PROJECT REPORT

INVENTORY MANAGEMENT SYSTEM FOR RETAILERS

Submitted by

PNT2022TMID03825

Harini M - 412519104039

Dharshini T - 412519104028

Om Arthy Sri B - 412519104088

Srinithi S - 412519104141

Project Report

1. INTRODUCTION

1.1 Project Overview:

Retailers struggle because they lack a structured strategy for keeping track of their inventory data. Because they only retain the inventory data in the logbook and are not properly organized, the admin finds it challenging to record the data promptly and safely. Managing inventory with this paperwork and manual processes is tedious and not secure. And it doesn't easily scale across multiple warehouses with lots of stock. Moreover,customer demand is constantly shifting. Keeping too much could result in obsolete inventory where you're unable to sell, while keeping too little could leave you unable to fulfill customer orders. To properly manage the inventory and run their business,the retailer needs a technique to effectively analyze conditions, avoid out-of-stock problems,avoid overstocking, and keep customers coming back.

1.2 Purpose:

The main goals are to effectively run the business, manage the inventory, prevent out-of-stock issues, avoid overstocking, and keep consumers coming back. With the right platform, operations can be automated, inventory management techniques can be improved, and customer experiences can be enhanced. We can create a programme that keeps track of the stock on hand and alerts the retailer when there is running low supply. The retailer must also constantly verify the stock count and must deliver the goods without delay and on time.

2. LITERATURE SURVEY

2.1 Existing problem:

Retailers struggle because they lack a structured strategy for keeping track of their inventory data. Because they only retain the inventory data in the logbook and are not properly organised, the admin finds it challenging to record the data promptly and safely. To properly manage the inventory and run their business, the retailer needs a technique to effectively analyze conditions, avoid out-of-stock problems, avoid overstocking, and keep customers coming back.

2.2 References:

Inventory management for retail companies: A literature review and current trends:

This article's objective is to examine and summarize a large body of literature on inventory management that includes numerous definitions and key ideas for the retail industry. To identify the primary trends and indicators of inventory management in Small and Medium-sized Enterprises, a systematic literature study was conducted (SMEs). The five-year study period between 2015 and 2019 focuses solely on the retail industry. The main findings of this study include the top inventory control and management models, the Key Performance Indicators (KPIs) for managing

them correctly, and the advantages and difficulties of selecting or implementing an effective system. According to this research, there are 22 crucial inventory management KPIs that must be taken into account when shops assess their inventories. Ten main indicators were established from them, including inventory level, actual inventory and its connection to the business' information system, shortage or shortage frequency, frequency of product reordering or replenishment, service level, replacement frequency, product availability, inventory excess, number of items on the shelf, and level of income or profit. These indications enable the business to monitor stock levels, manage them effectively, and present a high level of customer service and product availability.

Case Study on an Android App for Inventory Management System with Sales Prediction for Local Shopkeepers in India:

In this article, the author proposed the idea of developing a mobile app that offers all the functionality of a point-of-sale system and provides insight into potential future sales is a very affordable and practical solution to this issue. It will make it possible for store owners to monitor their current goods purchases and billing. They will be able to modify their investments in goods and supply because of the predictive sales analysis, ensuring maximum profitability. A store's customer base will grow if it has pertinent items that meet customer needs.According to the study, the crucial part of this mobile app is sales prediction and analysis and this can be done by using data mining algorithms to the customer information that has been gathered as well as the temporal data that the merchants have provided.Regression Analysis is the algorithm that works best in this situation.

Implementation of inventory management system:

The objective of this study is to develop a system that effectively manages all the data regarding the dealer, supplier, manufactured goods, and raw materials. The many issues with storing unstructured data can be resolved using MongoDB, which saves data row-by-row and supports indexing. Other NoSQL databases like Cassandra and Neo4j do not have this feature. Hence, the data is stored in MongoDB on the backend, and the frontend is created using Java on NetBeans to provide a good Graphical User Interface (GUI), allowing anyone to access the inventory without any technical knowledge. Every day, the entire E-Commerce department stores a tonne of data, which can occasionally lead to huge things going missing, bad inventory management, and losing sight of their database. The customers also play a significant role in creating this situation by altering the products in the cart, leaving the cart with things at any time, which causes issues at checkout, and frequently canceling the orders. A system that not only stores this changing data, but also manages it effectively, is absolutely necessary. The present project might assist in fixing the issue and providing the agreed-upon quality of customer service.

A Proficient Process for Systematic Inventory Management:

The proposed approach controls an organization's inventory and aids in a better analysis of data relating to the storage and sale of items so that pertinent insights can be drawn from it. The system can be used to generate sales reports daily, weekly, or monthly in the form of different visualization charts by storing and updating the details of the inventory, stock maintenance, and system. It suggests creating a system for storing data by keeping track of the stock information for products classified under different brands and categories. The system recognises the necessity for an input device, a locked enclosure, a computer device, a data store, and a portal site or application in order to provide a comprehensive environment for effective warehouse and inventory management. The portal, application, computing device, and data repository must all be connected via the internet or a dispersed network. The system offers a technique that will make use of the idea of data analysis to supply details on the most popular, lucrative, and uninteresting stocks.

Design and implementation of Store Management System:

Delivering an improving client experience is both an opportunity and a challenge in this era of technology expansion and growing customer demands. Store management hires a retailer to give in-store retail customers a more engaging and modern shopping experience. With the help of this innovation, the store administrators will be able to exchange data consistently with all other participants in the inventory network. Additionally, it can prevent "loss of offer" problems to a greater extent by acting proactively. The store management identifies and monitors items in the shelves and automatically alerts retailers when it’s time to restock. Automation will take the place of the human data collection process, allowing businesses to obtain data instantly and without encoding or manual involvement. This paper proposes a new feature that will notify the store manager when a shelf is empty and out of stock. So that the store manager can swiftly fill orders that are out of stock by informing the store

Github link:

Literature Survey -Github

2.3 Problem Statement Definition:

To develop an application that tracks the stock available and gets notified to the retailer about the low stock beforehand and also needs to regularly check the stock count and should deliver the products on time without delay.

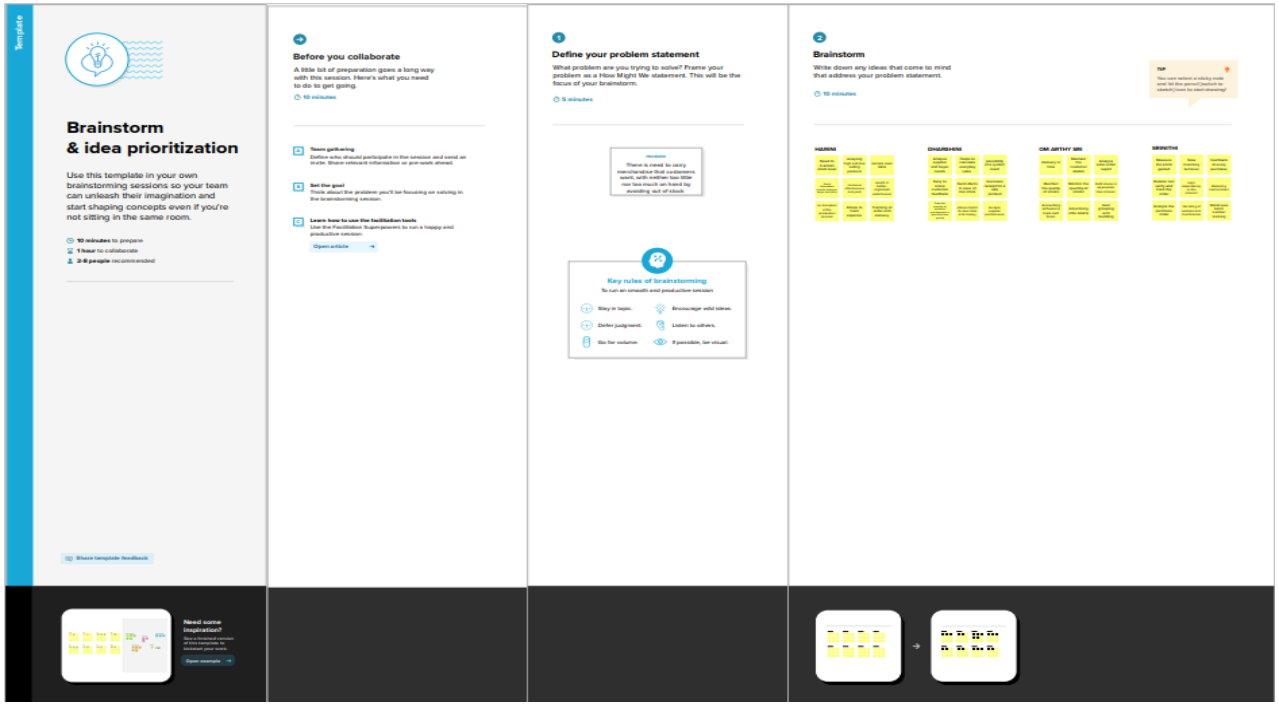
3. IDEATION & PROPOSED SOLUTION:

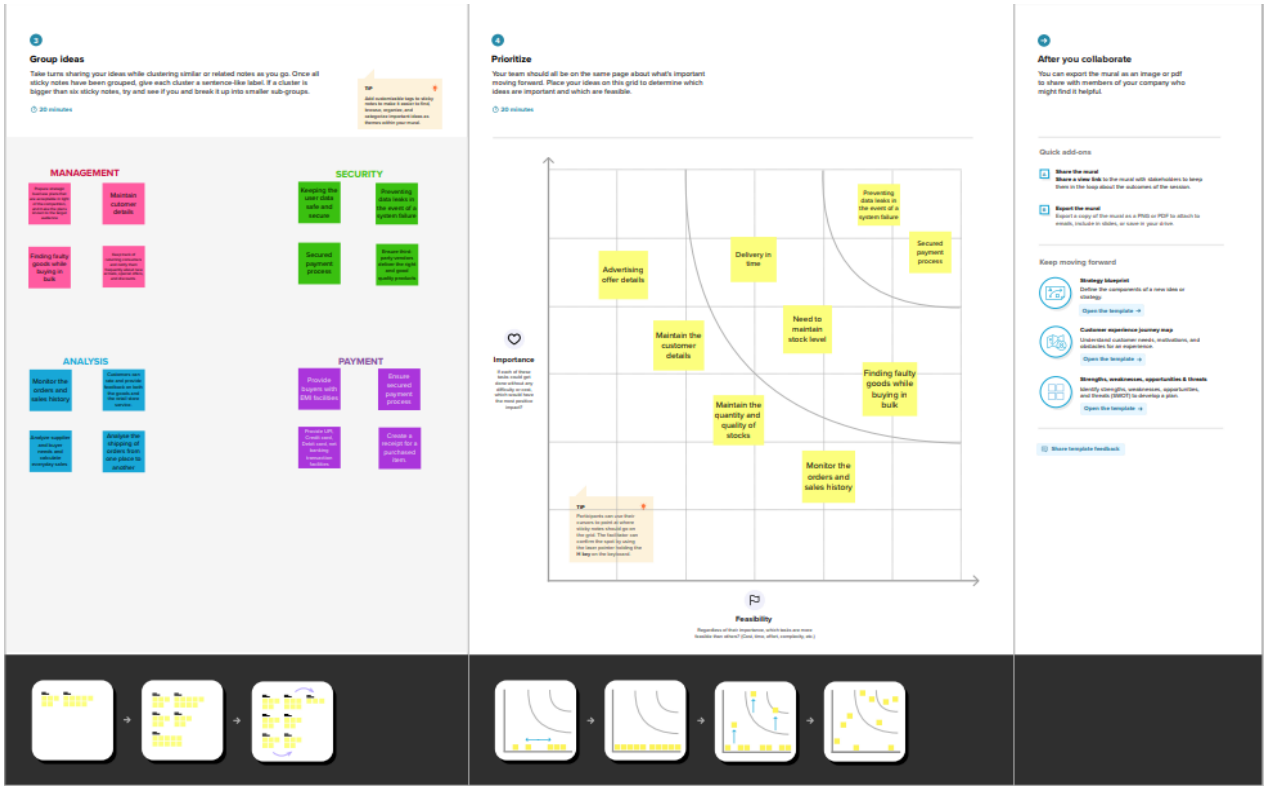
3.1 Empathy Map Canvas

Github link:

[Empathy Map - Github]

3.2 Ideation & Brainstorming:





Github link:

[Ideation & Brainstorming - Github]

3.3 Proposed Solution:

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Retailers struggle because they lack a structured strategy for keeping track of their inventory data. Because they only retain the inventory data in the logbook and are not properly organised, the admin finds it challenging to record the data promptly and safely. To properly manage the inventory and run their business, the retailer needs a technique to effectively analyse conditions, avoid out-of-stock problems, avoid overstocking, and keep customers coming back.
2.	Idea / Solution description	Processes can be automated, inventory management procedures can be improved, and customer experiences can be improved with the correct platform. Developing an application that tracks

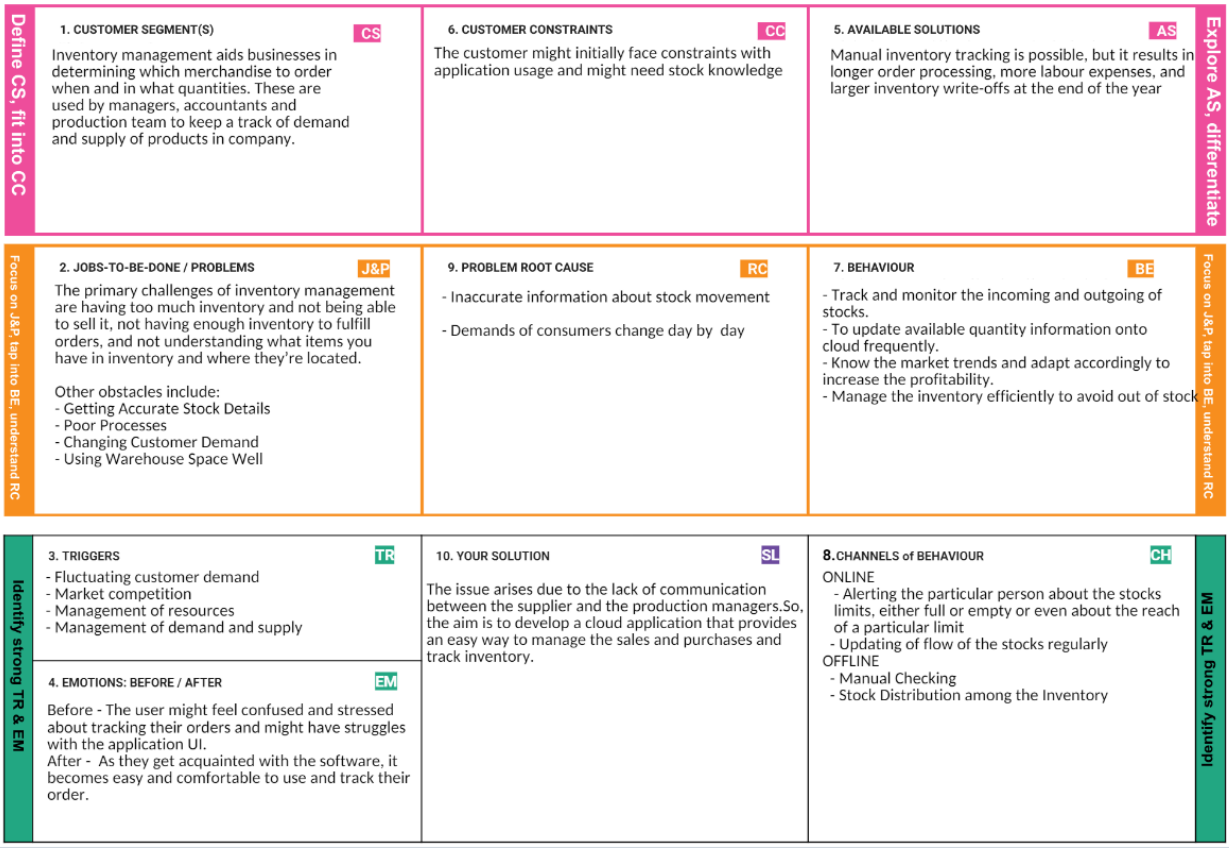
		the stock available and gets notified to the retailer about the low stock beforehand and also needs to regularly check the stock count and should deliver the products on time without delay.
3.	Novelty / Uniqueness	● Increased sales ● Track inventory across many sites ● Schedule production and distribution ● Online and offline order management ● Increased scalability and flexibility with a variety of add-ons ● Simple and affordable pricing. ● Increased customer satisfaction with end-to-end tracking ● Handle reorder points automatically ● Forecast demand
4.	Social Impact / Customer Satisfaction	Customers will be satisfied because less time will be wasted looking for a product that isn't available. Retailers' workloads will be kept to a minimum if the system is automated every day and every time a purchase is made. Receiving prompt and appropriate responses from the shops will increase customer satisfaction.
5.	Business Model (Revenue Model)	Demand and supply must be balanced, financial and operational planning must be combined, and high-level strategic plans must be connected to medium- and long-term company plans. Inventory management aids businesses in determining which stock to order when and in what quantities.
6.	Scalability of the Solution	A company will have the framework and real-time metrics it needs to stay competitive and meet its growth objectives with an automated inventory management system. There will be an improvement in corporate profitability and effectiveness. Implementing a system that anybody and anywhere may use to purchase goods can be advantageous.

		The stock can be updated daily and after every purchase to stop inventory shrinkage.

Github link:

Proposed Solution-Github

3.4 Problem Solution fit:



Github link:

Problem Solution fit-Github

4. REQUIREMENT ANALYSIS:

4.1 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
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	User Login	Login with username Login with Email Login with password
FR-4	Dashboard	View product availability, name of the product, stock keep unit, brand, retail price, product category, lot number, expiration date, vendor information, wholesale cost,etc.,
FR-5	Identification of the stock location	Provide number label for - shelf, rack and boxes
FR-6	Periodical stock checking	Automate the tracking of stock count
FR-7	Purchase management and Forecasting	Order review and placement, avoid risk stock, review product, priorities purchases based on an item's profitability, popularity, and lead time.
FR-8	Returns Management System	Examine for flaws or damage and return to the vendor if necessary. Add it to inventory counts if it is sellable.
FR-9	Markdown and promotion	Display product savings, Keep enough inventory on hand to satisfy demand.
FR-10	Calculating the death stock	Return the stock to the vendor for credits

Github link:

Functional Requirements-Github

4.2 Non-Functional requirements:

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

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	The success of deploying an inventory monitoring system in your company depends on usability. The system must be user-friendly and straightforward in the manner it presents all pertinent data and linkages, and its menus must include buttons that are simple to grasp. The software is not worthwhile if training takes hours for your workforce. Remember to pick a solution that makes inventory management simple.This variant is compatible with desktop browsers.
NFR-2	**Security**	It is the method for making sure that kept goods are safe and under the best management control. It is crucial for effective warehouse management because a warehouse's productivity and safety determines how well a firm performs. In this case, only authorized people with their username and password can access the system.
NFR-3	**Reliability**	The user must constantly receive accurate inventory status from the system. By routinely comparing the real levels to the levels shown in the system, any errors are fixed.
NFR-4	**Performance**	Every time a user requests a process, the system must successfully complete the tasks like updating the stocks in the database, adding new stocks, and deleting it. Every time the system is turned on, all of its features must be accessible to the customer. The system's calculations must adhere to the standards established by the customer and shouldn't change until the customer specifically requests a change.
NFR-5	**Availability**	Only the organization's admin will have access to the software, and it is he who will record information of the product as well as the customer details. A single item's inventory availability determines whether it is accessible for

		customer orders. The admin can also add, remove or update the stock and stock details respectively.
NFR-6	**Scalability**	The business will become considerably more scalable with the use of an automated inventory management system for inventory tracking, enabling it to capitalize on rising sales and maintain steady growth.

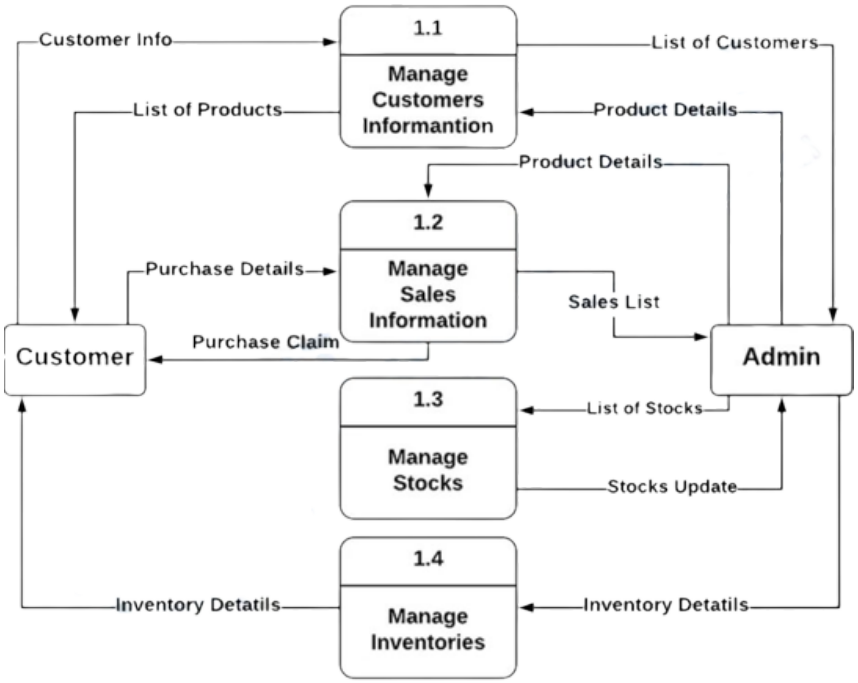
Github link:

Non-Functional Requirements-Github

5. PROJECT DESIGN:

5.1 Data Flow Diagrams:

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.

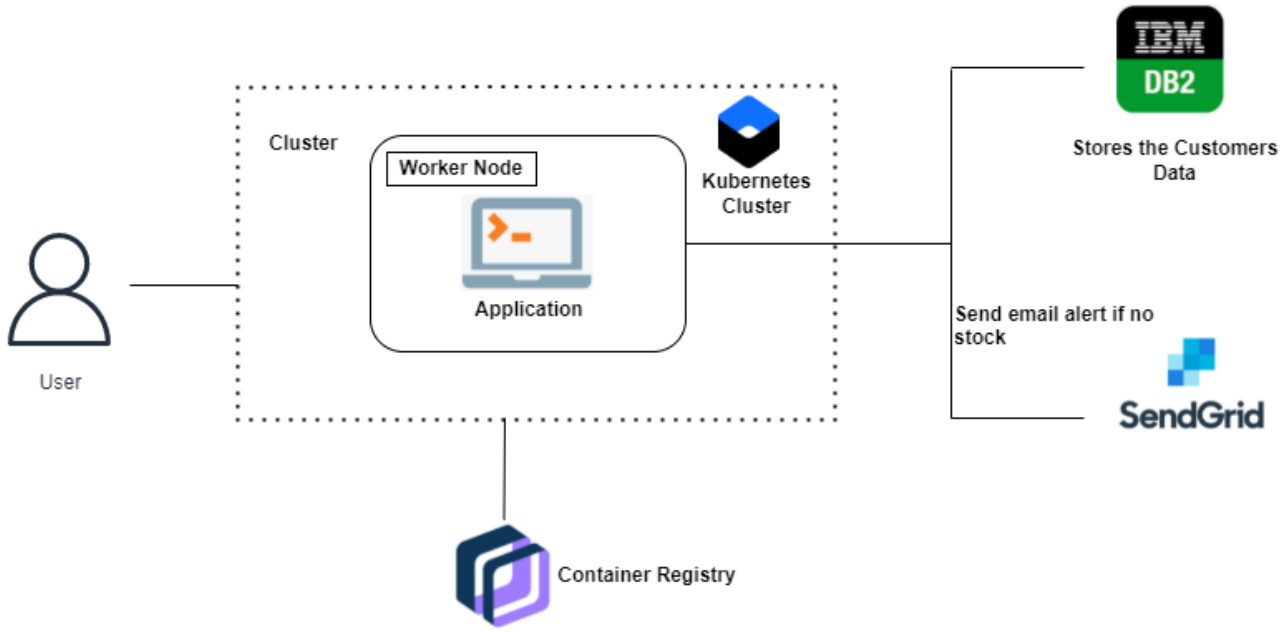


Github link:

Data flow Diagrams-Github

5.2 Solution & Technical Architecture:

Technical Architecture:



Solution Architecture:

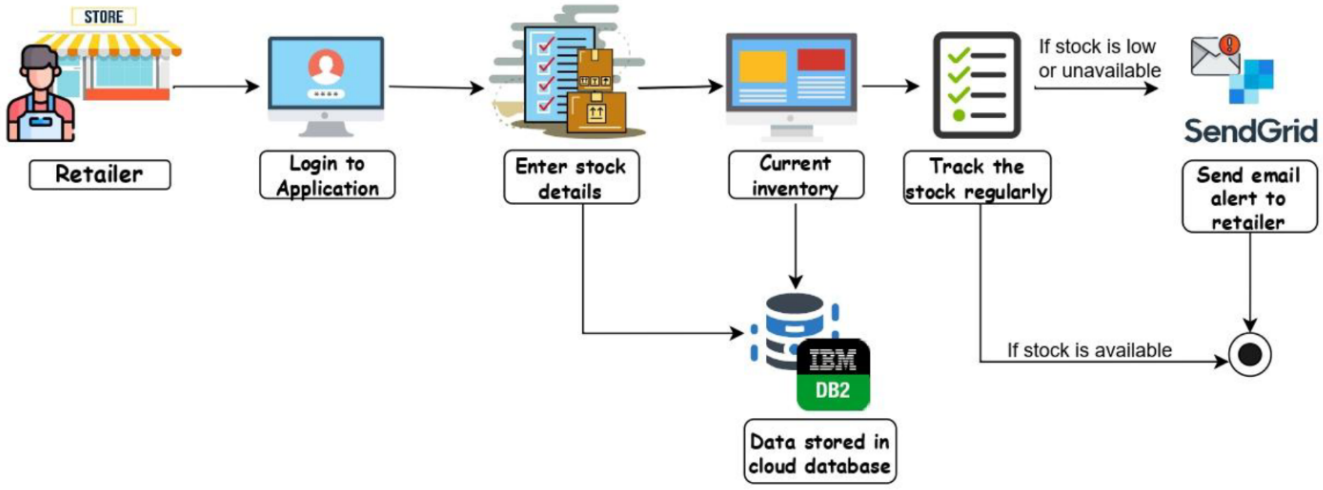


Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	Web UI, Mobile App, Chatbot etc.	HTML, CSS, JavaScript, React Js
2.	Application Logic - 1	Alert retailers in case of low stock	Python - Flask, Sen
3.	Application Logic - 2	Chatbot for CRM	IBM Watson Assistant
4.	Cloud Database	Database Service on Cloud	IBM DB2
5.	File Storage	File storage requirements	IBM Object Storage
6.	Infrastructure (Server / Cloud)	Cloud Server Configuration	Local, Cloud Foundry, Kubernetes.

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	Used open-source frameworks for styling webpage and building web application	Python flask, Bootstrap
2.	Security Implementations	To keep user data secure, verify ownership of the website, prevent attackers from creating a fake version of the site, and convey trust to users.	SSL, IBM Cloud security
3.	Scalable Architecture	3 – tier architecture MVC	Web server - HTML, CSS, JS Application server - Python Flask, Docker, Container Registry Database server - IBM DB2
4.	Availability	To evenly distribute network traffic to prevent failure caused by overloading a particular resource	IBM Load Balancer
5.	Performance	Use of local machine Cache memory	IBM Cloud, CDN

Github link:

Technology Architecture - Github

Solution Architecture - Github

5.3 User Stories:

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
		USN-3	As a user, I can register for the application through Facebook	I can register & access the dashboard with Facebook Login	Low	Sprint-2
		USN-4	As a user, I can register for the application through Gmail	I can register and access the dashboard with Gmail login	Medium	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering email & password	I can login to the application with verified Email and password	High	Sprint-1
	Dashboard	USN-6	As a user, I can view the available product list and the inventory data.	I can view the stock availability status	High	Sprint-2
		USN-7	As a user, I can view the order and track the shipping status	I can view the order and shipping status after making a purchase	Medium	Sprint-3
Customer (Web user)	Registration	USN-8	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
		USN-9	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
		USN-10	As a user, I can register for the application through Facebook	I can register & access the	Low	Sprint-2

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
				dashboard with Facebook Login		
		USN-11	As a user, I can register for the application through Gmail	I can register and access the dashboard with Gmail login	Medium	Sprint-1
	Login	USN-12	As a user, I can log into the application by entering email & password	I can login to the application with verified Email and password	High	Sprint-1
	Dashboard	USN-13	As a user, I can view the available product list and the inventory data.	I can view the stock availability status	High	Sprint-2
		USN-14	As a user, I can view the order and track the shipping status	I can view the order and shipping status after making a purchase	Medium	Sprint-3
Customer Care Executive	Chat bot	USN-15	As a customer care executive,I can view the complaints on chat bot and assist the users	I can view the complaints on the chat bot and assist the user with queries.	Medium	Sprint-4
Administrator	Alerts	USN-16	As an administrator, I would handle user registrations and maintenance of accounts	I can take care of registrations and maintenance of accounts	High	Sprint-3
		USN-17	As an administrator, I can refill the stock on receiving the alerts	I can refill stock if there's an alert	High	Sprint-3

Github link:

User Stories-Github

6. PROJECT PLANNING & SCHEDULING:

6.1 Sprint Planning & 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.	2	High	Harini M Dharshini T Om Arthy Sri B Srinithi S
Sprint-1		USN-2	As a user, I will receive confirmation email once I have registered for the application	1	High	Harini M Dharshini T Om Arthy Sri B Srinithi S
Sprint-2		USN-3	As a user, I can register for the application through Facebook	2	Low	Harini M Dharshini T Om Arthy Sri B Srinithi S
Sprint-1		USN-4	As a user, I can register for the application through Gmail	2	Medium	Harini M Dharshini T Om Arthy Sri B Srinithi S

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Login	USN-5	As a user, I can log into the application by entering email & password	1	High	Harini M Dharshini T Om Arthy Sri B Srinithi S
Sprint-2	Dashboard	USN-6	As a user, I can view the available product list and the inventory data		High	Harini M Dharshini T Om Arthy Sri B Srinithi S
Sprint-3		USN-7	As a user, I can view the order and track the shipping status		Medium	Harini M Dharshini T Om Arthy Sri B Srinithi S
Sprint-1	Registration	USN-8	As a user, I can register for the application by entering my email, password, and confirming my password		High	Harini M Dharshini T Om Arthy Sri B Srinithi S

Sprint-1		USN-9	As a user, I will receive confirmation email once I have registered for the application		High	Harini M Dharshini T Om Arthy Sri B Srinithi S
Sprint-2		USN-10	As a user, I can register for the application through Facebook		Low	Harini M Dharshini T Om Arthy Sri B Srinithi S
Sprint-1		USN-11	As a user, I can register for the application through Gmail		Medium	Harini M Dharshini T Om Arthy Sri B Srinithi S

Sprint	**Functional Requirement (Epic)**	**User Story Number**	**User Story / Task**	**Story Points**	**Priority**	**Team Members**
Sprint-1	Login	USN-12	As a user, I can log into the application by entering email & password		High	Harini M Dharshini T Om Arthy Sri B Srinithi S
Sprint-2	Dashboard	USN-13	As a user, I can view the available product list and the inventory data.		High	Harini M Dharshini T Om Arthy Sri B Srinithi S
Sprint-3		USN-14	As a user, I can view the order and track the shipping status		Medium	Harini M Dharshini T Om Arthy Sri B Srinithi S
Sprint-4	Chat bot	USN-15	As a customer care executive,I can view the complaints on chat bot and assist the users		Medium	Harini M Dharshini T Om Arthy Sri B Srinithi S
Sprint-3	Alerts	USN-16	As an administrator, I would handle user registrations and maintenance of accounts		High	Harini M Dharshini T Om Arthy Sri B Srinithi S
Sprint-3		USN-17	As an administrator, I can refill the stock on receiving the alerts		High	Harini M Dharshini T Om Arthy Sri B Srinithi S

Github link:

Sprint Planning & Estimation - Github

6.2 Sprint Delivery Schedule:

Project tracker:

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

Velocity:

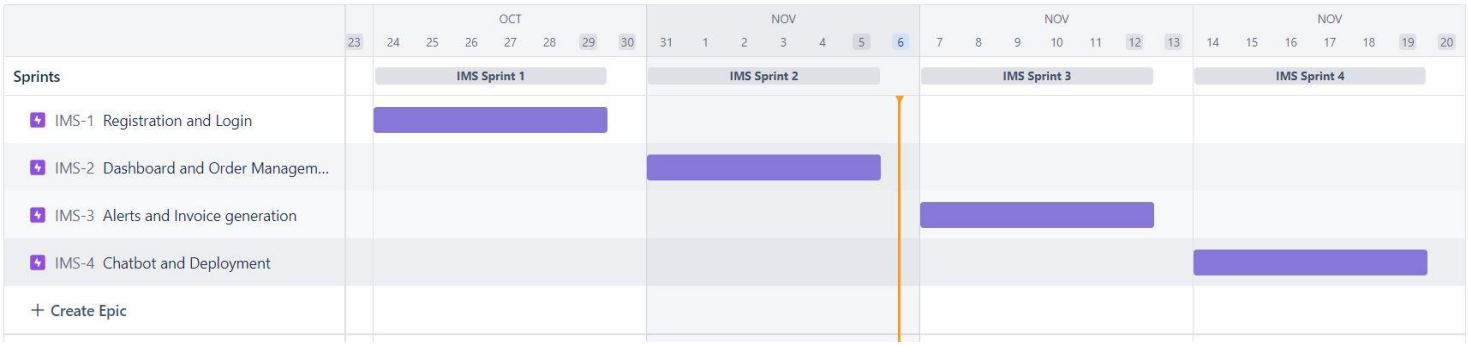
$$AV = \frac{\textit{Sprint Duration}}{\textit{Velocity}} = \frac{20}{6} = 3.33$$

Github link:

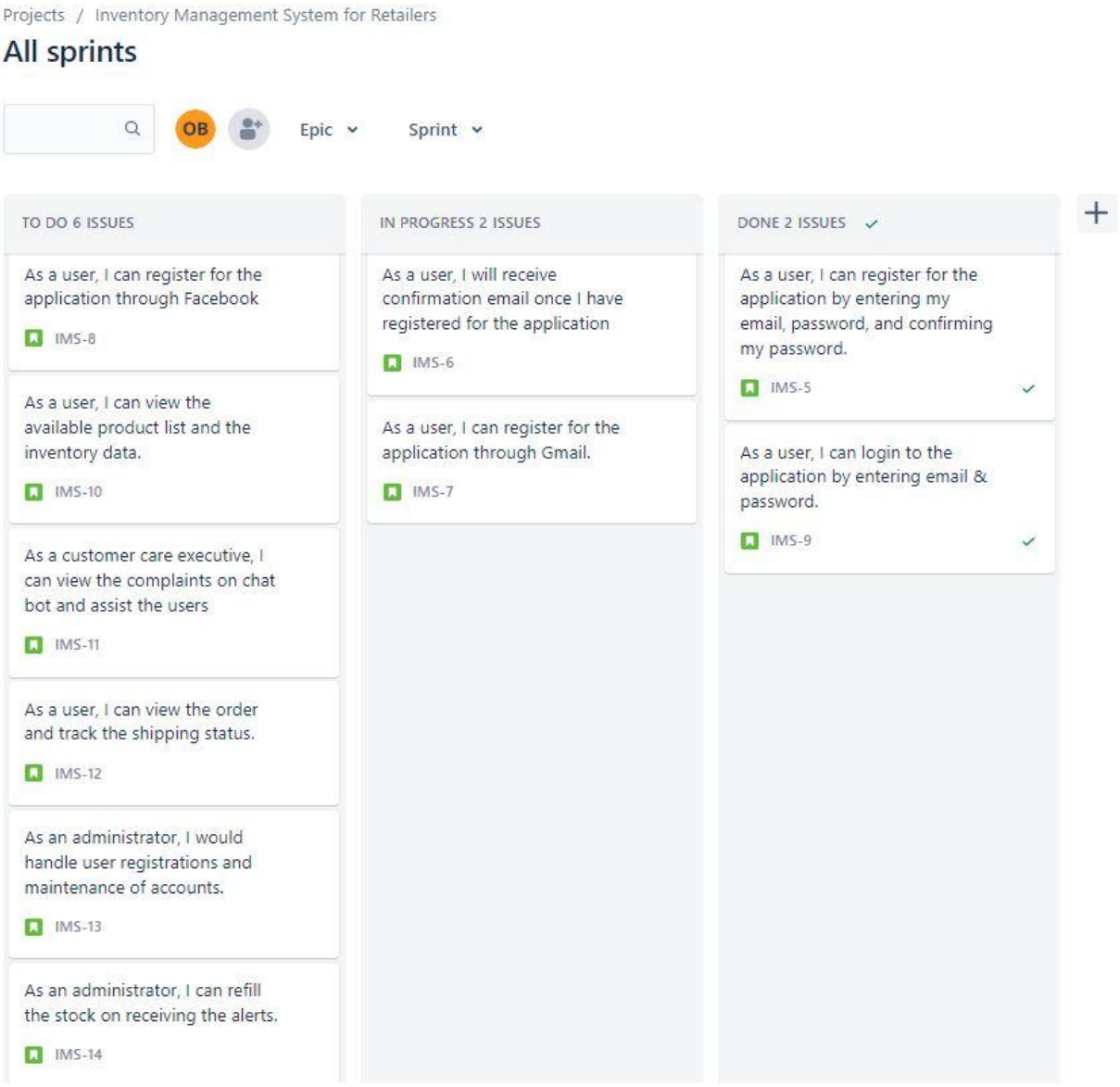
Sprint Delivery Schedule - Github

6.3 Reports from JIRA:

Roadmap:



Jira Board:



Github link:

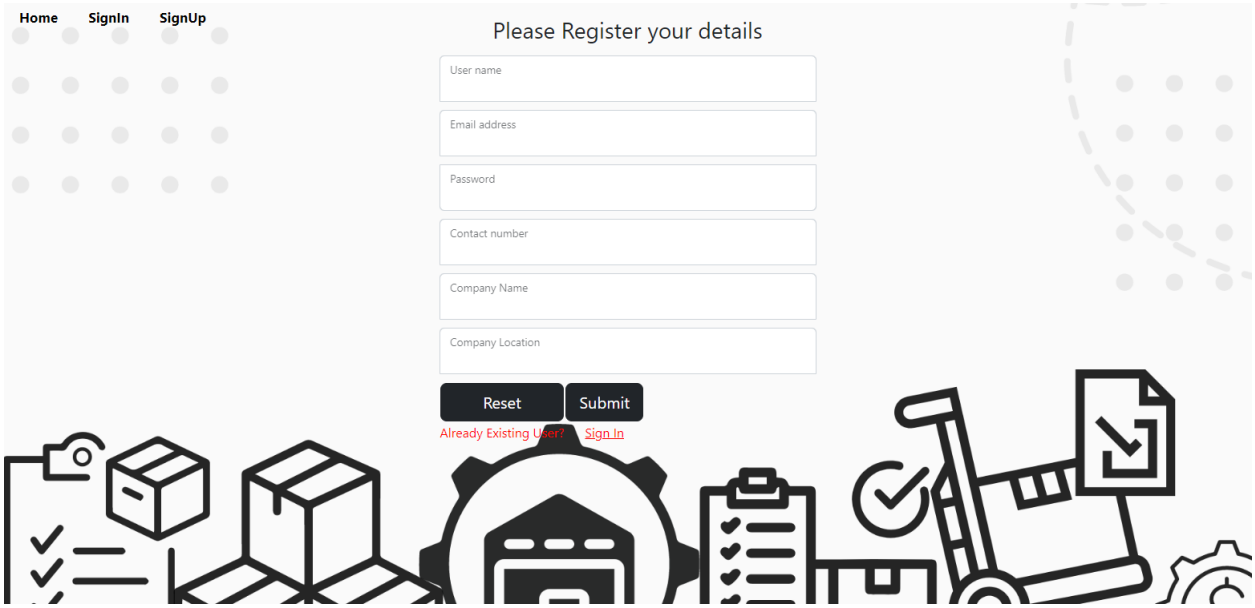
Jira Files -Github

7. CODING & SOLUTIONING:

7.1 Feature 1:

Home Page:

Sign Up Page:



Sign In Page:

CODE:

Home.html

home.html

templates > home.html > ...

```
<!doctype html>
<html lang="en">

<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <title>IMSFR</title>
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-Zenh87qX5JnK2Jl0vWa8Cl
  <link rel="canonical" href="https://getbootstrap.com/docs/5.2/examples/navbar-static/">
  <link href="/static/css/index.css" rel="stylesheet">
</head>

<body>
  <nav class="navbar navbar-expand-md navbar-light bg-transparent mb-4">
    <div class="container-fluid">
      <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarCollapse"
        aria-controls="navbarCollapse" aria-expanded="false" aria-label="Toggle navigation">
        <span class="navbar-toggler-icon"></span>
      </button>
      <div class="collapse navbar-collapse" id="navbarCollapse">
        <ul class="navbar-nav me-auto mb-2 mb-md-0">
          <li class="nav-item active">
            <a style="color: black; font-size: 17px; font-weight: 700; font-family:'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;" class="n
          </li>
          <li class="nav-item">
            <a style="color: black; font-size: 17px; font-weight: 700; font-family:'Segoe UI', Tahoma, Geneva, Verdana, sans-serif; padding-l
          </li>
          <li class="nav-item">
            <a style="color: black; font-size: 17px; font-weight: 700; font-family:'Segoe UI', Tahoma, Geneva, Verdana, sans-serif; padding-l
          </li>
        </ul>
      </div>
```

```
          <span class="navbar-toggler-icon"></span>
        </button>
        <div class="collapse navbar-collapse" id="navbarCollapse">
          <ul class="navbar-nav me-auto mb-2 mb-md-0">
            <li class="nav-item active">
              <a style="color: black; font-size: 17px; font-weight: 700; font-family:'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;" class="n
            </li>
            <li class="nav-item">
              <a style="color: black; font-size: 17px; font-weight: 700; font-family:'Segoe UI', Tahoma, Geneva, Verdana, sans-serif; padding-l
            </li>
            <li class="nav-item">
              <a style="color: black; font-size: 17px; font-weight: 700; font-family:'Segoe UI', Tahoma, Geneva, Verdana, sans-serif; padding-l
            </li>
          </ul>
        </div>
      </div>
    </nav>

    <main role="main" class="container">
      <div class="jumbotron">
        <h1>Servo-IN</h1>
        <p class="lead"><b>From us to you!!</b></p>
      </div>
    </main>

    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/js/bootstrap.bundle.min.js"
      integrity="sha384-OERcA2EqjJCMA+/3y+gxIOqMEjwtxJY7qPCqsdltbNJuaOe923+mo//f6V8Qbsw3"
      crossorigin="anonymous"></script>
  </body>

  </html>
```

Signup.html

signup.html

templates > signup.html > ...

```
<!doctype html>
<html lang="en">

<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <title>IMSFR</title>
  <!-- <link rel="canonical" href="https://getbootstrap.com/docs/5.2/examples/sign-in/"> -->
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css" rel="stylesheet"
    integrity="sha384-Zenh87qX5JnK2Jl0vWa8Ck2rdkQ2Bzep5IDxbcnCeuOxjzrPF/et3URy9Bv1WTRi" crossorigin="anonymous">
  <style>
    .bd-placeholder-img {
      font-size: 1.125rem;
      text-anchor: middle;
      -webkit-user-select: none;
      -moz-user-select: none;
      user-select: none;
    }

    @media (min-width: 768px) {
      .bd-placeholder-img-lg {
        font-size: 3.5rem;
      }
    }

    .b-example-divider {
      height: 3rem;
      background-color: rgba(0, 0, 0, .1);
      border: solid rgba(0, 0, 0, .15);
      border-width: 1px 0;
      box-shadow: inset 0 .5em 1.5em rgba(0, 0, 0, .1), inset 0 .125em .5em rgba(0, 0, 0, .15);
    }
```

signup.html

templates > signup.html > html > head

```
    .b-example-vr {
      flex-shrink: 0;
      width: 1.5rem;
      height: 100vh;
    }

    .bi {
      vertical-align: -.125em;
      fill: currentColor;
    }

    .nav-scroller {
      position: relative;
      z-index: 2;
      height: 2.75rem;
      overflow-y: hidden;
    }

    .nav-scroller .nav {
      display: flex;
      flex-wrap: nowrap;
      padding-bottom: 1rem;
      margin-top: -1px;
      overflow-x: auto;
      text-align: center;
      white-space: nowrap;
      -webkit-overflow-scrolling: touch;
    }
  </style>
  <link href="/static/css/signup.css" rel="stylesheet">
</head>
```

Signin.html

signin.html

templates > signin.html > ...

```
<!doctype html>
<html lang="en">

<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <title>IMSFR</title>
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css" rel="stylesheet"
    integrity="sha384-Zenh87qX5JnK2Jl0vWa8Ck2rdkQ2Bzep5IDxbcnCeuOxjzrPF/et3URy9Bv1WTRi" crossorigin="anonymous">
  <!-- <link rel="canonical" href="https://getbootstrap.com/docs/5.2/examples/sign-in/"> -->
  <style>
    .bd-placeholder-img {
      font-size: 1.125rem;
      text-anchor: middle;
      -webkit-user-select: none;
      -moz-user-select: none;
      user-select: none;
    }

    @media (min-width: 768px) {
      .bd-placeholder-img-lg {
        font-size: 3.5rem;
      }
    }

    .b-example-divider {
      height: 3rem;
      background-color: rgba(0, 0, 0, .1);
      border: solid rgba(0, 0, 0, .15);
      border-width: 1px 0;
      box-shadow: inset 0 .5em 1.5em rgba(0, 0, 0, .1), inset 0 .125em .5em rgba(0, 0, 0, .15);
    }
```

signin.html

templates > signin.html > html > body > div.form-signin.w-100.m-auto.text-center > form > table > tr > div.form-floating

```
<nav class="navbar navbar-expand-md navbar-light bg-transparent mb-4">
  <div class="container-fluid">
    <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarCollapse"
      aria-controls="navbarCollapse" aria-expanded="false" aria-label="Toggle navigation">
      <span class="navbar-toggler-icon"></span>
    </button>
    <div class="collapse navbar-collapse" id="navbarCollapse">
      <ul class="navbar-nav me-auto mb-2 mb-md-0">
        <li class="nav-item active">
          <a style="color: black; font-size: 17px; font-weight: 700; font-family:'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;" clas
        </li>
      </ul>
    </div>
  </div>
</nav>
<div class="form-signin w-100 m-auto text-center">
  <form method="POST">
    <h1 class="h3 mb-3 fw-normal">Please sign in</h1>
    <table>
  <tr>
          <div class="form-floating">
            <input type="email" class="form-control" name="email" id="email" placeholder="name@example.com" required>
            <label for="email">Email address</label>
          </div>
        </tr>
  <tr>
          <div class="form-floating">
            <input type="password" class="form-control" name="password" id="password" placeholder="Password" required>
            <label for="password">Password</label>
          </div>
        </tr>
        <tr>
```

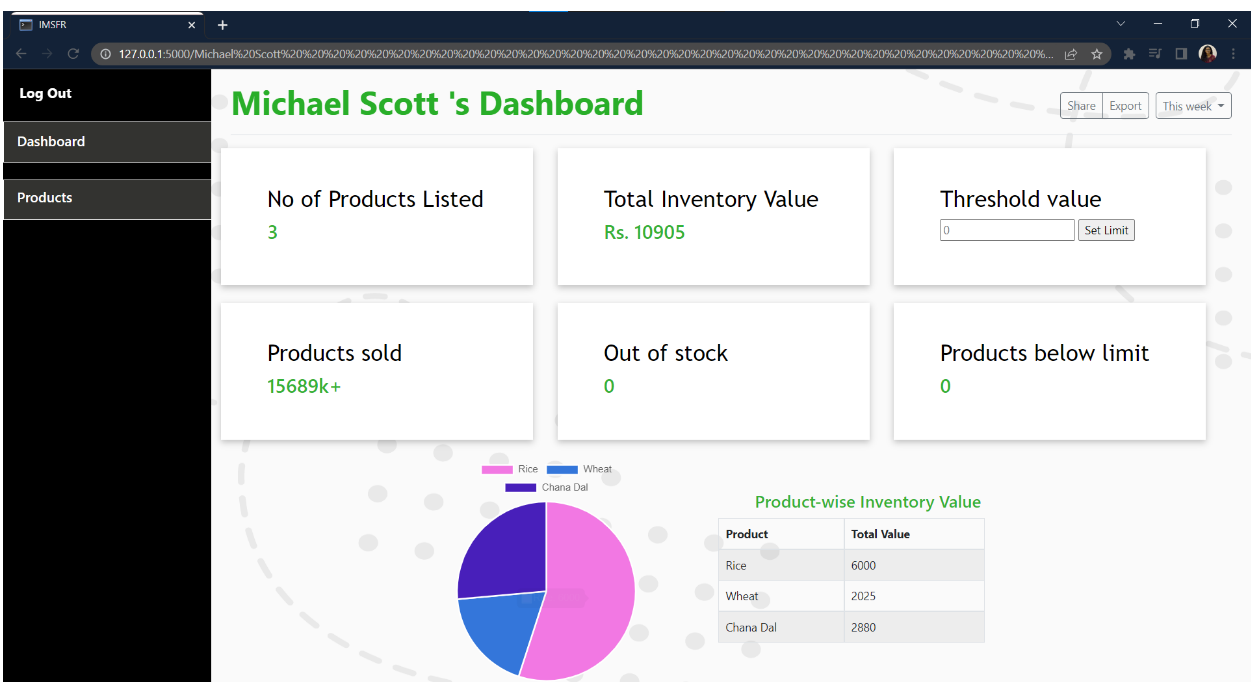
App.py

```
from app import app
from flask import request, redirect, url_for, render_template, session
import os
os.add_dll_directory(r'C:\Users\dharshini\AppData\Local\Programs\Python\Python311\Lib\site-packages\clidriver\bin')
os.add_dll_directory(r'C:\Users\dharshini\AppData\Local\Programs\Python\Python311\Lib\site-packages\clidriver\bin\icc64')
import ibm_db

from sendGrid import mailto, checkstatus

conn =ibm_db.connect("DATABASE=bludb;HOSTNAME=fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=32731;SE

@app.route("/")
def home():
    return render_template("home.html", title="Home")


@app.route("/signin", methods=('POST', 'GET'))
def signin():
    error = None
    if request.method == 'POST':
        email = request.form['email']
        password = request.form['password']

        sql = "SELECT username FROM users WHERE password = '{}' AND email = '{}'".format(password, email)
        stmt = ibm_db.exec_immediate(conn, sql)
        fetchUser = ibm_db.fetch_assoc(stmt)
        if fetchUser == False:
            error = "Incorrect Username/Password."

        if error is None:
            user = fetchUser["USERNAME"]
            session['loggedIn'] = True
```

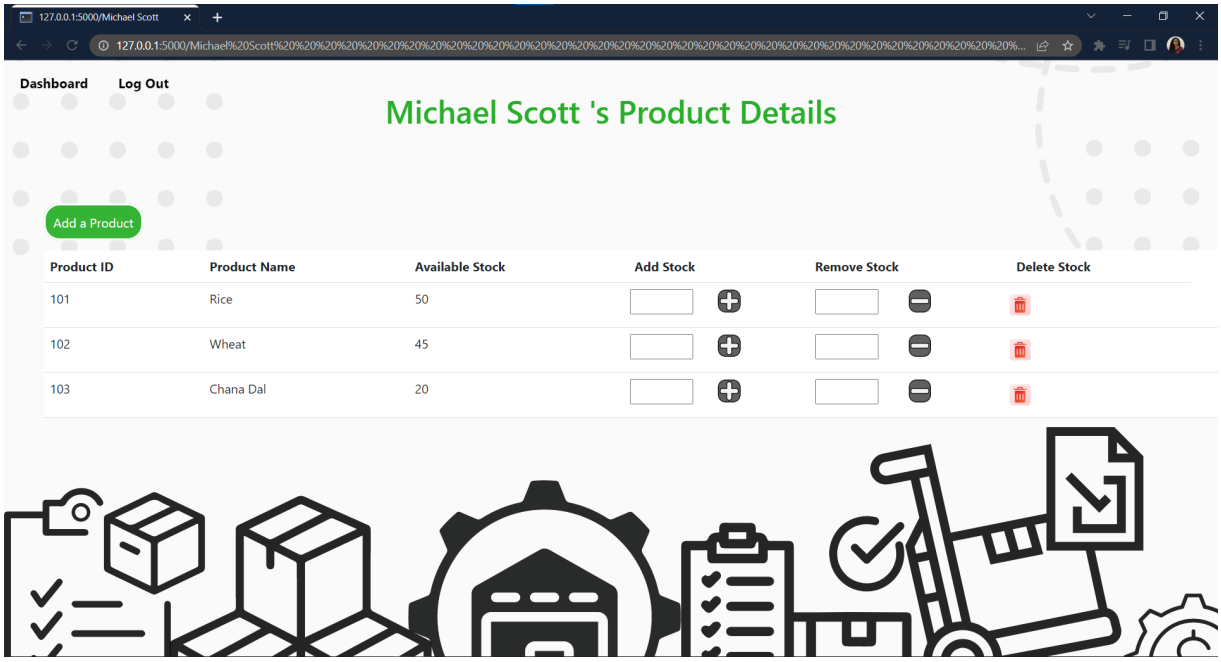
```
        fetchUser = ibm_db.fetch_assoc(stmt)
        if fetchUser == False:
            error = "Incorrect Username/Password."

        if error is None:
            user = fetchUser["USERNAME"]
            session['loggedIn'] = True
            session['id'] = user
            session['email'] = email
            return redirect(url_for('.dashboard', username=user))
    return render_template('signin.html', title='Sign In', error=error)


@app.route("/signup", methods=('POST', 'GET'))
def signup():
    if request.method == 'POST':
        username = request.form['username']
        email = request.form['email']
        password = request.form['password']
        contact = request.form['contact']
        companyname = request.form['companyname']
        companylocation = request.form['companylocation']
        checkUser = "SELECT * FROM users WHERE username = '{0}'".format(username)
        stmt = ibm_db.exec_immediate(conn, checkUser)
        findUser = ibm_db.fetch_assoc(stmt)
        if findUser == False:
            sql = "INSERT INTO users (email, username, password, contact, companyname, companylocation) VALUES ('{}', '{}', '{}', '{}', '{}', '
            ibm_db.exec_immediate(conn, sql)
            return render_template('home.html', title="Home", message="Registration Successful")
    return render_template("signup.html", title="Sign Up", error="Username aldready exists.")

```

7.2 Feature 2:

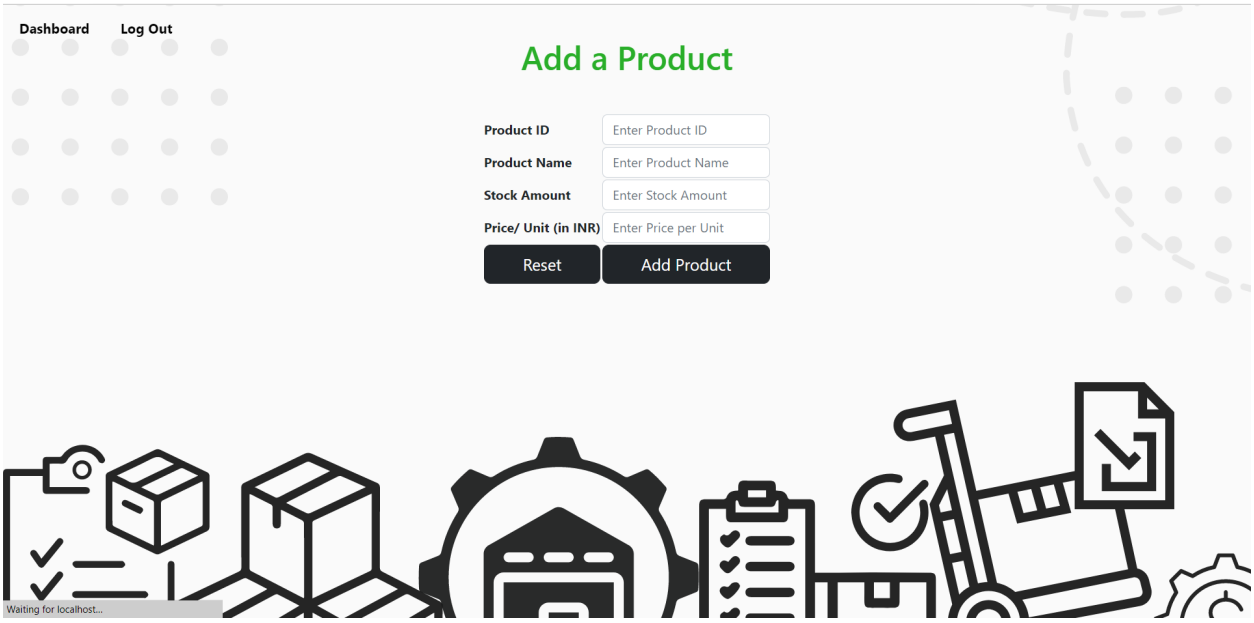
Dashboard:



Products Page:



Add Products Page:



CODE:

Dashboard.html

dashboard.html

templates > dashboard.html > html

```
<!doctype html>
<html lang="en">

<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <title>IMSFR</title>
  <link rel="canonical" href="https://getbootstrap.com/docs/5.2/examples/dashboard/">
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css" rel="stylesheet"
    integrity="sha384-Zenh87qX5JnK2Jl0vWa8Ck2rdkQ2Bzep5IDxbcnCeuOxjzrPF/et3URy9Bv1WTRi" crossorigin="anonymous">
    <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
  <style>
    html,
    body {
      height: 100%;
      background-image: url("\\static\\images\\dashboardbg.png");
      background-position: center;
      background-repeat: no-repeat;
      background-size: 100% 100%;
    }

    .bd-placeholder-img {
      font-size: 1.125rem;
      text-anchor: middle;
      -webkit-user-select: none;
      -moz-user-select: none;
      user-select: none;
    }

    @media (min-width: 768px) {
      .bd-placeholder-img-lg {
        font-size: 3.5rem;
```

dashboard.html

templates > dashboard.html > html

```
  <body>
    <header>
      <div>
        <nav class="navbar navbar-expand-md fixed-top navbar-light bg-transparent mb-4">
          <div class="container-fluid">
            <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarCollapse"
              aria-controls="navbarCollapse" aria-expanded="false" aria-label="Toggle navigation">
              <span class="navbar-toggler-icon"></span>
            </button>
            <div class="collapse navbar-collapse" id="navbarCollapse">
              <ul class="navbar-nav me-auto mb-2 mb-md-0">
                <li class="nav-item">
                  <a style="color: white; font-size: 17px; font-weight: 700; font-family:'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;"
                    href="{{ url_for('.logout') }}" class="nav-link">Log Out</a>
                </li>
              </ul>
            </div>
          </div>
        </nav>
      </div>
    </header>

    <div class="container-fluid">
      <div class="row">
        <nav id="sidebarMenu" class="col-md-3 col-lg-2 d-md-block sidebar collapse">
          <div class="position-sticky pt-3 sidebar-sticky">
            <ul class="nav flex-column">
              <li class="nav-item">
                <div class="navborder">
                  <a style="color: white; font-size: 17px;" class="nav-link" aria-current="page" href="#">
                    <span data-feather="home" class="align-text-bottom"></span>
                    Dashboard
```

Products.html

productsM.html

templates > productsM.html > ...

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css" rel="stylesheet"
        integrity="sha384-Zenh87qX5JnK2Jl0vWa8Ck2rdkQ2Bzep5IDxbcnCeuOxjzrPF/et3URy9Bv1WTRi" crossorigin="anonymous">
    <style>
        input[type="number"] {
            padding: 2px;
            width: 80px;
        }

        input[type="image"] {
            border: none;
            padding: 0;
            margin: 0;
        }

        .page-head {
            padding-top: 40px;
            text-align: center;
            color: rgb(41, 174, 41);
        }
    </style>
    <link href="/static/css/signin.css" rel="stylesheet">
</head>

<body>
    <nav class="navbar navbar-expand-md fixed-top navbar-light bg-transparent mb-4">
```

productsM.html

templates > productsM.html > ...

```
    </style>
    <link href="/static/css/signin.css" rel="stylesheet">
</head>

<body>
    <nav class="navbar navbar-expand-md fixed-top navbar-light bg-transparent mb-4">
        <div class="container-fluid">
            <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarCollapse"
                aria-controls="navbarCollapse" aria-expanded="false" aria-label="Toggle navigation">
                <span class="navbar-toggler-icon"></span>
            </button>
            <div class="collapse navbar-collapse" id="navbarCollapse">
                <ul class="navbar-nav me-auto mb-2 mb-md-0">
                    <li class="nav-item">
                        <a style="color: black; font-size: 17px; font-weight: 700; font-family:'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
                            href="{{ url_for('.dashboard', username=username) }}" class="nav-link">Dashboard</a>
                    </li>
                    <li class="nav-item">
                        <a style="color: black; font-size: 17px; font-weight: 700; font-family:'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
                            href="{{ url_for('.logout') }}" class="nav-link">Log Out</a>
                    </li>
                </ul>
            </div>
        </div>
    </nav>
    <div class="heading">
        <div class="page-head">
            <h1>{{username}}'s Product Details</h1>
        </div>
    </div>
    <div style="margin-left:50px;">
        <br><br><br><br>
        <a style="margin-bottom: 20px; color: black; font-size: 17px; font-weight: 700; font-family:'Segoe UI', Tahoma, Geneva, Verdana, sans-s
```

productsM.html

templates > productsM.html > ...

```
<div style="margin-left:50px;">
    <br>
    <table style="background-color: white;" class="table table-hover table-responsive">
        <tr>
            <th><label class="info head">Product ID</label></th>
            <th><label class="info head">Product Name</label></th>
            <th><label class="info head">Available Stock</label></th>
            <th><label class="info head"></label></th>
            <th><label style="margin-left:-195px;" class="info head">Add Stock</label></th>
            <!-- <th><label class="info head"></label></th> -->
            <th><label class="info head">Remove Stock</label></th>
            <th><label class="info head">Delete Stock</label></th>
        </tr>
        {% for product in products %}
        <tr>
            <td><label class="info">{{product['PRODUCTID']}}</label></td>
            <td><label class="info">{{product['PRODUCTNAME']}}</label></td>
            <td><label class="info">{{product['AVAILABLESTOCK']}}</label></td>
            <form method='POST'
                action="{{url_for('.editProduct', username=username, action='add', pid=product['PRODUCTID'])}}">
                <td><label class="info">
                        <input type="number" name="newstock" id="newstock" style="color: black;" required>
                    </label></td>
                <td>
                    <label style="margin-left: -90px;" class="info">
                        <input type="image" src="../static/add.png" width="30" height="30">
                    </label>
                </td>
            </form>
            </td>
            <form method='POST'
                action="{{url_for('.editProduct', username=username, action='remove', pid=product['PRODUCTID'])}}">
```

AddProduct.html

addProduct.html

templates > addProduct.html > html > body > form.form > div.form_input

```
{% block content %}
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="../static/form.css">
</head>

<body>
    <form class="form" method="POST">
        <h1>Add a Product</h1>
        <h2>{{username}}</h2>
        <div class="form_input">
            {% if error %}
                <label style="color: red;">{{error}}</label>
            {% endif %}
            <div class="form-element">
                <label for="productid">Product ID</label>
                <input type="text" name="pid" id="pid" placeholder="Enter Product ID" required>
            </div>
            <div class="form-element">
                <label for="productid">Product Name</label>
                <input type="text" name="pname" id="pname" placeholder="Enter Product Name" required>
            </div>
            <div class="form-element">
                <label for="productid">Stock Amount</label>
                <input type="number" name="stock" id="stock" placeholder="Enter Stock Amount" required>
            </div>
            <div class="form-element">
                <label for="productid">Price/Unit (in INR)</label>
```

App.py

app.py 4 ×

app.py > ...

```python
@app.route("/<username>/dashboard", methods=('POST', 'GET'))
@app.route("/<username>", methods=('POST', 'GET'))
def dashboard(username):
    fetchPrices = "SELECT p.productname,up.unitprice,up.availablestock FROM products p,userproducts up WHERE p.productid=up.productid AND up.user
    stmt = ibm_db.exec_immediate(conn, fetchPrices)
    productInfo = ibm_db.fetch_assoc(stmt)
    products=[]
    prices=[]
    while productInfo != False:
        products.append(productInfo['PRODUCTNAME'])
        prices.append(productInfo['UNITPRICE'] * productInfo['AVAILABLESTOCK'])
        productInfo = ibm_db.fetch_assoc(stmt)

    if request.method=='POST':
        th_value = request.form['threshold']
        sql = "INSERT INTO threshold_value (email, th_value) VALUES ('{}', '{}');".format(session['email'], th_value)
        ibm_db.exec_immediate(conn, sql)
        return redirect(url_for('.dashboard', username=username))
    return render_template("dashboard.html", username=username, success=True, overallValue=sum(prices))


@app.route("/<username>/manageProducts", methods=('POST', 'GET'))
def manageProducts(username):
    sql = "SELECT up.productid,p.productname,up.availablestock FROM products p, users u, userproducts up WHERE u.username=up.username AND p.produ
    stmt = ibm_db.exec_immediate(conn, sql)
    getProducts = ibm_db.fetch_assoc(stmt)
    products = []
    while getProducts != False:
        products.append(getProducts)
        getProducts = ibm_db.fetch_assoc(stmt)
    return render_template("productsM.html", username=username, success=True, products=products)

```

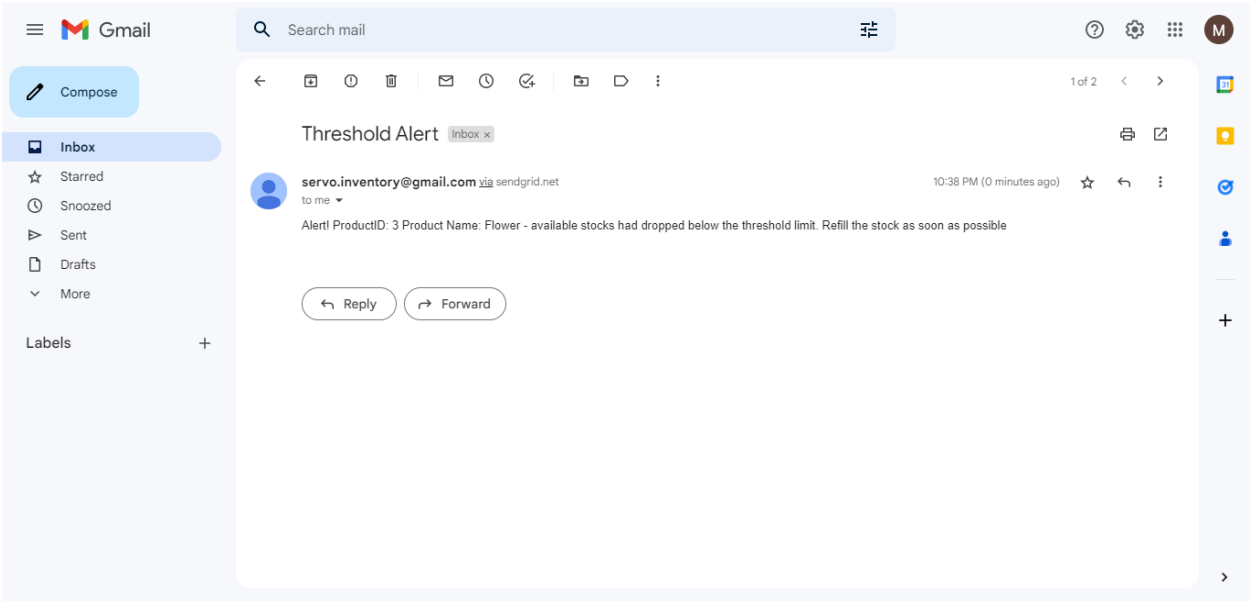
app.py 4 ×

app.py > ...

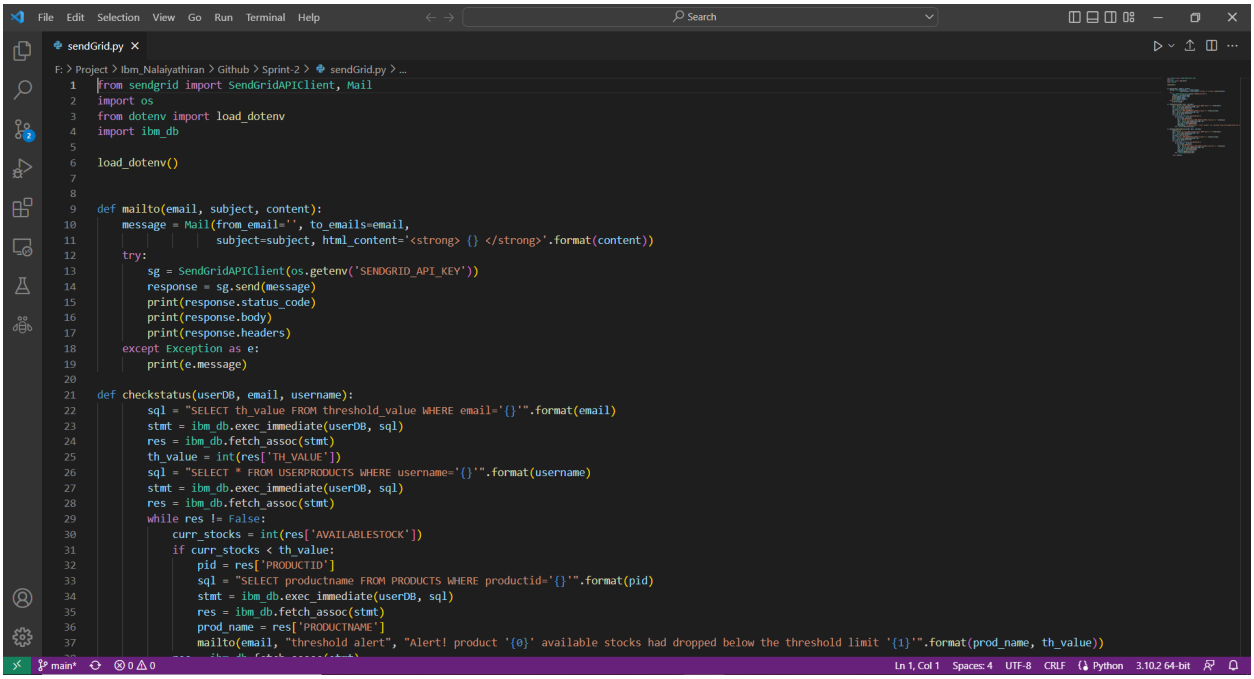
```python

@app.route("/<username>/manageProducts/edit=<pid>,action=<action>", methods=('POST', 'GET'))
def editProduct(username, pid, action):
    if request.method == 'POST':
        stock = int(request.form['newstock'])
        checkAvailable = "SELECT * FROM userproducts WHERE productid='{}' AND username='{}';".format(pid, username)
        statement = ibm_db.exec_immediate(conn, checkAvailable)
        productDetails = ibm_db.fetch_assoc(statement)
        available = productDetails['AVAILABLESTOCK']
        if action == "add":
            stock = available + stock
        else:
            stock = available - stock
            checkstatus(conn, session['email'], username)
        updateQuery = "UPDATE userproducts SET availablestock='{}' WHERE productid='{}' AND username='{}';".format(stock, pid, username)
        ibm_db.exec_immediate(conn, updateQuery)
    return redirect(url_for('.manageProducts', username=username))


@app.route("/<username>/manageProducts/delete=<pid>", methods=('POST', 'GET'))
def deleteProduct(username, pid):
    deleteQuery = "DELETE FROM userproducts WHERE productid='{}' AND username='{}';".format(pid, username)
    ibm_db.exec_immediate(conn, deleteQuery)
    return redirect(url_for('.manageProducts', username=username))


@app.route("/<username>/addProduct", methods=('POST', 'GET'))
def addProducts(username):
    error = None
    if request.method == 'POST':
        pid = request.form['pid']
        pname = request.form['pname']
```

Feature 3:

Email Alerts:

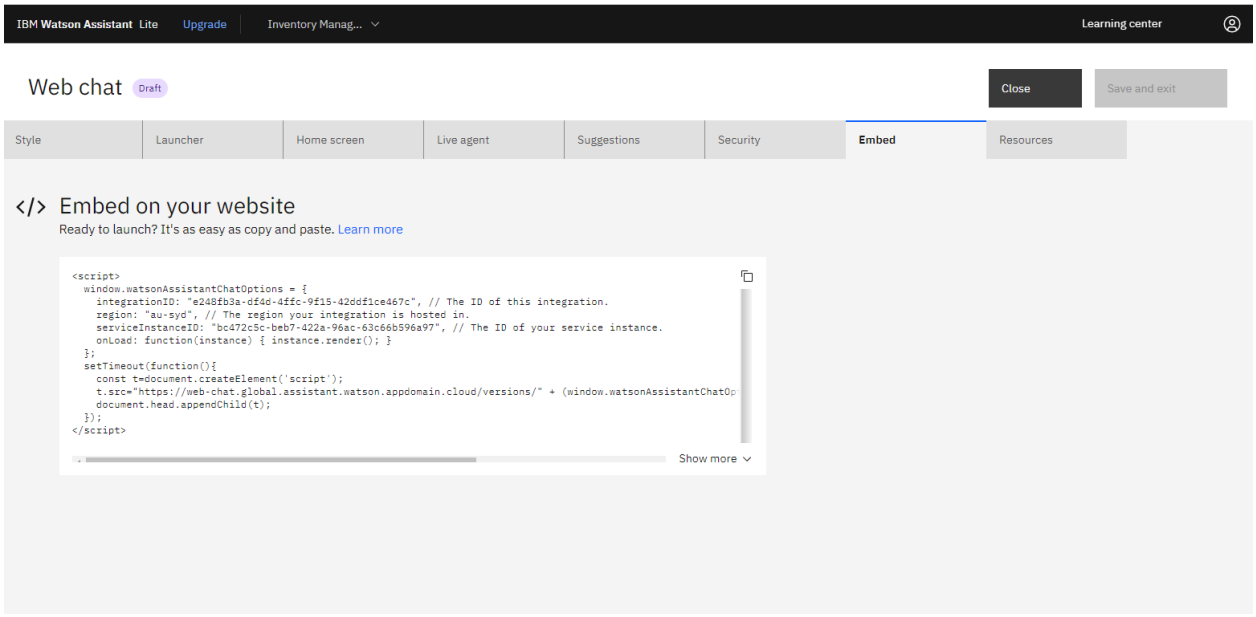


SendGrid Integration:

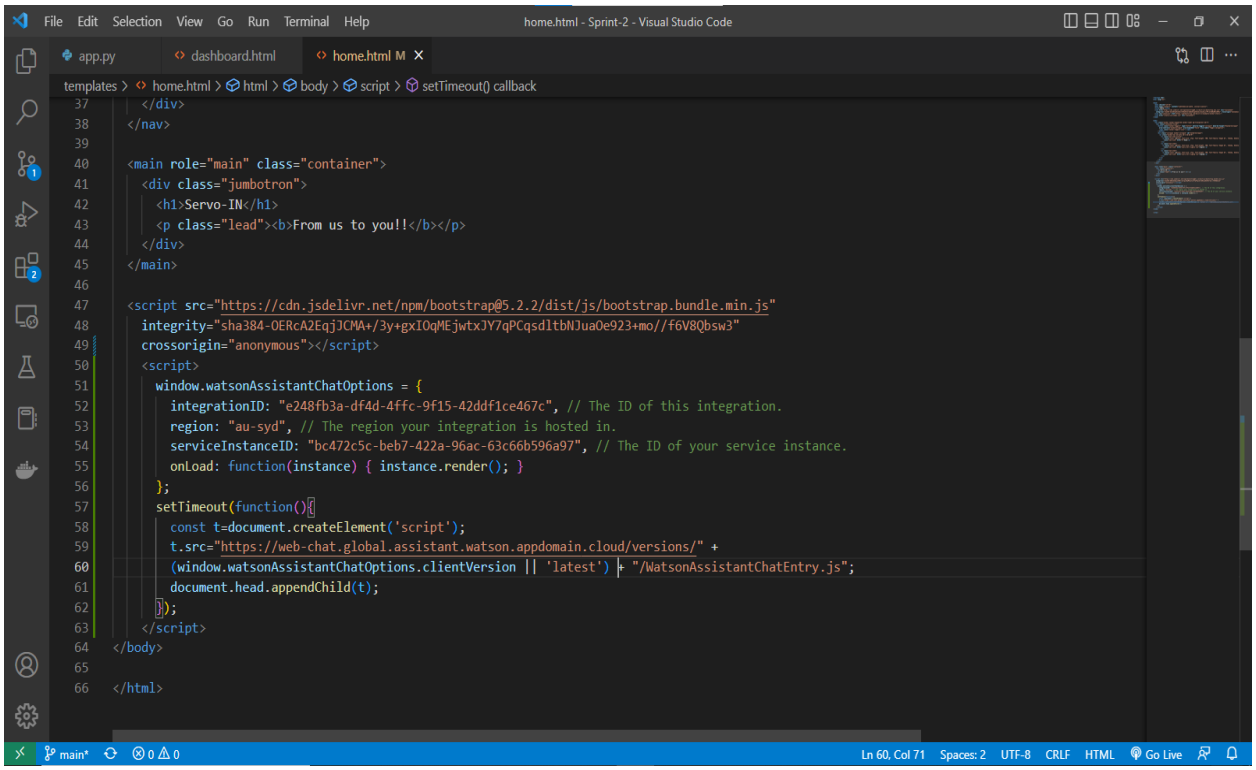


Feature 4:

Chatbot Integration:


IBM Watson Assistant Lite
Upgrade
Inventory Manag...
Learning center
Web chat
Draft
Close
Save and exit
Style
Launcher
Home screen
Live agent
Suggestions
Security
Embed
Resources
Embed on your website
Ready to launch? It's as easy as copy and paste. Learn more
<script>
window.watsonAssistantChatOptions = {
integrationID: "e248fb3a-df4d-4ffc-9f15-42ddf1ce467c", // The ID of this integration.
region: "au-syd", // The region your integration is hosted in.
serviceInstanceID: "bc472c5c-beb7-422a-96ac-63c66b596a97", // 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.watsonAssistantChatOp
document.head.appendChild(t);
});
</script>
Show more

Code:


File Edit Selection View Go Run Terminal Help
home.html - Sprint-2 - Visual Studio Code
app.py
dashboard.html
home.html M
templates > home.html > html > body > script > setTimeout() callback

</nav>
<main role="main" class="container">

<h1>Servo-IN</h1>
From us to you!!

</main>
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/js/bootstrap.bundle.min.js"
integrity="sha384-OERcA2EqjJCMA+/3y+gxIOqMEjwtxJY7qPCqsdltbNJuaOe923+mo//f6V8Qbsw3"
crossorigin="anonymous"></script>
<script>
window.watsonAssistantChatOptions = {
integrationID: "e248fb3a-df4d-4ffc-9f15-42ddf1ce467c", // The ID of this integration.
region: "au-syd", // The region your integration is hosted in.
serviceInstanceID: "bc472c5c-beb7-422a-96ac-63c66b596a97", // 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(t);
});
</script>
</body>
</html>
main*
Ln 60, Col 71
Spaces: 2
UTF-8
CRLF
HTML
Go Live

8. TESTING :

8.1 Test Cases:

					Team ID	PNT2022TMID03825						
					Project Name	Inventory Management System for Retailers						
Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Comments	TC for Automation(Y/N)	BUG ID	Executed By
HomePage_TC_OO1	Functional	Home Page	Displays sign in,signup actions to user and allows the access to chatbot	HTML, CSS, Python, Flask	1.Enter the URL in the browser and search it. 2.Home page of the website Servo-in is displayed in the browser.		Application displays the home page of website Servo-In	Working as expected	Pass				Dharshini T Om Arthy Sri B Srinithi S
SignUpPage_TC_OO2	Functional	SignUp Page	User enters his unique personel credentials and is redirected to sign in page	HTML, CSS, Python, Flask	1. Signup and Signin is present in the dashboard of the home page. 2. Click on the Signup button and it is redirected to signup page. 3. In Signup, the retailers must fill the required feilds like name, username, password, company name, company location. 4. New user is created and their records are stored in IBM DB2	Username:riya Email:riya@gmail.com Password:riya Contact number:9223467891 Company name:RRR Company Location:Chennai	1. User can fill the required details and those datas are stored in database. 2. After signup, the website is redirected to dashboard.	Working as expected	Pass	After sign up user is redirected to sign in page			Harini M Om Arthy Sri B Srinithi S
SignInPage_TC_OO3	Functional	SignIn page	User is authenticated with his email and password and is redirected to his customized dashboard	HTML, CSS, Python, Flask	1. Signup and Signin is present in the dashboard of the home page. 2. Click on the Signup button and it is redirected to signin page. 3.Enter Valid username/email in Email text box 4.Enter valid password in password text box 5.Click on login button	Username:riya@gmail.com password: riya	1. Signin page is displayed. 2. After successful signin, dashboard is redirected	Working as expected	Pass	After sign in user is redirected to dashboard page			Harini M Dharshini T Srinithi S
DashboardPage_TC_OO4	UI	Dashboard page	Displays the products in stock and out of stock and also displays the pictorial representation of instock products	HTML, CSS, Python, Flask	1.Enter URL and click go 2.Click on My Account dropdown button 3.Enter InValid username/email in Email text box 4.Enter valid password in password text box 5.Click on login button		Dashboard of the specific user is displayed along with product details, products price etc.	Working as expected	Pass	Dashboard displays all necessary details			Harini M Dharshini T Om Arthy Sri B
DisplayProductPage_TC_OO4	Functional	Display Products page	Displays the instock products	HTML, CSS, Python, Flask	1.Enter URL and click go 2.User can either login/signup. 3. Once after login/signup, user will be redirected to dashboard. 4. Product display page is shown here.	AddStock:10 RemoveStock:5	1. Products are displayed in this page	Working as expected	Pass				Harini M Dharshini T
AddProductPage_TC_OO5	Functional	Add Products page	Allows the user to add new products to the system	HTML, CSS, Python, Flask	1.Enter URL and click go 2.User can either login/signup. 3. Once after login/signup, user will be redirected to dashboard. 4. Retailers can add the product by clicking product button in dashboard. 5. Add product button is present in product display page and it is redirected to add product page on clicking it.	Product Id:1234 Product name:Clove StockAmount:5kg Price/Unit:40	1. Form to add products by user is displayed in this page	Working as expected	Pass				Om Arthy Sri B Srinithi S

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 [ProductName] project at the time of the release to User Acceptance Testing (UAT).

2. **Defect Analysis**

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

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	10	4	2	4	20
Duplicate	2	0	3	0	5
External	5	3	0	1	9
Fixed	12	2	5	20	39
Not Reproduced	1	0	1	0	2
Skipped	0	0	1	1	2
Won't Fix	0	5	2	1	8
Totals	30	14	14	27	85

3. Test Case Analysis

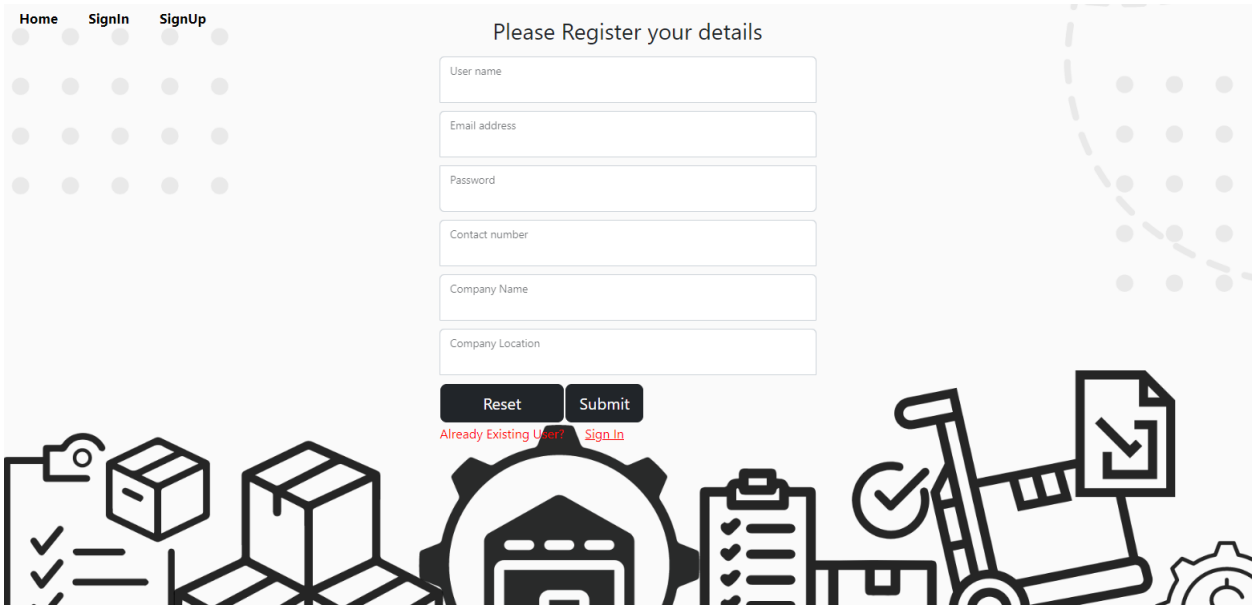
This report shows the number of test cases that have passed, failed, and untested

Section	Total Cases	Not Tested	Fail	Pass
Print Engine	8	0	0	8
Client Application	50	0	0	50
Security	3	0	0	3
Outsource Shipping	3	0	0	3
Exception Reporting	10	0	0	10
Final Report Output	5	0	0	5
Version Control	1	0	0	1

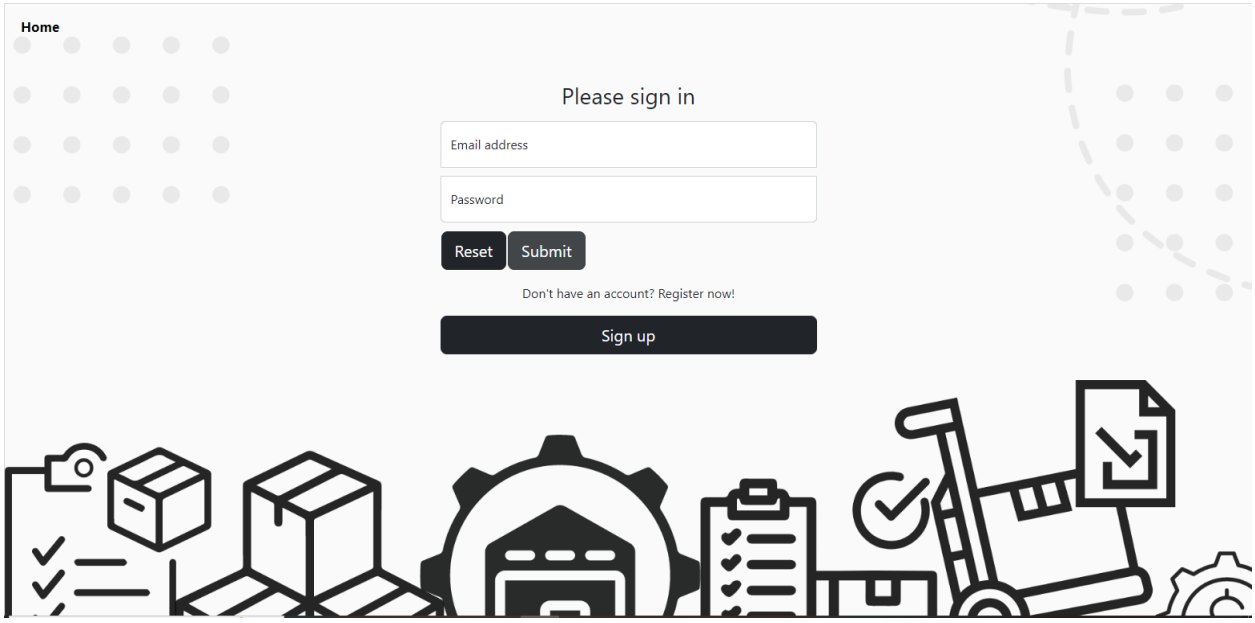
9. RESULTS :

Home Page:

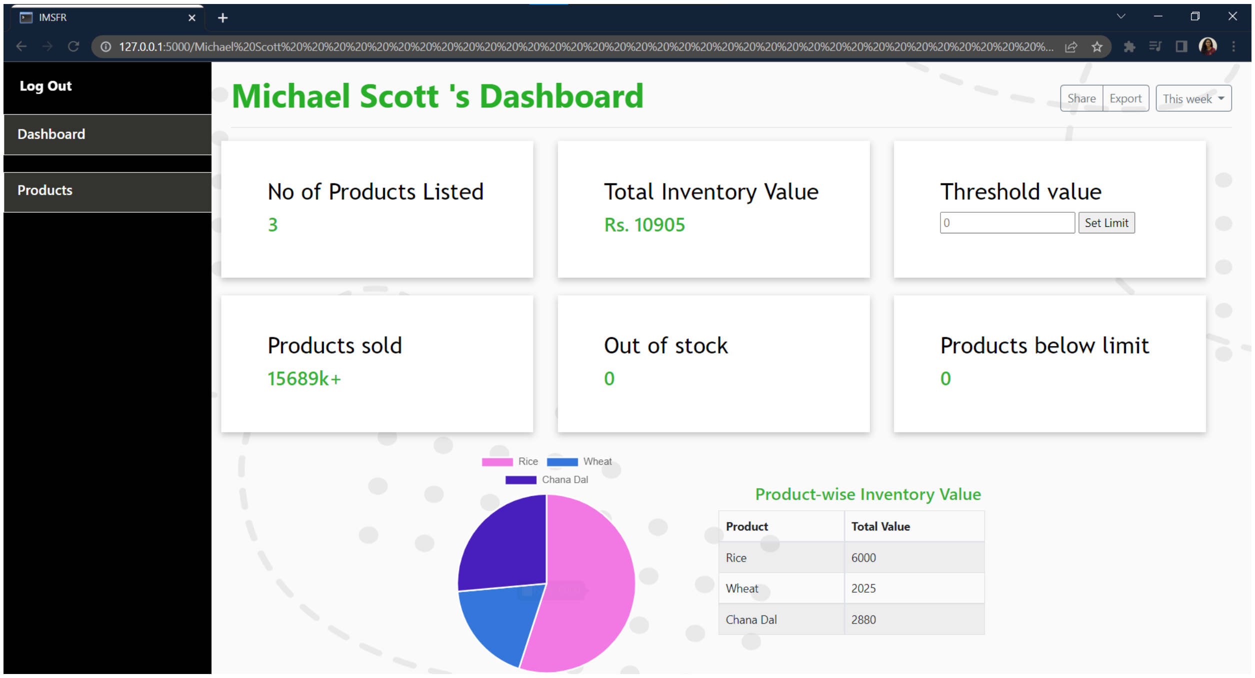
Sign Up Page:



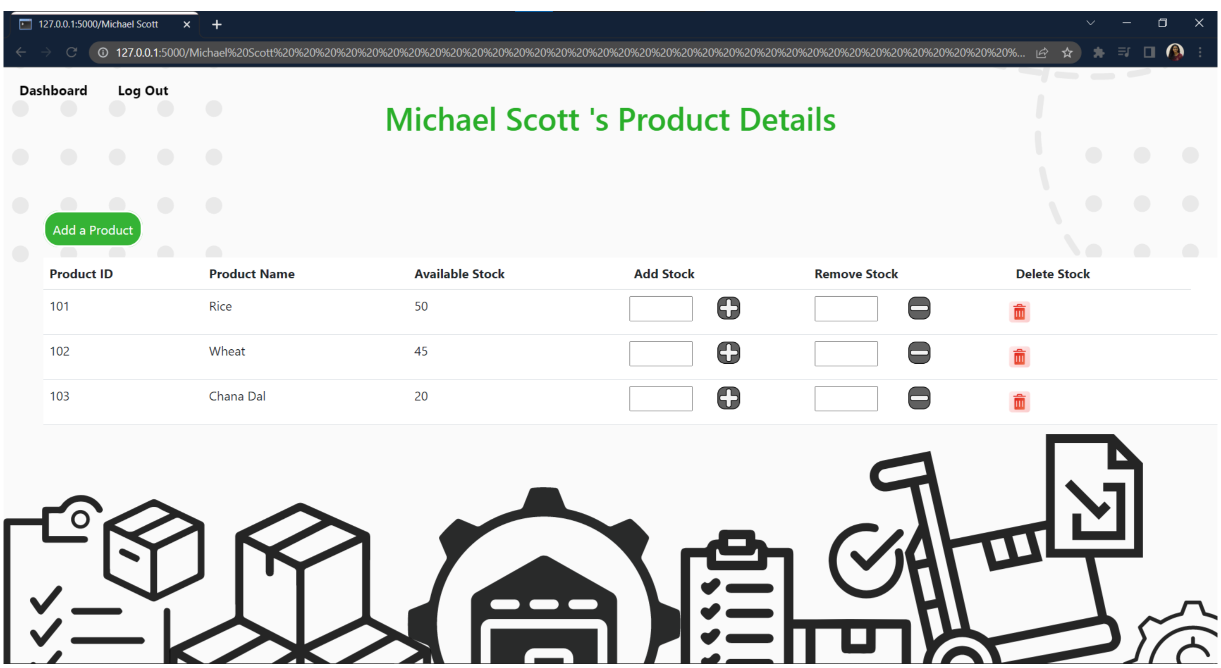
Sign In Page:



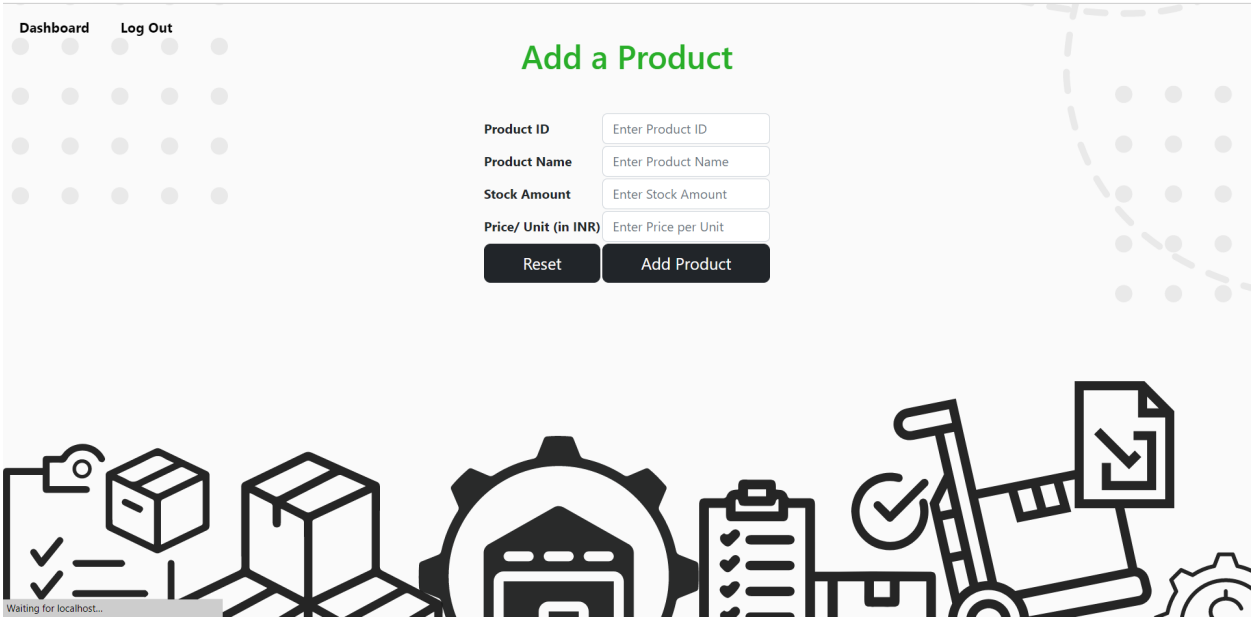
Dashboard:



Products Page:



Add Products Page:



Chatbot Integration in Home Page:

10.ADVANTAGES & DISADVANTAGES:

Advantages:

- Reduced Risk of Overselling
- Cost Savings
- Avoiding Stockouts and Excess Stock
- More Productivity
- Increased Profits
- Better Customer Experience

Disadvantages:

- Expensive for Small Businesses
- Complex to Learn
- Risk of System Crashes
- Malicious Hacks

11.CONCLUSION:

A highly difficult but crucial component of the supply chain is inventory management. Stock-related expenses, such as warehousing, carrying, and ordering costs, can be decreased with the aid of an efficient inventory management system.

12.FUTURE SCOPE:

Since we had very little prior understanding of the Inventory Management System when we began this project, we learned about the improvement capacity as we built it. Below are some of the areas of focus we might broaden for improved efficiency:

-> Designing an interactive user interface.

-> manage stock prudently

-> Making the system adaptable to any situation.

-> To enable product returns, a sales and buy return mechanism will be developed.

13.APPENDIX:

Source Code

addProduct.html:

addProduct.html ×

templates > addProduct.html > html > body > form.form > div.form_input

```
{% block content %}
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="../static/form.css">
</head>

<body>
    <form class="form" method="POST">
        <h1>Add a Product</h1>
        <h2>{{username}}</h2>
        <div class="form_input">
            {% if error %}
                <label style="color: red;">{{error}}</label>
            {% endif %}
            <div class="form-element">
                <label for="productid">Product ID</label>
                <input type="text" name="pid" id="pid" placeholder="Enter Product ID" required>
            </div>
            <div class="form-element">
                <label for="productid">Product Name</label>
                <input type="text" name="pname" id="pname" placeholder="Enter Product Name" required>
            </div>
            <div class="form-element">
                <label for="productid">Stock Amount</label>
                <input type="number" name="stock" id="stock" placeholder="Enter Stock Amount" required>
            </div>
            <div class="form-element">
                <label for="productid">Price/Unit (in INR)</label>
```

dashboard.html:

dashboard.html

templates > dashboard.html > html

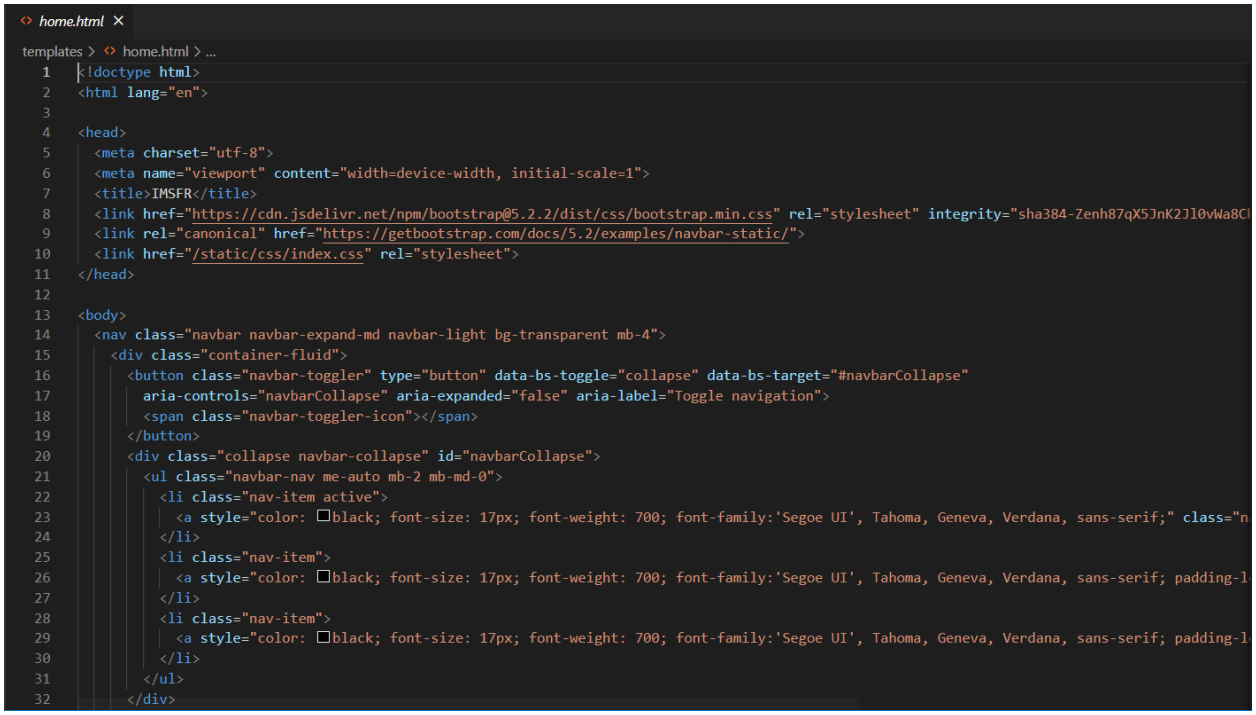
```
<!doctype html>
<html lang="en">

<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <title>IMSFR</title>
  <link rel="canonical" href="https://getbootstrap.com/docs/5.2/examples/dashboard/">
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css" rel="stylesheet"
    integrity="sha384-Zenh87qX5JnK2Jl0vWa8Ck2rdkQ2Bzep5IDxbcnCeuOxjzrPF/et3URy9Bv1WTRi" crossorigin="anonymous">
    <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
  <style>
    html,
    body {
      height: 100%;
      background-image: url("\\static\\images\\dashboardbg.png");
      background-position: center;
      background-repeat: no-repeat;
      background-size: 100% 100%;
    }

    .bd-placeholder-img {
      font-size: 1.125rem;
      text-anchor: middle;
      -webkit-user-select: none;
      -moz-user-select: none;
      user-select: none;
    }

    @media (min-width: 768px) {
      .bd-placeholder-img-lg {
        font-size: 3.5rem;
```

dashboard.html

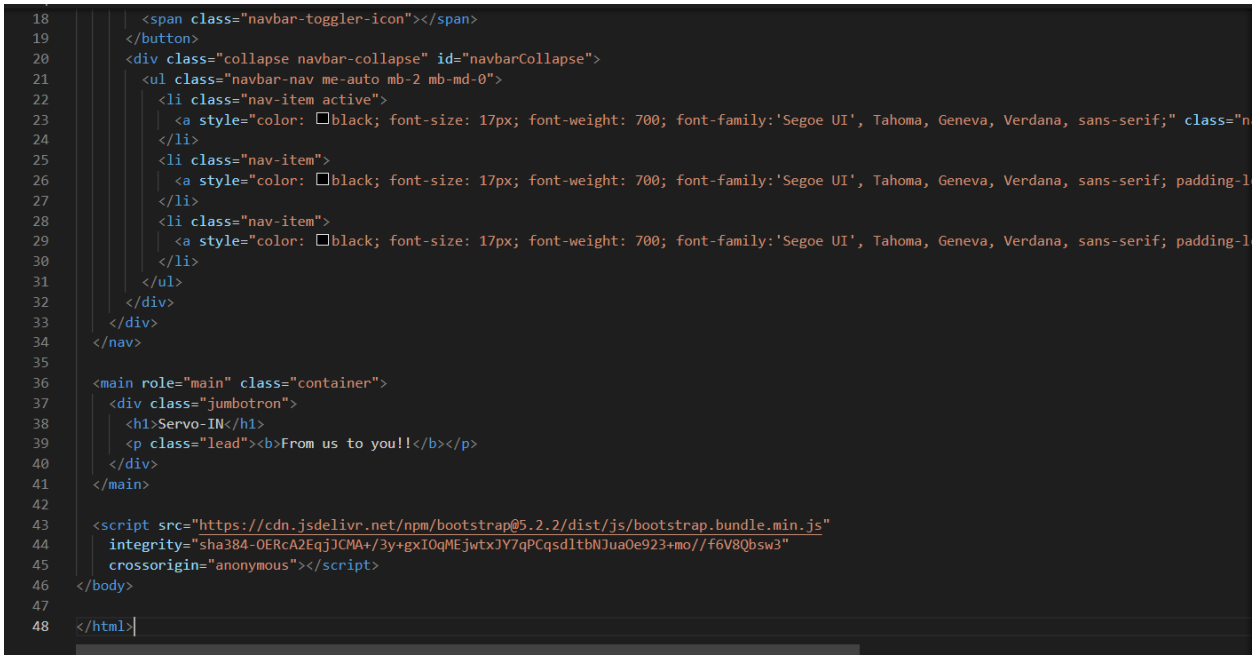
templates > dashboard.html > html

```
<body>
  <header>
    <div>
      <nav class="navbar navbar-expand-md fixed-top navbar-light bg-transparent mb-4">
        <div class="container-fluid">
          <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarCollapse"
            aria-controls="navbarCollapse" aria-expanded="false" aria-label="Toggle navigation">
            <span class="navbar-toggler-icon"></span>
          </button>
          <div class="collapse navbar-collapse" id="navbarCollapse">
            <ul class="navbar-nav me-auto mb-2 mb-md-0">
              <li class="nav-item">
                <a style="color: white; font-size: 17px; font-weight: 700; font-family:'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;"
                  href="{{ url_for('.logout') }}" class="nav-link">Log Out</a>
              </li>
            </ul>
          </div>
        </div>
      </nav>
    </div>
  </header>

  <div class="container-fluid">
    <div class="row">
      <nav id="sidebarMenu" class="col-md-3 col-lg-2 d-md-block sidebar collapse">
        <div class="position-sticky pt-3 sidebar-sticky">
          <ul class="nav flex-column">
            <li class="nav-item">
              <div class="navborder">
                <a style="color:white; font-size: 17px;" class="nav-link" aria-current="page" href="#">
                  <span data-feather="home" class="align-text-bottom"></span>
                  Dashboard
```

home.html:



```
templates > home.html > ...
1  <!doctype html>
2  <html lang="en">
3
4  <head>
5    <meta charset="utf-8">
6    <meta name="viewport" content="width=device-width, initial-scale=1">
7    <title>IMSFR</title>
8    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-Zenh87qX5JnK2Jl0vWa8Ck
9    <link rel="canonical" href="https://getbootstrap.com/docs/5.2/examples/navbar-static/">
10   <link href="/static/css/index.css" rel="stylesheet">
11 </head>
12
13 <body>
14   <nav class="navbar navbar-expand-md navbar-light bg-transparent mb-4">
15     <div class="container-fluid">
16       <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarCollapse"
17         aria-controls="navbarCollapse" aria-expanded="false" aria-label="Toggle navigation">
18         <span class="navbar-toggler-icon"></span>
19       </button>
20       <div class="collapse navbar-collapse" id="navbarCollapse">
21         <ul class="navbar-nav me-auto mb-2 mb-md-0">
22           <li class="nav-item active">
23             <a style="color: black; font-size: 17px; font-weight: 700; font-family:'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;" class="n
24           </li>
25           <li class="nav-item">
26             <a style="color: black; font-size: 17px; font-weight: 700; font-family:'Segoe UI', Tahoma, Geneva, Verdana, sans-serif; padding-l
27           </li>
28           <li class="nav-item">
29             <a style="color: black; font-size: 17px; font-weight: 700; font-family:'Segoe UI', Tahoma, Geneva, Verdana, sans-serif; padding-l
30           </li>
31         </ul>
32       </div>
```



```
18         <span class="navbar-toggler-icon"></span>
19       </button>
20       <div class="collapse navbar-collapse" id="navbarCollapse">
21         <ul class="navbar-nav me-auto mb-2 mb-md-0">
22           <li class="nav-item active">
23             <a style="color: black; font-size: 17px; font-weight: 700; font-family:'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;" class="n
24           </li>
25           <li class="nav-item">
26             <a style="color: black; font-size: 17px; font-weight: 700; font-family:'Segoe UI', Tahoma, Geneva, Verdana, sans-serif; padding-l
27           </li>
28           <li class="nav-item">
29             <a style="color: black; font-size: 17px; font-weight: 700; font-family:'Segoe UI', Tahoma, Geneva, Verdana, sans-serif; padding-l
30           </li>
31         </ul>
32       </div>
33     </div>
34   </nav>
35
36   <main role="main" class="container">
37     <div class="jumbotron">
38       <h1>Servo-IN</h1>
39       <p class="lead"><b>From us to you!!</b></p>
40     </div>
41   </main>
42
43   <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/js/bootstrap.bundle.min.js"
44     integrity="sha384-OERcA2EqjJCMA+/3y+gxIOqMEjwtxJY7qPCqsdltbNJuaOe923+mo//f6V8Qbsw3"
45     crossorigin="anonymous"></script>
46 </body>
47
48 </html>
```

products.html:

productsM.html

templates > productsM.html > ...

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css" rel="stylesheet"
        integrity="sha384-Zenh87qX5JnK2Jl0vWa8Ck2rdkQ2Bzep5IDxbcnCeuOxjzrPF/et3URy9Bv1WTRi" crossorigin="anonymous">
    <style>
        input[type="number"] {
            padding: 2px;
            width: 80px;
        }

        input[type="image"] {
            border: none;
            padding: 0;
            margin: 0;
        }

        .page-head {
            padding-top: 40px;
            text-align: center;
            color: rgb(41, 174, 41);
        }
    </style>
    <link href="/static/css/signin.css" rel="stylesheet">
</head>

<body>
    <nav class="navbar navbar-expand-md fixed-top navbar-light bg-transparent mb-4">
```

productsM.html

templates > productsM.html > ...

```
    </style>
    <link href="/static/css/signin.css" rel="stylesheet">
</head>

<body>
    <nav class="navbar navbar-expand-md fixed-top navbar-light bg-transparent mb-4">
        <div class="container-fluid">
            <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarCollapse"
                aria-controls="navbarCollapse" aria-expanded="false" aria-label="Toggle navigation">
                <span class="navbar-toggler-icon"></span>
            </button>
            <div class="collapse navbar-collapse" id="navbarCollapse">
                <ul class="navbar-nav me-auto mb-2 mb-md-0">
                    <li class="nav-item">
                        <a style="color: black; font-size: 17px; font-weight: 700; font-family:'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
                            href="{{ url_for('.dashboard', username=username) }}" class="nav-link">Dashboard</a>
                    </li>
                    <li class="nav-item">
                        <a style="color: black; font-size: 17px; font-weight: 700; font-family:'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
                            href="{{ url_for('.logout') }}" class="nav-link">Log Out</a>
                    </li>
                </ul>
            </div>
        </div>
    </nav>
    <div class="heading">
        <div class="page-head">
            <h1>{{username}}'s Product Details</h1>
        </div>
    </div>
    <div style="margin-left:50px;">
        <br><br><br><br>
        <a style="margin-bottom: 20px; color: black; font-size: 17px; font-weight: 700; font-family:'Segoe UI', Tahoma, Geneva, Verdana, sans-s
```

productsM.html

templates > productsM.html > ...

```
<div style="margin-left:50px;">
    <br>
    <table style="background-color: white;" class="table table-hover table-responsive">
        <tr>
            <th><label class="info head">Product ID</label></th>
            <th><label class="info head">Product Name</label></th>
            <th><label class="info head">Available Stock</label></th>
            <th><label class="info head"></label></th>
            <th><label style="margin-left:-195px;" class="info head">Add Stock</label></th>
            <!-- <th><label class="info head"></label></th> -->
            <th><label class="info head">Remove Stock</label></th>
            <th><label class="info head">Delete Stock</label></th>
        </tr>
        {% for product in products %}
        <tr>
            <td><label class="info">{{product['PRODUCTID']}}</label></td>
            <td><label class="info">{{product['PRODUCTNAME']}}</label></td>
            <td><label class="info">{{product['AVAILABLESTOCK']}}</label></td>
            <form method='POST'
                action="{{url_for('.editProduct', username=username, action='add', pid=product['PRODUCTID'])}}">
                <td><label class="info">
                        <input type="number" name="newstock" id="newstock" style="color:black;" required>
                    </label></td>
                <td>
                    <label style="margin-left: -90px;" class="info">
                        <input type="image" src="../static/add.png" width="30" height="30">
                    </label>
                </td>
            </form>
            </td>
            <form method='POST'
                action="{{url_for('.editProduct', username=username, action='remove', pid=product['PRODUCTID'])}}">
                <td><label class="info">
```

signin.html:

signin.html

templates > signin.html > ...

```
<!doctype html>
<html lang="en">

<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <title>IMSFR</title>
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css" rel="stylesheet"
    integrity="sha384-Zenh87qX5JnK2Jl0vWa8Ck2rdkQ2Bzep5IDxbcnCeuOxjzrPF/et3URy9Bv1WTRi" crossorigin="anonymous">
  <!-- <link rel="canonical" href="https://getbootstrap.com/docs/5.2/examples/sign-in/"> -->
  <style>
    .bd-placeholder-img {
      font-size: 1.125rem;
      text-anchor: middle;
      -webkit-user-select: none;
      -moz-user-select: none;
      user-select: none;
    }

    @media (min-width: 768px) {
      .bd-placeholder-img-lg {
        font-size: 3.5rem;
      }
    }

    .b-example-divider {
      height: 3rem;
      background-color: rgba(0, 0, 0, .1);
      border: solid rgba(0, 0, 0, .15);
      border-width: 1px 0;
      box-shadow: inset 0 .5em 1.5em rgba(0, 0, 0, .1), inset 0 .125em .5em rgba(0, 0, 0, .15);
    }
```

signin.html

templates > signin.html > html > body > div.form-signin.w-100.m-auto.text-center > form > table > tr > div.form-floating

```
<nav class="navbar navbar-expand-md navbar-light bg-transparent mb-4">
  <div class="container-fluid">
    <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarCollapse"
      aria-controls="navbarCollapse" aria-expanded="false" aria-label="Toggle navigation">
      <span class="navbar-toggler-icon"></span>
    </button>
    <div class="collapse navbar-collapse" id="navbarCollapse">
      <ul class="navbar-nav me-auto mb-2 mb-md-0">
        <li class="nav-item active">
          <a style="color: black; font-size: 17px; font-weight: 700; font-family:'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;" clas
        </li>
      </ul>
    </div>
  </div>
</nav>
<div class="form-signin w-100 m-auto text-center">
  <form method="POST">
    <h1 class="h3 mb-3 fw-normal">Please sign in</h1>
    <table>
  <tr>
          <div class="form-floating">
            <input type="email" class="form-control" name="email" id="email" placeholder="name@example.com" required>
            <label for="email">Email address</label>
          </div>
        </tr>
  <tr>
          <div class="form-floating">
            <input type="password" class="form-control" name="password" id="password" placeholder="Password" required>
            <label for="password">Password</label>
          </div>
        </tr>
        <tr>
```

signup.html:

signup.html

templates > signup.html > ...

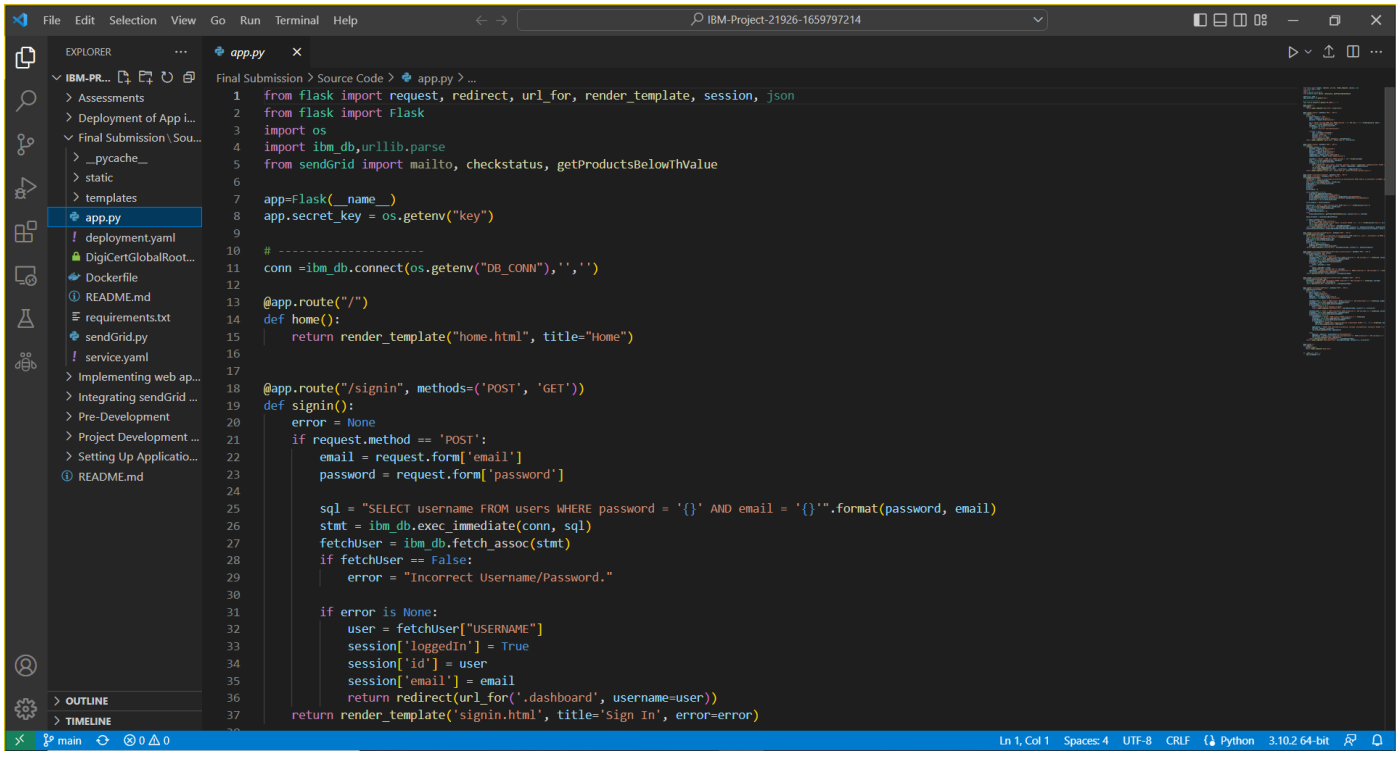
```
<!doctype html>
<html lang="en">

<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <title>IMSFR</title>
  <!-- <link rel="canonical" href="https://getbootstrap.com/docs/5.2/examples/sign-in/"> -->
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css" rel="stylesheet"
    integrity="sha384-Zenh87qX5JnK2Jl0vWa8Ck2rdkQ2Bzep5IDxbcnCeuOxjzrPF/et3URy9Bv1WTRi" crossorigin="anonymous">
  <style>
    .bd-placeholder-img {
      font-size: 1.125rem;
      text-anchor: middle;
      -webkit-user-select: none;
      -moz-user-select: none;
      user-select: none;
    }

    @media (min-width: 768px) {
      .bd-placeholder-img-lg {
        font-size: 3.5rem;
      }
    }

    .b-example-divider {
      height: 3rem;
      background-color: rgba(0, 0, 0, .1);
      border: solid rgba(0, 0, 0, .15);
      border-width: 1px 0;
      box-shadow: inset 0 .5em 1.5em rgba(0, 0, 0, .1), inset 0 .125em .5em rgba(0, 0, 0, .15);
    }
```

signup.html

templates > signup.html > html > head

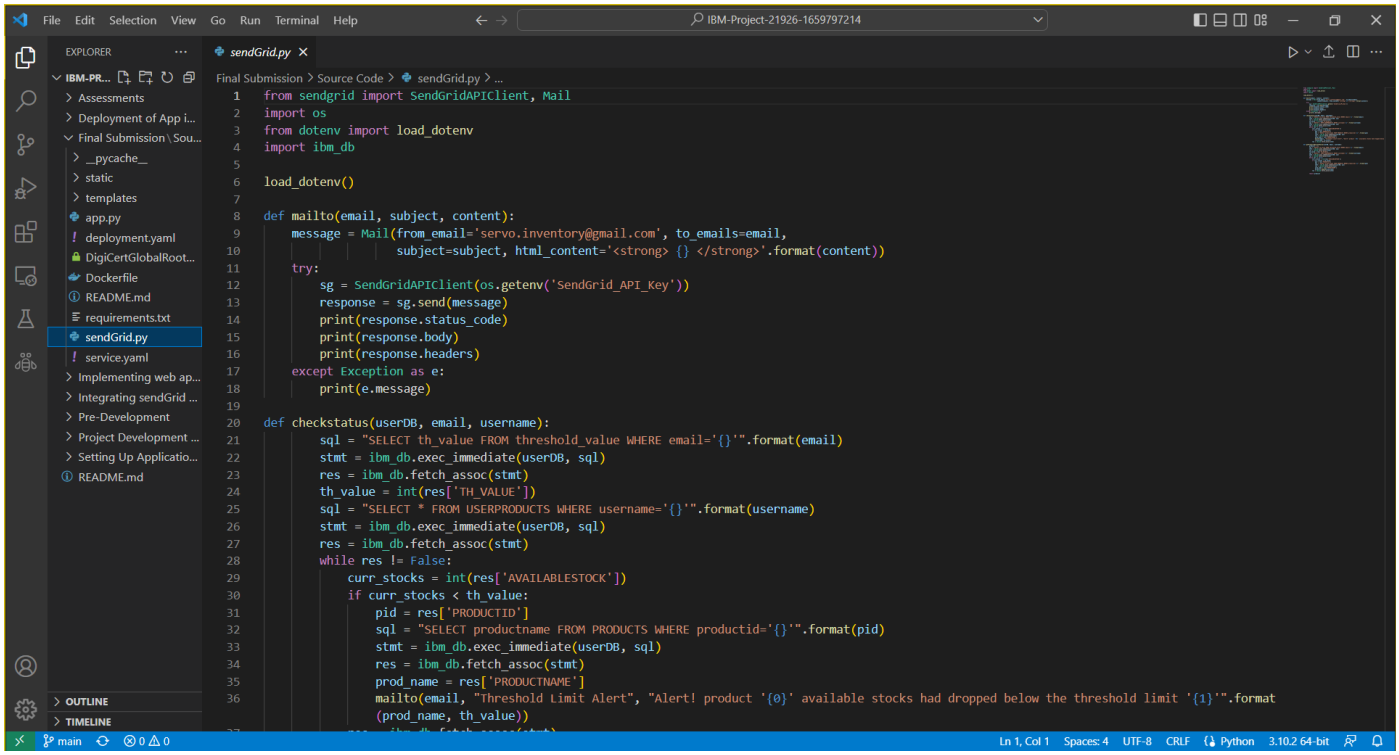
```
    .b-example-vr {
      flex-shrink: 0;
      width: 1.5rem;
      height: 100vh;
    }

    .bi {
      vertical-align: -.125em;
      fill: currentColor;
    }

    .nav-scroller {
      position: relative;
      z-index: 2;
      height: 2.75rem;
      overflow-y: hidden;
    }

    .nav-scroller .nav {
      display: flex;
      flex-wrap: nowrap;
      padding-bottom: 1rem;
      margin-top: -1px;
      overflow-x: auto;
      text-align: center;
      white-space: nowrap;
      -webkit-overflow-scrolling: touch;
    }
  </style>
  <link href="/static/css/signup.css" rel="stylesheet">
</head>
```

app.py:

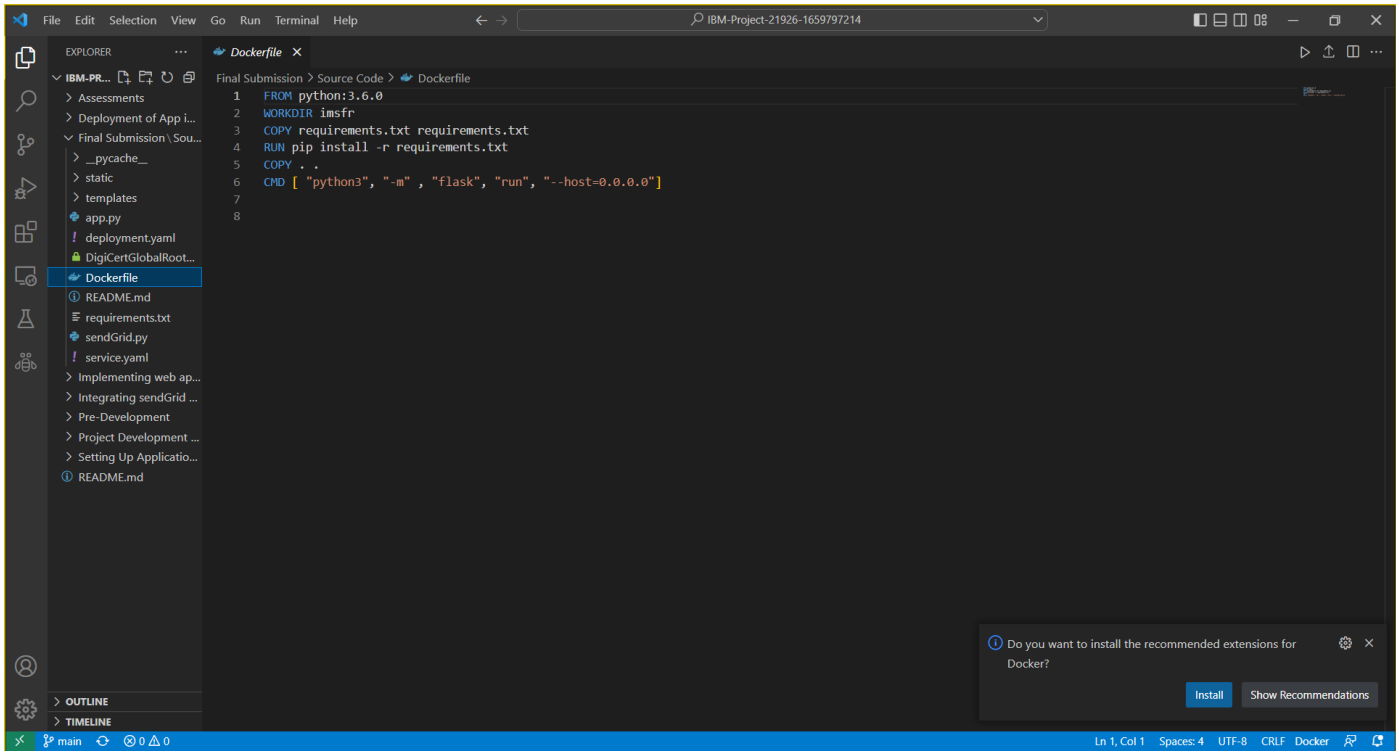



```
from flask import request, redirect, url_for, render_template, session, json
from flask import Flask
import os
import ibm_db,urllib.parse
from sendGrid import mailto, checkstatus, getProductsBelowThValue

app=Flask(__name__)
app.secret_key = os.getenv("key")

# ---------------------
conn =ibm_db.connect(os.getenv("DB_CONN"),'','')

@app.route("/")
def home():
    return render_template("home.html", title="Home")


@app.route("/signin", methods=('POST', 'GET'))
def signin():
    error = None
    if request.method == 'POST':
        email = request.form['email']
        password = request.form['password']

        sql = "SELECT username FROM users WHERE password = '{}' AND email = '{}'".format(password, email)
        stmt = ibm_db.exec_immediate(conn, sql)
        fetchUser = ibm_db.fetch_assoc(stmt)
        if fetchUser == False:
            error = "Incorrect Username/Password."

        if error is None:
            user = fetchUser["USERNAME"]
            session['loggedIn'] = True
            session['id'] = user
            session['email'] = email
            return redirect(url_for('.dashboard', username=user))
    return render_template('signin.html', title='Sign In', error=error)
```

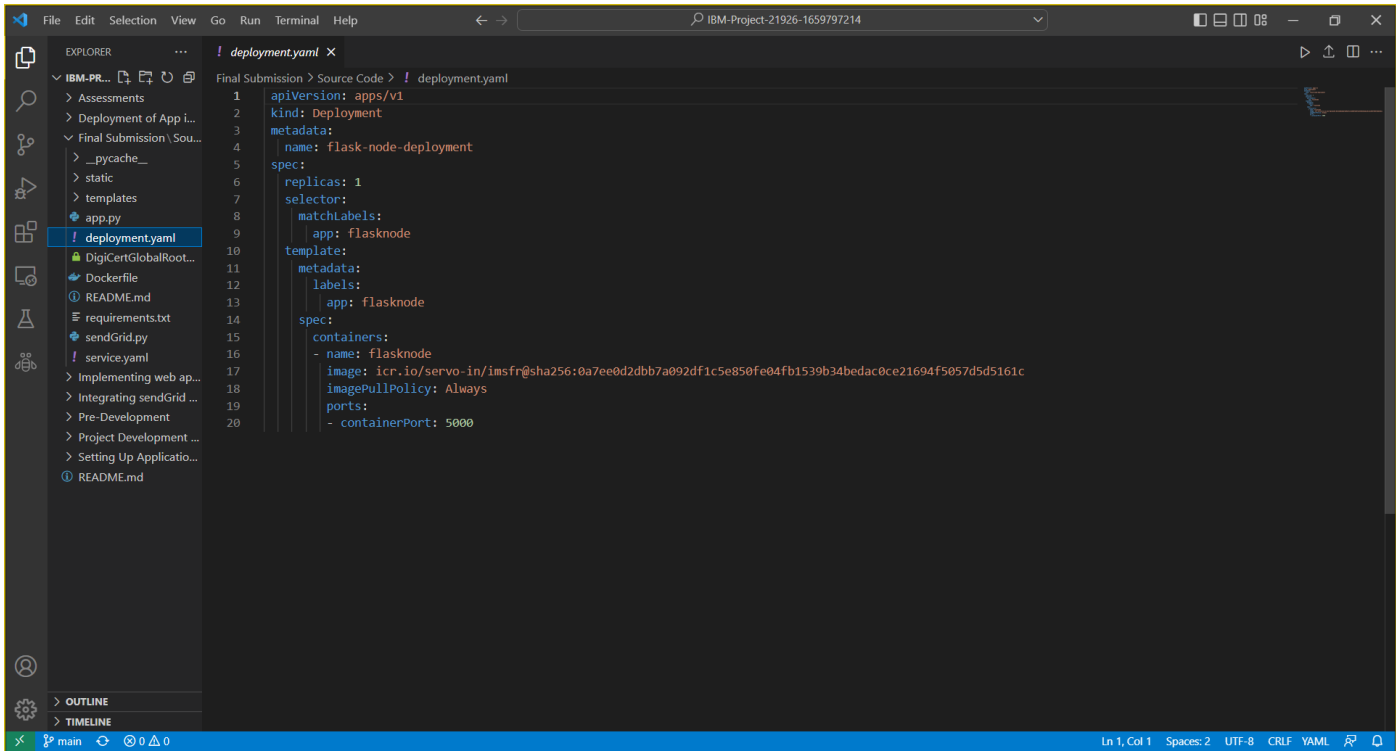
sendGrid.py:



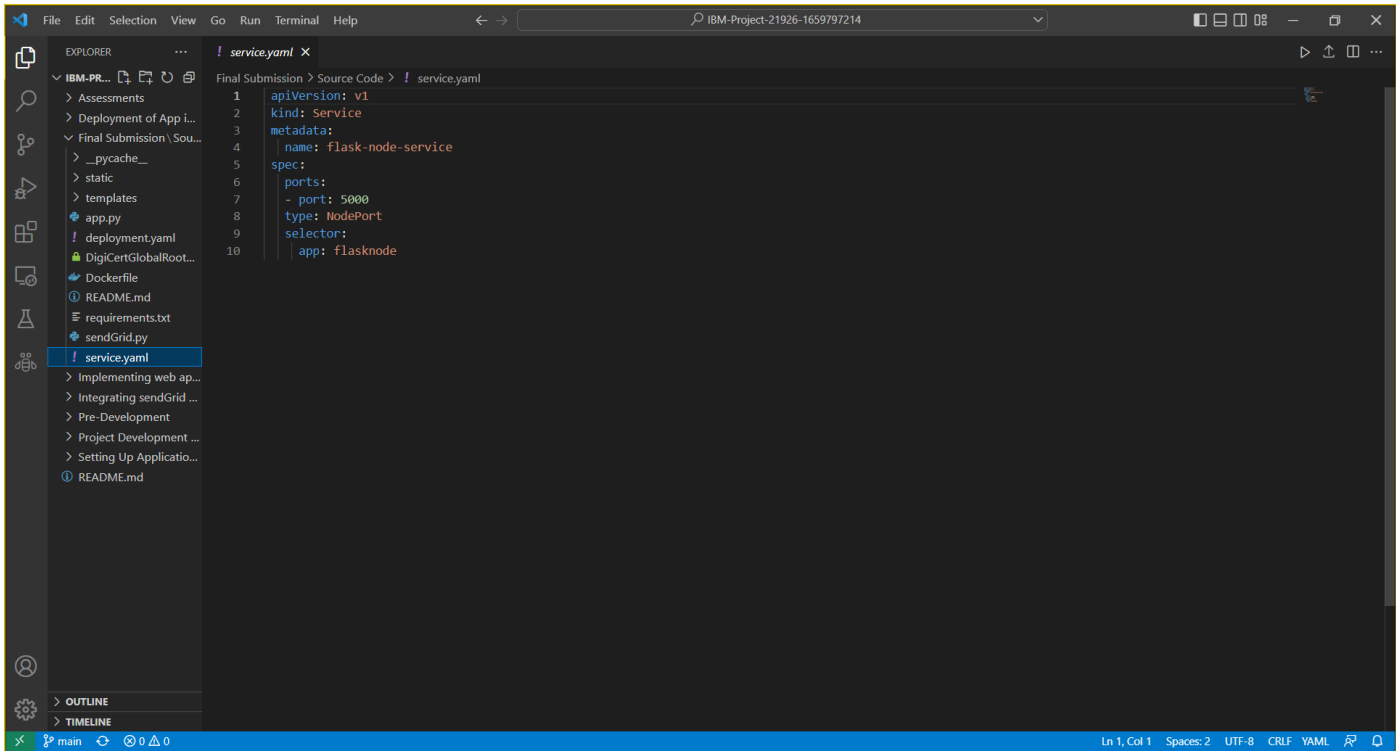
DockerFile:



deployment.yaml:



service.yaml:



Github link:

Source Code-Github

GitHub & Project Demo Link

Github Link:

Inventory Management System for Retailers - Team Id (PNT2022TMID03825)

Project Demo Link:

https://youtu.be/k1cI3D0U2_k