



INVENTORY MANAGEMENT FOR RETAIER CLOUD BASED APPLICATION



A MINI PROJECT REPORT

Submitted by

SOMASUNDARAM D	(AC19UEC119)
SURYA PRAKASAM	(AC19UEC130)
THIRUNAUKARSU	(AC19UEC134)
RANJITH S	(AC19UEC104)

*In partial fulfilment for the award of the
degree of*

BACHELOR OF ENGINEERING

in

**ELECTRONICS AND COMMUNICATION
ENGINEERING**

ADHIYAMAAN COLLEGE OF ENGINEERING (AUTONOMOUS)

Dr. M.G.R NAGAR, HOSUR- 635 130

ANNA UNIVERSITY: : CHENNAI- 600 025

NOVEMBER 2022

ANNA UNIVERSITY: CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this main project report “**PROJECT ON INVENTORY MANAGEMENT FOR RETAILERS**” is the Bonafede work of “**SOMASUNDARAM D(AC19UEC119), SURYA PRAKASAM B(AC19UEC130), THIRUNAUKARSU J(AC19UEC134), RANJITH S (AC19UEC104)**” who carried out the project under my supervision.

SIGNATURE

**Dr. S. SUMATHI, M.E., Ph.D.,
HEAD OF THE DEPARTMENT**

PROFESSOR,

Department of ECE,
Adhiyamaan College of Engineering,
(Autonomous) Dr.
M.G.R. Nagar,
Hosur – 635 130.

SIGNATURE

Ms.M.MAHARAJA

MENTOR

ASSISTANT PROFESSOR,

Department of ECE,
Adhiyamaan College of Engineering,
(Autonomous) Dr.
M.G.R. Nagar,
Hosur – 635 130.

Submitted for Main project VIVA-VOCE Examination held on _____ at
ADHIYAMAAN COLLEGE OF ENGINEERING (AUTONOMOUS), Hosur.

INTERNAL EXAMINER

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

First and foremost, we thank the almighty for granting us the wisdom, strength and grace to complete the report and for being with us in every step that we look in order to complete the project successfully.

We are grateful to our beloved Principal **Dr. G. RANGANATH, M.E., Ph.D.**, Principal, Adhiyamaan College of Engineering (Autonomous), Hosur, for providing the opportunity to do this work in the premises.

We acknowledge our heartfelt gratitude to **Dr.S.SUMATHI, ME., Ph.D.**, Professor and Head of the Department, Department of Electronics and communication Engineering, Adhiyamaan College of engineering (Autonomous), Hosur, for her guidance and valuable suggestions and encouragement throughout this project.

We are highly indebted to **Mrs.M.MAHARAJA, Assistant** Professor, Department of Electronics and communication Engineering, Adhiyamaan College of engineering (Autonomous), Hosur, whose immense support, encouragement and valuable guidance made us to complete this project successfully.

We also extend our thanks to Project Coordinator and all Staff Members for their support in completing the project successfully.

Finally, we would like to thank our parents, without their motivations and support would not have been possible for us to complete this project successful.

ABSTRACT

Retail inventory management is the process of ensuring you carry merchandise that shoppers want, with neither too little nor too much on hand. By managing inventory, retailers meet customer demand without running out of stock or carrying excess supply.

In practice, effective retail inventory management results in lower costs and a better understanding of sales patterns. Retail inventory management tools and methods give retailers more information on which to run their businesses. Applications have been developed to help retailers track and manage stocks related to their own products. The System will ask retailers to create their accounts by providing essential details. Retailers can access their accounts by logging into the application.

Once retailers successfully log in to the application they can update their inventory details, also users will be able to add new stock by submitting essential details related to the stock. They can view details of the current inventory. The System will automatically send an email alert to the retailers if there is no stock found in their accounts. So that they can order new stock

TITLE

1. INTRODUCTION

1.1 Project Overview

1.2 Purpose

2. LITERATURE SURVEY

2.1 Existing problem

2.2 References

2.3 Problem Statement Definition

3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas

3.2 Ideation & Brainstorming

3.3 Proposed Solution

3.4 Problem Solution fit

4. REQUIREMENT ANALYSIS

4.1 Functional requirement

4.2 Non-Functional requirements

5. PROJECT DESIGN

5.1 Data Flow Diagrams

5.2 Solution & Technical Architecture

5.3 User Stories

6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

6.2 Sprint Delivery Schedule

6.3 Reports from JIRA

7. CODING & SOLUTIONING

7.1 Feature 1

7.2 Feature 2

7.3 Database Schema (if Applicable)

8. TESTING

8.1 Test Cases

8.2 User Acceptance Testing

9. RESULTS

9.1 Performance Metrics

10.ADVANTAGES & DISADVANTAGES

11.CONCLUSION

12.FUTURE SCOPE

13.APPENDIX Source Code

13.1 GitHub & Project Demo Link

CHAPTER 1

1. INTRODUCTION

Retail inventory management is the process of ensuring you carry merchandise that shoppers want, with neither too little nor too much on hand. By managing inventory, retailers meet customer demand without running out of stock or carrying excess supply.

Inventory management is the process of tracking and managing inventory in a retail environment. It includes the tracking of inventory levels, orders, and sales. It also involves the management of stock levels, pricing, and promotions. Inventory management is a critical part of retail operations.

It helps retailers to keep track of their inventory, ensure that they have the right products in stock, and manage their stock levels.

It also helps retailers to optimize their stock levels and pricing. Inventory management is a complex process. It requires the use of multiple software applications and data sources.

Retailers need to have a clear understanding of their inventory levels and the products they sell.

They also need to be able to track and manage their inventory in real-time. The first step in inventory management is to track inventory levels.

This can be done manually or through the use of an inventory management system. Inventory management systems are designed to track and manage inventory in real-time.

They provide retailers with the ability to view their inventory levels, stock levels, and sales. They also allow retailers to manage their stock levels and pricing. Inventory management systems can be used to track inventory in a number of ways.

1.1 Project Overview

Inventory management is the process of tracking and managing inventory in a retail environment. It includes the tracking of inventory levels, orders, and sales. It also involves the management of stock levels, pricing, and promotions. Inventory management is a critical part of retail operations. It helps retailers to keep track of their inventory, ensure that they have the right products in stock, and manage their stock levels.

1.2 Purpose

They provide retailers with the ability to view their inventory levels, stock levels, and sales. They also allow retailers to manage their stock levels and pricing. Inventory management systems can be used to track inventory in a number of ways.

It also helps retailers to optimize their stock levels and pricing. Inventory management is a complex process. It requires the use of multiple software applications and data sources.

CHAPTER 2

2. LITERATURE SURVEY

A literature review is a comprehensive summary of previous research on a topic. The literature review surveys scholarly articles, books, and other sources relevant to a particular area of research. The review should enumerate, describe, summarize, objectively evaluate and clarify this previous research. It should give a theoretical base for the research and help you (the author) determine the nature of your research. The literature review acknowledges the work of previous researchers, and in so doing, assures the reader that your work has been well conceived. It is assumed that by mentioning a previous work in the field of study, that the author has read, evaluated, and assimilated that work into the work at hand.

2.1 EXISTING PROBLEM

They introduce Agent technology into domestic storage management and uses the autonomy, reactivity and sociality of Agent to realize the seamless connection among enterprises by defining interaction and cooperation mechanisms among different Agents. This paper mainly designs a storage management system model based on multi-Agent and describes main Agent cooperation processes of the system.

In the design of storage management system model based on multi-Agent in this paper, we use a hierarchical federation multi-Agent system organization structure and the cooperation among Agents is based on improved contract net protocol, which enhances system performance on the whole. Next, we will analyse from Agent performance and system processing efficiency. The autonomy of Agent in the model is mainly manifested.

2.2 REFERENCE

1. Aditya A. Pande, Sabahudin, “Study of Material Management Techniques on Construction project”, International Journal of Informative & Futuristic Research, ISSN: 2347-1697, Vol.2 (3), May 2015, pp.34793486.
2. Smangele Raphaelle, Gomathy Nathan and Chitra, “Inventory Management. A Case Study”, International Journal of Emerging Research in Management & Technology, ISSN: 2278-9359, Vol.3 (3) June 2014, pp.94-102.
3. Ashwini Patil, Smite V. Pat Askar, “Analysing Material Management Techniques on Construction Project”, International Journal of Engineering and Innovative Technology (IJEIT), Vol.3 (4), Jan 2013, pp.96-100.
4. "Integrations and Apps for Online Inventory Management. Software Trade Gecko". www.tradegecko.com. Retrieved 2015-11-24.

2.3 PROBLEM STATEMENT DEFINITION

Retail inventory management is the process of ensuring carry merchandise that shoppers want, with neither too little nor too much on hand. By managing inventory, retailers meet customer demand without running out of stock or carrying excess supply. Retail inventory management results in lower costs and a better understanding of sales patterns. Retail inventory management tools and methods give retailers more information with which to run their businesses.

CHAPTER 3

3. IDEATION & PROPOSED SOLUTION

Ideation is the process of forming ideas from conception to implementation, most often in a business setting. Ideation is expressed via graphical, written, or verbal methods, and arises from past or present knowledge, influences, opinions, experiences, and personal convictions.

PROPOSED SOLUTION

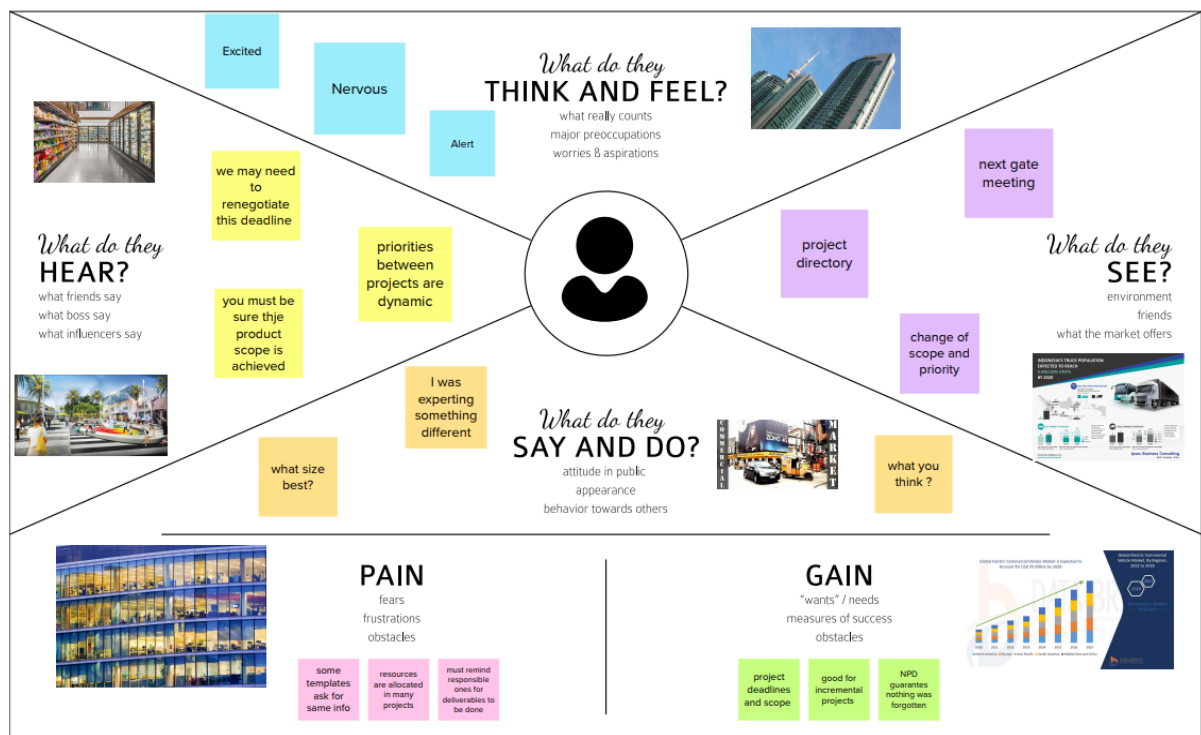
Effective retail inventory management results in lower costs and a better understanding of sales patterns. Retail inventory management tools and methods give retailers more information on which to run their businesses.

The application they can update their inventory details, also users will be able to add new stock by submitting essential details related to the stock. They can view details of the current inventory. The System will automatically send an email alert to the retailers if there is no stock found in their accounts. So that they can order new stock.

- Retailers track and manage stocks.
- Updating the inventory details.
- Adding new stock by submitting essential details.
- System will automatically send an email alert to the retailers if there is no stock found in their accounts.
- Can order new stock.

3.1 EMPATHY MAP CANVAS

An empathy map is a collaborative tool teams can use to gain a deeper insight into their customers. Much like a user persona, an empathy map can represent a group of users, such as a customer segment. The empathy map was originally created by Dave Gray and has gained much popularity within the agile community.



what do they think and feel:

- Alert
- Nervous
- Hands on stock maintain
- Exited to Implement
- Analysis the purchase
- Customer Satisfaction

what do they See:

- Next gate meeting
- Project directory
- Change of scope and priority

what do they Say and Do:

- Whom to approach
- What size best?
- I was expecting something different
- What you think?

Pains and Gains:

- Some templates ask for same info
- Resources are allocated in main project
- Must remind responsible ones for deliverables to be done
- Analyse the sales
- Project deadlines and scope
- Good for incremental project
- NPD guarantees nothing was forgotten
- Hike of Income
- Growth of retail shop

what do they Here:


- We may need to renegotiate this deadline
- Priorities between projects are dynamic
- You must be sure that product scope is achieved

3.1 IDEATION AND BRAINSTROMING

Ideation is often closely related to the practice of brainstorming, a specific technique that is utilized to generate new ideas. A principal difference between ideation and brainstorming is that ideation is commonly more thought of as being an individual pursuit, while brainstorming is almost always a group activity.

Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem solving. Prioritizing volume over value, out-of-the-box ideas are welcome and built upon, and all participants are encouraged to collaborate, helping each other develop a rich number of creative solutions.

Step-1: Team Gathering, Collaboration and Select the Problem Statement



Brainstorm & idea prioritization

Executing a brainstorm isn't unique; holding a productive brainstorm is. Great brainstorms are ones that set the stage for fresh and generative thinking through simple guidelines and an open and collaborative environment. Use this when you're just kicking-off a new project and want to hit the ground running with big ideas that will move your team forward.

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

[Share template feedback](#)

Before you collaborate

A little bit of preparation goes a long way with this session. Here's what you need to do to get going.

10 minutes

- A Team gathering**
Create 5 HMW statements before the activity to propose them to the team.
- B Set the goal**
Set a clear goal for the session. This could be to generate a list of ideas, to solve a specific problem, or to explore a new concept. Make sure the goal is achievable and measurable.
- C Learn how to use the facilitation tools**
Use the Facilitation Superpowers to run a happy and productive session.

[Open article](#)

1 Define your problem statement

Retail inventory management is the process of ensuring you carry merchandise that shoppers want, with neither too little nor too much on hand. By encouraging inventory practices most customer demand without running out of stock or carrying excess supply. In practice, effective retail inventory management results in lower costs and a better understanding of sales patterns. Retail inventory management tools and methods give retailers more information on which to run their businesses.

5 minutes

PROBLEM

How might we [your problem statement]?

Key rules of brainstorming

To run a smooth and productive session

- Stay in topic.
- Encourage wild ideas.
- Defer judgement.
- Listen to others.
- Go for volume.
- If possible, be visual.

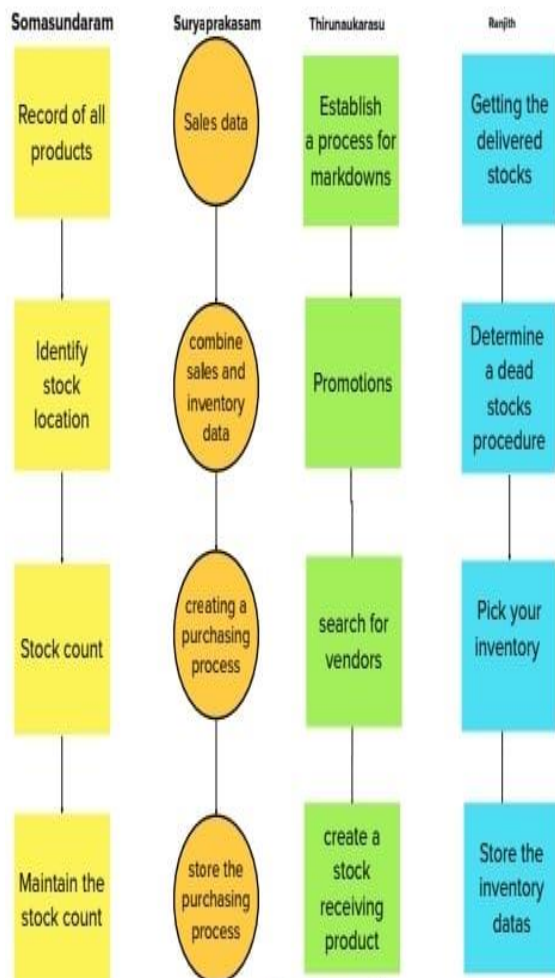
Step-2: Brainstorm, Idea Listing and Grouping

2

Brainstorm solo

Have each participant begin in the "solo brainstorm space" by silently brainstorming ideas and placing them into the template. This "silent-storming" avoids group-think and creates an inclusive environment for introverts and extroverts alike. Set a time limit. Encourage people to go for quantity.

10 minutes



INVENTORY
MANAGEMENT FOR
RETAILERS

3

Brainstorm as a group

Have everyone move their ideas into the "group sharing space" within the template and have the team silently read through them. As a team, sort and group them by thematic topics or similarities. Discuss and answer any questions that arise. Encourage "Yes, and..." and build on the ideas of other people along the way.

15 minutes

TIP

You can use the Voting session tool above to focus on the strongest ideas.



Step-3:Idea Prioritization

4

Decide your focus

Give each person two icons to vote which idea should your team focus on.

🕒 5 minutes

Person 1



Person 1



Person 3



Person 4



idea should be focus



After you collaborate

A brainstorm like this typically results in a handful of promising ideas that you can carry forward and act upon.

Quick add-ons



Cluster related ideas

Look for patterns or similarities in the standout ideas. Could any be combined together to form a stronger concept? Cluster similar ideas and label each cluster with a theme.



Vote on the most promising ideas

Narrow your focus to only the strongest few ideas by holding a **Voting Session**. Give each person 2 votes

Keep moving forward



2x2 Prioritization matrix

Build shared understanding and make collective decisions for moving ideas forward.

[Open the template →](#)



Storyboarding

Show existing and/or future consumer experiences through the act of sketching.

[Open the template →](#)



Pre-mortem

Harness the collective experience and wisdom of the team, before the project even starts.

[Open the template →](#)

[Share template feedback](#)

3.3 PROPOSED SOLUTION

Applications have been developed to help retailers track and manage stocks related to their own products. The System will ask retailers to create their accounts by providing essential details. Retailers can access their accounts by logging into the application. Once retailers successfully log in to the application they can update their inventory details, also users will be able to add new stock by submitting essential details related to the stock. They can view details of the current inventory. The System will automatically send an email alert to the retailers if there is no stock found in their accounts. So that they can order new stock.

S.no	Parameter	Description
1	Problem statement (problem to be solved)	By managing inventory, retailers meet customer demand without running out of stock or carrying excess supply.
2	Idea / Solution description	We developed a business model application that provides the inventory details. We built it with a graphical user interface. The applications have been developed to solve the retailer's problem. The application will track the sales pattern of the products in the shop. Managing and updating live stocks, as well as counting the cycle and measurement of products, Users will also be able to add new stocks by submitting essential details related to the stock. If the product is sold out of 75% of its stock, the system will automatically send an email and SMS alert to the user to update the stock, and the retailer can forward the mail to whole sellers to place the order.
3	Novelty / Uniqueness	Our application is developed with many features. They are <ul style="list-style-type: none">• Tracking the sales pattern.• Managing and updating live stocks.• Cycle counting of products.

		<ul style="list-style-type: none"> • Measurement of products.
4	Social Impact / Customer Satisfaction	Tracking the sales pattern and managing and updating live stocks, counting the cycle and measurement of products. It accelerates retail sales and profits. The retailer can focus on business growth.
5	Business Model (financial Benefit)	<ul style="list-style-type: none"> • Improve the accuracy of inventory management. • You can save both time and money. • It improves warehouse organisation. • It improves customer retention and engagement. • This ensures more profitability.
6	Scalability of Solution	Analysing inventory details and tracking a sales pattern. Stock management based on customer demand leads to retailer and customer satisfaction.

3.4 PROBLEM SOLUTION FIT

Problem-Solution canvas is a tool for entrepreneurs, marketers and corporate innovators, which helps them identify solutions with higher chances for solution adoption, reduce time spent on solution testing and get a better overview of current situation.

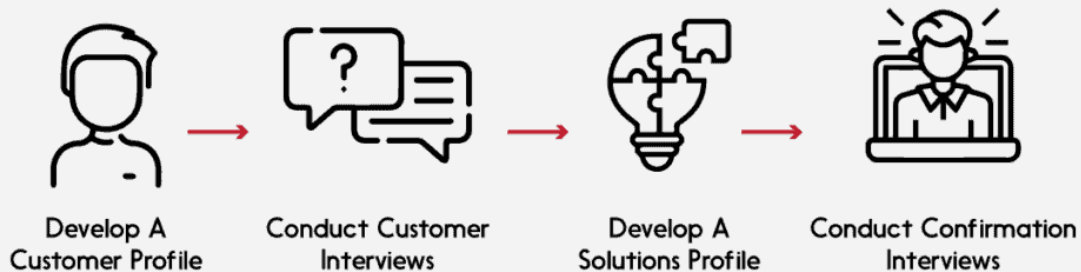
Problem-Solution fit canvas 2.0

Purpose / Vision : Inventory Management for Retails

Define CS, fit into	1. CUSTOMER SEGMENTS <p>Management of inventory process makes them to work efficient and time efficient , and also they know their options to be selected while ordering their stocks and products in time without running out stocks</p>	6. CUSTOMER <p>Management of inventory can be monitor Through only smart device and all the data are stored in the multiple device</p>	5.AVAILABLE SOLUTIONS <p>using smart mobility is the concept of connecting the each and every products in the inventory system .</p>	Explore AS,
Focus on J&P, tap into BE, understand	2. JOBS-TO-BE-DONE / PROBLEMS <ol style="list-style-type: none"> 1. Products with neither too little nor too much in hand 2. Waste in expiring products 3. Without stock losing of customers 4. Focus on customers product purchase details 	9. PROBLEM ROOT CAUSE <ol style="list-style-type: none"> 1. The uncontrolled orders with out proper stock data and sales patten. 2. Not monitoring on expiring goods and products on daily bases . 	7.BEHAVIOUR <p>waiting for stocks becomes very tension and may be it takes very long time to deliver the products on time so the sales rate become decrease due delay and also the products do not get sell will get expiry</p>	Focus on J&P, tap into BE, understand
Identify strong TR & EM	3. TRIGGERS <ol style="list-style-type: none"> 1. one work at a time 2. Lack in concentration 3. Lack in time management 4. EMOTIONS: BEFORE/AFTER <ol style="list-style-type: none"> 1. Stress 2. Anxiety 3. Physical illness 	10.YOUR SOLUTION <p>Applications have been developed to help retailers track and manage stocks related to their own products . The System will ask retailers to create their accounts by providing essential details. Retailers can access their accounts by logging into the application . Once retailers successfully log in to the application . Once retailers successfully log in to the application they can update their inventory details, also users will be able to add new stock by submitting essential details related to the stock . They can view details of the current inventory. The System will automatically send an email alert to the retailers when stock reaches to the certain limit in their accounts. So that they can order new stock.</p>	8.CHANNELS of BEHAVIOUR <ol style="list-style-type: none"> 1.ONLINE <p>The products are monitored regularly by fixing time .</p> 2.OFFLINE <p>Orders are placed on based the notification through the mail .</p> 	Extract online & offline CHOT BE



How To Achieve Problem-Solution Fit?



4.1 FUNCTIONAL REQUIREMENT

Requirements analysis, also called requirements engineering, is the process of determining user expectations for a new or modified product. These features, called requirements, must be quantifiable, relevant and detailed. In software engineering, such requirements are often called functional specifications.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form Registration through Gmail Registration through LinkedIn
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	Processing the Transactions	Online payment
FR-4	Authentication	Sent through Email
FR-5	Reporting	Through App (or) Through Email

4.2. NON-FUNCTIONAL REQUIREMENT

Nonfunctional Requirements (NFRs) define system attributes such as security, reliability, performance, maintainability, scalability, and usability. They serve as constraints or restrictions on the design of the system across the different backlogs.

FR No.	Non-Functional Requirement	Description
NFR-1	Usability	Tracking the inventory, sales and stock of products.
NFR-2	Security	Granting the permission for only authenticated users to access the portal.
NFR-3	Reliability	Updating process is fails enables to retrieve the stocks.
NFR-4	Performance	Quick access of the product through application.
NFR-5	Availability	Arrival of new updates does not impact the application or product details.
NFR-6	Scalability	Supports multiple users to access at a time without interference.

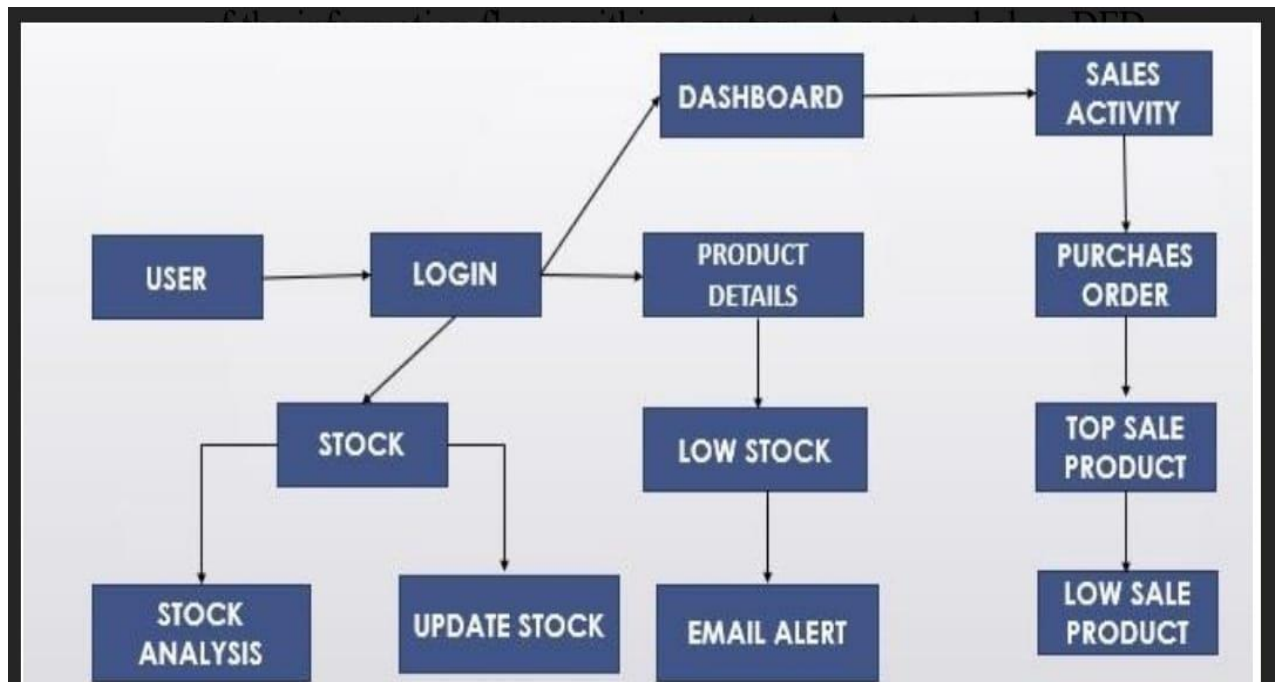
5. PROJECT DESIGN

Project design is an early phase of the project lifecycle where ideas, processes, resources, and deliverables are planned out. A project design comes before a project plan as it's a broad overview whereas a project plan includes more detailed information.

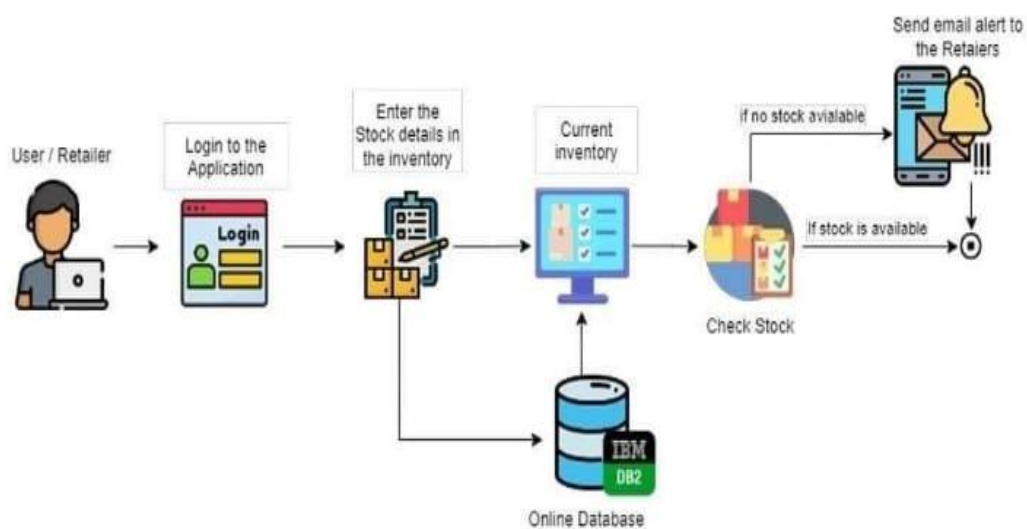
5.1. DATA FLOW DIAGRAMS

A data flow diagram (DFD) is a graphical or visual representation using a standardized set of symbols and notations to describe a business's operations through data movement. They are often elements of a formal methodology such as Structured Systems Analysis and Design Method (SSADM).

DATA FLOW DIAGRAMS:



5.2 SOLUTION ARCHITECTURE



5.3 USER STORIES

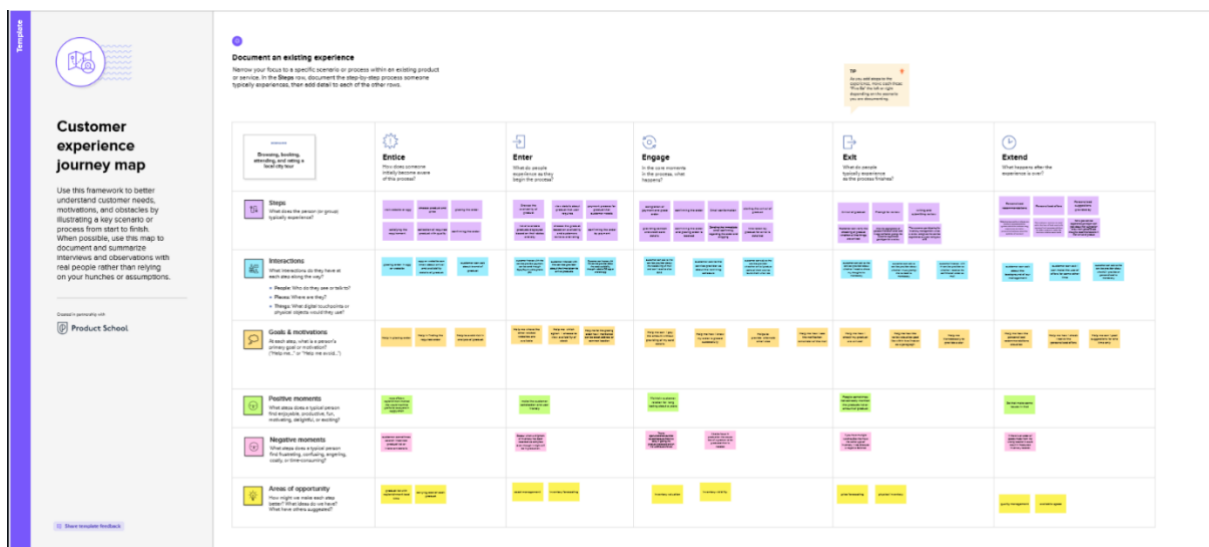
A user story is an informal, general explanation of a software feature written from the perspective of the end user or customer. The purpose of a user story is to articulate how a piece of work will deliver a particular value back to the customer.

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		Medium	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering email and password		High	Sprint-1
User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release

Customer (Web user)	Login	USN-6	As a user, I can login into application by entering my email and password.	Able to access my account / dashboard	High	Sprint-1
Customer Care Executive		USN-7	It can be used, easily access and responsible	I can access by easily Through application	High	Sprint-1
Administrator		USN-8	As an Administrator I can Update the application	I can fix the bug which Arises for the SFcustomers and users of the application	Medium	Sprint-1

5.4 CUSTOMER JOURNEY MAP

A customer journey map is a visual storyline of every engagement a customer has with a service, brand, or product. The creation of a journey map puts the organization directly in the mind of the consumer, so they can see and understand their customer's processes, needs, and perceptions.



6. PROJECT PLANNING & SCHEDULING

The process of planning primarily deals with selecting the appropriate policies and procedures in order to achieve the objectives of the project. Scheduling converts the project action plans for scope, time cost and quality into an operating timetable.

6.1. SPRINT PLANNING AND ESTIMATION

In Scrum Projects, Estimation is done by the entire team during Sprint Planning Meeting. The objective of the Estimation would be to consider the User Stories for the Sprint by Priority and by the Ability of the team to deliver during the Time Box of the Sprint.

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	Somasundaram D Thirunaukarsu J
Sprint-2	Connecting user data to web application	USN-2	As a user, I will receive confirmation email once I have registered for the application	1	High	Somasundaram D Ranjith s

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-3	Connecting web application to object storage	USN-3	As a user, I can register for the application through Facebook	2	Low	Somasundaram D Thirunakarsu J
Sprint-4	Integrating all the technologies in application	USN-4	As a user, I can register for the application through Gmail	2	Medium	Somasundaram D Ranjith s
Testing phase	Analysis of risk	USN-5	As a user, I can log into the application by entering email & password	1	High	Somasundaram D Surya parakasam B
	Debugging		Resolving the error		High	Somasundaram D Ranjith s
	Testing		Testing the application		High	Somasundaram D Thirunakarsu J

6.2 SPRINT DELIVERY SCHEDULE

The deliverables of a sprint aren't as predictable as they are for other projects. Sprint participants have produced sketches and drawings, writing, photographs, comic strips, videos and fully coded working prototypes. The answer is whatever's right to answer the problem.

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	5 Nov 2022
Sprint-3	20	6 Days	07Nov 2022	12 Nov 2022	20	12 Nov 2022

Sprint-4	20	6 Days	14Nov 2022	19 Nov 2022	20	19 Nov 2022
----------	----	--------	------------	-------------	----	-------------

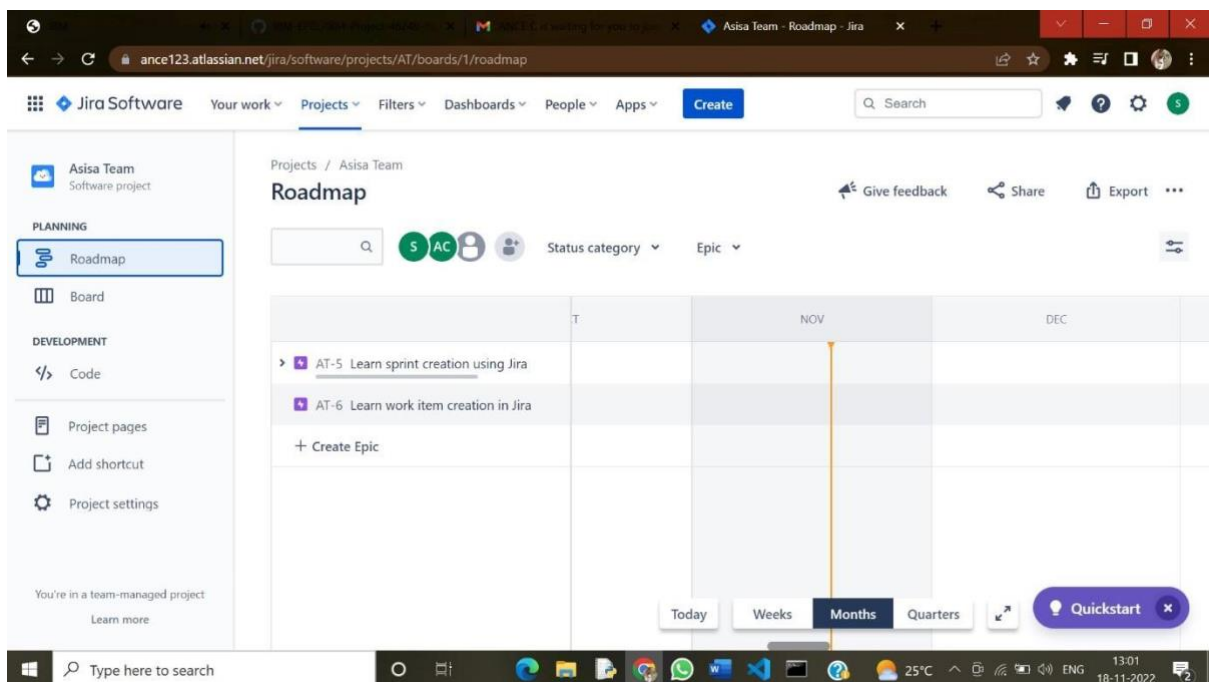
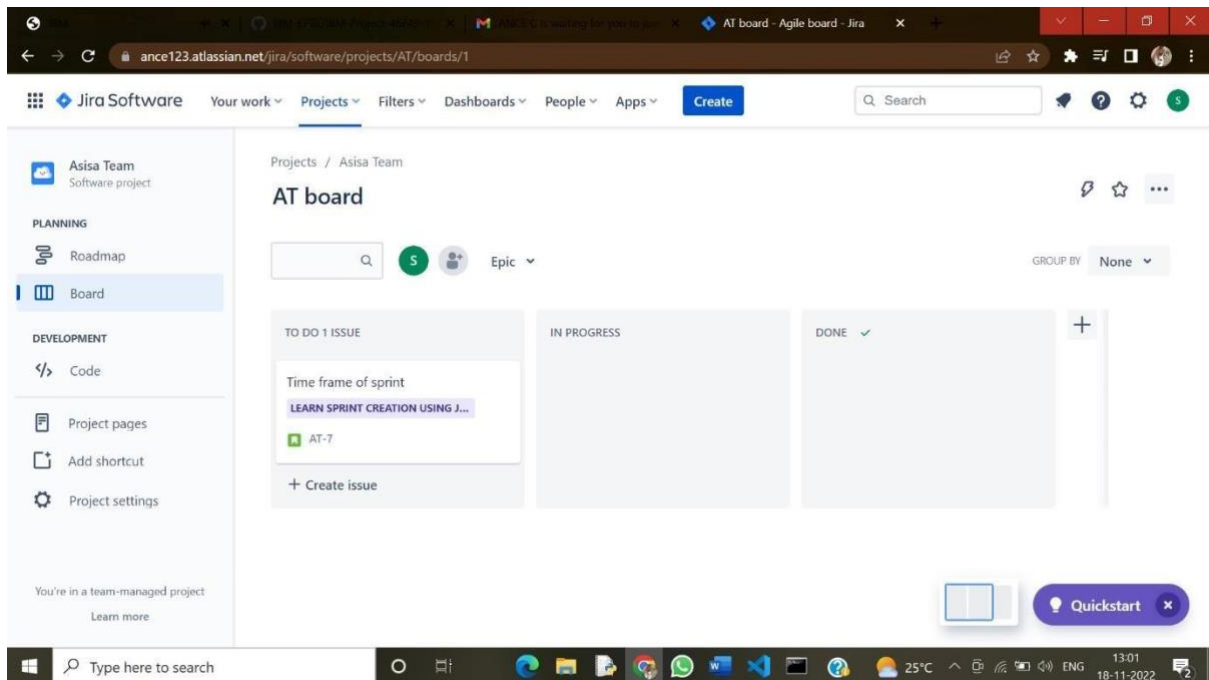
6.3 MILESTONE AND ACTIVITY LIST

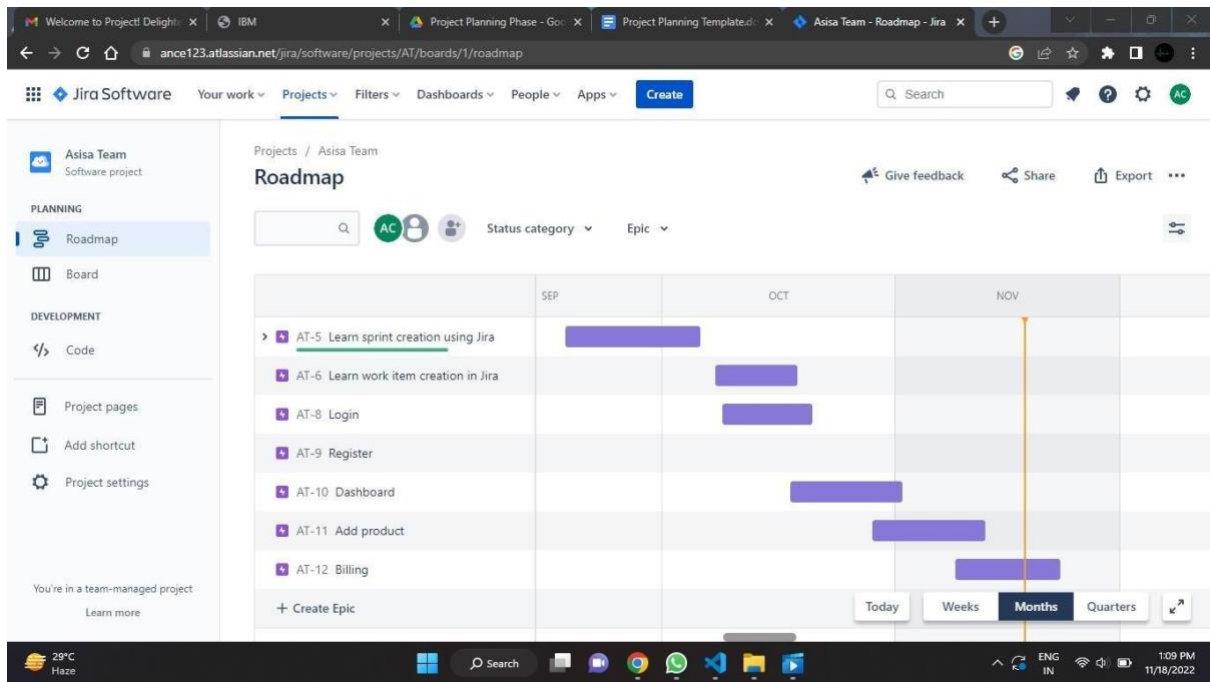
A milestone list is a project management document that identifies all project milestones. A milestone is a significant event or a point in a project. It represents nothing more than a moment in time; hence, when scheduling, milestones should be assigned zero duration.

TITLE	DESCRIPTION	DATE
Literature survey & information gathering	Collect the relevant use cases and refer to existing solutions	19 SEPTEMBER 2022
Prepare empathy map	Prepare Empathy Map canvas and list of problem statements	19 SEPTEMBER 2022
Ideation	List the ideas by organizing the brainstorming session and prioritize the top 3 ideas based on the feasibility & importance	19 SEPTEMBER 2022
Problem solution fit	Prepare problem - solution fit document & solution architecture	07 OCTOBER 2022
Proposed Solution	Preparing the new idea for our problem statement	07 OCTOBER 2022

Customer journey	Prepare the customer journey maps to understand the user interactions & experiences with the application	07 NOVEMBER 2022
Solution requirement	Prepare the Functional Requirement Document	07 NOVEMBER 2022
Data flow diagrams	Prepare the Data Flow Diagrams	07 NOVEMBER 2022
Technology architecture	Prepare Technology Architecture of the solution	07 NOVEMBER 2022
Prepare Milestone & activity list	Prepare the Milestone & activity list of the project	08 NOVEMBER 2022
Sprint Delivery Plan	Prepare the plan for all the sprints in the project	08 NOVEMBER 2022
Project development – delivery of sprint – 1,2,3 & 4	Develop & submit the developed code by testing it	18 NOVEMBER 2022

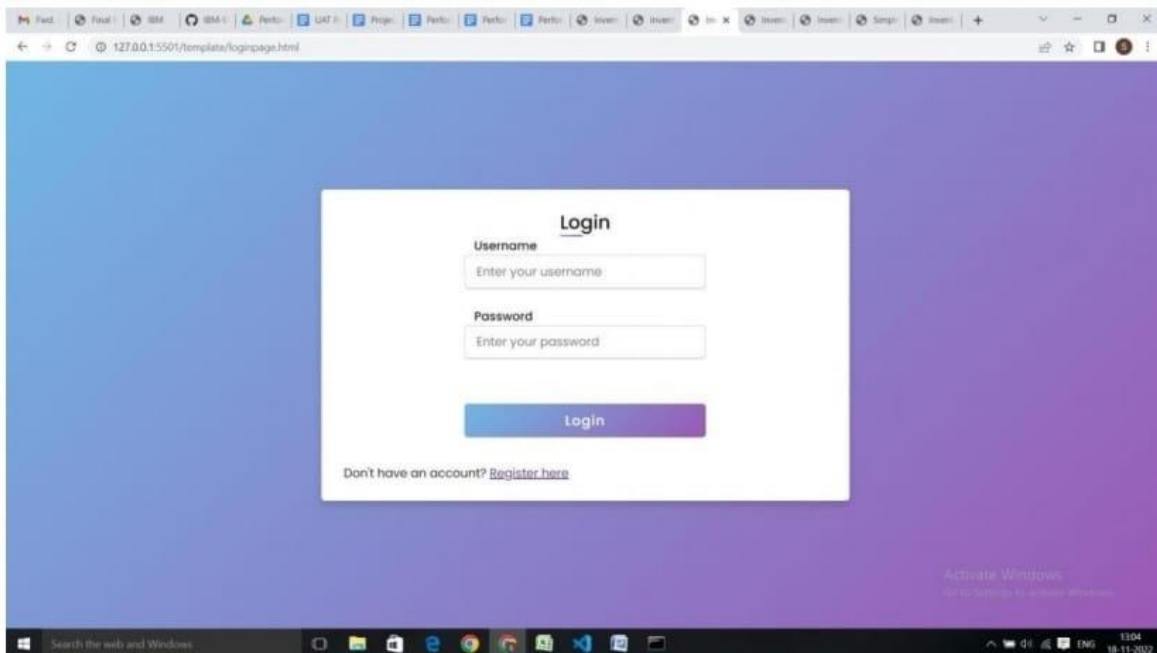
6.2. REPORT FROM JIRA





7. CODING & SOLUTIONING

HOME PAGE



```
@import
url('https://fonts.googleapis.com/css2?family=Poppins:wght@200;300;400;500;600;700&dis
play=swap');

*{

margin: 0;

padding: 0;

box-sizing: border-box;

font-family: 'Poppins',sans-serif;

}


.container{

display: flex;

justify-content: center;

align-items: center;

padding:25px;

min-height:calc( 100vh - 88px) ;

line-height: 2rem;

}


.container .title{

font-size: 25px;

font-weight: 500;

position: relative;

text-align:center;

}
```



```
.container form{  
  padding:20px;  
  width:700px;  
  background: #fff;  
  box-shadow: 0 5px 10px rgba(0,0,0,.1);  
}
```

```
form .input-box span.details{  
  display: block;  
  font-weight: 500;  
  margin-bottom: 5px;  
  
}
```

```
.user-details .input-box input{  
  height: 45px;  
  width: 100%;  
  outline: none;  
  font-size: 16px;  
  border-radius: 5px;  
  padding-left: 15px;  
  border: 1px solid #ccc;  
  border-bottom-width: 2px;  
  transition: all 0.3s ease;  
  
}
```

```
.user-details .input-box input:focus,  
.user-details .input-box input:valid{  
    border-color: #f2f546;  
}
```

```
form .category{  
    display: flex;  
    width: 80%;  
    margin: 14px 0 ;  
    justify-content: space-between;  
}
```

```
form .category label{  
    display: flex;  
    align-items: center;  
    cursor: pointer;  
}
```

```
form .category label .dot{  
    height: 18px;  
    width: 18px;  
    border-radius: 50%;  
    margin-right: 10px;  
    background: #d9d9d9;  
    border: 5px solid transparent;  
    transition: all 0.3s ease;  
}
```

```
#dot-1:checked ~ .category label .one,
#dot-2:checked ~ .category label .two,
#dot-3:checked ~ .category label .three{
    background: #d9e94c;
    border-color: #d9d9d9;
}

form input[type="radio"]{
    display: none;
}

form .button{
    height: 55px;
    margin: 35px 0;
}

form .button input{
    height: 100%;
    width: 50%;
    border-radius: 5px;
    border: none;
    color: #fff;
    font-size: 18px;
    font-weight: 500;
    letter-spacing: 1px;
    cursor: pointer;
    transition: all 0.3s ease;
    background: linear-gradient(135deg, #71b7e6, #9b59b6);
```

```

    margin-left: 25%;
}

form .button input:hover{
    /* transform: scale(0.99); */
    background: linear-gradient(-135deg, #71b7e6, #fff23d);
}

@media(max-width: 584px){
    .container{
        max-width: 100%;
    }
    form .user-details .input-box{
        margin-bottom: 15px;
        width: 100%;
    }
    form .category{
        width: 100%;
    }
    .content form .user-details{
        max-height: 300px;
        overflow-y: scroll;
    }
    .user-details::-webkit-scrollbar{
        width: 5px;
    }

```

```

}

@media(max-width: 459px){

.container .content .category{

    flex-direction: column;

}

}

```

7.1. FEARTURE 1

REGISTER PAGE

```
<!DOCTYPE html>
```

```
<html lang="en" dir="ltr">
```

```
<head>
```

```

<meta charset="UTF-8">

<title> Inventory Managment System - Register </title>

<link rel="stylesheet" href="{{ url_for('static',filename='reg.css') }}">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

</head>

<body>

<div class="container">

    <div class="mx-4 mt-2 text-danger" style="color: red;">{{ msg }}</div>

    <div class="title">Registration</div>

    <div class="content">

        <form method="POST" action="">

            <div class="user-details">

                <div class="input-box">

                    <span class="details">Full Name</span>

                    <input type="text" placeholder="Enter your name" name="full" required>

                </div>

                <div class="input-box">

                    <span class="details">Username</span>

                    <input type="text" placeholder="Enter your username" name="user" required>

                </div>

                <div class="input-box">

                    <span class="details">Email</span>

                    <input type="text" placeholder="Enter your email" name="email" required>

                </div>

                <div class="input-box">

```

```

    <span class="details">Phone Number</span>

    <input type="text" placeholder="Enter your number" name="phone" required >

</div>

<div class="input-box">

    <span class="details">Password</span>

    <input type="text" placeholder="Enter your password" name="password" required>

</div>

<div class="input-box">

    <span class="details">Confirm Password</span>

    <input type="text" placeholder="Confirm your password" name="confirm" required>

</div>

</div>

<div class="button">

    <input type="submit" value="Register">

</div>

<p>have an account? <a href="{{ url_for('home') }}">login</a></p>

</form>

</div>

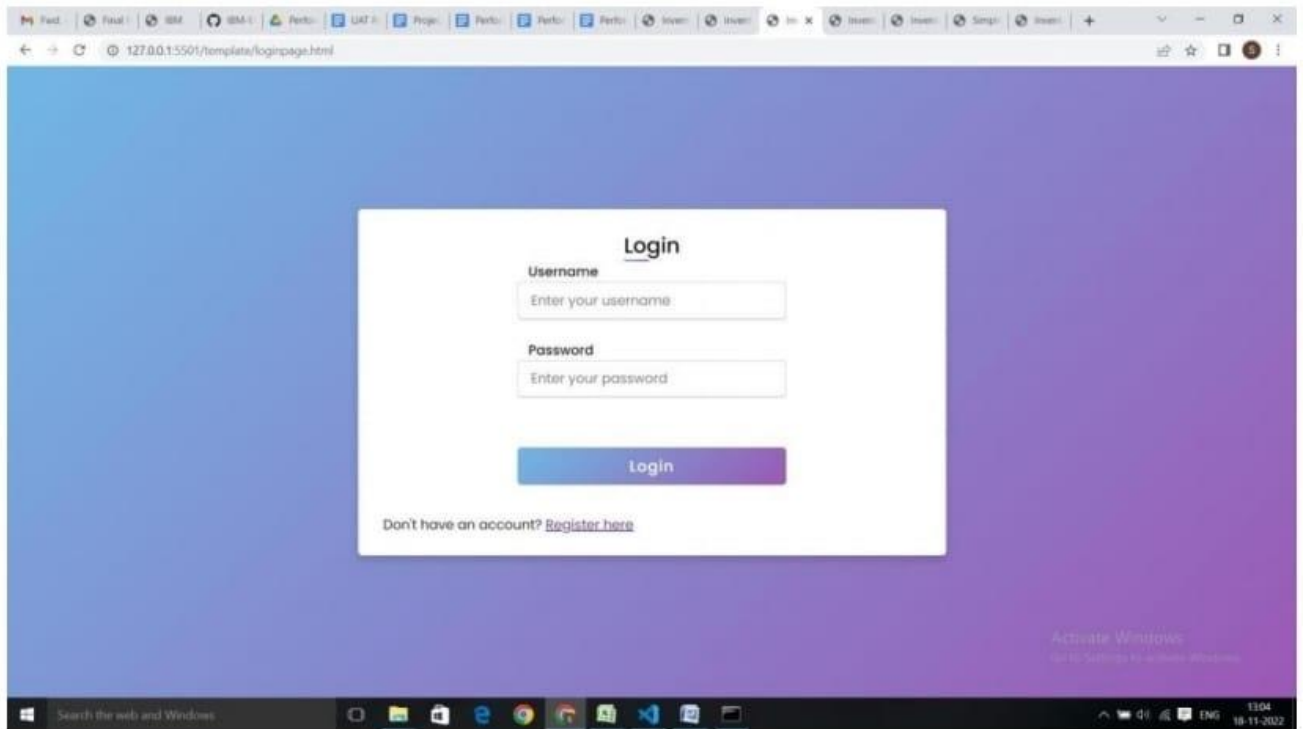
</div>

</body>

</html>

```

LOGIN PAGE



```
<!DOCTYPE html>
```

```
<html lang="en" dir="ltr">
```

```
<head>
```

```
  <meta charset="UTF-8">
```

```
  <title> Inventory Managment System - Login</title>
```

```
  <link rel="stylesheet" href="{{ url_for('static',filename='reg.css') }}">
```

```
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
```

```
</head>
```

```
<body>
```

```
  <div class="container">
```

```
    <div class="title">Login</div>
```

```
    <div class="content">
```



```
<div class="mx-4 mt-2 text-danger" style="color: red;">{{ msg }}</div>
```

```
<form method="POST" action="">
```

```
<div class="login">
```

```
<div class="input-box">
```

```
<span class="detailslog">Username</span>
```

```
<input type="text" placeholder="Enter your username" name="user" required>
```

```
</div>
```

```
<br>
```

```
<div class="input-box">
```

```
<span class="detailslog">Password</span>
```

```
<input type="text" placeholder="Enter your password" name="password"
required>
```

```
</div>
```

```
<br>
```

```
<div class="button1">
```

```
<input type="submit" value="Login">
```

```
</div>
```

```
</div>
```

<p>Don't have an account? Register here</p>

</form>

</div>

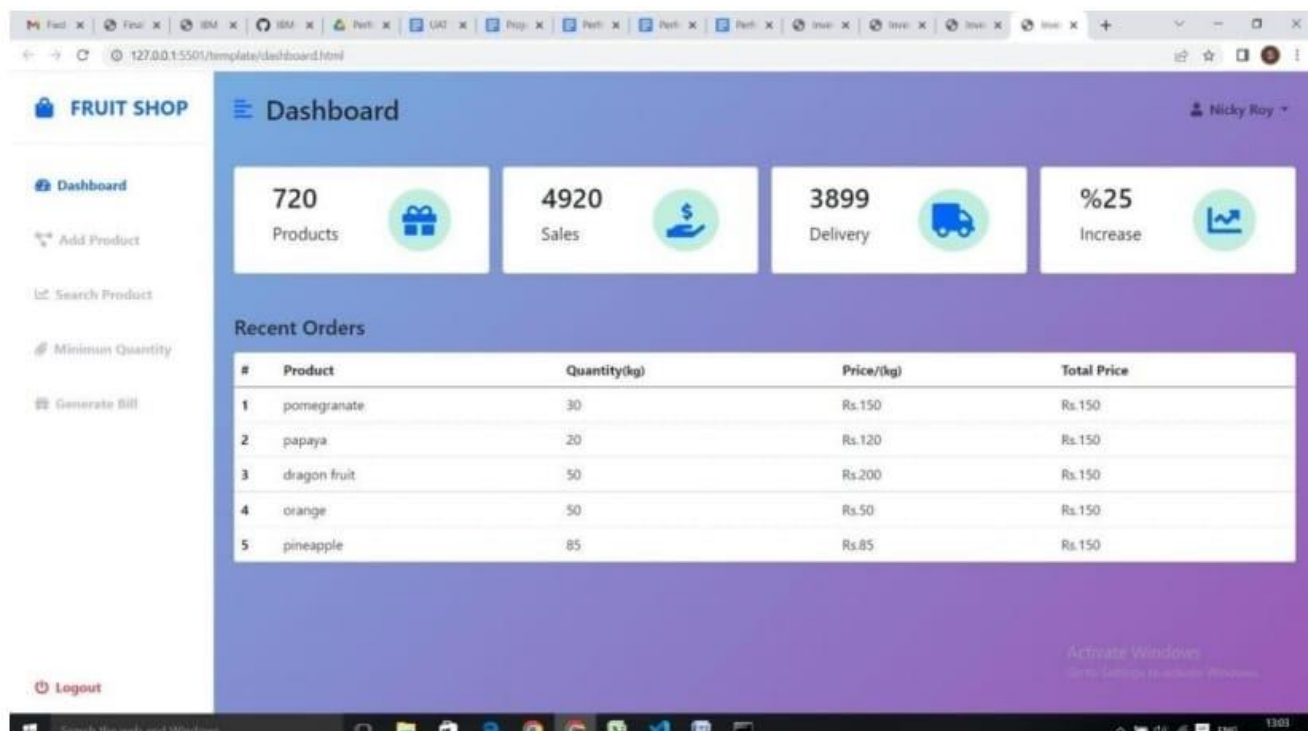
</div>

</body>

</html>

7.2. FEARTURE 2

DASHBOARD



<!DOCTYPE html>

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

```
<head>
```

```
<meta charset="UTF-8" />
```

```
<title> Inventory Managment System - Dashboard </title>
```

```
<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@5.0.0-beta3/dist/css/bootstrap.min.css"
rel="stylesheet" />
```

```
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/5.15.3/css/all.min.css" />
```

```
<link rel="stylesheet" href="{{ url_for('static',filename='dash.css') }}" />
```

```
</head>
```

```
<body>
```

```
<div class="d-flex" id="wrapper">
```

```
<!-- Sidebar -->
```

```
<div class="bg-white" id="sidebar-wrapper">
```

```
<div class="sidebar-heading text-center py-4 primary-text fs-4 fw-bold text-uppercase
border-bottom"><i
```

```
class="fas fa-shopping-bag"></i> &nbsp; Fruit Shop</div>
```

```
<div class="list-group list-group-flush my-3">
```

```
<div>
```

```

        <a href="{{ url_for('dashboard') }}" class="list-group-item list-group-item-action
bg-transparent second-text active"><i

            class="fas fa-tachometer-alt me-2"></i>Dashboard</a>

        <a href="{{ url_for('addproduct') }}"

            class="list-group-item list-group-item-action bg-transparent second-text fw-
bold"><i

                class="fas fa-project-diagram me-2"></i>Add Product</a>

            <a href="{{ url_for('searchproduct') }}" class="list-group-item list-group-item-action
bg-transparent second-text fw-bold"><i

                class="fas fa-chart-line me-2"></i>Search Product</a>

            <a href="{{ url_for('minimum') }}" class="list-group-item list-group-item-action bg-
transparent second-text fw-bold"><i

                class="fas fa-paperclip me-2"></i>Minimun Quantity</a>

        <a href="{{ url_for('billing') }}"

            class="list-group-item list-group-item-action bg-transparent second-text fw-
bold"><i

                class="fas fa-gift me-2"></i>Generate Bill</a>

            <a href="{{ url_for('viewbill') }}"

                class="list-group-item list-group-item-action bg-transparent second-text fw-
bold"><i

                    class="fas fa-gift me-2"></i>View Bill</a>

        </div>

        <div>

            <a href="{{ url_for('register') }}"

                class="list-group-item list-group-item-action bg-transparent text-danger fw-bold
"><i

                    class="fas fa-power-off me-2"></i>Logout</a>

        </div>

```

```

    </div>

</div>

<!-- /#sidebar-wrapper -->

<!-- Page Content -->

<div id="page-content-wrapper">

    <nav class="navbar navbar-expand-lg navbar-light bg-transparent py-4 px-4">

        <div class="d-flex align-items-center">

            <i class="fas fa-align-left primary-text fs-4 me-3" id="menu-toggle"></i>

            <h2 class="fs-2 m-0">Dashboard</h2>

        </div>

        <button class="navbar-toggler" type="button" data-bs-toggle="collapse"

            data-bs-target="#navbarSupportedContent"                aria-
controls="navbarSupportedContent"

            aria-expanded="false" aria-label="Toggle navigation">

            <span class="navbar-toggler-icon"></span>

        </button>

        <div class="collapse navbar-collapse" id="navbarSupportedContent">

            <ul class="navbar-nav ms-auto mb-2 mb-lg-0">

                <li class="nav-item dropdown">

                    <a class="nav-link dropdown-toggle second-text fw-bold" href="#"

id="navbarDropdown"

                    role="button" data-bs-toggle="dropdown" aria-expanded="false">

                        <i class="fas fa-user me-2"></i>Nicky Roy

```

```

        </a>

        <ul class="dropdown-menu" aria-labelledby="navbarDropdown">

            <li><a          class="dropdown-item"          href="{{          url_for('home')
}}">Logout</a></li>

        </ul>

    </li>

</ul>

</div>

</nav>

<div class="container-fluid px-4">

    <div class="mx-4 mt-2 text-danger">{{ msg }}</div>

    <div class="row g-3 my-2">

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

            <div class="p-3 bg-white shadow-sm d-flex justify-content-around align-items-
center rounded">

                <div>

                    <h3 class="fs-2">{{ count }}</h3>

                    <p class="fs-5">Products</p>

                </div>

                <i class="fas fa-apple-alt fs-1 primary-text border rounded-full secondary-bg
p-3"></i>

            </div>

        </div>

    </div>

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

```

```
<div class="p-3 bg-white shadow-sm d-flex justify-content-around align-items-center rounded">
```

```
<div>
```

```
<h3 class="fs-2">{{ amount }}</h3>
```

```
<p class="fs-5">Sales Amount</p>
```

```
</div>
```

```
<i class="fas fa-hand-holding-usd fs-1 primary-text border rounded-full secondary-bg p-3"></i>
```

```
</div>
```

```
</div>
```

```
<div class="col-md-3">
```

```
<div class="p-3 bg-white shadow-sm d-flex justify-content-around align-items-center rounded">
```

```
<div>
```

```
<h3 class="fs-2">{{ low }}</h3>
```

```
<p class="fs-5">Minimum Quantity </p>
```

```
</div>
```

```
<i class="fab fa-microblog fs-1 primary-text border rounded-full secondary-bg p-3"></i>
```

```
</div>
```

```
</div>
```

```
<div class="col-md-3">
```

```
<div class="p-3 bg-white shadow-sm d-flex justify-content-around align-items-center rounded">
```

```
<div>
```

```

        <h3 class="fs-2">{{ bill_count }}</h3>

        <p class="fs-5">Bills</p>

    </div>

    <i class="fas fa-file-invoice fs-1 primary-text border rounded-full secondary-
bg p-3"></i>

    </div>

</div>

</div>

</div>

<div class="row my-5 card">

    <h3 class="fs-4 m-3 text-center">Products</h3>

    <div class="col">

        <table class="table bg-white rounded shadow-sm table-hover text-center ">

            <thead>

                <tr>

                    <th scope="col">Product</th>

                    <th scope="col">Quantity<small>(kg)</small></th>

                    <th scope="col">Price<small>/</small></th>

                </tr>

            </thead>

            <tbody>

                { % for i in datas % }

                <tr>

                    <td>{{ i['product'] }}</td>

```



```

        <td>{{ i['stock'] }}</td>

        <td>{{ i['price'] }}</td>

    </tr>

    {% endfor %}

</tbody>

</table>

</div>

</div>

</div>

</div>

</div>

<!-- /#page-content-wrapper -->


<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-
beta3/dist/js/bootstrap.bundle.min.js"></script>

<script>

    var el = document.getElementById("wrapper");

    var toggleButton = document.getElementById("menu-toggle");

    toggleButton.onclick = function () {

        el.classList.toggle("toggled");

    };

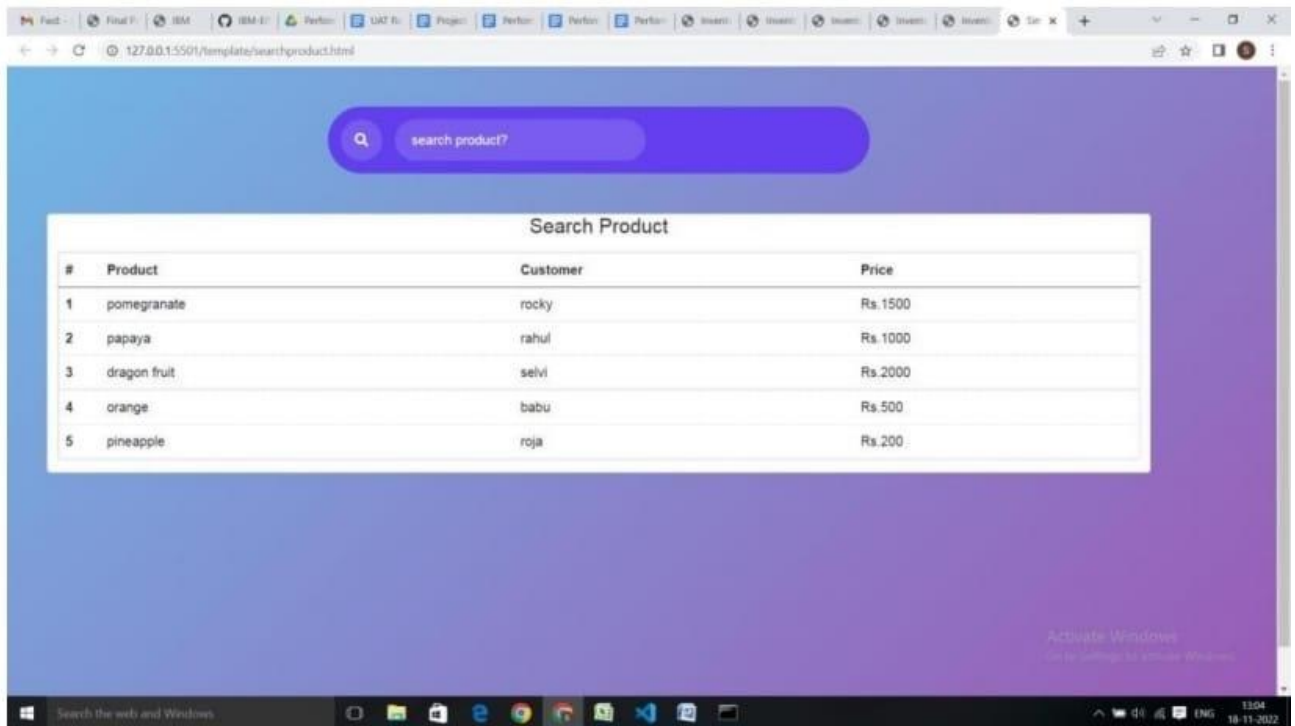
```

</script>

</body>

</html>

PRODUCT



<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title> Inventory Managment System - Search Product </title>

<link rel="stylesheet" href="{{ url_for('static',filename='search.css') }}">

<script src="https://kit.fontawesome.com/b99e675b6e.js"></script>

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-beta3/dist/css/bootstrap.min.css" rel="stylesheet" />

```
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.15.3/css/all.min.css" />
```

```
<link rel="stylesheet" href="{{ url_for('static',filename='dash.css') }}" />
```

```
</head>
```

```
<body>
```

```
<div class="d-flex" id="wrapper">
```

```
<!-- Sidebar -->
```

```
<div class="bg-white" id="sidebar-wrapper">
```

```
<div class="sidebar-heading text-center py-4 primary-text fs-4 fw-bold text-uppercase border-bottom"><i
```

```
class="fas fa-shopping-bag"></i> &nbsp; Fruit Shop</div>
```

```
<div class="list-group list-group-flush my-3">
```

```
<div>
```

```
<a href="{{ url_for('dashboard') }}" class="list-group-item list-group-item-action bg-transparent second-text "><i
```

```
class="fas fa-tachometer-alt me-2"></i>Dashboard</a>
```

```
<a href="{{ url_for('addproduct') }}"
```

```
class="list-group-item list-group-item-action bg-transparent second-text fw-bold"><i
```

```
class="fas fa-project-diagram me-2"></i>Add Product</a>
```

```
<a href="{{ url_for('searchproduct') }}" class="list-group-item list-group-item-action bg-transparent second-text fw-bold active"><i
```

```
class="fas fa-chart-line me-2"></i>Search Product</a>
```

```
<a href="{{ url_for('minimum') }}" class="list-group-item list-group-item-action bg-transparent second-text fw-bold"><i
```

```

        class="fas fa-paperclip me-2"></i>Minimun Quantity</a>
<a href="{ { url_for('billing') } }"
        class="list-group-item list-group-item-action bg-transparent second-text fw-
bold"><i
        class="fas fa-gift me-2"></i>Generate Bill</a>
<a href="{ { url_for('viewbill') } }"
        class="list-group-item list-group-item-action bg-transparent second-text fw-
bold"><i
        class="fas fa-gift me-2"></i>View Bill</a>
</div>
<div>
<a href="{ { url_for('home') } }"
        class="list-group-item list-group-item-action bg-transparent text-danger fw-
bold "><i
        class="fas fa-power-off me-2"></i>Logout</a>
</div>
</div>
<!-- /#sidebar-wrapper -->

<!-- Page Content -->
<div id="page-content-wrapper">
    <nav class="navbar navbar-expand-lg navbar-light bg-transparent py-4 px-4">
        <div class="d-flex align-items-center">
            <i class="fas fa-align-left primary-text fs-4 me-3" id="menu-toggle"></i>
            <h2 class="fs-2 m-0"></h2>
        </div>

```

```

        <button class="navbar-toggler" type="button" data-bs-toggle="collapse"
            data-bs-target="#navbarSupportedContent"
controls="navbarSupportedContent"
            aria-expanded="false" aria-label="Toggle navigation">

            <span class="navbar-toggler-icon"></span>

        </button>

        <div class="collapse navbar-collapse" id="navbarSupportedContent">

        </div>

    </nav>

    <div class="container mt-5">

        <form method="POST" action="">

            <div class="search_box w-50 m-auto">

                <button class="search_btn" type="submit"><i class="fas fa-
search"></i></button>

                <input type="text" class="input_search" name="search" placeholder="search
product?">

            </div>

        </form>

        <div class="row my-5 card">

```

```
<h3 class="mt-3 mb-5 text-center">Product</h3>
```

```
<table class="table bg-white rounded shadow-sm table-hover text-center"
cellspacing="0" width="100%">
```

```
<thead>
```

```
<tr>
```

```
<th scope="col">Product</th>
```

```
<th scope="col">Quantity<small> &nbsp;(kg)</small></th>
```

```
</tr>
```

```
</thead>
```

```
<tbody>
```

```
{% if product %}
```

```
<tr>
```

```
<td>{{ product['PRODUCT'] }}</td>
```

```
<td>{{ product['STOCK'] }}</td>
```

```
</tr>
```

```
{% else %}
```

```
<tr>
```

```
<td colspan='2'>no data</td>
```

```
</tr>
```

```
{% endif %}
```

```

        </tbody>

    </table>

</div>

</div>

</div>

</div>

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-
beta3/dist/js/bootstrap.bundle.min.js"></script>

<script>

    var el = document.getElementById("wrapper");

    var toggleButton = document.getElementById("menu-toggle");

    toggleButton.onclick = function () {

        el.classList.toggle("toggled");

    };

</script>

</body>

</html>

[11:50 am, 19/11/2022] Live Your Life: View billing.html
[11:50 am, 19/11/2022] Live Your Life: <!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">

<title> Inventory Managment System - View Bill </title>


<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-beta3/dist/css/bootstrap.min.css"
rel="stylesheet" />

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/5.15.3/css/all.min.css" />

<link rel="stylesheet" href="{{ url_for('static',filename='dash.css') }}" />

</head>

<body >


<div class="d-flex" id="wrapper">

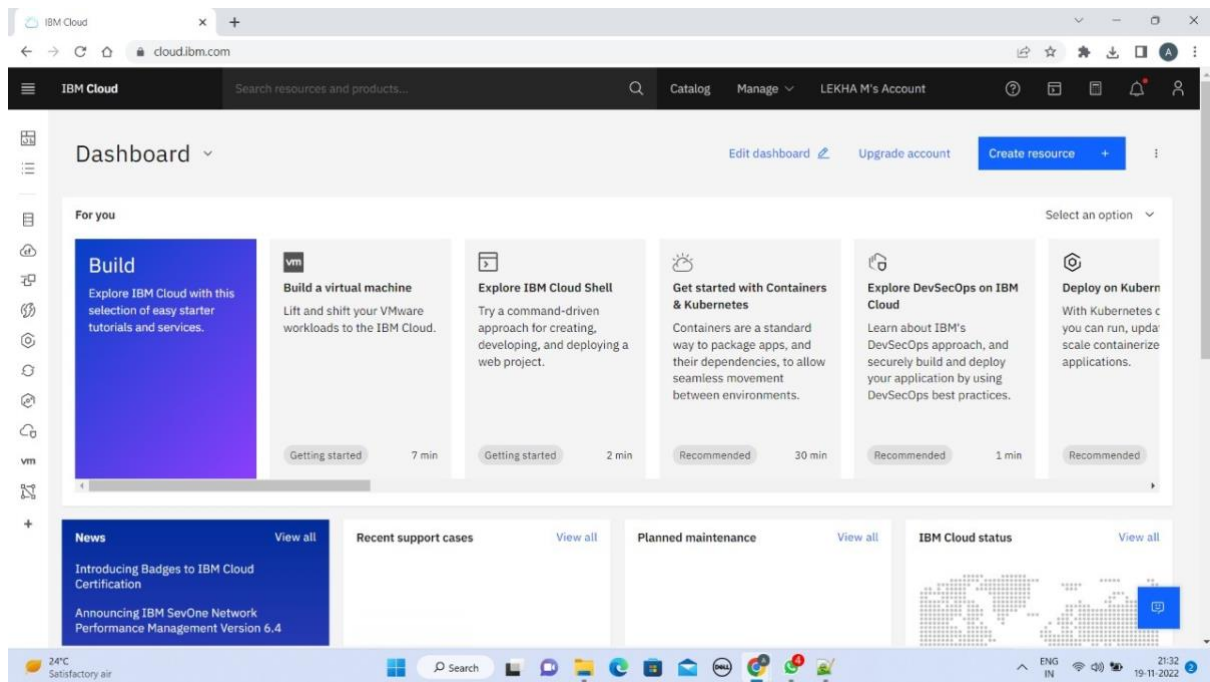
    <!-- Sidebar -->

    <div class="bg-white" id="sidebar-wrapper">

```

7.3 DATABASE SCHEMA

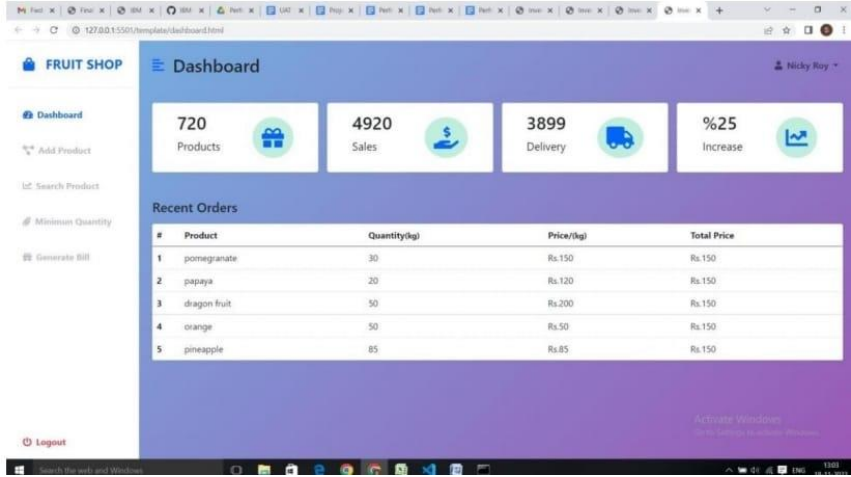
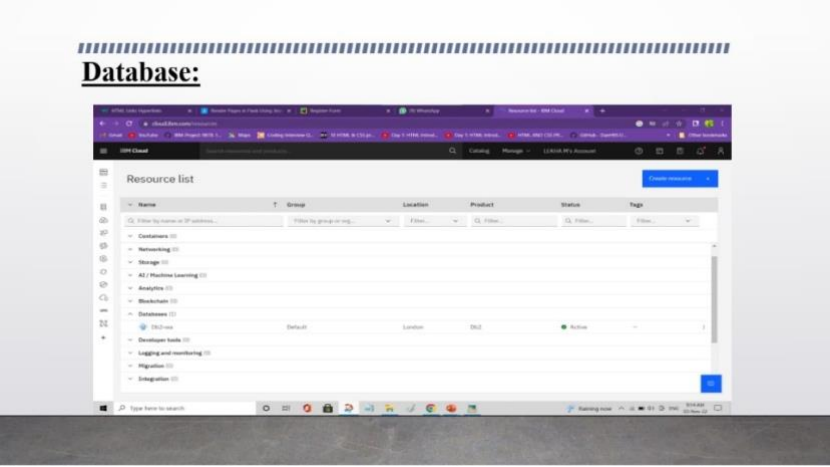
CLOUD ACCOUNT CREATION

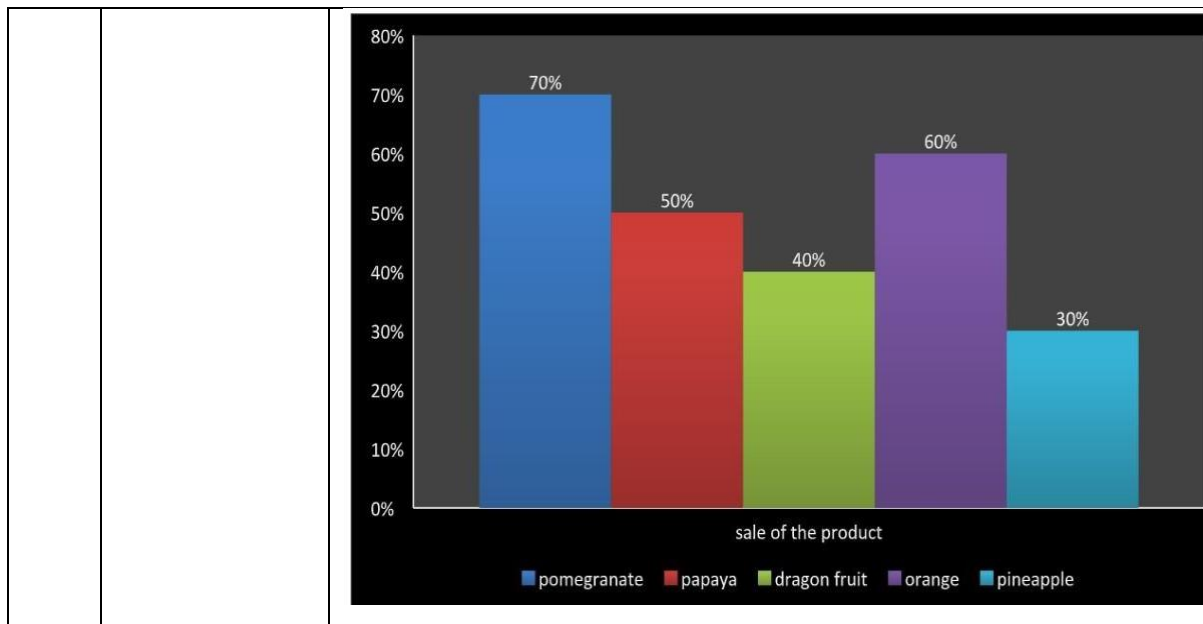


8.TESTING

In general, testing is finding out how well something works. In terms of human beings, testing tells what level of knowledge or skill has been acquired. In computer hardware and software development, testing is used at key checkpoints in the overall process to determine whether objectives are being met.

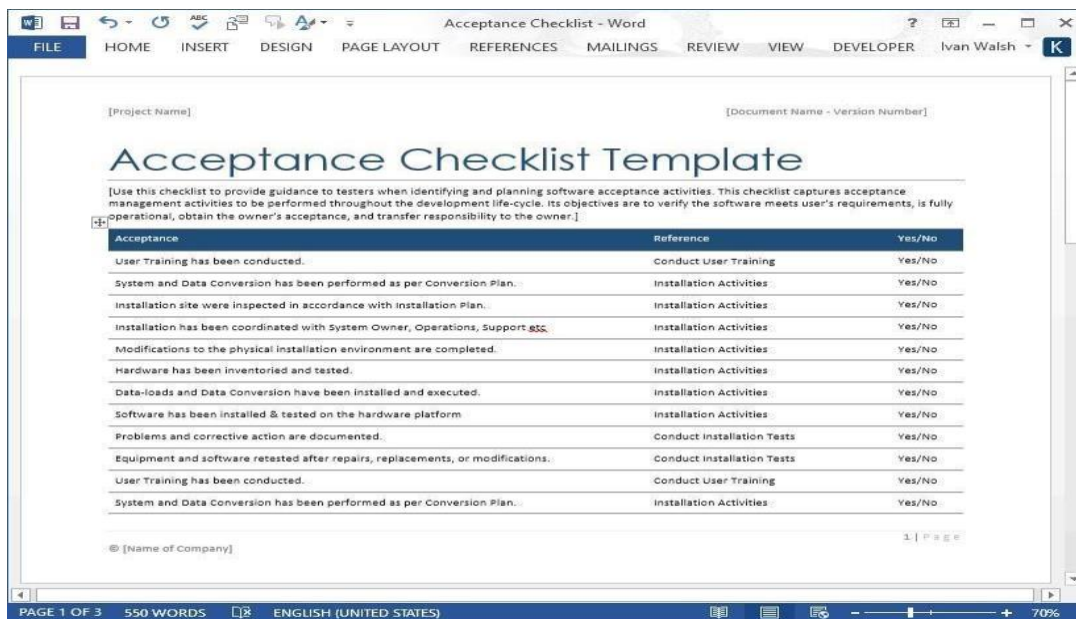
8.1. TEST CASE

S.No	Parameter	Screenshot / Values
1.	Dashboard design	
2.	Data Responsiveness	<ul style="list-style-type: none"> • Responds quickly to unpredictable demand. • Have a lower utilization rate. • Utilise an excess buffer capacity. <p>Invest in lead time reduction.</p>
3.	Amount Data to Rendered (DB2 Metrics)	7
4.	Utilization of Data Filters	
5.	Effective User Story	<ul style="list-style-type: none"> • Dashboard is used to accessing the other pages. • Add product page is to add the product they need. • Minimum quantity page is to view the present quantity and minimum quantity of the product they need. <p>Search product page is used to search the product quickly.</p>



8.2. USER ACCEPTANCE TESTING

Step	Procedures	Expected Result	Result
1	Insert admin, username, and password	Save the insert data into database	Success
2	Insert correct username, password for login	Verify the admin	Success
3	Click 'Register,' 'Login' button	Application redirect admin to Login page after register and Main page after login	Success
4	Repeat step 2 and 3 for login using false username, password	Application display error message	Success
5	Update Admin Account	New update data saved into database	Success
6	Log Out Account	Log out redirected to Login page	Success
Precondition		No credentials are currently login	
Post-condition		New and updated Admin name, username, and password saved in	



9.RESULT

9.1. PERFORMANCE TESTING

Inventory Performance is a measure of how effectively and efficiently inventory is used and replenished. The goal of inventory performance metrics is to compare actual on-hand dollars versus forecasted cost of goods sold.

- Weeks on Hand. ...
- Inventory Turnover Rate. ...
- Days on Hand. ...
- Stock to Sales Ratio. ...
- Sell-through Rate. ...
- Backorder Rate. ...
- Accuracy of Forecast Demand. ...
- Rate of Return.

10.ADVANTAGE AND DISADVANTAGE

Advantage:

- To maintain the right amount of stocks
- To a more organized warehouse
- It saves time and money

- Improves efficiency and productivity
- A well-structured inventory management system leads to improved customer retention:
- It leads It helps Avoid lawsuits and regulatory fines
- Schedule maintenance
- Reduction in holding costs
- Flexibility

Disadvantage:

- Bureaucracy
- Impersonal touch
- Production problem
- Increased space is need to hold the inventory
- Complexity

11.CONCLUSION

In conclusion as you can see the importance of inventory management is very serious, it is one of the most important aspects of any business. The aspect of this part of the business is whether or not you can satisfy the demand of your customers if you aren't sure if you have all the materials available to make the final product Without having the proper inventory management, they would not be able to supply their customers with their ordered ambulance. And this product is what their entire business is based on, so it is of great importance When they are choosing from the different types of programs or automated systems to help with keeping records accurate, needs to keep in mind that the customer is not concerned with which materials are needed to complete the finished product, but the product is operating as promised based on the contract. In addition, the plans for the maintenance of having proper inventory levels need to be in place and also adjusted when the company grows and as the business dictates implements the new suggestions, they will be on the right track to having a well-established business.

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

SOURCE CODE:

```
from flask import Flask, render_template, request, redirect, url_for, flash
app = Flask(__name__)
import ibm_db
from flask_login import login_user, current_user, logout_user,
login_required, LoginManager, UserMixin
import datetime
from sendgrid import SendGridAPIClient

from sendgrid.helpers.mail import Mail


dsn_hostname = "9938aec0-8105-433e-8bf9-0fbb7e483086.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud" # e.g.: "54a2f15b-5c0f-46df-8954-7e38e612c2bd.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud"
dsn_uid = "krh91100" # e.g. "abc12345"
dsn_pwd = "gSPyVtDpdim5wKGL" # e.g. "7dBZ3wWt9XN6$o0J"


dsn_driver = "{IBM DB2 ODBC DRIVER}"
dsn_database = "bludb" # e.g. "BLUDB"
dsn_port = "32459" # e.g. "32733"
dsn_protocol = "TCPIP" # i.e. "TCPIP"
dsn_security = "SSL" # i.e. "SSL"
```

```

dsn = (
    "DRIVER={0};"
    "DATABASE={1};"
    "HOSTNAME={2};"
    "PORT={3};"
    "PROTOCOL={4};"
    "UID={5};"
    "PWD={6};"
    "SECURITY={7};").format(dsn_driver,    dsn_database,    dsn_hostname,    dsn_port,
    dsn_protocol, dsn_uid, dsn_pwd,dsn_security)

```

```

print(dsn)

```

```

try:
    conn = ibm_db.connect(dsn, "krh91100","gSPyVtDpdim5wKGL" "9938aec0-8105-433e-
8bf9-0fbb7e483086.clogj3sd0tgtu0lqde00.databases.appdomain.cloud")
    print ("Connected to database: ", dsn_database, "as user: ", dsn_uid, "on host: ",
    dsn_hostname)

```

```

except:
    print ("Unable to connect: ", ibm_db.conn_errormsg() )

```

```

SECRET_KEY = 'Vibinprakash'

```

```

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

```

```

def home():

```

```

    if request.method == 'POST':

```

```

        name=request.form.get('user')
        password=request.form.get('password')
        print(name,password)
        sql = "SELECT * FROM users WHERE user_name =? AND password=?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt,1,name)
        ibm_db.bind_param(stmt,2,password)
        ibm_db.execute(stmt)
        account = ibm_db.fetch_assoc(stmt)
        print (account)
        if account:

```

```

            return redirect(url_for('dashboard'))

```

```

        else:
            msg='invalid user name and password'

```

```

        return render_template('loginpage.html',msg=msg)

    else:

        return render_template('loginpage.html')

@app.route("/register",methods=['GET', 'POST'])
def register():
    if request.method == 'POST':
        name=request.form.get('full')
        user_name=request.form.get('user')
        email=request.form.get('email')
        phone=request.form.get('phone')
        password=request.form.get('password')
        confirm=request.form.get('confirm')
        print(confirm)
        if password==confirm:

            sql ="SELECT id FROM users ORDER BY ID DESC limit 1"
            stm=ibm_db.exec_immediate(conn,sql)
            while ibm_db.fetch_row(stm) != False:
                count=ibm_db.result(stm,0)
                print(count)

            insert=f"insert into users values ({int(count)+1}, '{name}', '{user_name}', '{email}',
            '{password}', '{phone}')"
            table=ibm_db.exec_immediate(conn,insert)
            return redirect(url_for('home'))

        else:
            msg='invalid user name and password'
            return render_template('register.html',msg=msg)

    else:

        return render_template('register.html')

@app.route("/searchproduct",methods=['GET', 'POST'])
def searchproduct():
    if request.method == 'POST':
        Product=request.form.get('search')
        sql = "SELECT * FROM products WHERE product =?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt,1,Product)

```



```

        ibm_db.execute(stmt)
        account = ibm_db.fetch_assoc(stmt)
        print (account)
        return render_template('searchproduct.html',product=account)
    else:
        return render_template('searchproduct.html')

@app.route("/viewbill",methods=['GET', 'POST'])
def viewbill():
    sql ="SELECT * FROM billing"

    stmt = ibm_db.exec_immediate(conn, sql)

    bill=[]
    amount=0

    while ibm_db.fetch_row(stmt) != False:
        dic=dict()
        dic['invoice']=ibm_db.result(stmt, 1)
        dic['product']=ibm_db.result(stmt, 3)
        dic['price']=ibm_db.result(stmt, 4)
        price=ibm_db.result(stmt, 4)
        dic['quantity']=ibm_db.result(stmt,5)
        quantity=ibm_db.result(stmt,5)
        dic['total']=int(price)*int(quantity)
        total=int(price)*int(quantity)
        amount +=total
        bill.append(dic)

    print(bill)
    print(amount)
    return render_template('viewbill.html',datas=bill)

@app.route("/minimum",methods=['GET', 'POST'])
def minimum():
    sql ="SELECT * FROM products"
    # stmt = ibm_db.prepare(conn, sql)
    # ibm_db.bind_param(stmt,1,name)
    # ibm_db.bind_param(stmt,2,password)
    stmt = ibm_db.exec_immediate(conn, sql)

    datas=[]
    while ibm_db.fetch_row(stmt) != False:
        dic=dict()
        dic['product']=ibm_db.result(stmt, 0)

```

```

        dic['stock']=ibm_db.result(stmt, 1)
        dic['price']=ibm_db.result(stmt, 2)
        dic['alert']=ibm_db.result(stmt, 3)
        datas.append(dic)
    print(datas)
    # ibm_db.execute(stmt)
    # account = ibm_db.fetchall(stmt)
    # print (account)
    return render_template('minimum.html',datas=datas)

@app.route("/dashboard",methods=['GET', 'POST'])
def dashboard():
    sql="SELECT * FROM products"
    stmt = ibm_db.exec_immediate(conn, sql)

    datas=[]
    low=0
    count=0
    while ibm_db.fetch_row(stmt) != False:
        dic=dict()
        dic['product']=ibm_db.result(stmt, 0)
        dic['stock']=ibm_db.result(stmt, 1)
        stock=ibm_db.result(stmt, 1)
        dic['price']=ibm_db.result(stmt, 2)
        dic['alert']=ibm_db.result(stmt, 3)
        alert=ibm_db.result(stmt, 3)
        if int(stock)< int(alert):
            low += 1
        datas.append(dic)
        count+=1
    print(datas)
    sql="SELECT * FROM billing"

    stmt = ibm_db.exec_immediate(conn, sql)

    bill=[]
    amount=0
    bill_count=0
    while ibm_db.fetch_row(stmt) != False:
        dic=dict()
        dic['invoice']=ibm_db.result(stmt, 1)
        dic['product']=ibm_db.result(stmt, 3)
        dic['price']=ibm_db.result(stmt, 4)
        price=ibm_db.result(stmt, 4)
        dic['quantity']=ibm_db.result(stmt,5)
        quantity=ibm_db.result(stmt,5)

```

```

        dic['total']=int(price)*int(quantity)
        total=int(price)*int(quantity)
        amount +=total
        bill_count+=1
        bill.append(dic)

    print(bill)
    print(amount)
    return
render_template('dashboard.html',datas=datas,low=low,amount=amount,count=count,bill_count=bill_count)

@app.route("/billing",methods=['GET', 'POST'])
def billing():
    date=datetime.datetime.today().date()
    if request.method == 'POST':
        invoice=request.form.get('invoice')
        date=request.form.get('date')
        product=request.form.get('product')
        quantity=request.form.get('quantity')
        price=request.form.get('price')
        insert=f"insert into billing values ( {invoice[-1]}, '{invoice}', '{date}', '{product}',
        {int(quantity)}, {int(price)})"
        table=ibm_db.exec_immediate(conn,insert)
        sql = "SELECT * FROM products WHERE product =?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt,1,product)
        ibm_db.execute(stmt)
        account = ibm_db.fetch_assoc(stmt)
        try:
            count=account['STOCK']
            alert=account['Alert']
            update_count=int(count)-int(quantity)
            if int(update_count) < int(alert):
                print('email code')

                #write sendgrid email code here

            message = Mail(

                from_email='shajalraj333@gmail.com',
                to_emails='shijonida2@gmail.com',
                subject='Sending with Twilio SendGrid is Fun',

                html_content='<strong>and easy to do anywhere, even with Python</strong>')
            try:

```

```

        sg
=SendGridAPIClient('SG.ugbhjeYMQkKqLEAXNHd3ig.ovmpPK1LNdf_oXybocXoEx
qsjavVbSEefk0NvjqCEJs')
        response = sg.send(message)
        print(response.status_code)
        print(response.body)
        print(response.headers)
    except Exception as e:
        print(e)

        update=f"UPDATE products SET  stock = {update_count} WHERE  product =
'{product}'"
        table=ibm_db.exec_immediate(conn,update)
        return redirect(url_for('dashboard'))
    except:

        return redirect(url_for('dashboard'))

    else:
        return render_template('billing.html',date=date,invoice=invoice_no())

@app.route("/addproduct",methods=['GET', 'POST'])
def addproduct():
    if request.method == 'POST':
        Product=request.form.get('Product')
        Stock=request.form.get('Stock')
        Price=request.form.get('Price')
        Alert=request.form.get('Alert')
        print(Product,Stock,Price,Alert)
        insert=f"insert into products values ( '{Product}', {int(Stock)}, {int(Price)}, {int(Alert)})"
        table=ibm_db.exec_immediate(conn,insert)

        return redirect(url_for('dashboard'))
    else:
        return render_template('addproduct.html')

def invoice_no():
    sql ="SELECT  id FROM billing ORDER BY ID DESC limit 1"
    stm=ibm_db.exec_immediate(conn,sql)
    while ibm_db.fetch_row(stm) != False:
        count=ibm_db.result(stm,0)
    if count:

```

```
        return f'bill00{int(count)+1}'  
    else:  
        return 'bill001'
```

```
app.run(debug=True)
```

GitHub & Project Demo Link:

<https://github.com/IBM-EPB>