Inventory Management System For Retailers

PTN2022TMID27239

1. **INTRODUCTION**
    1. Project Overview
    2. Purpose
2. **LITERATURE SURVEY**
    1. Existing problem
    2. References
    3. Problem Statement Definition
3. **IDEATION & PROPOSED SOLUTION**
    1. Empathy Map Canvas
    2. Ideation & Brainstorming
    3. Proposed Solution
    4. Problem Solution fit
4. **REQUIREMENT ANALYSIS**
    1. Functional requirement
    2. Non-Functional requirements
5. **PROJECT DESIGN**
    1. Data Flow Diagrams
    2. Solution & Technical Architecture
    3. User Stories
6. **PROJECT PLANNING & SCHEDULING**
    1. Sprint Planning & Estimation
    2. Sprint Delivery Schedule
    3. Reports from JIRA
7. **CODING & SOLUTIONING**
    1. Feature 1
    2. Feature 2
8. **TESTING**
    1. Test Cases
    2. User Acceptance Testing
9. **RESULTS**
    1. Performance Metrics
10. **ADVANTAGES & DISADVANTAGES**
11. **CONCLUSION**
12. **FUTURE SCOPE**
13. **APPENDIX**

Source Code

GitHub & Project Demo Link

1. INTRODUCTION

1.1 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

1.2 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.

2. LITERATURE SURVEY

2.1 Existing problem:

MISMANAGED ORDER MANAGEMENT: Managing customer order to avoid overselling and running out of stock is one of the most difficult tasks. Delivering the orders on time and addressing their complaints if the 7 order has some issues will have a big role to play in the reviews and rating a brand is going to get.

EXPANDING PRODUCT RANGE: Growing a product line and newly set up warehouse demand effective management of inventory stocks. Manually updating the stock list and tracking orders without real-time data will end up in mismanagement. All inventory managers need to view orders, and shipping details, and track the inventory stocks to allocate deliveries with the highest demands.

LACK OF A CENTRALIZED INVENTORY HUB: Imagine switching multiple tabs for customer order details and tracking real time data. This leads to inventory managers getting frustrated and slow delivery of results. Without a unified dashboard, all conversation, order information, and delivery agents tracking will not flow into a single inbox.

ANALYZING THE MARKET DEMAND: A look into the market of how the product is doing is very important as the demand accelerates the production followed by growth in inventory stocks. No analysis of the most selling or demand areas will end up having product shortages in the market leading to lesser customer satisfaction and losing out bon brand value.

OUTDATED PRODUCTS: Updating products over time happens inevitably to keep the products fresh and stay 8 relevant to the trends with matching buyer's expectations in the market.so, in this process at the old products that are unsold need to be recorded for easy clearance and need to make way for the new ones.

S.no	TITLE	AUTHOR	ABSTRACT	MERITS	DEMERITS
1	Inventory management for retail companies: A literature review and current trends	Cinthya Vanessa Muñoz Macas, Jorge Andrés Espinoza Aguirre, Mario Peña	In recent years, the correct management of inventories has become a fundamental pillar for achieving success in enterprises. Unfortunately, studies suggesting the investment and adoption of advanced inventory management and control systems are not easy to find. In this context, this article aims to analyze and present an extensive literature concerning inventory management, containing multiple definitions and fundamental concepts for the retail sector. A systematic literature review was carried out to determine the main trends and indicators of inventory management in Small and Medium-sized Enterprises (SMEs). This research covers five years, between 2015 and 2019, focusing specifically on the retail sector. The primary outcomes of this study are the leading inventory management systems and models, the Key Performance Indicators (KPIs) for their correct management, and the benefits and challenges for choosing or adopting an efficient inventory control and management system. Findings	1) All the KPIs identified allow knowing the effectiveness of inventory control and management carried out within retail companies. 2) The product availability is related to the inventory information provided to the customer, through which the customer verifies the service quality. 3) Price calculations can be presented in real-time. 4) RFID stock counts allow inventory levels to be evaluated every day considering each stock line in every area of the store. 5) This item-level tagging tool is able to reduce the technology breach and give the retailers both the accuracy and the ease of use which are needed in order to help their merchandising plan and store display performance. 6)field of vision is not needed for the item registration, various products are able to be registered with a single can, also tickets can be read from quite a great range. 7)In the retail store,there are four main ideas on which the procedures are	1) A retail store must have the same data in all its records, that is, the data that has been recorded in the information system must be the same data that is physically held.This is necessary due to continuous inconsistencies that exist between the physical inventory record and the inventory that appears in the system, incurring operational consequences. 2) Among the strategies used by retailers to minimize the effect on operational activities caused by inventory, several different errors can be detected, such as storing additional items or increasing the frequency of restocking of stores with the purpose of maintaining a high level of inventory. 3)A lack of products can be caused by various factors, including differences between product costs, which creates the possibility of a shortage of an expensive product and an excess of cheap products.

			indicate that SMEs do not invest resources in sophisticated systems; instead, a simple Enterprise Resource Planning (ERP) system or even programs such as Excel or manual inventories are mainly used.	based on: improving stock exactness, out of stock management, products localization, and loss recognition.	4) Retailers suffer from product misplacement problems. 5) One of the causes of inventory inaccuracy is making incorrect deliveries, driving an increase in the return of products . 6)Inventory problems imply low adaptability and a lack of functionality in the retailer's SC. 7)inaccuracy in the inventory record affects the operational performance of a retailer. 8)Poor service level results from having inadequate inventory control parameters.
2	Towards Intelligent Retail: Automated On-Shelf Availability Estimation Using a Depth Camera	Annalisa Milella	Efficient management of on-shelf availability and inventory is a key issue to achieve customer satisfaction and reduce the risk of profit loss for both retailers and manufacturers. Conventional store audits based on physical inspection of shelves are labor-intensive and do not provide reliable assessment. This paper describes a novel framework for automated shelf monitoring, using a consumer-grade depth sensor. The aim is to develop a low-cost embedded system for early detection of out-of-stock situations with particular regard	1) In the last decade, advanced sensor-based technologies mainly using Auto-ID systems, weight sensors and imaging devices have been proposed for automatic stock monitoring and inventory. 2) a novel framework for online shelf monitoring using a depth sensor is proposed. It can generate a 3D point cloud of the shelf and products therein. 3) In order to cope with the high variability of store environments, the use of machine learning	1)If OOS conditions occur repeatedly, customer satisfaction is reduced with potentially negative effects for both retailers and manufacturers. 2)weight sensors entail high installation costs. 3)sensors can only determine the number of products stacked on the shelf without accounting for possible product misplacements, as they do not allow for product identification and tracking

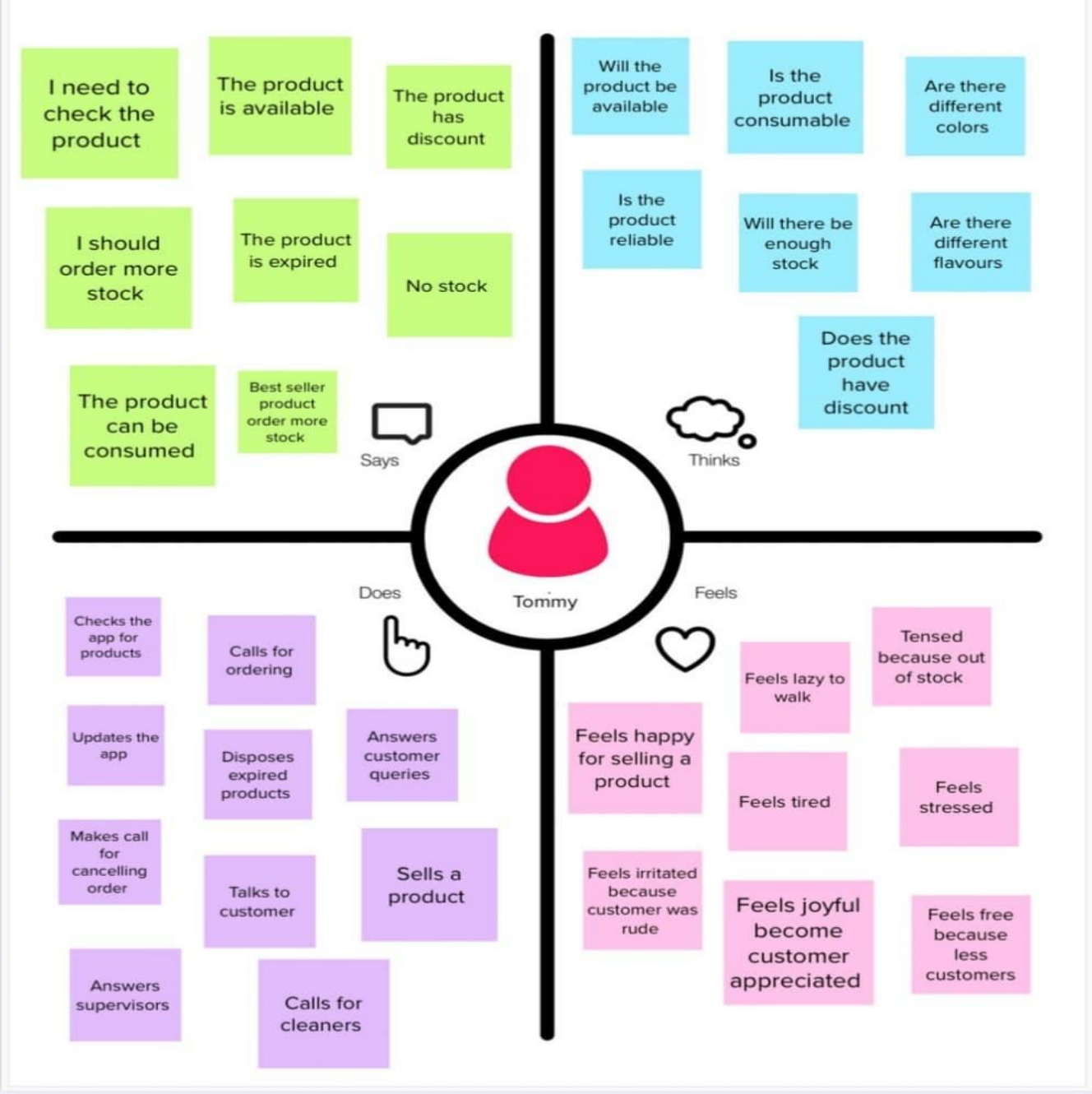
			to perishable goods stored in countertop shelves, refrigerated counters, baskets or crates. The proposed solution exploits 3D point cloud reconstruction and modelling techniques, including surface fitting and occupancy grids, to estimate product availability, based on the comparison between a reference model of the shelf and its current status. No a priori knowledge about the product type is required, while the shelf reference model is automatically learnt based on an initial training stage. The output of the system can be used to generate alerts for store managers, as well as to continuously update product availability estimates for automated stock ordering and replenishment and for e-commerce apps. Experimental tests performed in a real retail environment show that the proposed system is able to estimate the on-shelf availability percentage of different fresh products with a maximum average discrepancy with respect to the actual one of about 5.0%.	techniques has been proposed, to ensure higher robustness and accuracy,	
3	Inventory Management Optimization Model with	Yusuf Sutanto, Riyanarto Sarno	Although the physical internet inventory control	1)WMS improves inventory control by minimizing the manual	1)Based on the qualitative data collected, key drivers leading to

	Database Synchronization through Internet Network (A Simulation Study)		model better than the classical model, but the model of inventory management with database synchronization better than the physical Internet. With the aproach database synchronization, several hubs can be combined into one hubs, and the plant can also sending goods directly to the retailer. In addition to optimizing transport and inventory, this analysis allows to choose a dynamic source when an order is placed: source substitution. Although this calculation is working on a computer simulation, the main intent of this paper is to define new research model inventory controlling better than classical inventory model and physical internet inventory model, which is inventory management model in synchronized database.	operation because it is associated with auto-ID data capture technology. 2) To obtain a high level of supply chain visibility, responsiveness, and flexibility, IoT is the best technology available. 3) IoT technology allows the reduction in the lead-time between data captured and real-time decision-making, enabling the supply chain to react to any dynamic changes on a real-time basis. 4) IoT enables remote management of operations, better collaboration with partners, and provides accurate information for more effective decision-making.	the poor performance of thereverse logistics operationsgaps were identified. 2) Poor traceability among returns. 3) No visibility and No integration the existing WMS. 4) Loading Error and Handling issues. 5) Delayed Pick up and Counting errors
4	Case Study on an Android App for Inventory Management System with Sales Prediction for Local Shopkeepers in India	Tejal Tandel, Sayali Wagal, Nisha Singh, Rujata Chaudhari, Vishal Badgujar	a mobile application that provides all the features of a point-of-sale system as well as gives future sales insights. It will enable shopkeepers to manage their current product purchases and invoicing. The predictive sales analysis will help them to modify their investments on products and supplies thereby ensuring maximum profits. If a shop houses relevant products that cater to	1)good percentage of people in India have access to smartphones and that percentage will greatly increase in the coming 2-3 years. With such favorable circumstances, an Android app is ought to flourish and attract a wider customer base over a period of time. Thus it is advantageous to have a mobile application	1) The technique used in this paper for data mining and prediction reports isfuzzy logic. 2) Fuzzy logic isused when the outcome is uncertain.

			customer needs, its customer reach will increase. The Economic Times published an article in the May of 2019, which stated that the number of smartphone users in India is expected to rise by 84% to 859 million by 2022 from 468 million in 2017. It is safe to assume that a large population of shop owners will have smartphones in the following years. Hence, equipping the local shopkeepers with a mobile application will prove instrumental since it will give them exposure to all the aforementioned benefits.	2) customers will avail of the experience of accessing the right products at the right time and will stay informed about new product 3) the app will not only assist in bringing about social empowerment and development but will also present profitable business opportunities to app development companies. 4) the future scopes of this app is that the processing can be taken over the cloud so that the app consumes less memory space but functions speedily and efficiently.	
5	Streamlining Reverse Logistics through IoT driven Warehouse Management System	The process of managing the return of goods (Reverse logistics) is critical for the company, yet the most undermanaged business function. From the operational process, prospective warehouse management and 3PL (third party logistics party) play a crucial role in managing RL (Reverse Logistics) process. In the entire process, the WMS	Leena Wanganoo	1) Omni-channel can be effective and responsive to customer needs because it exploits the strengths of both online and onsite retail channels of the supply chain 2) Omni-channel retailing refers to the use of a variety of distribution channels to fulfill the customers' orders.	1) The process of return is complex and there are challenges faced byall the key players inthe process 2) The process of reverse logistics managed manually managed on legacy systems. Retailer pays least attention to post-purchase activities like reverse logistics management. Poor 3)Management of reverse logistics leads – loss in revenue and customer dissatisfaction.

3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas

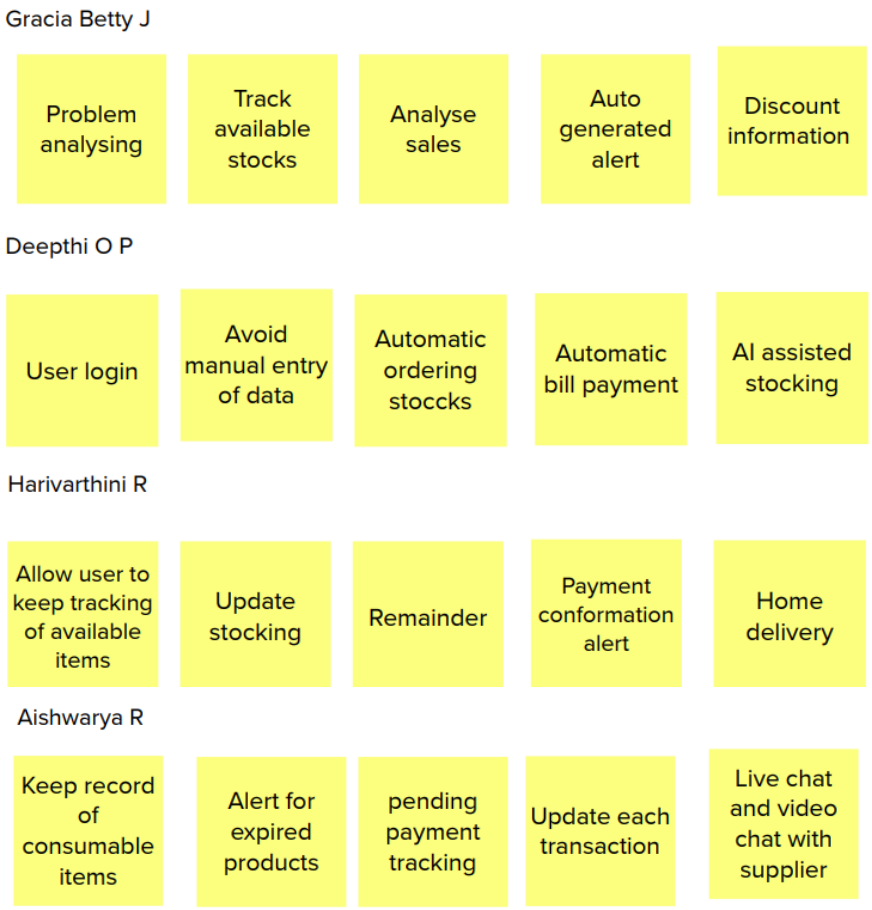


3.2 Ideation & Brainstorming

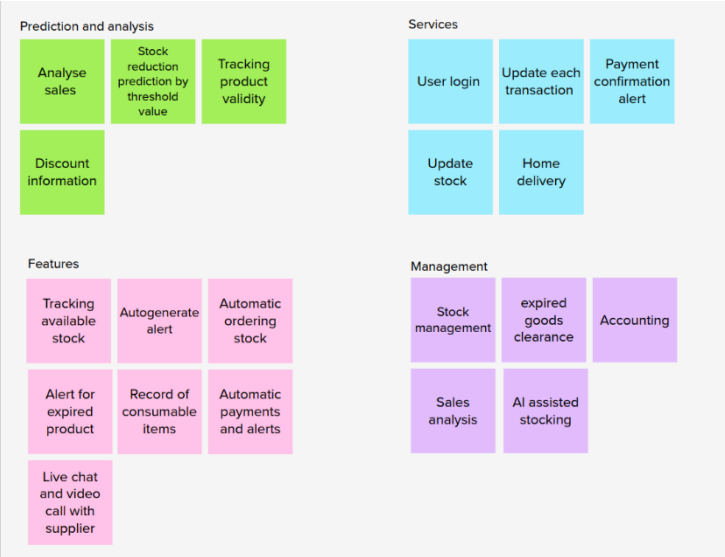
Brainstorm

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

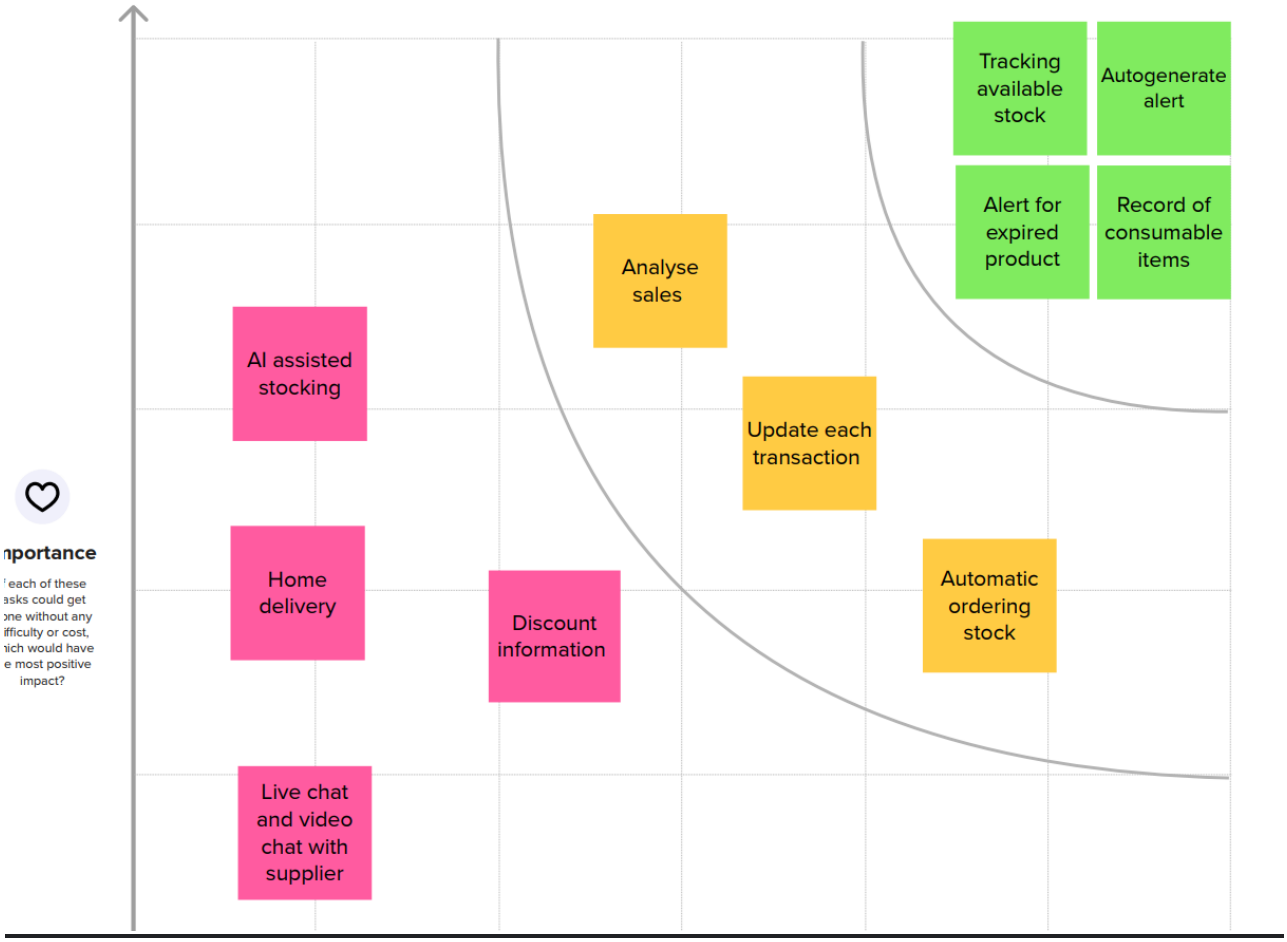
10 minutes



Prioritize the Ideas



Grouping



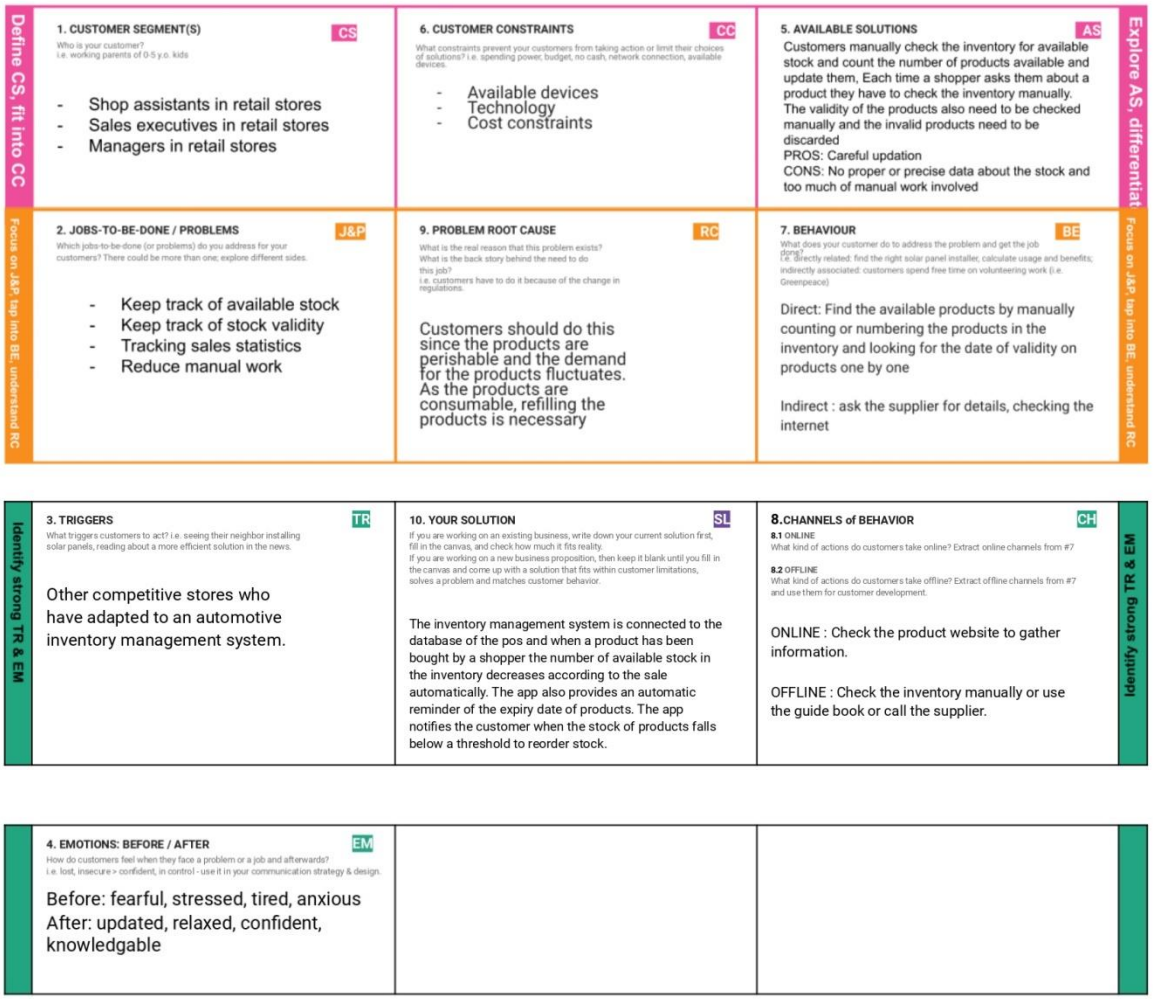
3.3 Proposed Solution

S.No	PARAMETER	DESCRIPTION
1.	Problem statement	Inventory Management system for retail
2.	Idea/ Solution Description	The inventory management system can be used by the workers in the retail shop to check for the stock of products and their validity. When a customer approaches them for information about a particular item they can just look into the app and check the quantity of stock that is available and the validity of the products. By using the user does not need to go to the inventory directly to check for information about each product. This also allows the user to update the stock in a timely manner.
3.	Novelty/Uniqueness	Manual entry and editing data and checking for validity of products
4.	Social Impact/ Customer Satisfaction	Customer need not worry about insufficient or invalid goods
5.	Business Model/ Revenue Model	Revenue is generated from the number ofusers who access the app, In app purchases.
6.	Scalability of the solution	As the app is based on cloud it is scalable.

3.4 Problem Solution fit

Project Title: INVENTORY MANAGEMENT SYSTEM FOR RETAIL
Team ID: PNT2022TMID27239

Project Design Phase-I - Solution Fit Template



4. REQUIREMENT ANALYSIS

4.1 Functional requirement

FR No.	Functional Requirement(Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration requirement	User can register through their email id.
FR-2	User Login	Login through User name and password.
FR-3	Records of the product	Product name Product description Product ID Product expiry date Stock count
FR-4	Updating of inventory details	Manual entry into form fields
FR-5	Monitoring stocks	Audit monitoring through incoming and outgoing stocks.

4.2 Non-functional Requirements:

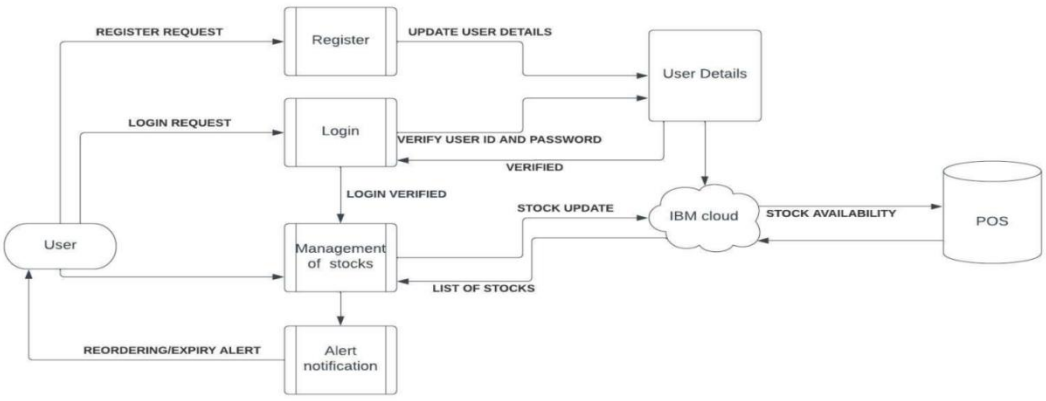
FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	Once retailers successfully login into the application they can update their inventory details, also they will be able to add new stock by submitting essential details related to the stock. They can also deleted and edit the entered details. The users can check the app for information about each product.
NFR2	**Security**	Application have been developed to help retailers track and manage their stocks. The system will ask the retailers to login into their account using their respective email id

NFR-3	**Reliability**	It will be reliable to use, that it provides easy automatic updation with accuracy.
NFR-4	**Performance**	User will be provided with easy inventory tracking. By the automatic alerts for the stock reduction and expiring products the man power will be considerably reduced then time and costs will also be reduced. This improves the inventory management performance.
NFR-5	**Availability**	Inventory management system is designed to monitor product availability, alerting and providing easy tracking of stock for the shop assistants who login into their respective account.
NFR-6	**Scalability**	The ability of the inventory management system can store and maintain large amount of data. It stores data in kilobytes and it will not utilize more storage for storing data.

5. PROJECT DESIGN

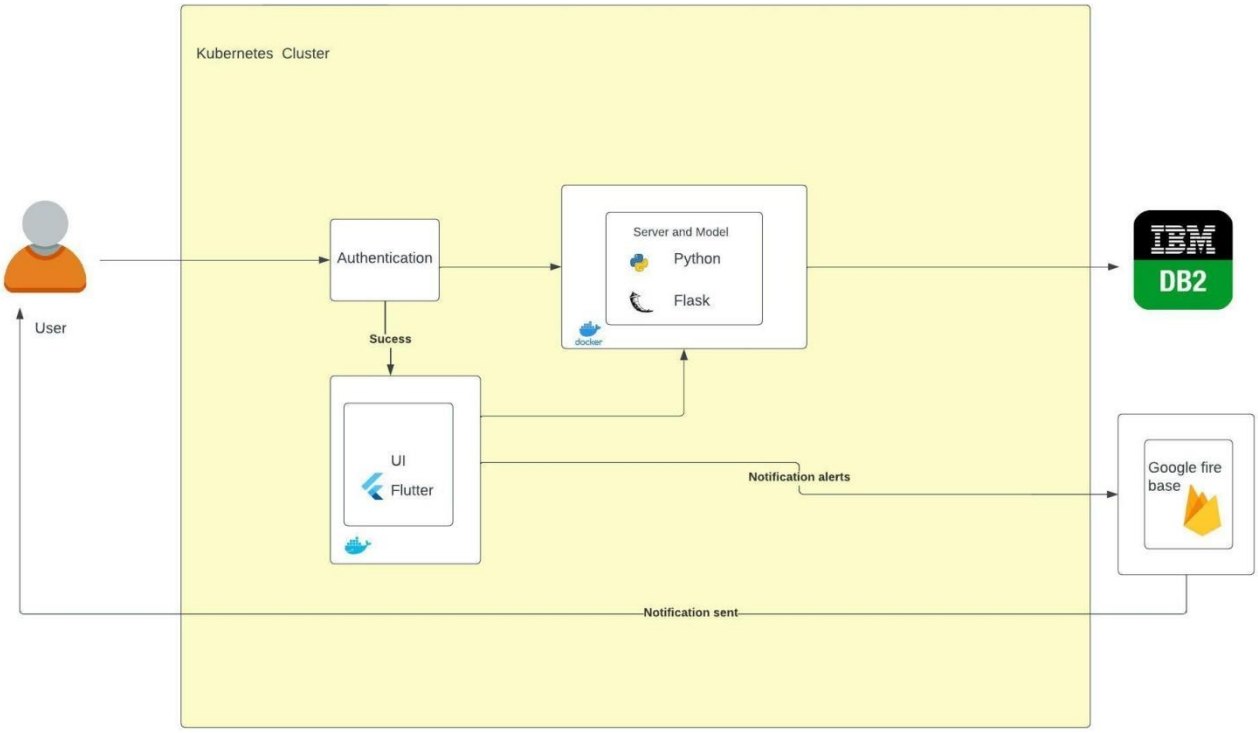
5.1 Data Flow Diagrams

Data Flow Diagrams:

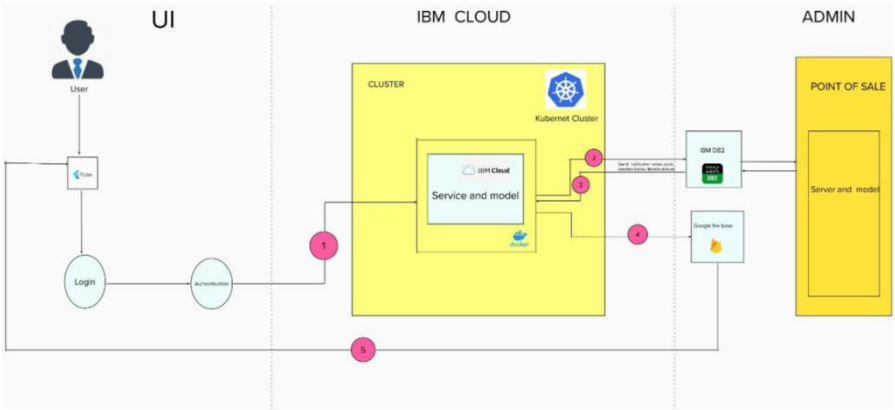


5.2 Solution & Technical Architecture

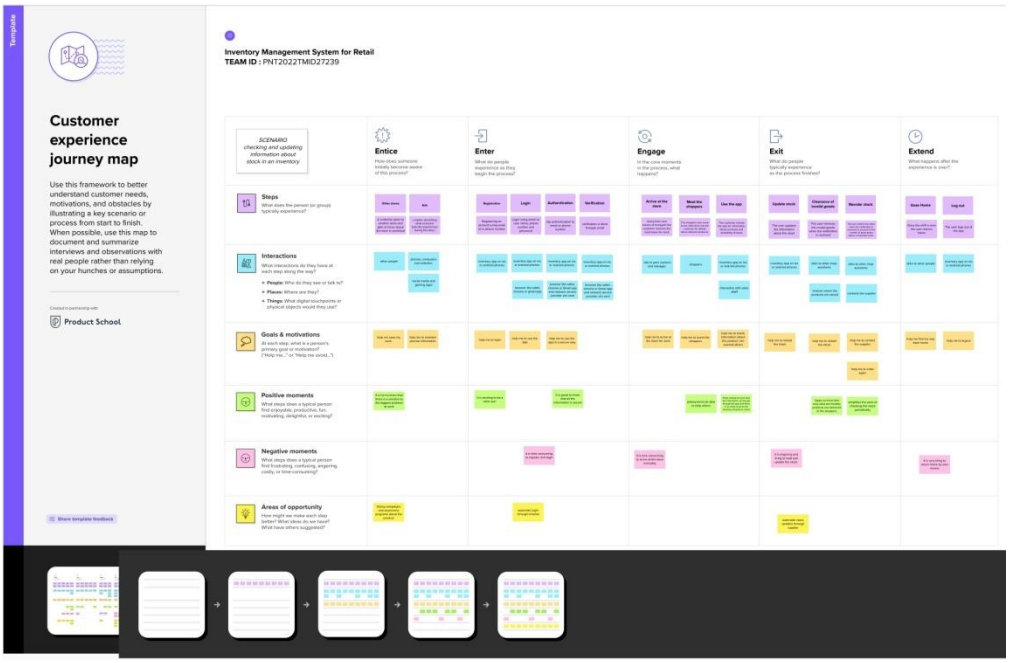
Solution Architecture



Technical Architecture



5.3 User Stories



6. PROJECT PLANNING AND SCHEDULING

6.1 Spring Planning and Estimation:

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my Password or by entering phone number and confirming by otp	2	High	Gracia Betty,Deepthi
Sprint-1		USN-2	As a user, I can register for the application through E-mail or phone number	1	Medium	Gracia Betty,Harivarthini
Sprint-1	Confirmation	USN-3	As a user, I will receive confirmation email or otp once I have registered for the application	1	Medium	Harivarthini,Aishwarya

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Login	USN-4	As a user, I can log into the application by entering email & password or using phone number and otp.	2	High	Aishwarya,Deepthi
Sprint-2	Dashboard	USN-5	As a user, I can view the products which are available	4	High	Deepthi,Harivarthini
Sprint-3	Stock Update	USN-7	Once the product reaches the threshold level as a user ,I will be getting the notification to reorder the stock.	5	High	Harivarthini,Gracia Betty
Sprint-4	Expiry update	USN-8	As a user, I will be notified about the expiry date of the products	5	High	Aishwarya,Deepthi

6.2 Sprint Delivery Schedule:

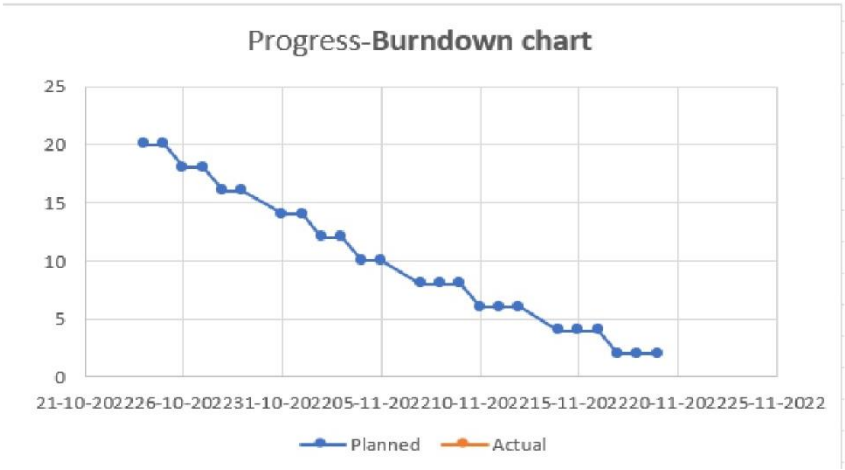
Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	6	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	4	6 Days	31 Oct 2022	05 Nov 2022		05 Nov 2022
Sprint-3	5	6 Days	07 Nov 2022	12 Nov 2022		12 Nov 2022
Sprint-4	5	6 Days	14 Nov 2022	19 Nov 2022		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)

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

AV=20/6 =3.33

6.3 Reports:



7. CODING AND SOLUTIONING

7.1 FEATURE ONE:

This feature allows the user to add, delete and edit the products. The user can store information about the name of the product, the description of the product, the quantity of stock available and the validity of a product.

Product.html

```
<!DOCTYPE html>
<html>
<head>
{% if title%}
    <title>Inventory {{title}}</title>
  {% else %}
    <title>Inventory</title>
  {% endif %}
  <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css
">
  <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"></script>
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.3/umd/popper.min.js
"></script>
  <script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/js/bootstrap.min.js"><
/script>
</head>
<body>
        <nav class="navbar navbar-expand-lg navbar-light bg-light">
        <a class="navbar-brand" href="#"><b>Inventory Management
System</b></a>
```

```
        <button class="navbar-toggler" type="button" data-toggle="collapse"
data-target="#navbarNavAltMarkup" aria-controls="navbarNavAltMarkup" aria-
expanded="false" aria-label="Toggle navigation">
          <span class="navbar-toggler-icon"></span>
        </button>
        <div class="collapse navbar-collapse" id="navbarNavAltMarkup">
          <div class="navbar-nav">
            <a class="nav-item nav-link" href="/home">Home</a>
            <a class="nav-item nav-link active" href="/Product">Product <span
class="sr-only">(current)</span></a>
            <a class="nav-item nav-link" href="/stock">Stock</a>
            <!-- <a class="nav-item nav-link" href="/Location">Location</a> --
>
            <!-- <a class="nav-item nav-link" href="/ProductMovement">Product
Movement</a>      -->
          </div>
        </div>
      </nav>
  </br>
<div class="container">
  <h2>Product Information</h2>

 <div class="float-md-right">
  <button type="button" class="btn btn-primary" data-toggle="modal" data-
target="#exampleModal">ADD PRODUCT</button></div>
  <div class="modal fade" id="exampleModal" tabindex="-1" role="dialog" aria-
labelledby="exampleModalLabel" aria-hidden="true">
  <div class="modal-dialog" role="document">
  <div class="modal-content">
      <div class="modal-header">
        <h5 class="modal-title" id="exampleModalLabel">ADD PRODUCT</h5>
        <button type="button" class="close" data-dismiss="modal" aria-
label="Close">
          <span aria-hidden="true">&times;</span>
        </button>
      </div>
      <div class="modal-body">
            <div class="col-md-4">
        <form class="form-group" action = "{{ url_for('addProduct') }}" method
= "POST">
            <div class="form-group">
              Name
               <input class="form-group" type="text" name="pn"
placeholder="Product Name" id="p_name" required>
            </div>
             <div class="form-group">
              Description
```

```
              <input class="form-group" type="text" name="pd"
placeholder="Product Description" id="P_description" required>
            </div>
             <div class="form-group">
              QTY
              <input class="form-group" type="text" name="pq"
placeholder="Product QTY and Validity" id="P_QTY" required>
            </div>
            <!-- <div class="form-group">
              Validity
              <input class="form-group" type="text" name="pv"
placeholder="Product Validity" id="P_QTY" required>
            </div> -->
      </div>
      <div class="modal-footer">
        <input class="btn btn-success" class="form-control"  type="Submit"
value="Submit" />
        <button type="button" class="btn btn-secondary" data-
dismiss="modal">Close</button>
      </div>
    </div>
  </form></div>
  </form>
  </div>
 </div>

 </br>
</br>

  <table class="table table-striped">
    <thead>
      <tr>
        <th class="text-info">Product id</th>
        <th class="text-info">Product name</th>
        <th class="text-info">Product Description</th>
        <th class="text-info">Qty and Validity</th>
        <!-- <th class="text-info">Validity</th> -->
        <th class="text-success">Edit</th>
        <th class="text-danger">Delete</th>
      </tr>
    </thead>
    <tbody>
       {% for row in rows %}
            <tr>
               <td>{{row["productID"]}}</td>
               <td>{{row["productName"]}}</td>
               <td> {{ row["productDescription"]}}</td>
               <td>{{row['QTY']}}</td>
```

```
                <!-- <td>{{row['Validity']}}</td> -->
                <td>
                     <a><button class="btn btn-primary"
OnClick='showModal({{row["productID"]}},"{{row["productName"]}}","{{row["produ
ctDescription"]}}","{{row["QTY"]}}");'>Edit</button></a></td>
                        <td> <a
href='deleteProduct/{{row["productID"]}}'><button class="btn btn-
danger">Delete</button></a>
                     </td>

            </tr>
         {% endfor %}
    </tbody>
  </table>
</div>
 <div class="modal fade" id="myModal">
      <div class="modal-dialog">
      <div class="modal-content">

         <!-- Modal Header -->
         <div class="modal-header">
            <h4 class="modal-title">Edit Product</h4>
            <button type="button" class="close" data-
dismiss="modal">&times;</button>
         </div>

         <!-- Modal body -->
         <div class="modal-body">
            <form class="form-group"  action = "{{ url_for('editProduct') }}"
method = "POST">
               <div class="form-group">
                  <input type="text" readonly class="form-control-plaintext
 form-control-lg" name="ProductID" id="ProductID" style="display: none;"
value="00">
               </div>
               <div class="form-group">
                  <label for="NEWProductName" class="sr-only"></label>
                  <input class="form-control form-control-lg" type="text"
name="NEWProductName" placeholder="New Product Name" id="NEWProductName"
required>
               </div>
                <div class="form-group">
                  <label for="NEWProductDescription" class="sr-only"></label>
                  <input class="form-control form-control-lg" type="text"
name="NEWProductDescription" placeholder="NEW Product Description"
id="NEWProductDescription" required>
               </div>
               <div class="form-group">
```

```
                <label for="NEWProductQty" class="sr-only"></label>
                <input class="form-control form-control-lg" type="text"
name="NEWProductQty" placeholder="NEW Product QTY and Validity"
id="NEWProductQty" required>
              <!-- </div>
              <div class="form-group">
               <label for="NEWProductValidity" class="sr-only"></label>
               <input class="form-control form-control-lg" type="text"
name="NEWProductValidity" placeholder="NEW Product Validity"
id="NEWProductQty" required>
            </div> -->
              <button type="submit" class="btn btn-success mb-2 font-weight-
bold">EDIT PRODUCT</button>
           </form>
        </div>
      </div>
    </div>
  </div>
 </div>
</body>
</html>
<script type="text/javascript">

    function showModal(id,oldname,oldDescription,oldqty){
            $('#ProductID').val(id);
            $('#NEWProductName').val(oldname);
            $('#NEWProductDescription').val(oldDescription);
            $('#NEWProductQty').val(oldqty);
            $('#myModal').modal('toggle');
        }
   </script>
</script>
```

Product.py

```
@app.route("/Product")
def Product():
   con = sql.connect("database.db")
   con.row_factory = sql.Row

   cur = con.cursor()
   cur.execute("select * from Product")

   rows = cur.fetchall();
   return  render_template('Product.html',rows = rows)

                                #ADD Product
```

```
@app.route('/addProduct',methods = ['POST'])
def addProduct():
   if request.method == 'POST':
      try:
         pn = request.form['pn']
         pd = request.form['pd']
         pq = request.form['pq']


         with sql.connect("database.db") as con:
            cur = con.cursor()
            cur.execute("INSERT INTO Product
(productName,productDescription,QTY) VALUES (?,?,?)",(pn,pd,pq) )

            con.commit()
            msg = "Record added"
      except:
         con.rollback()
         msg = "error in  operation"

      finally:

         return redirect(url_for('Product')+"?msg="+msg)
         con.close()
                                  #Edit Product
@app.route('/editProduct',methods = ['POST'])
def editProduct():
   if request.method == 'POST':
      try:
         productID = request.form['ProductID']
         productName = request.form['NEWProductName']
         productDescription=request.form['NEWProductDescription']
         ProductQty=request.form['NEWProductQty']
         cur.execute("UPDATE Product SET productName = ?,productDescription =
?, QTY = ? WHERE productID =
?",(productName,productDescription,ProductQty,productID) )

         con.commit()
         msg = "Product Edited "
      except:
         con.rollback()
         msg = "error in operation"

      finally:
         return redirect(url_for('Product')+"?msg="+msg)
         con.close()

                                 #Delete Product
```

```
@app.route('/deleteProduct/<productID>')
def deleteProduct(productID):
      try:
            cur.execute("DELETE FROM Product WHERE productID =
?",(productID,))

            con.commit()
            msg = "Product Deleted"
      except:
            con.rollback()
            msg = "error in operation"

      finally:
            return redirect(url_for('Product')+"?msg="+msg)
            con.close()
```

7.2 FEATURE ONE:

This feature displays the name of the product , the quantity of stock available and the validity of the products.

Stock.html

```
<!DOCTYPE html>
<html>
<head>
  <title>Stock balance</title>
  <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css
">
  <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"></script>
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.3/umd/popper.min.js
"></script>
  <script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/js/bootstrap.min.js"><
/script>
</head>
<body>
             <table class="table table-striped">
    <thead>
      <tr>
        <!-- <th class="text-info">Location Name</th> -->
        <th class="text-info">Product Name</th>
           <th class="text-info">Product QTY and Validity</th>
           <!-- <th class="text-info">Product Validity</th> -->
           <th class="btn btn-danger"><a href="/home"><button>Home
Page</button></a></th>
```

```
      </tr>
    </thead>
    <tbody>
     {% for row in rows %}
            <tr>
               <td>{{row["productName"]}}</td>
               <td>{{row["QTY"]}}</td>
               <!-- <td>{{row["Validity"]}}</td> -->

            </tr>
         {% endfor %}
    </tbody>
  </table>
</div>

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

8. TESTING

8.1 Test Cases:

				Date	5-Apr-22						
				Team ID	PNT2022TMID27239						
				Project Name	Inventory Management System for Retailers						
				Maximum Marks	4 marks						
Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Commnets	TC for Automation(Y/N)	BUG ID	Executed By
LoginPage_TC_001	Functional	Home Page	Verify user is able to see the Login/Signup popup when user clicks on My account button		1.Enter URL and click go 2.Click on My Account dropdown button 3.verify login/Signup popup displayed or not	Inventorymanagement localhost	Login/Signup popUp should display	Working as expected	Pass				
LoginPage_TC_002	UI	Home Page	Verify the UI elements in Login/Signup popup		1.Enter URL and click go 2.Verify login/Signup popup with below UI elements: a.email text box b.password text box c.Login button d.New customer? Create account link	Inventorymanagement localhost	Application should show below UI elements: a.email text box b.password text box c.Login button with blue colour d.New customer? Create account link e.Last password? Recovery password link	Working as expected	Fail	Steps are not clear to follow		BUG 1234	Deepchi OP
LoginPage_TC_003	Functional	Home page	Verify user is able to log into application with Valid credentials		1.Enter URL and click go 2.Enter Valid username/email in Email text box 3.Enter valid password in password text box 4.Click on login button	Username: gracie@gmail.com password: 123	User should navigate to user account homepage	Working as expected	pass				Gracia Retty J
LoginPage_TC_004	Functional	Login page	Verify user is able to log into application with InValid credentials		1.Enter URL and click go 2.Enter InValid username/email in Email text box 3.Enter valid password in password text box 4.Click on login button	Username: alabhvidit6@gmail.com password: abi123	Application should show 'Incorrect email or password ' validation message.	Working as expected	Fail	need to verify		bug 1235	Abhwanya R
LoginPage_TC_005	Functional	Login page	Verify user is able to log into application with Valid credentials		1.Enter URL and click go 2.Enter Valid username/email in Email text box 3.Enter invalid password in password text box 4.Click on login button	Username: alabhish@gmail.com password: abi 1234hai2	Application should show 'Incorrect email or password ' validation message.	Working as expected	pass				Harika chinil R
LoginPage_TC_006	Functional	Login page	Verify user is able to log into application with InValid credentials		1.Enter URL and click go 2.Enter InValid username/email in Email text box 3.Enter invalid password in password text box 4.Click on login button	Username: bala password: abi 1234hai2	Application should show 'Incorrect email or password ' validation message.	Working as expected	Pass				Harika chinil R

8.2 User Acceptance Testing:

1. Purpose of Document

The purpose of this document is to briefly explain the test coverage and open issues of the Inventory Management system for Retailers project at the time of the release to User Acceptance Testing (UAT).

2. Defect Analysis

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

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	8	7	1	2	18
Duplicate	2	0	2	0	4
External	2	3	1	2	8
Fixed	12	1	5	17	35
Not Reproduced	0	0	1	0	1
Skipped	0	0	1	1	2
Won't Fix	0	5	2	1	8
Totals	24	16	13	23	76

3. Test Case Analysis

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

Section	Total Cases	Not Tested	Fail	Pass
Print Engine	6	0	0	6
Client Application	55	0	0	55
Security	4	0	0	4
Outsource Shipping	0	0	0	0
Exception Reporting	8	0	0	8
Final Report Output	4	0	0	4
Version Control	2	0	0	2

9. RESULTS:

9.1 Performance Metrics:

Performance metrics are known as numbers and data representing organizations' abilities, actions, and overall quality. Various forms of performance metrics include profit, sales, customer happiness, return on investment, customer reviews, general quality, personal reviews, along with reputation in marketplaces. Take note that performance metrics can be various when they are viewed through many different industries. Performance metrics play an important role in any organization's success. It is necessary for any organization to choose its performance metrics, then paying attention to those areas since performance metrics support and guarantee an organization's success. Some key success elements are helpful in case they are tracked and acknowledged. Business measurements have to be controlled carefully to ensure they are providing the key answers, and the right questions are asked.

10. ADVANTAGES AND DISADVANTAGES

ADVANTAGES:

- Reduces manual work
- Timely update of products
- Helps in maintain the products in the store up to date

DISADVANTAGES:

- Manual entry of stock data
- Expensive to implement
- Needs learning to use

11. CONCLUSION:

It was a wonderful learning experience for me while working on this project. This project took me through the various phases of project development and gave me real insight into the world of software engineering.

12. FUTURE SCOPE:

The scope of an inventory system can cover many needs, including valuing the inventory, measuring the change in inventory and planning for future inventory levels. The value of the inventory at the end of each period provides a basis for financial reporting on the balance sheet. Measuring the change in inventory allows the company to determine the cost of inventory sold during the period. This allows the company to plan for future inventory needs.

13. APPENDIX

Login.html

```
<!-- Store this code in 'login.html' file inside the 'templates' folder -->

<html>
    <head>
        <meta charset="UTF-8">
        <title> Login </title>

        <style>
            .border{
    padding: 80px 50px;
    width: 350px;
    height: 500px;
    border: 1px solid #236B8E;
    border-radius: 0px;
    background-color: rgb(8, 38, 49);
}
            .header{
    padding: 5px 105px;
    width: 150px;
    height: 70px;
    background-color:rgb(8, 38, 49);


}
.png2 {
    width: 100px;
    height: 100px;
    border-radius: 50%;
    align-content: center;
    /*margin-left: 35%;*/
    background-color: white;
}
.msg {
    color: white;
    font-size: 10px;
    text-align: center;
    }
    .btn {
    padding: 10px 40px;
    background-color: #feffff;
    color: #060606;
    font-style: monospace;
    font-weight: bold;
    border-radius: 10px;
}

.textbox{
```

```
    padding: 10px 40px;
    background-color: #e6edf0;
    caret-color: #100e0e;
    /*border-radius: 10px;*/
}

::placeholder {
    color: #0b0b0b;
    opacity: 1;
    font-style: oblique;
    font-weight: bold;
}

.word{
    color: #FFFFFF;
    font-family:Arial, Helvetica, sans-serif ;
    /*font-weight: bold;*/
}

.bottom{
    color: #ffffff;
    font-style: oblique;
    font-weight: bold;
}

        </style>
    </head>
    <body></br></br></br></br></br>
        <div align="center">
        <div align="center" class="border">
            <div class="header">
                <h1 class="word">Login</h1>
            </div></br></br></br>
            <!-- <img src="img/logo.png" class="png2" /> -->
            <img src="../static/logo.png" class="png2"/>

            <h2 class="word">
                <form  method="POST" action="/afterlogin">
                <div class="msg">YOUR TIMELY UPDATED FRIEND</div>
                    <input id="username" name="_id" type="text"
placeholder="Enter Your Username" class="textbox"/></br></br>
                    <input id="password" name="psw" type="password"
placeholder="Enter Your Password" class="textbox"/></br></br></br>
                    <input type="submit" class="btn" value="Sign
In"></br></br>
                </form>
            </h2>
```

```
            <p class="bottom">Don't have an account? <a class="bottom"
href="{{url_for('register')}}"> Sign Up here</a></p>
        </div>
        </div>
    </body>
</html>
```

Register.html

```
<!-- Store this code in 'register.html' file inside the 'templates' folder -->

<html>
    <head>
        <meta charset="UTF-8">
        <title> Register </title>
        <style>
            .border{
    padding: 80px 50px;
    width: 350px;
    height: 500px;
    border: 1px solid #236B8E;
    border-radius: 0px;
    background-color: rgb(8, 38, 49);
}
            .header{
    padding: 5px 105px;
    width: 150px;
    height: 70px;
    background-color:rgb(8, 38, 49);



}
.png2 {
    width: 100px;
    height: 100px;
    border-radius: 50%;
    align-content: center;
    /*margin-left: 35%;*/
    background-color: white;
}
.msg {
    color: white;
    font-size: 10px;
    text-align: center;
    }
    .btn {
    padding: 10px 40px;
    background-color: #feffff;
```

```
    color: #060606;
    font-style: monospace;
    font-weight: bold;
    border-radius: 10px;
}

.textbox{
    padding: 10px 40px;
    background-color: #e6edf0;
    caret-color: #100e0e;
    /*border-radius: 10px;*/
}

::placeholder {
    color: #0b0b0b;
    opacity: 1;
    font-style: oblique;
    font-weight: bold;
}

.word{
    color: #FFFFFF;
    font-family:Arial, Helvetica, sans-serif ;
    /*font-weight: bold;*/
}

.bottom{
    color: #ffffff;
    font-style: oblique;
    font-weight: bold;
}

        </style>
    </head>
    <body></br></br></br></br></br>
        <div align="center">
        <div align="center" class="border">
            <div class="header">
                <h1 class="word">Register</h1>
            </div></br></br></br>
            <h2 class="word">
                <form  method="POST" action="/afterreg">

                    <input id="username" name="username" type="text"
placeholder="Enter Your Username" class="textbox"/></br></br>
                    <input id="password" name="password" type="password"
placeholder="Enter Your Password" class="textbox"/></br></br>
```

```
                    <input id="email" name="email" type="text"
placeholder="Enter Your Email ID" class="textbox"/></br></br>
                    <input type="submit" class="btn" value="Sign Up"></br>
                </form>
            </h2>
            <p class="bottom">Already have an account? <a class="bottom"
href="{{url_for('login')}}"> Sign In here</a></p>
        </div>
        </div>
    </body>
</html>
```

Home.html

```
<!DOCTYPE html>
<html>
<head>
  {% if title%}
    <title>Inventory {{title}}</title>
  {% else %}
    <title>Inventory</title>
  {% endif %}
  <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css
">
  <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"></script>
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.3/umd/popper.min.js
"></script>
  <script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/js/bootstrap.min.js"><
/script>
  <style>
    .mobile {
    position: relative;
    width: 320px;
    height: 500px;
    background: rgb(8, 38, 49);
    }

.border{
    padding: 80px 50px;
    width: 350px;
    height: 500px;
    border: 1px solid #236B8E;
    border-radius: 0px;
    background-color: rgb(8, 38, 49);
}
```

```
    .bottom{
    color: #18353d;
    font-style: oblique;
    font-family: 'Times New Roman', Times, serif;
    font-weight: bold;
}
.png2 {
    width: 100px;
    height: 100px;
    border-radius: 50%;
    align-content: center;
    /*margin-left: 35%;*/
    background-color: white;
}

.btn {
    padding: 10px 40px;
    background-color: #304c4c;
    color: #fffefe;
    font-style: monospace;
    font-weight: bold;
    border:#060606;
    border-radius: 10px;
}

  </style>
</head>
<body>


<nav class="navbar navbar-expand-lg navbar-light bg-light">
  <a class="navbar-brand" href="#"><b>Inventory Management System</b></a>
  <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarNavAltMarkup" aria-controls="navbarNavAltMarkup" aria-
expanded="false" aria-label="Toggle navigation">
    <span class="navbar-toggler-icon"></span>
  </button>
  <div class="collapse navbar-collapse" id="navbarNavAltMarkup">
    <div class="navbar-nav">
      <a class="nav-item nav-link active" class="home" href="/home">Home <span
class="sr-only">(current)</span></a>
      <a class="nav-item nav-link" class="Product" href="/Product">Product</a>
      <!-- <a class="nav-item nav-link" href="/Location">Location</a> -->
      <!-- <a class="nav-item nav-link" href="/ProductMovement">Product
Movement</a> -->
       <a class="nav-item nav-link" href="{{url_for('stock')}}">Stock</a>

    </div>
```

```
  </div>
</nav>
<div class="text-center">
 <img  class="png2" src="../static/logo.png"  />
 <h1 class="bottom">
  Hi {{session.username}}!!</br></br> Welcome to Inventory Manager</br> YOUR
TIMELY UPDATED FRIEND
</h1></br></br></br>
<a href="{{url_for('login')}}" class="btn">Logout</a>

 </div>

</body>

</html>
```

Product.html

```
<!DOCTYPE html>
<html>
<head>
{% if title%}
    <title>Inventory {{title}}</title>
  {% else %}
    <title>Inventory</title>
  {% endif %}
  <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css
">
  <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"></script>
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.3/umd/popper.min.js
"></script>
  <script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/js/bootstrap.min.js"><
/script>
</head>
<body>
        <nav class="navbar navbar-expand-lg navbar-light bg-light">
        <a class="navbar-brand" href="#"><b>Inventory Management
System</b></a>
        <button class="navbar-toggler" type="button" data-toggle="collapse"
data-target="#navbarNavAltMarkup" aria-controls="navbarNavAltMarkup" aria-
expanded="false" aria-label="Toggle navigation">
          <span class="navbar-toggler-icon"></span>
        </button>
        <div class="collapse navbar-collapse" id="navbarNavAltMarkup">
          <div class="navbar-nav">
            <a class="nav-item nav-link" href="/home">Home</a>
```

```
            <a class="nav-item nav-link active" href="/Product">Product <span
class="sr-only">(current)</span></a>
            <a class="nav-item nav-link" href="/stock">Stock</a>
            <!-- <a class="nav-item nav-link" href="/Location">Location</a> --
>
            <!-- <a class="nav-item nav-link" href="/ProductMovement">Product
Movement</a>      -->
          </div>
        </div>
      </nav>
  </br>
<div class="container">
  <h2>Product Information</h2>

 <div class="float-md-right">
  <button type="button" class="btn btn-primary" data-toggle="modal" data-
target="#exampleModal">ADD PRODUCT</button></div>
  <div class="modal fade" id="exampleModal" tabindex="-1" role="dialog" aria-
labelledby="exampleModalLabel" aria-hidden="true">
  <div class="modal-dialog" role="document">
  <div class="modal-content">
      <div class="modal-header">
        <h5 class="modal-title" id="exampleModalLabel">ADD PRODUCT</h5>
        <button type="button" class="close" data-dismiss="modal" aria-
label="Close">
          <span aria-hidden="true">&times;</span>
        </button>
      </div>
      <div class="modal-body">
            <div class="col-md-4">
        <form class="form-group" action = "{{ url_for('addProduct') }}" method
= "POST">
            <div class="form-group">
              Name
               <input class="form-group" type="text" name="pn"
placeholder="Product Name" id="p_name" required>
            </div>
             <div class="form-group">
              Description
               <input class="form-group" type="text" name="pd"
placeholder="Product Description" id="P_description" required>
            </div>
             <div class="form-group">
              QTY
               <input class="form-group" type="text" name="pq"
placeholder="Product QTY and Validity" id="P_QTY" required>
            </div>
            <!-- <div class="form-group">
```

```
              Validity
               <input class="form-group" type="text" name="pv"
placeholder="Product Validity" id="P_QTY" required>
            </div> -->
      </div>
      <div class="modal-footer">
        <input class="btn btn-success" class="form-control"  type="Submit"
value="Submit" />
        <button type="button" class="btn btn-secondary" data-
dismiss="modal">Close</button>
      </div>
    </div>
  </form></div>
  </form>
  </div>
 </div>

 </br>
</br>

  <table class="table table-striped">
    <thead>
      <tr>
        <th class="text-info">Product id</th>
        <th class="text-info">Product name</th>
        <th class="text-info">Product Description</th>
        <th class="text-info">Qty and Validity</th>
        <!-- <th class="text-info">Validity</th> -->
        <th class="text-success">Edit</th>
        <th class="text-danger">Delete</th>
      </tr>
    </thead>
    <tbody>
       {% for row in rows %}
            <tr>
               <td>{{row["productID"]}}</td>
               <td>{{row["productName"]}}</td>
               <td> {{ row["productDescription"]}}</td>
               <td>{{row['QTY']}}</td>
               <!-- <td>{{row['Validity']}}</td> -->
               <td>
                    <a><button class="btn btn-primary"
OnClick='showModal({{row["productID"]}},"{{row["productName"]}}","{{row["produ
ctDescription"]}}","{{row["QTY"]}}");'>Edit</button></a></td>
                      <td> <a
href='deleteProduct/{{row["productID"]}}'><button class="btn btn-
danger">Delete</button></a>
                    </td>
```

```
            </tr>
          {% endfor %}
      </tbody>
    </table>
</div>
 <div class="modal fade" id="myModal">
      <div class="modal-dialog">
      <div class="modal-content">

         <!-- Modal Header -->
         <div class="modal-header">
            <h4 class="modal-title">Edit Product</h4>
            <button type="button" class="close" data-
dismiss="modal">&times;</button>
         </div>

         <!-- Modal body -->
         <div class="modal-body">
            <form class="form-group"  action = "{{ url_for('editProduct') }}"
method = "POST">
               <div class="form-group">
                  <input type="text" readonly class="form-control-plaintext
 form-control-lg" name="ProductID" id="ProductID" style="display: none;"
value="00">
               </div>
               <div class="form-group">
                  <label for="NEWProductName" class="sr-only"></label>
                  <input class="form-control form-control-lg" type="text"
name="NEWProductName" placeholder="New Product Name" id="NEWProductName"
required>
               </div>
                <div class="form-group">
                  <label for="NEWProductDescription" class="sr-only"></label>
                  <input class="form-control form-control-lg" type="text"
name="NEWProductDescription" placeholder="NEW Product Description"
id="NEWProductDescription" required>
               </div>
               <div class="form-group">
                  <label for="NEWProductQty" class="sr-only"></label>
                  <input class="form-control form-control-lg" type="text"
name="NEWProductQty" placeholder="NEW Product QTY and Validity"
id="NEWProductQty" required>
               <!-- </div>
               <div class="form-group">
                <label for="NEWProductValidity" class="sr-only"></label>
```

```
                <input class="form-control form-control-lg" type="text"
name="NEWProductValidity" placeholder="NEW Product Validity"
id="NEWProductQty" required>
            </div> -->
              <button type="submit" class="btn btn-success mb-2 font-weight-
bold">EDIT PRODUCT</button>
            </form>
         </div>
       </div>
     </div>
   </div>
 </div>
</body>
</html>
<script type="text/javascript">

    function showModal(id,oldname,oldDescription,oldqty){
            $('#ProductID').val(id);
            $('#NEWProductName').val(oldname);
            $('#NEWProductDescription').val(oldDescription);
            $('#NEWProductQty').val(oldqty);
            $('#myModal').modal('toggle');
         }
   </script>
</script>
```

Stock.html

```
<!DOCTYPE html>
<html>
<head>
  <title>Stock balance</title>
  <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css
">
  <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"></script>
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.3/umd/popper.min.js
"></script>
  <script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/js/bootstrap.min.js"><
/script>
</head>
<body>
            <table class="table table-striped">
    <thead>
      <tr>
        <!-- <th class="text-info">Location Name</th> -->
        <th class="text-info">Product Name</th>
           <th class="text-info">Product QTY and Validity</th>
```

```
          <!-- <th class="text-info">Product Validity</th> -->
          <th class="btn btn-danger"><a href="/home"><button>Home
Page</button></a></th>

      </tr>
    </thead>
    <tbody>
     {% for row in rows %}
            <tr>
               <td>{{row["productName"]}}</td>
               <td>{{row["QTY"]}}</td>
               <!-- <td>{{row["Validity"]}}</td> -->

            </tr>
         {% endfor %}
    </tbody>
  </table>
</div>

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

Home.py

```
from flask import Flask,render_template,redirect, url_for
from flask import request
import sqlite3 as sql




from cloudant.client import Cloudant

client=Cloudant.iam('71ba514f-7497-4541-b670-2cca557572d7-
bluemix','jbwiZagjHPeEaY_Ff_WeZowtqr_cjHjP5fUsGtkaqfDH',connect=True)
my_database=client.create_database('my_database')
stock_database=client.create_database('stock_database')

con = sql.connect("database.db",check_same_thread=False)
con.row_factory = sql.Row
```

```
cur = con.cursor()

app = Flask(__name__)


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


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


@app.route("/home")
def home():
    return render_template('home.html')
    #stock balance
@app.route("/stock")
def stock():
   con = sql.connect("database.db")
   con.row_factory = sql.Row

   cur = con.cursor()
   cur.execute("select * from Product")

   rows = cur.fetchall();
   return  render_template('stock.html',rows = rows)


#register
@app.route('/afterreg',methods=['POST'])
def afterreg():
    x=[x for x in request.form.values()]
    print(x)
    data={
        '_id':x[1],
        'name':x[0],
        'psw':x[2]
    }
    print(data)
    query={'_id':{'$eq':data['_id']}}
    docs=my_database.get_query_result(query)
    print(docs)

    print(len(docs.all()))

    if(len(docs.all())==0):
```

```
        url=my_database.create_document(data)
        return render_template('login.html',pred="Registeration successfull,
Please login your details")
    else:
        return render_template('register.html',pred="You re already member,
Please login using r details")

#login
@app.route('/afterlogin',methods=['POST'])
def afterlogin():
   user = request.form['_id']
   passw = request.form['psw']
   print(user,passw)
   query = {'_id': {'$eq': user}}
   docs = my_database.get_query_result(query)
   print(docs)
   print(len(docs.all()))
   # if(len(docs,all())==0):
   #  return render_template('login.html',pred="The username is not found.")
   # else:
   if((user==docs[0][0]['_id'] and passw==docs[0][0]['psw'])):
      return render_template('home.html')
   else:
      print('Invalid User')



                                #Product Page
@app.route("/Product")
def Product():
   con = sql.connect("database.db")
   con.row_factory = sql.Row

   cur = con.cursor()
   cur.execute("select * from Product")

   rows = cur.fetchall();
   return  render_template('Product.html',rows = rows)

                               #ADD Product
@app.route('/addProduct',methods = ['POST'])
def addProduct():
   if request.method == 'POST':
      try:
         pn = request.form['pn']
         pd = request.form['pd']
         pq = request.form['pq']
```

```
        with sql.connect("database.db") as con:
            cur = con.cursor()
            cur.execute("INSERT INTO Product
(productName,productDescription,QTY) VALUES (?,?,?)",(pn,pd,pq) )

            con.commit()
            msg = "Record added"
      except:
         con.rollback()
         msg = "error in  operation"

      finally:

         return redirect(url_for('Product')+"?msg="+msg)
         con.close()
                                  #Edit Product
@app.route('/editProduct',methods = ['POST'])
def editProduct():
   if request.method == 'POST':
      try:
         productID = request.form['ProductID']
         productName = request.form['NEWProductName']
         productDescription=request.form['NEWProductDescription']
         ProductQty=request.form['NEWProductQty']
         cur.execute("UPDATE Product SET productName = ?,productDescription =
?, QTY = ? WHERE productID =
?",(productName,productDescription,ProductQty,productID) )

         con.commit()
         msg = "Product Edited "
      except:
         con.rollback()
         msg = "error in operation"

      finally:
         return redirect(url_for('Product')+"?msg="+msg)
         con.close()

                                #Delete Product
@app.route('/deleteProduct/<productID>')
def deleteProduct(productID):
      try:
            cur.execute("DELETE FROM Product WHERE productID =
?",(productID,))

            con.commit()
            msg = "Product Deleted"
```

```
      except:
            con.rollback()
            msg = "error in operation"

      finally:
            return redirect(url_for('Product')+"?msg="+msg)
            con.close()



if __name__ == '__main__':
    app.run(debug=True)
```

connect.py

```
import sqlite3 as sql


conn = sql.connect('database.db')


conn.execute('UPDATE TABLE product_movement (movementID INTEGER PRIMARY KEY,
productName TEXT, Timing timestamp, Validity DATE,QTY INTEGER)')
print ("Table productmovement Done")



conn.execute("INSERT INTO Balance
(locationName,productName,QTY)VALUES('Mumbai','STEEL','10')")
print ("Table balance c successfully")
```

GITHUB LINK:

https://github.com/IBM-EPBL/IBM-Project-9589-1659024443.git

DEMO VIDEO LINK:

https://drive.google.com/file/d/1k_U_jhsEN7JlOKZrjb4udZM9jB_iR6i_/view?usp=share_link