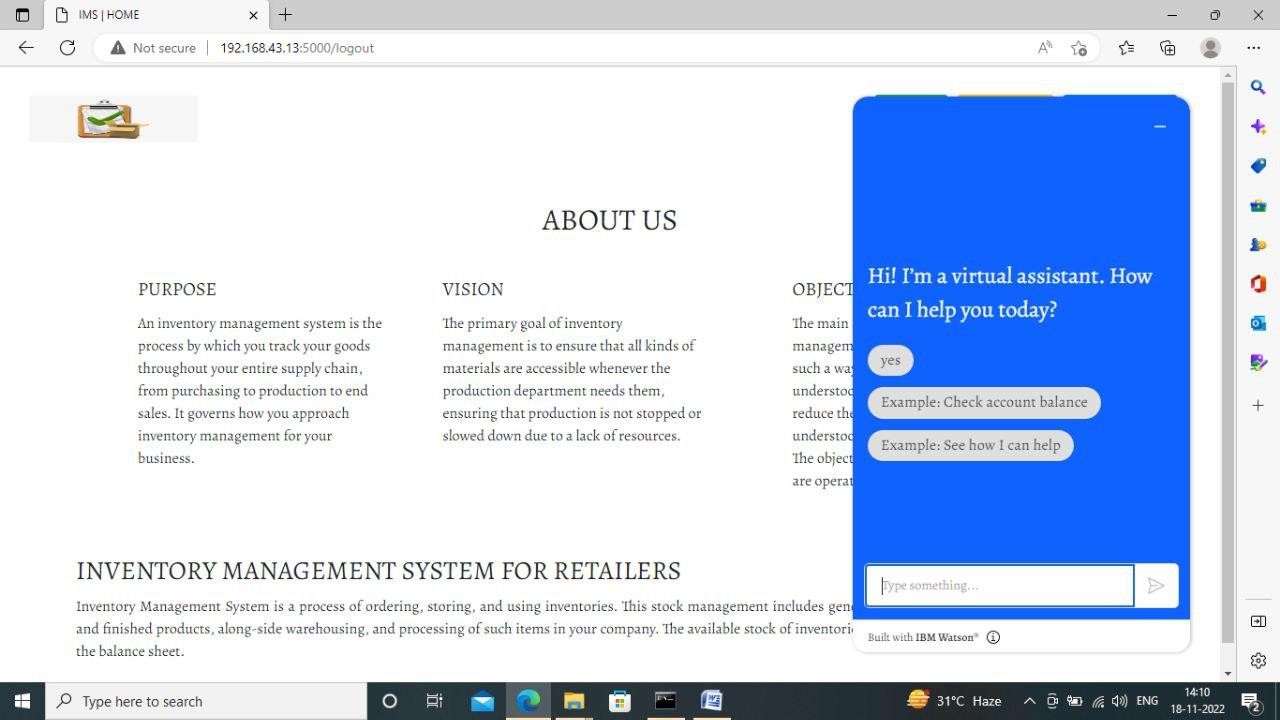
Case 4: Test if the details are updated to the database when requested.

# RESULTS

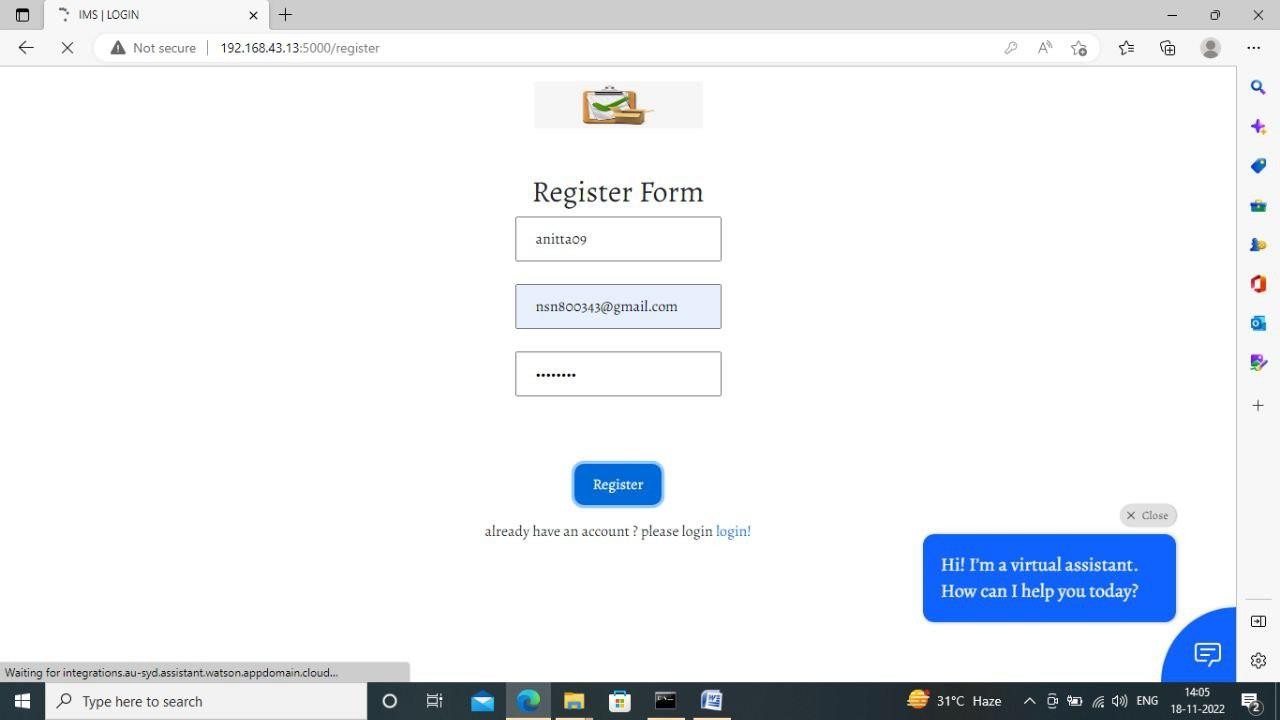
**HOME PAGE**



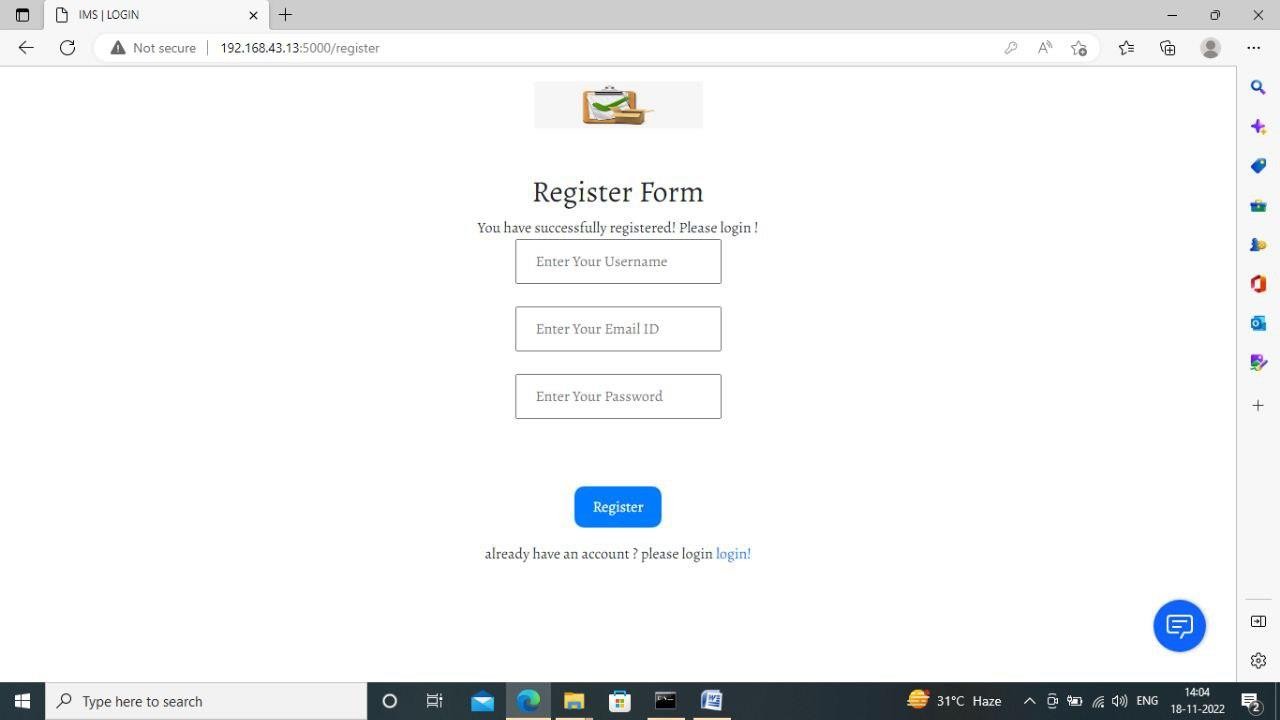
# HOMEPAGE WITH VIRTUAL ASSISTANT



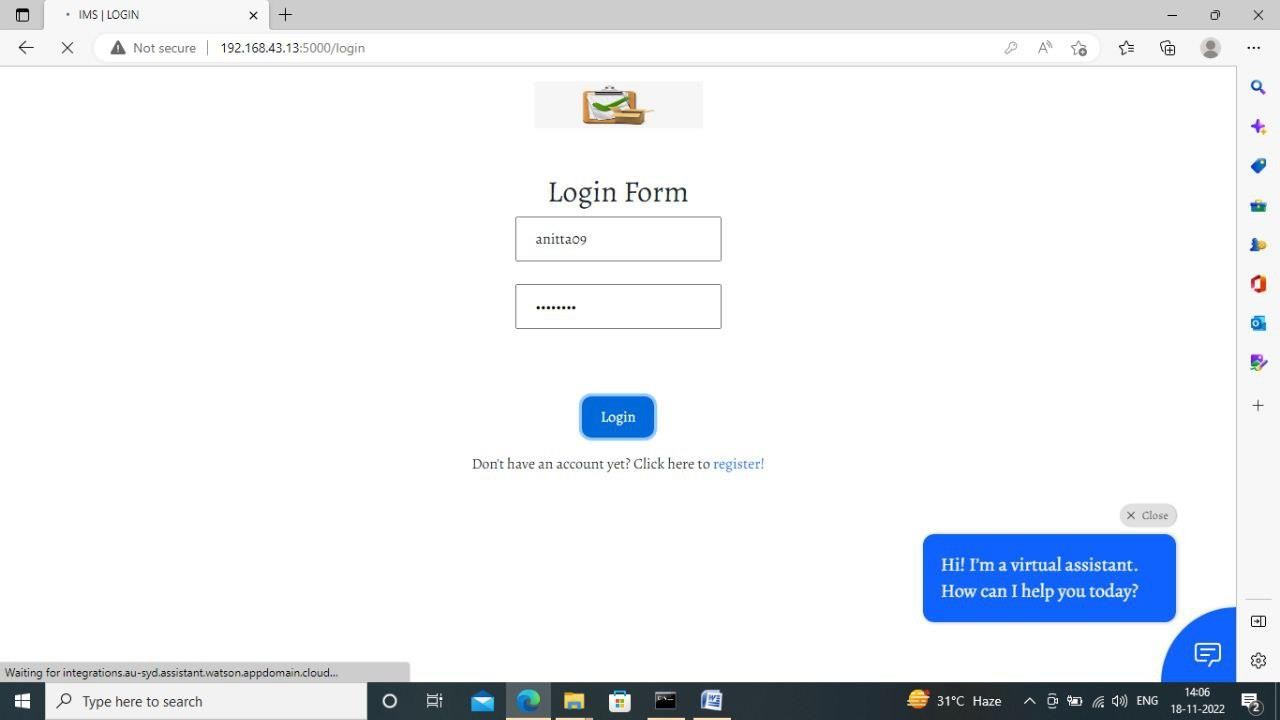
**REGISTRATION FORM**

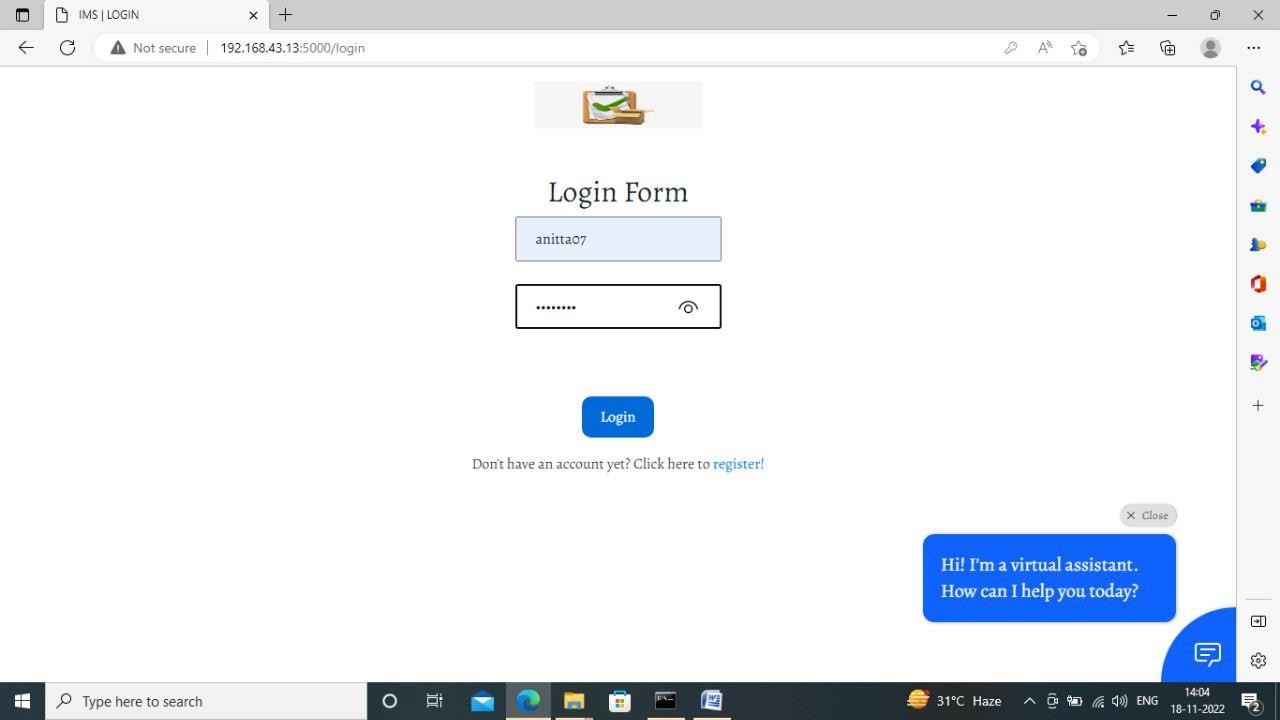


# REGISTRATION SUCCESSFULLY

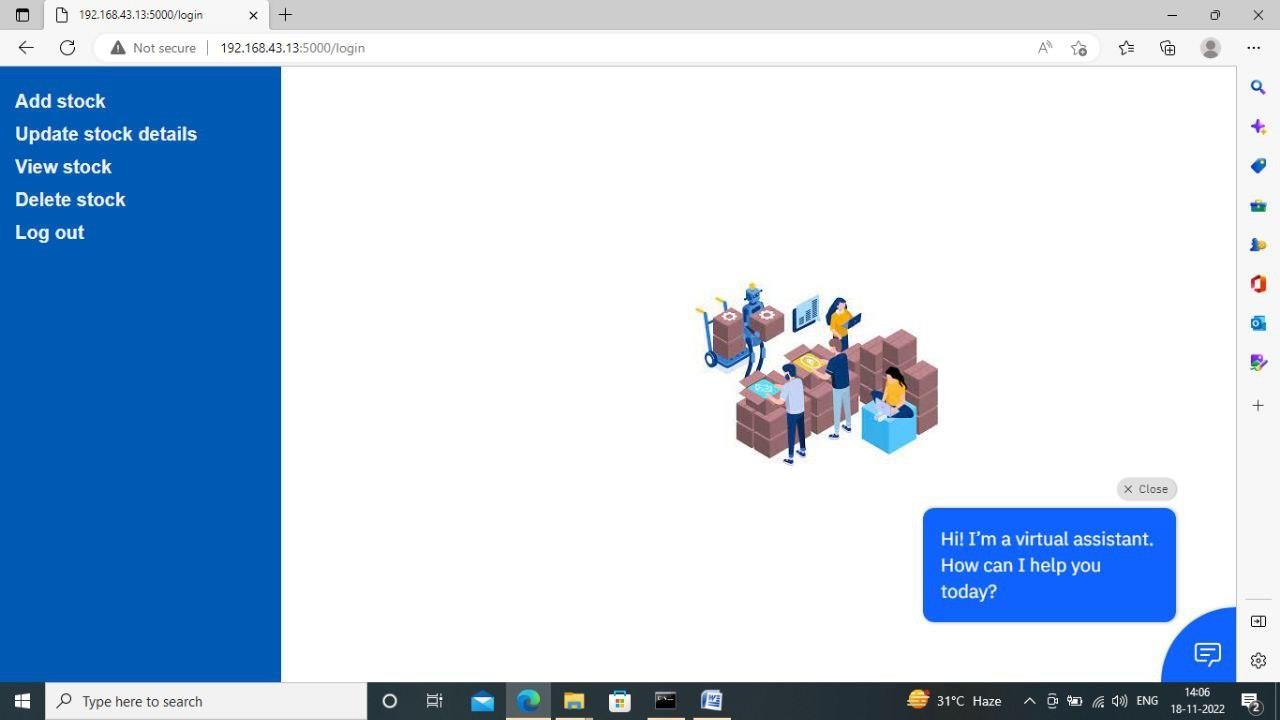


**LOGIN FORM**

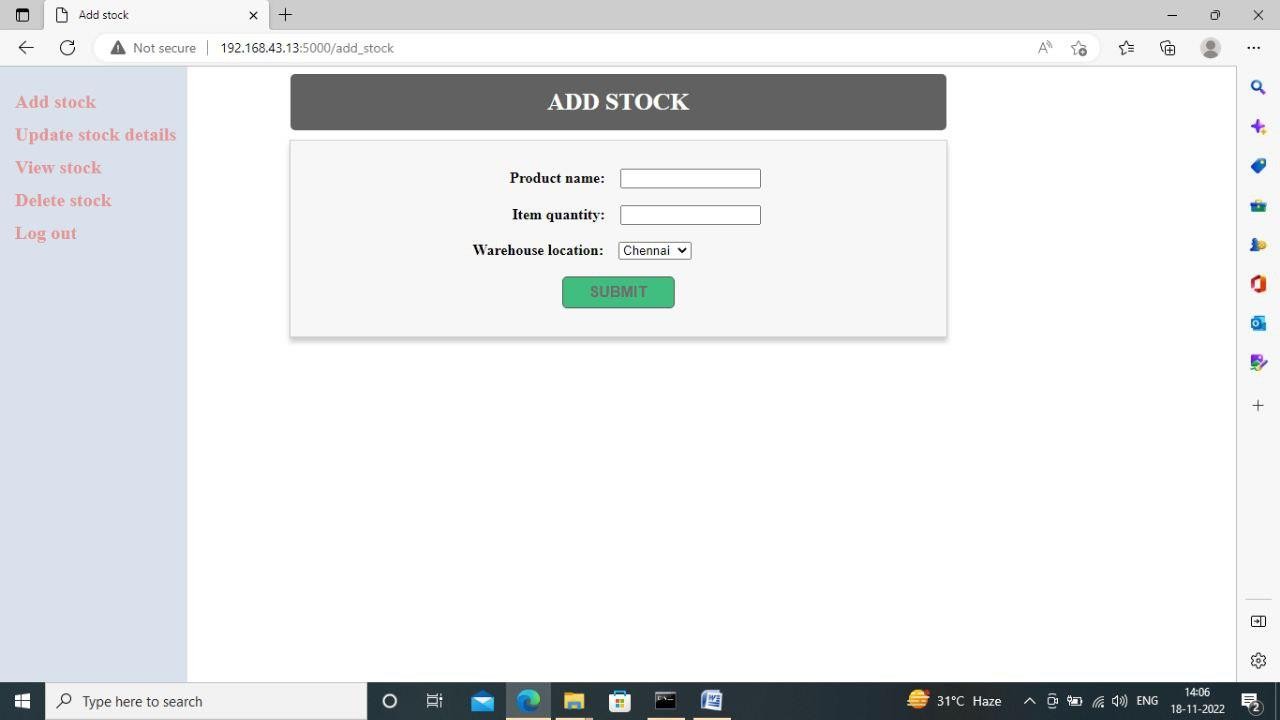




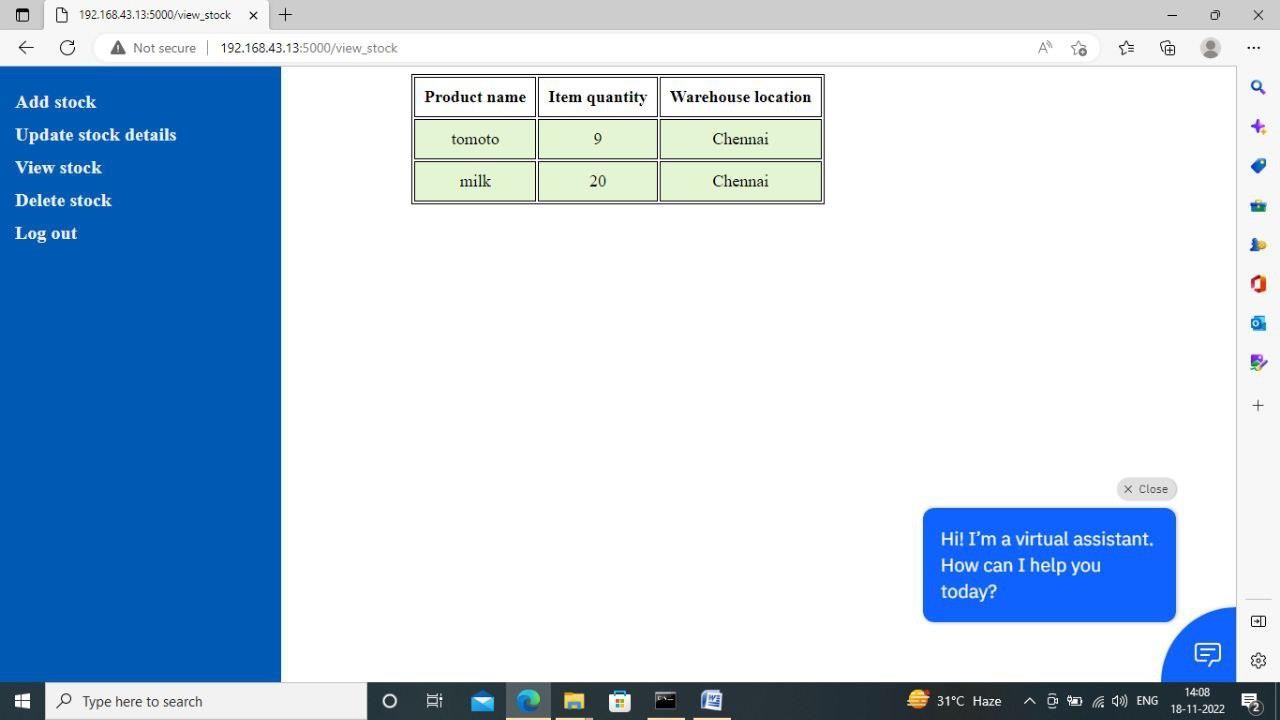
# DASHBOARD



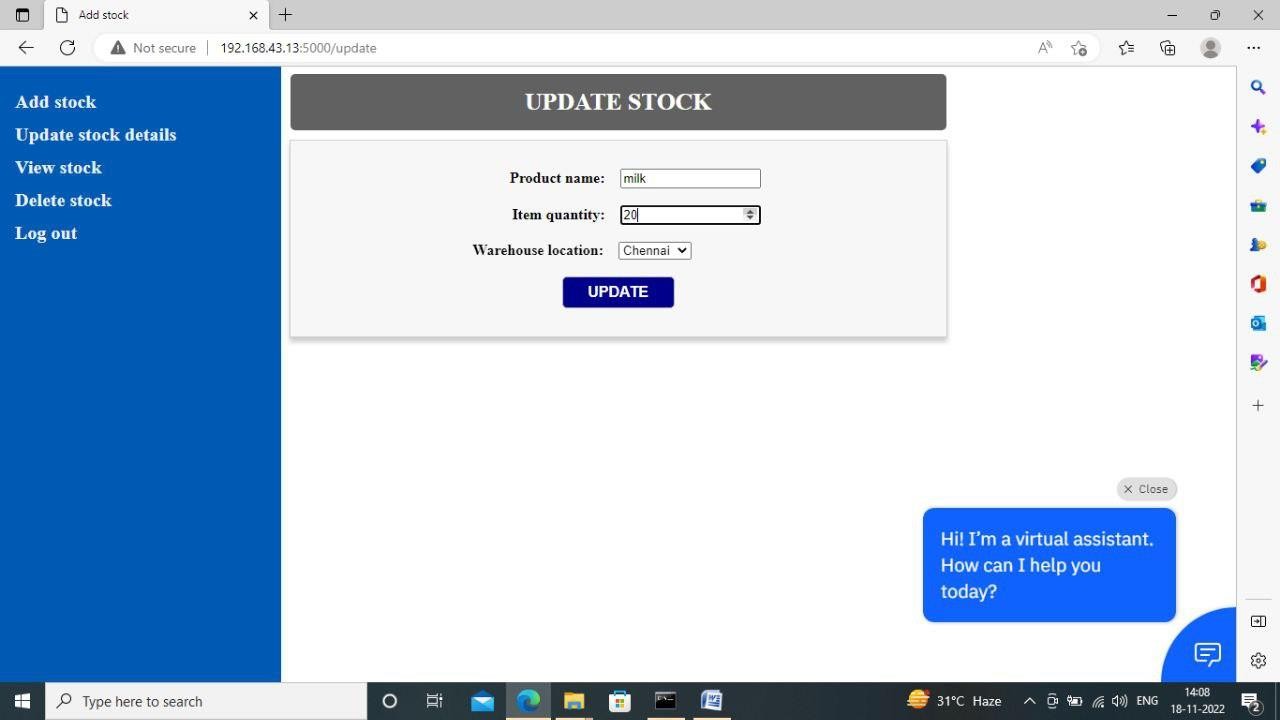
**ADDSTOCK**



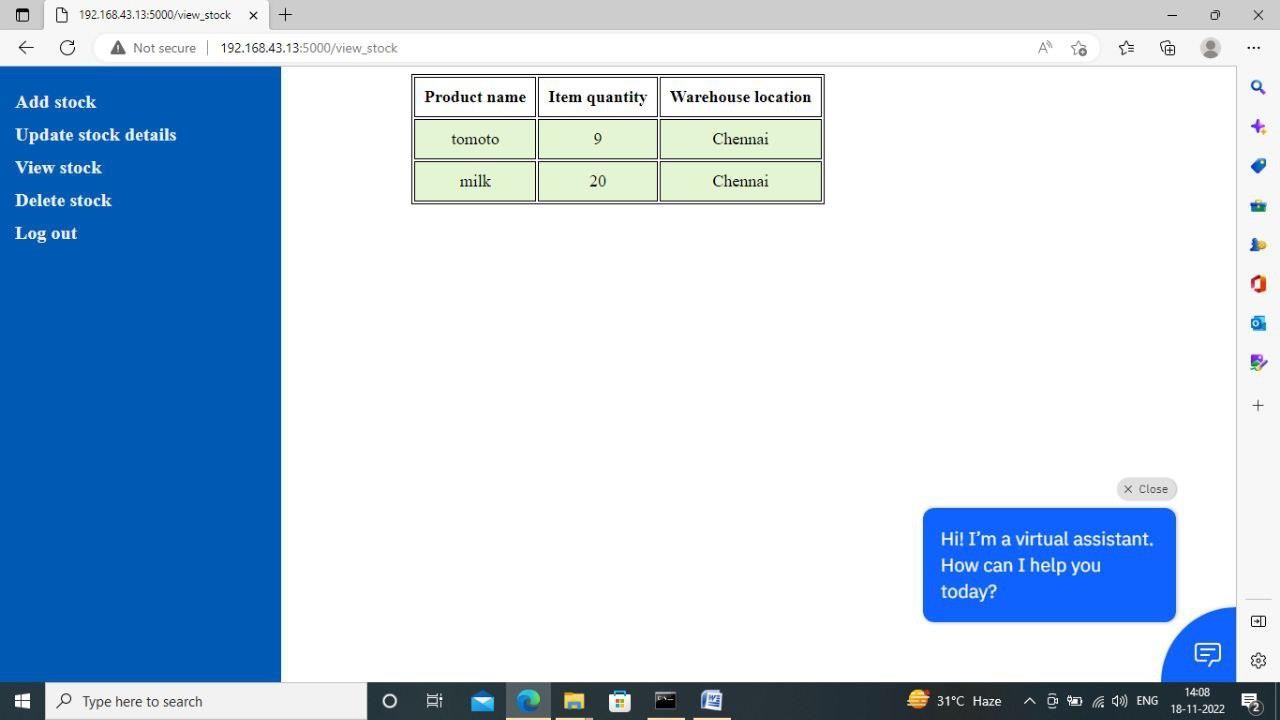
# VIEW STOCK



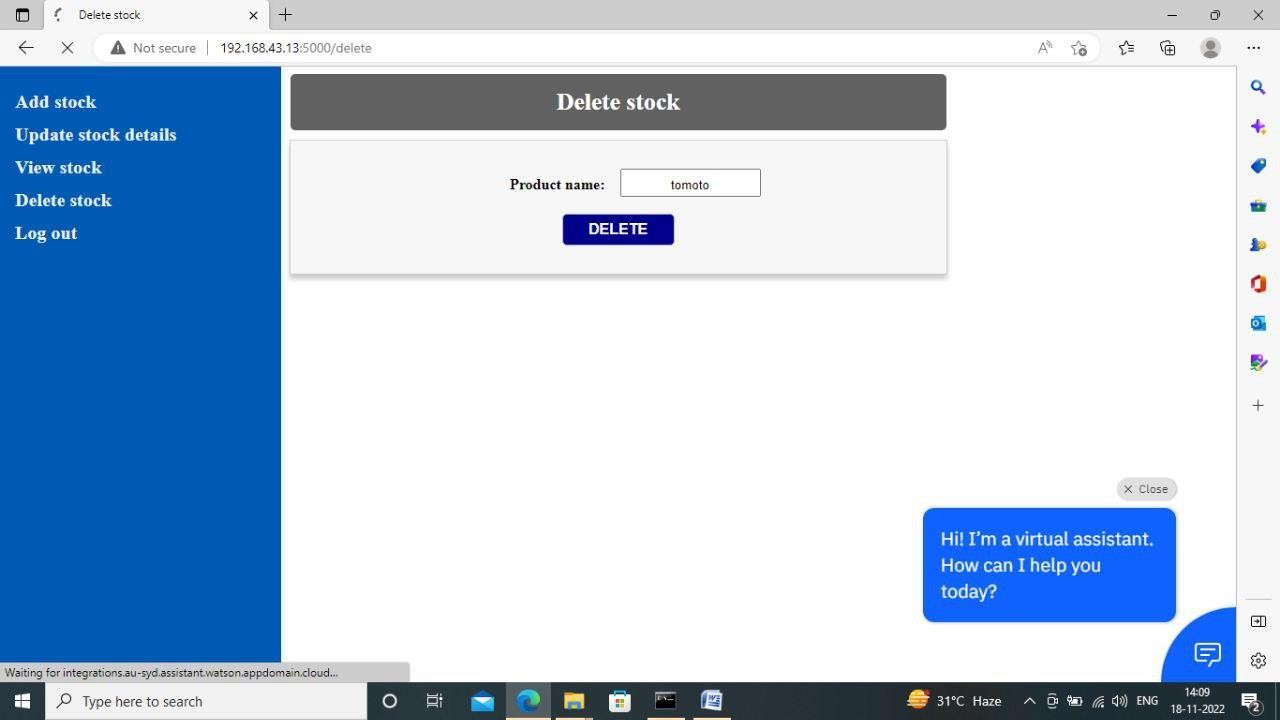
**UPDATE STOCK**

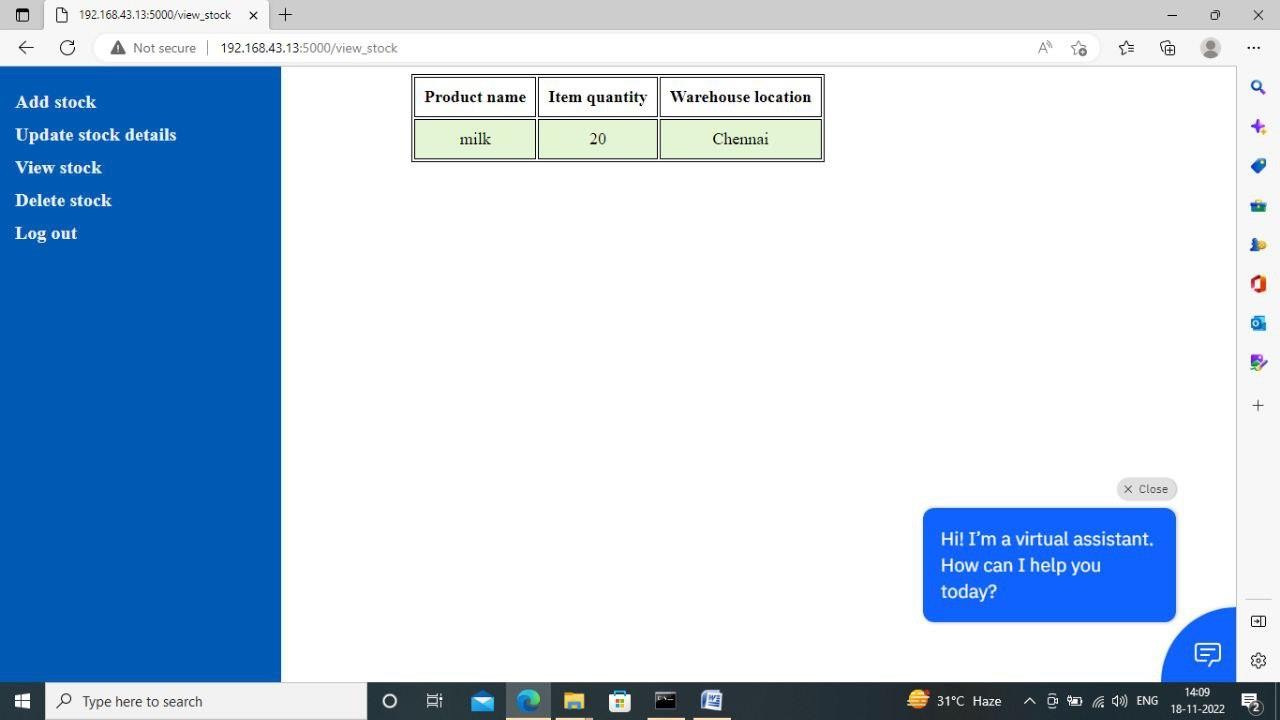


# UPDATESTOCKDETAILS

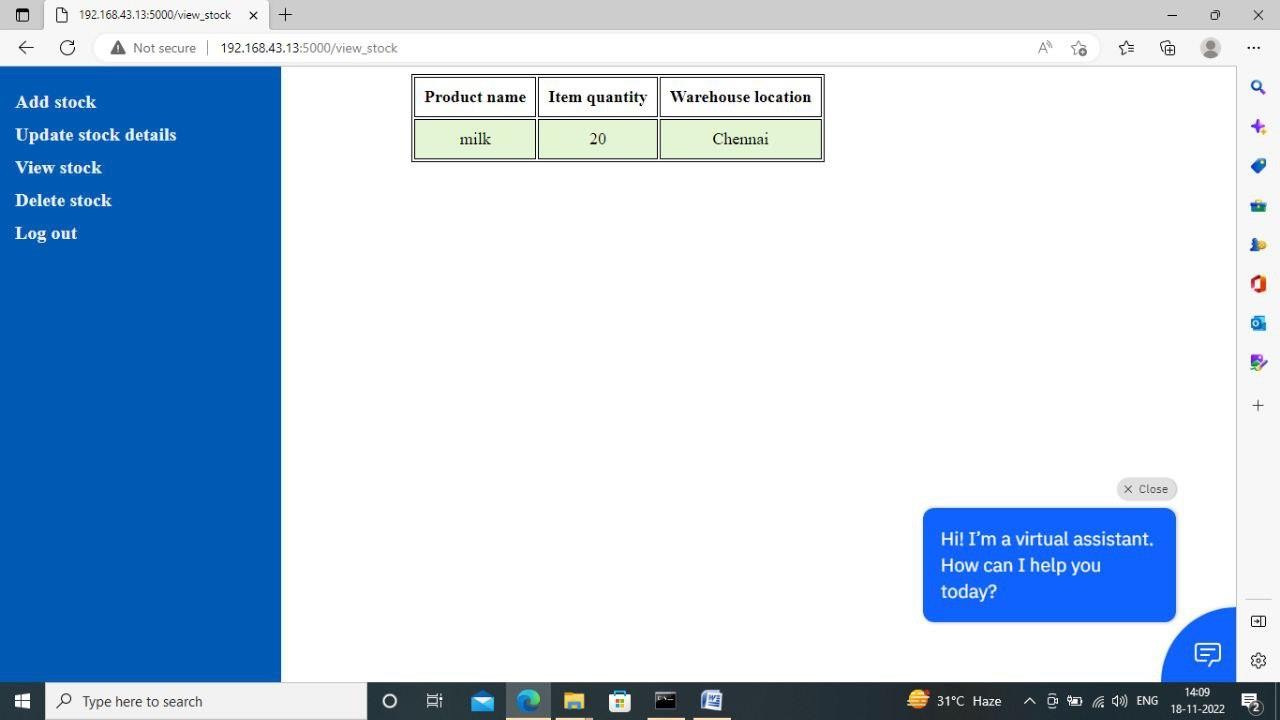


**DELETE STOCK**





# LOGOUT



* 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.

# 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.

# 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.

.

# 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.

# APPENDIX

**App.py**

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

import ibm\_db import sqlite3 as sql import re

app = Flask(name) app.secret\_key = 'a'

conn = ibm\_db.connect("DATABASE=bludb;HOSTNAME=1bbf73c5-d84a- 4bb0-85b9- ab1a4348f4a4.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud;PORT=322 86;SECURITY=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=w jy24066;PWD=3w6H3sui635KMvWX",'','')

print(conn)

print("Connecting Successful!!!!!!!!")

@app.route('/') def homer()

return render\_template('home.html')

@app.route('/login',methods =['GET', 'POST']) def login():

global userid msg = ''

if request.method == 'POST' : username = request.form['username'] password = request.form['password']

sql = "SELECT \* FROM users WHERE username =? AND password=?" stmt = ibm\_db.prepare(conn, sql)

ibm\_db.bind\_param(stmt,1,username) ibm\_db.bind\_param(stmt,2,password) ibm\_db.execute(stmt)

account = ibm\_db.fetch\_assoc(stmt) print (account)

if account: session['loggedin'] = True

session['id'] = account['USERNAME'] userid= account['USERNAME'] session['username'] = account['USERNAME'] msg = 'Logged in successfully !'

return render\_template('dashboard.html', msg = msg) else:

msg = 'Incorrect username / password !' return render\_template('login.html', msg = msg)

@app.route('/register', methods =['GET', 'POST']) def registet():

msg = ''

if request.method == 'POST' :

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

sql = "SELECT \* FROM users WHERE username =?" stmt = ibm\_db.prepare(conn, sql) ibm\_db.bind\_param(stmt,1,username) ibm\_db.execute(stmt)

account = ibm\_db.fetch\_assoc(stmt) print(account)

if account:

msg = 'Account already exists !'

elif not re.match(r'[^@]+@[^@]+\.[^@]+', email): msg = 'Invalid email address !'

elif not re.match(r'[A-Za-z0-9]+', username):

msg = 'name must contain only characters and numbers !' else:

insert\_sql = "INSERT INTO users VALUES (?, ?, ?)" prep\_stmt = ibm\_db.prepare(conn, insert\_sql) ibm\_db.bind\_param(prep\_stmt, 1, username)

ibm\_db.bind\_param(prep\_stmt, 2, email)

ibm\_db.bind\_param(prep\_stmt, 3, password) ibm\_db.execute(prep\_stmt)

msg = 'Please fill out the form !' if request.method == 'POST':

msg = 'You have successfully registered! Please login !' return render\_template('register.html', msg = msg)

@app.route('/add\_stock',methods=['GET','POST'])

def add\_stock(): msg=''

if request.method == "POST": prodname=request.form['prodname'] quantity=request.form['quantity'] warehouse\_location=request.form['warehouse\_location'] sql='SELECT \* FROM product WHERE prodname =?' stmt = ibm\_db.prepare(conn, sql) ibm\_db.bind\_param(stmt,1,prodname) ibm\_db.execute(stmt)

acnt=ibm\_db.fetch\_assoc(stmt) print(acnt)

if acnt:

msg='Product already exits!!' else:

insert\_sql='INSERT INTO product VALUES (?,?,?)' pstmt=ibm\_db.prepare(conn, insert\_sql) ibm\_db.bind\_param(pstmt,1,prodname) ibm\_db.bind\_param(pstmt,2,quantity) ibm\_db.bind\_param(pstmt,3,warehouse\_location) ibm\_db.execute(pstmt)

msg='You have successfully added the products!!' return render\_template("dashboard.html")

else:

msg="fill out the form first!"

return render\_template('add\_stock.html',meg=msg)

@app.route('/delete\_stock',methods=['GET','POST']) def delete\_stock():

if(request.method=="POST"): prodname=request.form['prodname']

sql2="DELETE FROM product WHERE prodname=?" stmt2 = ibm\_db.prepare(conn, sql2) ibm\_db.bind\_param(stmt2,1,prodname) ibm\_db.execute(stmt2)

flash("Product Deleted", "success") return render\_template("dashboard.html")

@app.route('/update\_stock',methods=['GET','POST']) def update\_stock():

mg=''

if request.method == "POST": prodname=request.form['prodname'] quantity=request.form['quantity'] quantity=int(quantity) print(quantity)

print(type(quantity)) warehouse\_location=request.form['warehouse\_location'] sql='SELECT \* FROM product WHERE prodname =?' stmt = ibm\_db.prepare(conn, sql) ibm\_db.bind\_param(stmt,1,prodname) ibm\_db.execute(stmt)

acnt=ibm\_db.fetch\_assoc(stmt)

print(acnt)

if acnt:

insert\_sql='UPDATE product SET quantity=?,warehouse\_location=?

WHERE prodname=? '

pstmt=ibm\_db.prepare(conn, insert\_sql) ibm\_db.bind\_param(pstmt,1,quantity) ibm\_db.bind\_param(pstmt,2,warehouse\_location) ibm\_db.bind\_param(pstmt,3,prodname) ibm\_db.execute(pstmt)

mg='You have successfully updated the products!!' limit=5 print(type(limit))

if(quantity<=limit):

("Please update the quantity of the product {}, Atleast {} number of pieces must be added!".format(prodname,10))

return render\_template("dashboard.html",meg=mg)

else:

mg='Product not found!!'

else:

msg="fill out the form first!"

return render\_template('update\_stock.html',meg=msg)

@app.route('/view\_stock') def view\_stock():

sql = "SELECT \* FROM product"

stmt = ibm\_db.prepare(conn, sql) result=ibm\_db.execute(stmt) print(result)

products=[]

row = ibm\_db.fetch\_assoc(stmt) print(row)

while(row):

products.append(row)

row = ibm\_db.fetch\_assoc(stmt) print(row)

products=tuple(products) print(products)

if result>0:

return render\_template('view.html', products = products) else:

msg='No products found'

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

@app.route('/delete') def delete():

return render\_template('delete\_stock.html')

@app.route('/update') def update():

return render\_template('update\_stock.html')

@app.route('/logout') def logout():

session.pop('loggedin', None) session.pop('id', None) session.pop('username', None) return render\_template('home.html')

if name == 'main': app.run(host='0.0.0.0')

# home.html

<!DOCTYPE html>

<html>

<head>

<meta name="viewport" content="width=device-width, initial-scale=1">

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font- awesome/4.7.0/css/font-awesome.min.css">

<style> body {

font-family: "Lato", sans-serif;

}

/\* Fixed sidenav, full height \*/

.sidenav { height: 100%; width: 300px; position: fixed; z-index: 1;

top: 0;

left: 0;

background-color: #0059b3; overflow-x: hidden;

padding-top: 20px;

}

/\* Style the sidenav links and the dropdown button \*/

.sidenav a{

padding: 6px 8px 6px 16px; text-decoration: none;

font-size: 20px;

color: rgb(239, 239, 239); display: block;

border: none; background: none; width: 100%;

text-align: left; cursor: pointer; outline: none;

}

/\* On mouse-over \*/

.sidenav a:hover{ color: #111;

}

/\* Some media queries for responsiveness \*/ @media screen and (max-height: 450px) {

.sidenav {padding-top: 15px;}

.sidenav a {font-size: 18px;}

}

</style>

</head>

<body>

> Deeps:

<div class="sidenav">

<a href="{{url\_for('add\_stock') }}"><strong>Add stock<strong></a>

<a href="{{url\_for('update') }}"><strong>Update stock details<strong></a>

<a href="{{url\_for('view\_stock') }}"><strong>View stock<strong></a>

<a href="{{url\_for('delete')}}"><strong>Delete stock<strong></a>

<a href="{{url\_for('logout') }}"><strong>Log out<strong></a>

</div>

<nav>

<script> window.watsonAssistantChatOptions = {

integrationID: "4bd6f313-33d4-4e87-8825-22b90b8e3c2c", // The ID of this integration.

region: "au-syd", // The region your integration is hosted in.

serviceInstanceID: "60e1396a-421f-4091-b39a-a23a546843e8", // The ID of your service instance.

onLoad: function(instance) { instance.render(); }

};

setTimeout(function(){

const t=document.createElement('script'); t.src="https://web-

chat.global.assistant.watson.appdomain.cloud/versions/" + (window.watsonAssistantChatOptions.clientVersion || 'latest') + "/WatsonAssistantChatEntry.js";

document.head.appendChild(

});

</script>

</nav>

</body>

</html>

# GITHUB:

**https://github.com/IBM-EPBL/IBM-Project-22616-1659855139**

# DEMOLINK:

https://drive.google.com/file/d/1AK8O8wViuLqYAF6woPEGLZX5y80MBjtA/view?usp=share\_link