HX8001- PROFESSIONAL READINESS FOR INNOVATION, EMPLOYABILITY AND NTREPRENEURSHIP

INVENTORY MANAGEMENT SYSTEM FOR RETAILERS

TEAM ID: PNT2022TMID41025

TEAM MEMBERS:

ABARNA P
CHANDRU P
GANESAN S
MAHALAKSHMI V
RAVEENA R

In partial fulfilment for the award of the degree

Of

BACHELOR OF ENGINEERING

In

COMPUTER SCIENCE AND ENGINEERING

SHREE SATHYAM COLLEGE OF ENGINEERIGN AND TECHNOLOGY

ANNA UNIVERSITY: CHENNAI 600 025

NOV-DEC 2022

TABLE OF CONTENT

APPENDIX

1. INTRODUCTION

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

1.1. PROJECT OVERVIEW

Inventory management system is an application which is helpful for business operate. It is a cloud based web application that is specifically implemented to make the lives of warehouse workers much easier. It is an inventory management system for all the retailers out there in the market where they can manage, add and delete and track their goods that are being imported and exported through all locations. By managing inventory, retailers meet customer demand without running out of stock or carrying excess supply. This results in lower costs and gives them a better understanding on sales patterns. Inventory management is a challenging problem area in supply chain management. Companies need to have inventories in warehouses in order to fulfil customer demand, meanwhile these inventories have holding costs and this is frozen fund that can be lost. Therefore, the task of inventory management is to find the quantity of inventories that will fulfil the demand, avoiding overstocks.

1.2. PURPOSE:

The purpose is to help retailers track and manage stocks related to their own products. The system will ask the retailers to create their accounts by providing essential details. Once retailers login successfully into 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 their inventory whenever they wish and we have used Send Grid email service which sends an alert to retailers through email If there is no stock found in their accounts. And they can order new stock at that time.

2.LITERATURE SURVEY

2.1. EXISTING PROBLEM

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

2.2. REFERENCES

[1] Sahari, Salawati, Tinggi, Michael & Kadri, Norlina. (2012 firms: Impact on performance. SIU Journal of Management, vol.2,iss.1, pp.59–72.). Inventory management in Malaysian construction.

[2] Soni, Anita. (2012). Inventory management of engineering goods industry in Punjab: An empirical analysis. International Journal of Multidisciplinary Research, vol.2,iss.2, pp.247–261.

[3] Panigrahi, Ashok K. (2013). Relationship between inventory management and profitability: An empirical analysis of Indian cement companies. Asia Pacific Journal of Marketing & Management Review, vol.2,iss.7,
pp.107–120.

[4] Madishetti, Srinivas & Kibona, Deogratias. (2013). Impact of inventory management on the profit ability of
SMEs in Tanzania. International Journal of Research in Commerce & Management, vol.4,iss.2, pp.1–6.

[5] Srinivasa Rao Kasisomayajula(2014) “An Analytical Study on Inventory Management in Commercial Vehicle
Industry in India”, International Journal of Engineering Research, Vol.3, Iss.6, pp.378-383.

[6] Edwin Sitienei, Florence Memba(2015-16) “ The Effect of Inventory Management on Profitability of Cement
Manufacturing Companies in Kenya: A Case Study of Listed Cement Manufacturing Companies in Kenya”
International Journal of Management and Commerce Innovations Vol. 3, Iss.2, pp. 111- 119.

[7] Ajaro Tony Martins, Cynthia Ejike, Joy Nwokoro, Solomon (2022) “Inventory Management Features, objective,pros and cons, form https://www.ukessama.com.

2.3. PROBLEM STATEMENT DEFINITION`

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

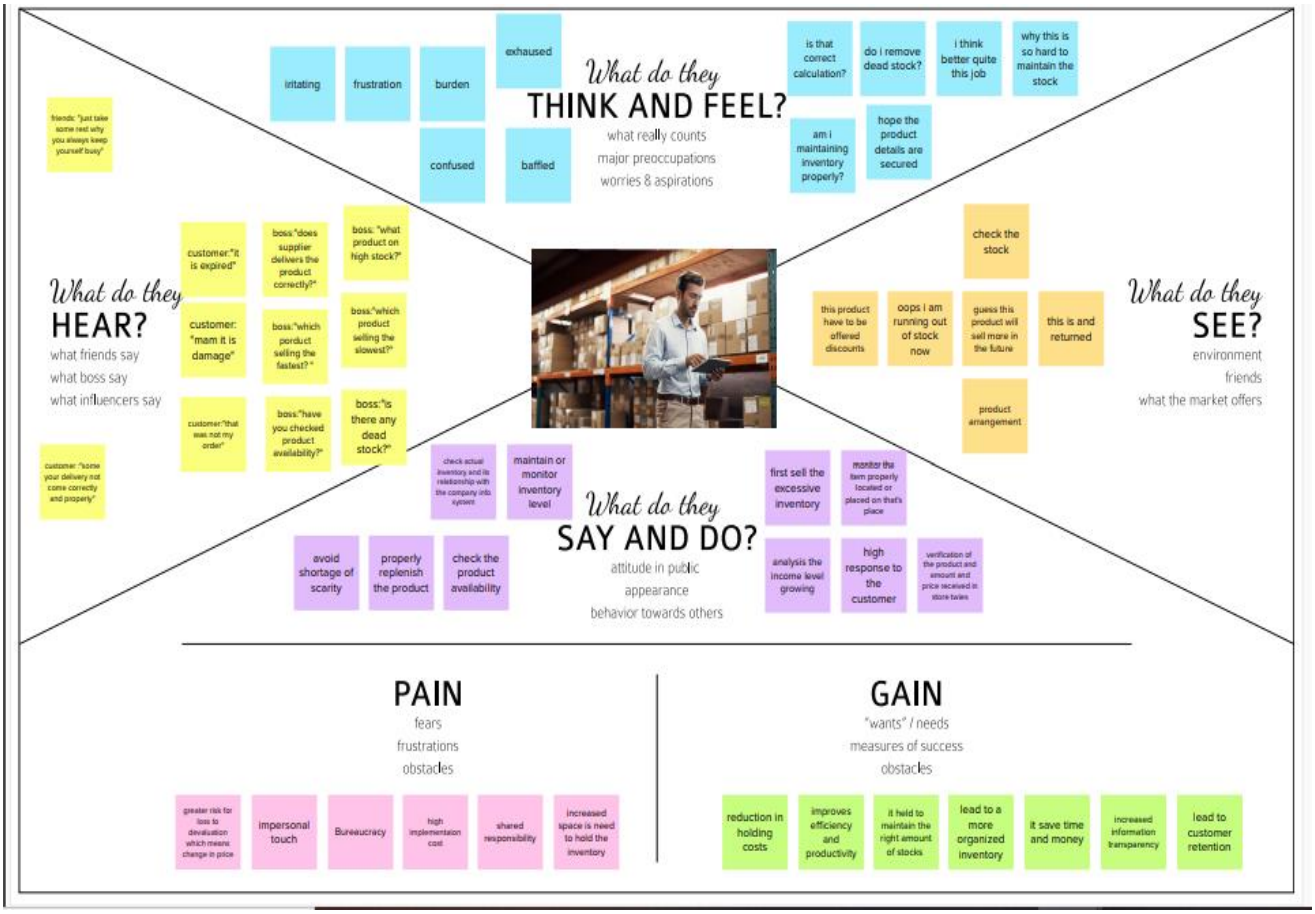
Problem Statement (PS)	Iam	I'm trying to	But	Because	Which makes me feel
PS-1	Retailer	monitor stock levels, and analyze situations effectively, avoid out-of-stock situations, avoid overstocking, retain customers	is time-consuming, redundant and vulnerable to errors	Using manual inventory tracking procedures across different software and spreadsheets.	Frustrated, stressed and sad

3. IDEATION & PROPOSED SOLUTION

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

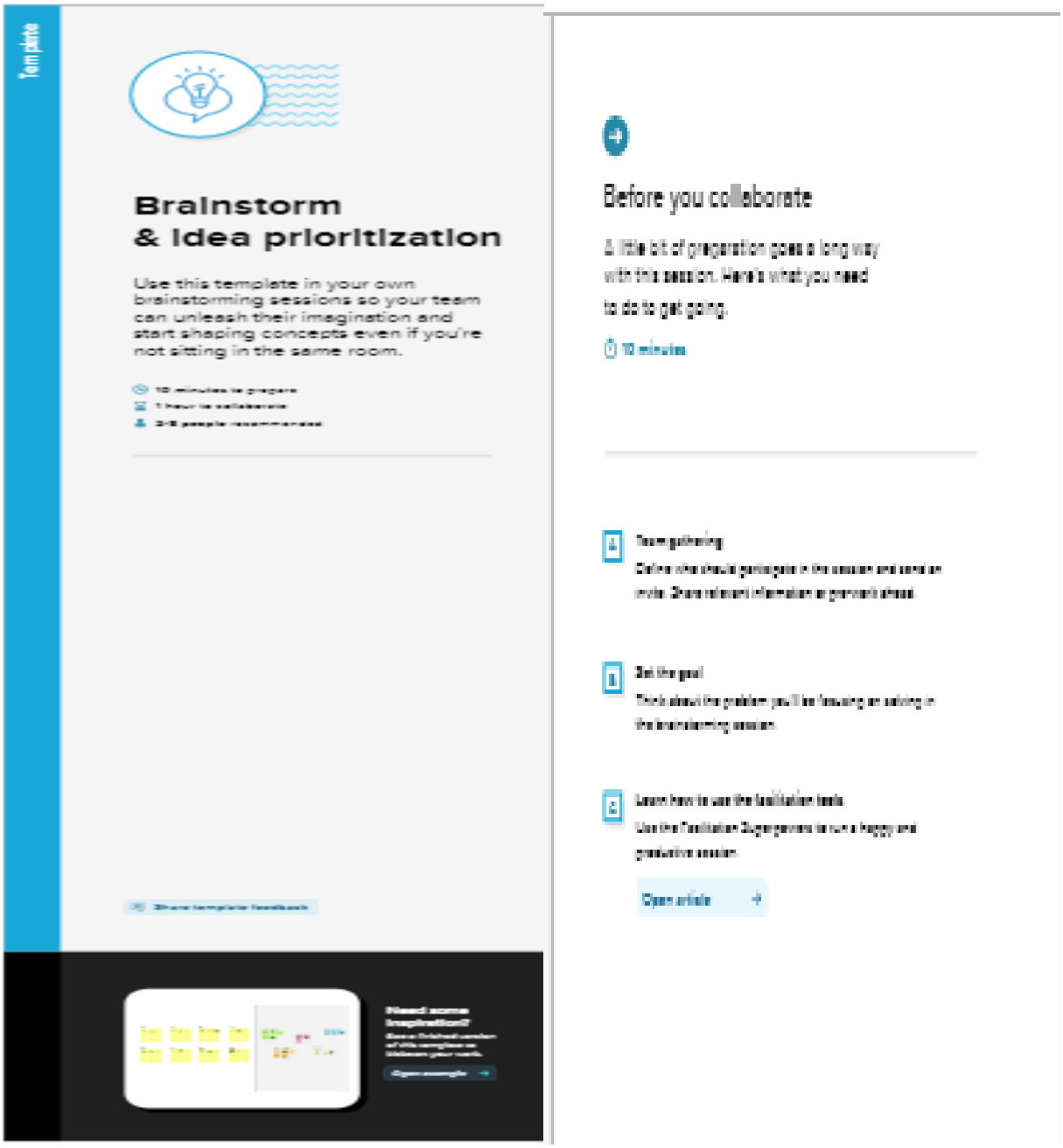
3.1. EMPATHY MAP CANVAS

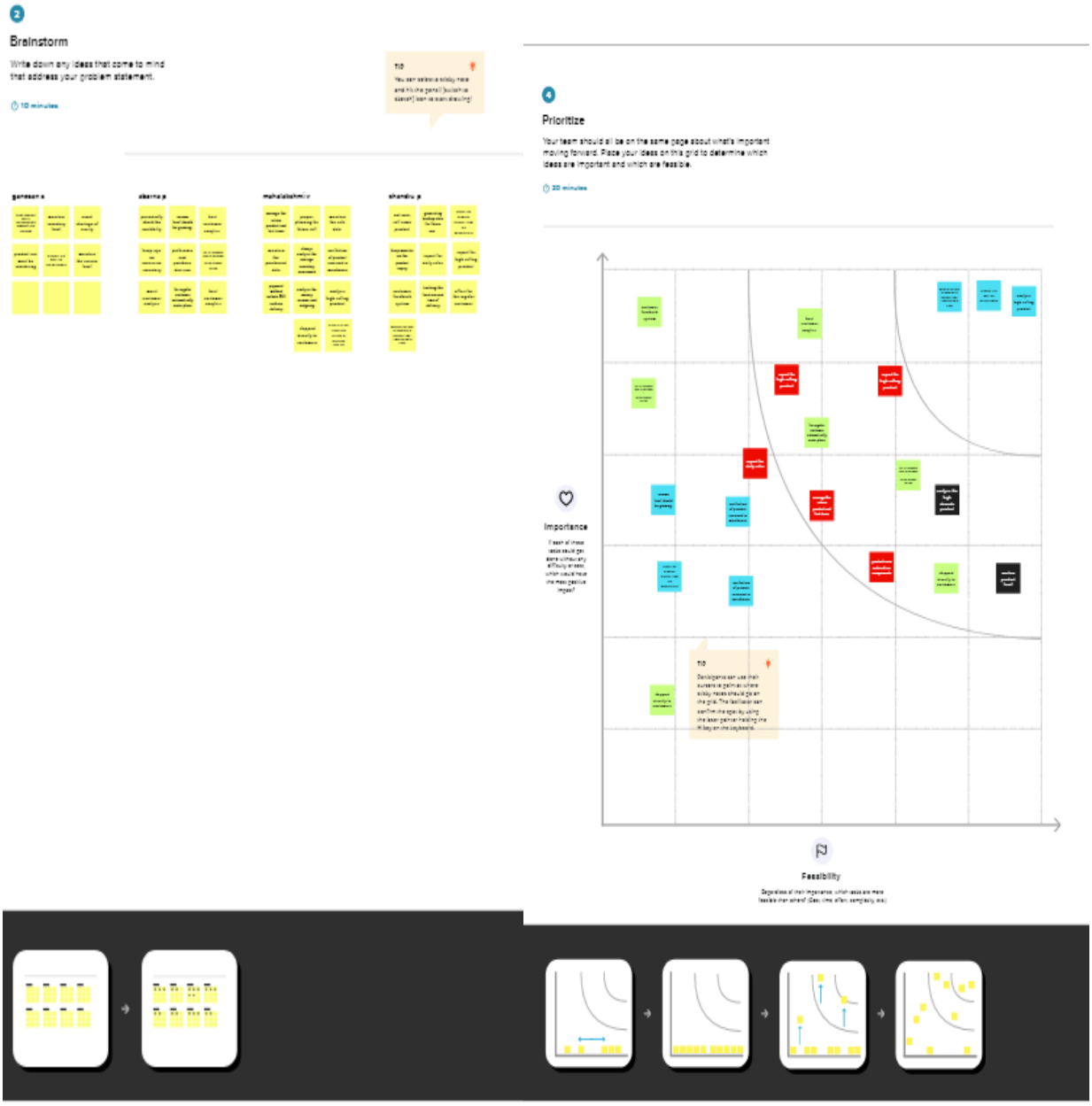
An empathy map canvas helps brands provide a better experience for users by helping teams understand the perspectives and mindset of their customers. Using a template to create an empathy map canvas reduces the preparation time and standardizes the process so you create empathy map canvases of similar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



3.2. IDEATION & BRAINSTORMING

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




2
Brainstorm
Write down any ideas that come to mind that address your problem statement.
10 minutes
4
Prioritize
Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.
20 minutes
Importance
Feasibility

3.3. PROPOSED SOLUTION

PREPROCESSING PHASES

S.No.	Parameter	Description
1.	Problem Statement	The retailer need a way to monitor stock levels, and analyze situations effectively, avoid out-of-stock situations, avoid overstocking, retain customers so the he/she can maintain the inventory efficiently and successfully run their business.
2.	Idea / Solution description	• Economic order quantity which is a quantity of inventory which can reasonably be ordered economically at time, • ABC analysis means always better control; the inventory is classified into 3 categories according to the inventory value and cost significance. • VED analysis; the item are classified on the basis of their criticality to the production services. V stands for vital items without which the production process would come to standstill. E denotes Essential item whose stock out would adversely affect the efficiency of the production system.D stands for desirable item which are required but do not immediately causes a loss of production • Just In Time: having the right items of the right quality and quantity in the right place at the right time. • Stock keeping unit: every product at the store has a unique code. This help in identification
3.	Novelty / Uniqueness	• Seamless CRM Integrations • Boosted Sales. • Online and Offline Order Management. • Increased customer satisfaction with end-to-end tracking. • Increased scalability and flexibility with a host of available add-ons. . • Simple and affordable pricing.
4.	Social Impact / Customer Satisfaction	By providing service to the small and large scale retailers.
5.	Business Model (Revenue Model)	Inventory management helps companies identify which and how much stock to order at what time. It tracks inventory from purchase to the sale of goods. The practice identifies and responds to trends to ensure there's always enough stock to fulfill customer orders and proper warning of a shortage.
6.	Scalability of the Solution	profitability of the business increase and efficiency of doing business increase

3.4. PROBLEM SOLUTION FIT

➢ To develop a system that will enhance the monitoring of the sales and inventory

➢ To develop a module that can generate monthly sales and inventory report.

➢ To develop security in terms of keeping the records of the inventory

➢ To develop a system that can monitor the stocks inventory in a fast and efficient manner.

➢ To accurately record, compute and produce a report of sales.

Project Title: Inventory Management System For Retailers **Team ID: PNT2022TMId41025**

Define CS, fit into CC

1. CUSTOMER SEGMENT(S) CS

- Retailers
- Small enterprises

2. CUSTOMER CONSTRAINTS CC

- Network Connection
- Proper stock knowledge
- Manual data entry accuracy

3. AVAILABLE SOLUTIONS AS

- Manual Inventory Tracking
- slower order processing, higher labor costs and larger inventory write-offs at the end of the year small mistakes can amount to a big profit-loss

Explore AS, differentiate

Identify strong TR & EM

7. TRIGGERS TR

- Increasing customer demand
- Market competition
- Insufficient Order Management

8. EMOTIONS: BEFORE / AFTER EM

- Before: frustrated and stress mentally and physically
- After: happier, relief, confident.

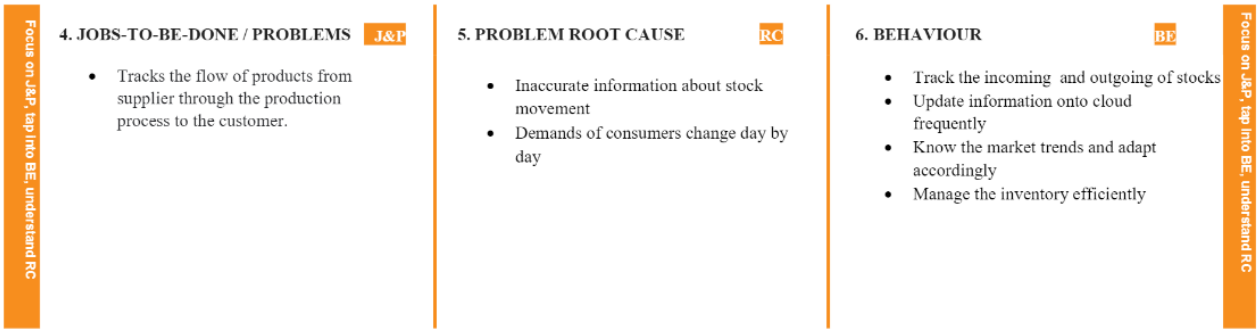
9. YOUR SOLUTION SOLN

- Developing a cloud application which helps the customer to create and manage both sales and purchase orders, and track inventory.
- provide a option for graphical view of sales

10. CHANNELS of BEHAVIOUR CB

1. **ONLINE**
   - Alerting the particular person about the stocks limits, either full or empty or even about the reach of a particular limit
   - Updating of flowing of the stocks regularly
2. **OFFLINE**
   - Manual Checking
   - Stock Distribution among the Inventory

Extract Online and Offline CH of BE



4. REQUIREMENT ANALYSIS

4.1. FUNCTIONAL REQUIREMENTS

Functional Requirements:

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form Registration through Gmail Registration through LinkedIN
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	User login	Login with username Login with password
FR-4	Centralized Record of all product	Product name, Stock keep unit, brand, retail price, product category, lot number , expire date, vendor details, wholesale cost, minimum reorder amount, case quantity amount, reorder lead time
FR-5	Stock location identification	Provide number label for- Shelf, Rack and Boxes
FR-6	Periodical stock checking	Physical counting and Cycle counting
FR-7	Integration of sales and inventory data	sales administration and database upkeep FIFO,LILO according to the goods
FR-8	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, ABC,FSC,XYZ,JIT techniques
FR-9	Markdown and promotion	Show product discount, Maintain enough stock on hand to meet demand.
FR-9	Markdown and promotion	Show product discount, Maintain enough stock on hand to meet demand.
FR-10	Management of Receiving Stock	Accurately recording goods on an inventory
FR-11	Returns Management System	Check for damage or defects and return to vendor as needed If sellable add it to inventory counts
FR-12	Determination of death stock	Return to the vendor for credits
FR-13	Inventory KPIs(Key Performance Indicator)	Sale KPIs, Receive KPIs, Operational KPIs, Employee KPIs

4.2 NON-FUNCTIONAL REQUIREMENTS

Non-functional Requirements:

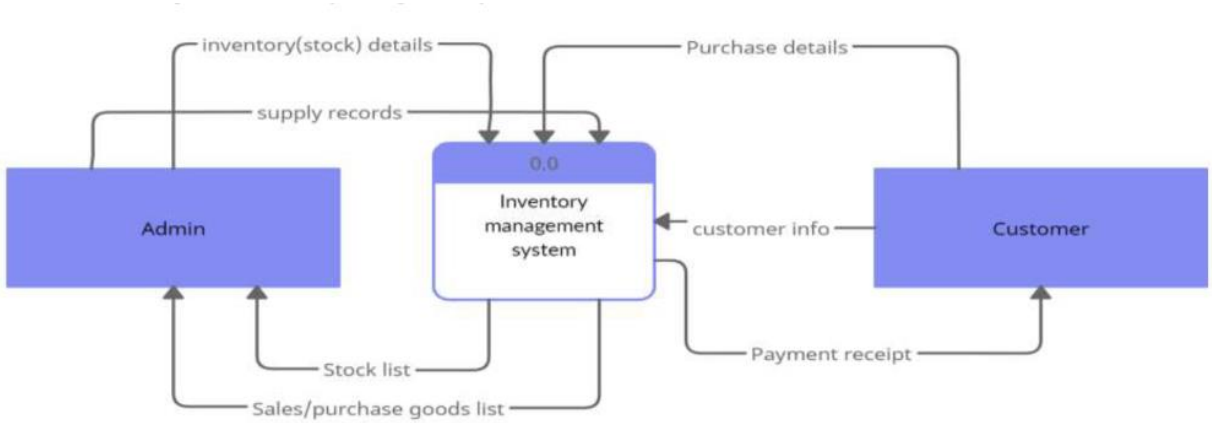
Following are the non-functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	This system must be easy to use by both managers and chefs, such that they do not need to read an extensive number of manuals; it must be quickly accessible by both managers and chefs; it must be intuitive and simple in the way it displays all relevant data and relationships; and the menus of the system are easily navigable by the users with buttons that are easy to understand.
NFR-2	**Security**	The security requirements deal with the primary security. Only authorized users can access the system with user name and password of administrator.
NFR-3	**Reliability**	The system must give accurate inventory status to the user continuously. Any inaccuracies are corrected by regularly comparing the actual levels to the levels displayed in the system. The system must successfully add any recipe, ingredients, vendors, or special occasions given by the user and provide estimations and inventory status in relevance to the newly updated entities.
NFR-4	**Performance**	The system must not lag, because the workers using it don't have downtime to wait for it to complete an action. The system must successfully complete updating the databases, adding new recipes, ingredients, vendors, and occasions every time the user requests such a process. All the functions of the system must be available to the user every time the system is turned on. The calculations performed by the system must comply with the norms set by the user and should not vary unless explicitly changed by the user.
NFR-5	**Availability**	The software will be available only to administrator of the organization and the product as well as customer details will be recorded by him. He can add customers, Update and delete them as well as add new products and manage them
NFR-6	**Scalability**	The ability of a system to handle a growing amount of work.

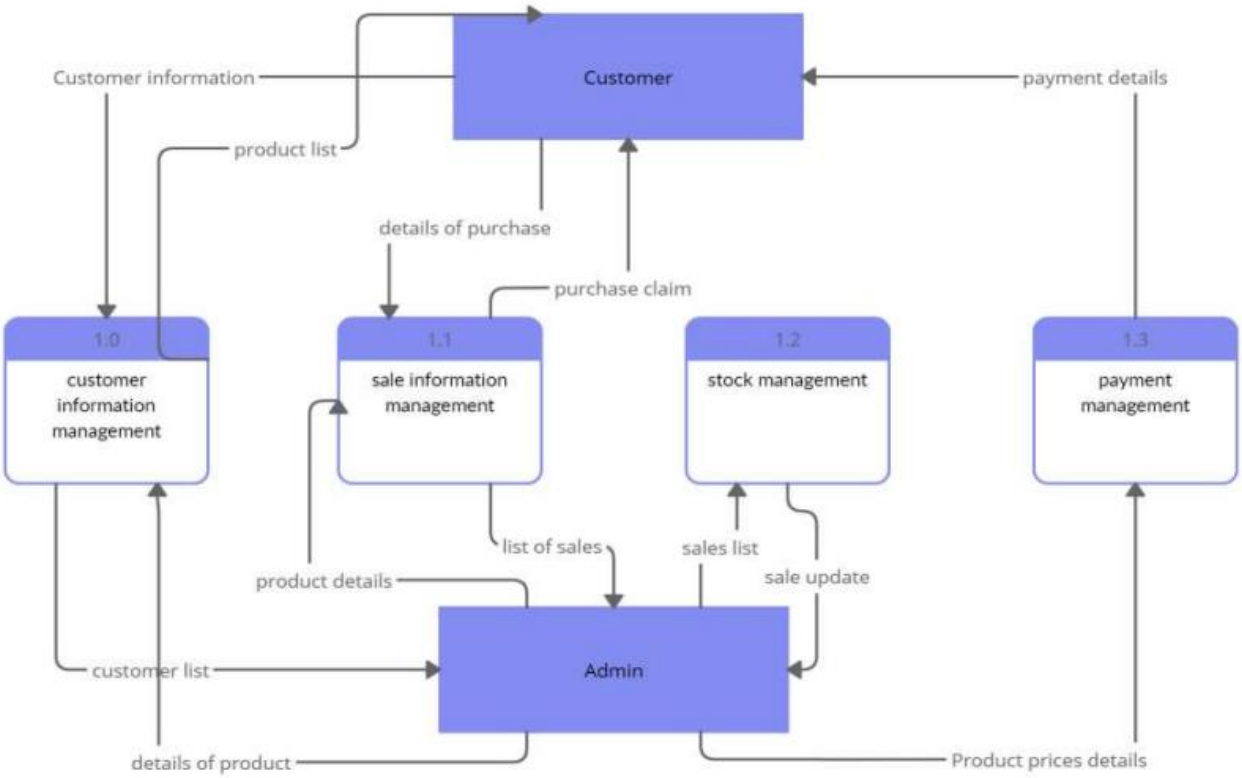
5. PROJECT DESIGN

5.1DATAFLOWDIAGRAMS

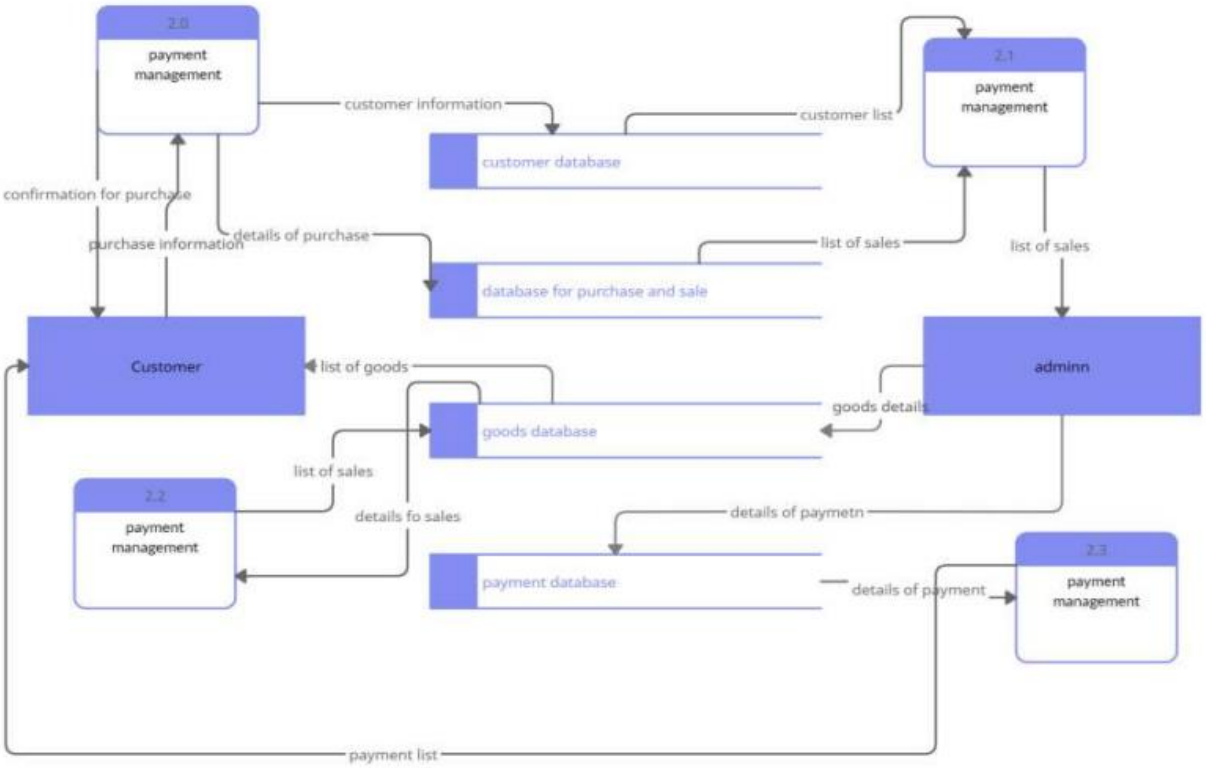
Level 0 Data Flow Diagram for inventory management system for retailers:



Level 1 Data Flow Diagram for inventory management system for retailers:



Level 2 Data Flow Diagram for inventory management system for retailers:



5.2SOLUTION & TECHNICAL ARCHITECTURE

• There was no efficient solution available in many companies these days.

•Every process was based on paper work.

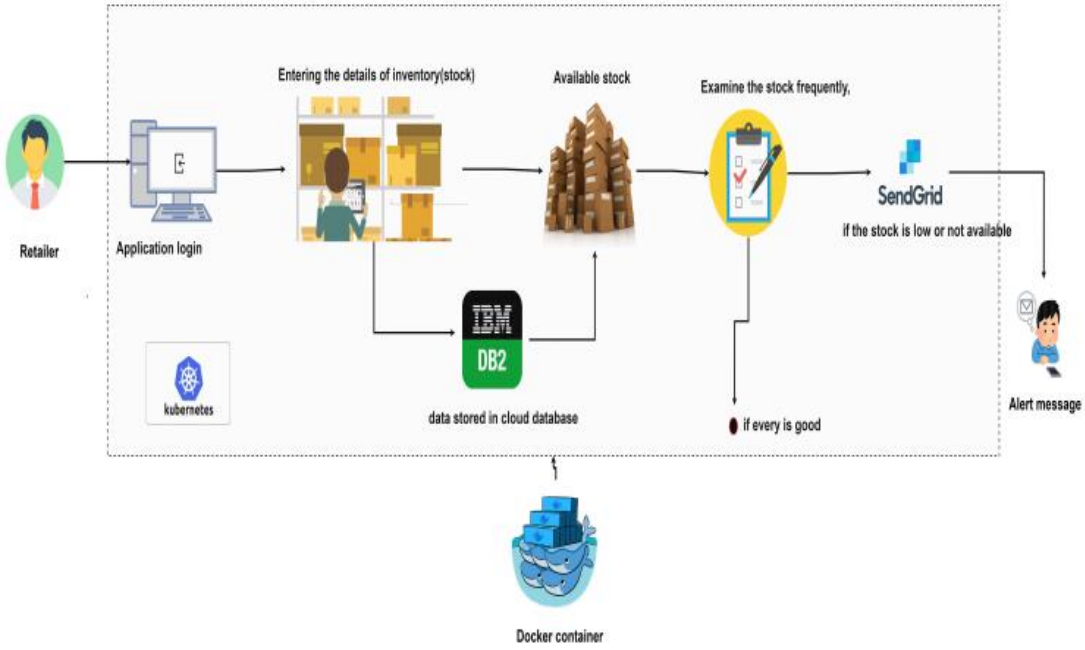
• Human fault rate were high.

• Tracing the inventory losses were not possible

• There was no efficient login system.

• After the computer age, every process is started to be integrated into computer environment.

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



-

1solution architecture diagram

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 receive conformation email and click confirm button	Medium	Sprint-1
	Login	USN-5	As a user, I can sign in to the application by giving my email & password	I can access my account	High	Sprint-1
	Dashboard	USN-6	As a user, it displays the stock , current sale demand product	I can see available stock , daily sale	High	Sprint-2
Customer (Web user)	Application	USN-7	As a user, I can register, sign in, and shop the products simply	I can access account anywhere	High	Sprint-3
Customer Care Executive	Update inventory details	USN- 8	To monitor the track of inventory and availability	I can improve the productivity	High	Sprint-4
Administrator	Update purchased stock	USN-9	To update purchased goods in database	I can update the new purchased product	High	Sprint-3
Customer care executive	Customer feedback verification	USN-10	To get a clear understanding about our application and for the convenience of the user	I can fulfil the customer expectations	High	Sprint-4
	Inventory control	USN-11	To avoid stock overflow and run out	I can alert mail if stock run out	Medium	Sprint- 2
administrator	Quality checking	USN-12	To maintain the product and improving the customer relationship	I can improve my product quality	High	Sprint-4

6 .PROJECT PLANNING & SCHEDULING

6.1SPRINT PLANNING & ESTIMATION

Sprint 1:

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

Sprint 2:

1. We added all the html templates needed for our project.
2. We styled those pages using CSS and Bootstrap
3. We wrote Queries to connect IBM Cloud Database
4. Finished all the Fetching and Posting Stuff of IBM Cloud Database Integration.

Sprint 3:

1. Integration of Send grid into our application

Sprint 4:

1. Deploying the application using Docker and Kubernetes

6.2SPRINT DELIVERY SCHEDULE

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	5	High	S Ganesan P abarna V mahalakshmi P chandru
Sprint-1		USN-2	As a user, I will receive confirmation email once I have registered for the application	4	High	S Ganesan P abarna V mahalakshmi P chandru
Sprint-1		USN-3	As a user, I can register for the application through Gmail	3	Medium	S Ganesan P abarna
Sprint-1	Login	USN-4	As a user, I can log into the application by entering email & password	4	High	S Ganesan P abarna V mahalakshmi P chandru
Sprint-1	Dashboard	USN-5	As a user, I can see the stock in hand and how much stock will be received and check other details.	4	High	S Ganesan P abarna V mahalakshmi P chandru
Sprint-2	Customer details	USN-6	As a user, I can see the customer details like name, company, location, and so on.	3	Low	P abarna V mahalakshmi
Sprint-2	Search	USN-7	As a user, I can search the products and customer and so on	1	Low	P abarna V mahalakshmi
Sprint-2	Purchase order management	USN-10	As a user, I can enter the newly purchased stock and add or remove the stocks. and upload the purchased details as well.	5	Medium	V mahalakshmi P chandru
Sprint-3	Stocks	USN-11	As a user, I can see the stock level, fast-moving, and death stocks.	4	High	S Ganesan P abarna V mahalakshmi P chandru
Sprint-3	Report	USN-12	As a user, I can see the report of the stock	1	Low	V mahalakshmi P chandru
Sprint-3	Notification	USN-13	As a user, it is good if I get a notification for low stock.	2	Medium	S Ganesan P abarna
Sprint-3	Supplier	USN-14	As a user, I can see the supplier details for a better understanding.	3	Low	V mahalakshmi P chandru
Sprint-2	Profile	USN-15	As a user, I can see my profile and give my details after registering as well.	1	Low	P abarna V mahalakshmi
Sprint-3	bill	USN-16	As a user, I like to print the product the sold now and maintain it.	4	Medium	S Ganesan P abarna
Sprint-3	Chatbot	USN-17	As a customer care executive,I can view the complaints on chat box, As a customer, I should be able solve and reply for the customers queries and as a customer, I can close the complaint after assisting	4	High	S Ganesan P abarna V mahalakshmi P chandru
Sprint-4	Containerization	USN-18	As a user, I can access the software with high performance	10	High	S Ganesan P abarna V mahalakshmi P chandru

Project Tracker, Velocity & Burndown Chart: (4 Marks)

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	31 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:

Sprint Duration : 6 Days

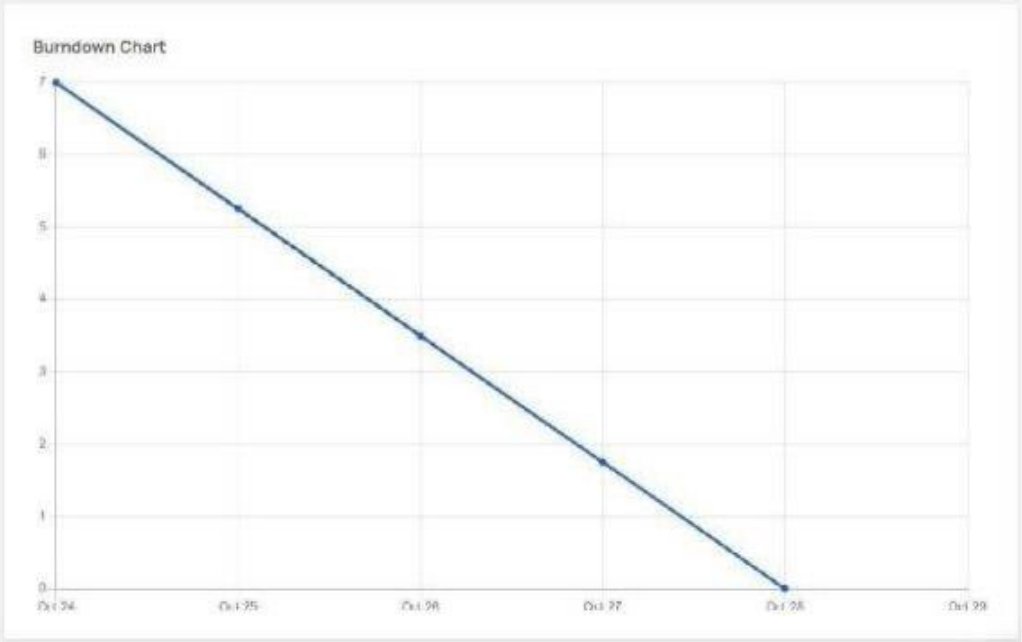
Velocity of the Team : 20 (points per sprint)

Team's Average Velocity :

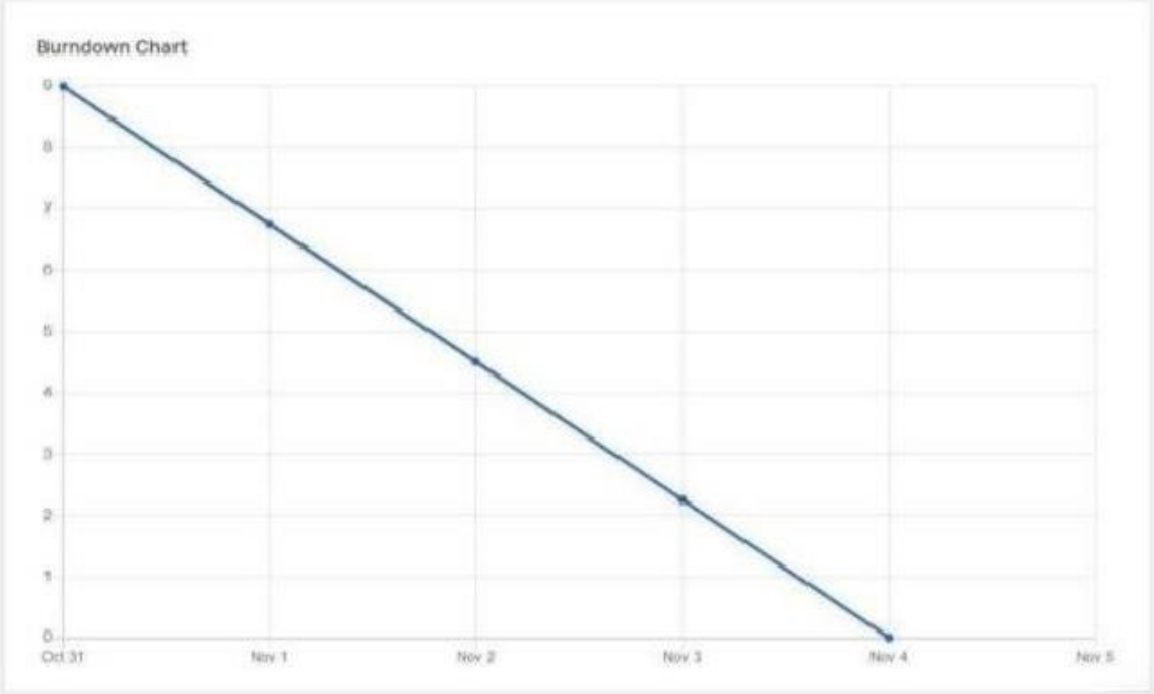
$$AV = \text{story points} / velocity\ sprint\ duration$$
$$= 206$$
$$= 3.3$$

Burndown Chart:

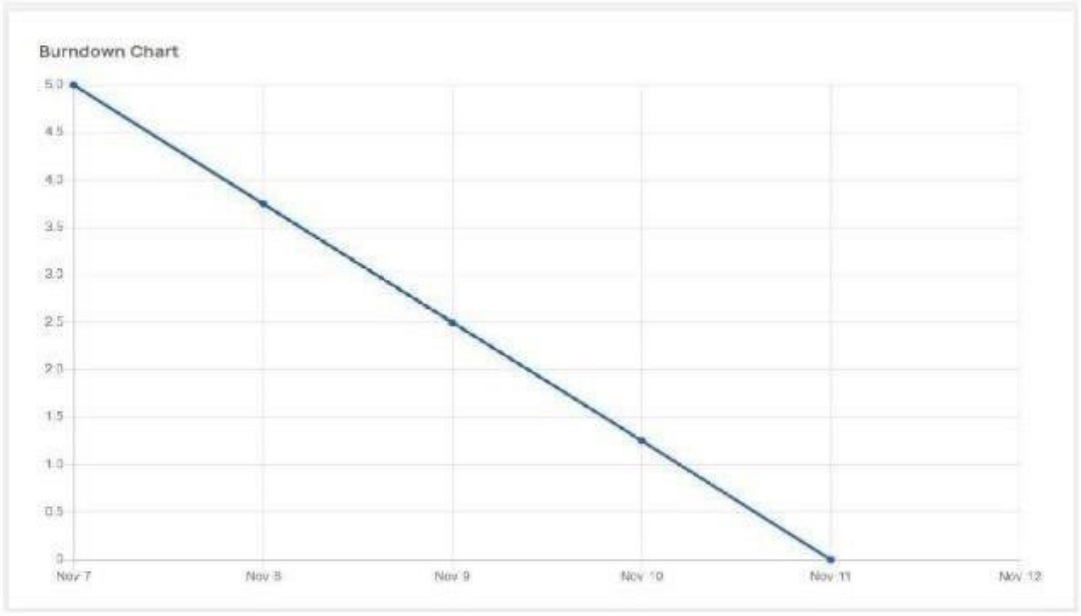
Sprint - 1



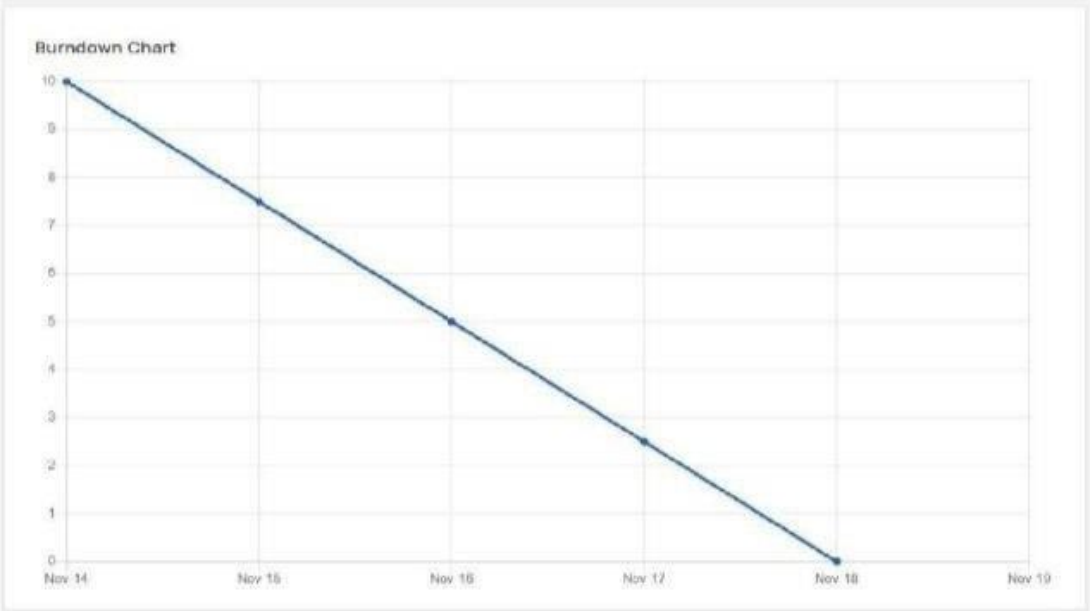
Sprint – 2



Sprint - 3



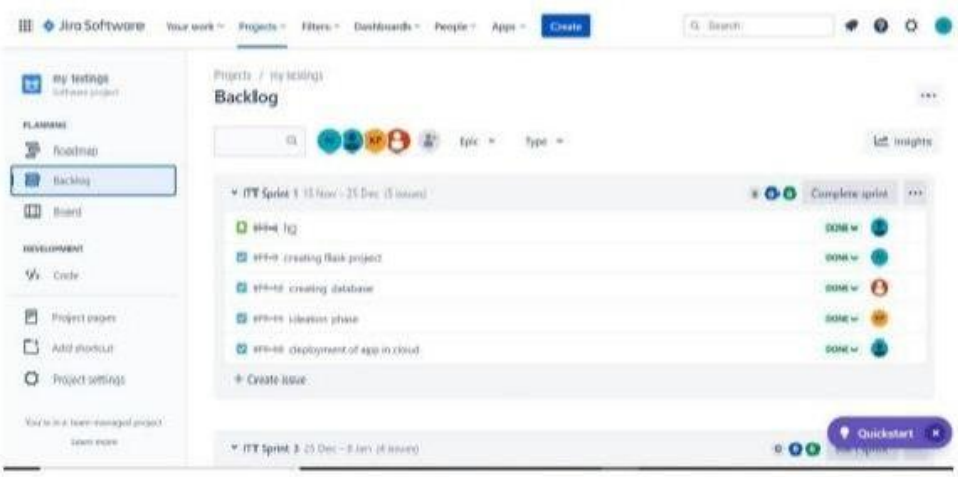
Sprint – 4 Project Tool: JIRA



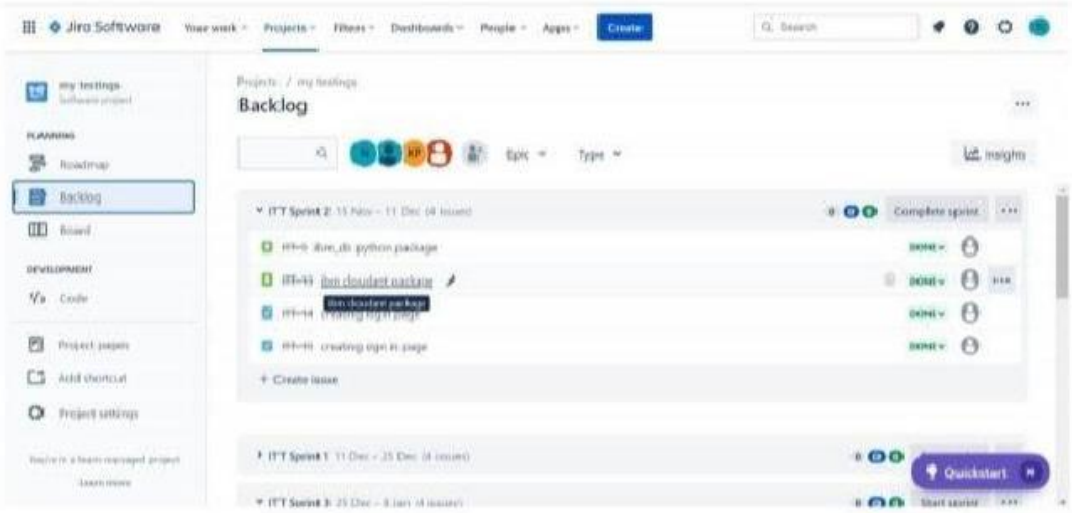
6.3 REPORTS FROM JIIRA

IT organization have the challenge of ensuring system uptime, supporting users, and managing inventory of both hardware and software. IT teams gain significant efficiencies when one tool can support multiple business operations. According to Gartner, mastering the discipline of effective asset management is a huge cost savings for companies.

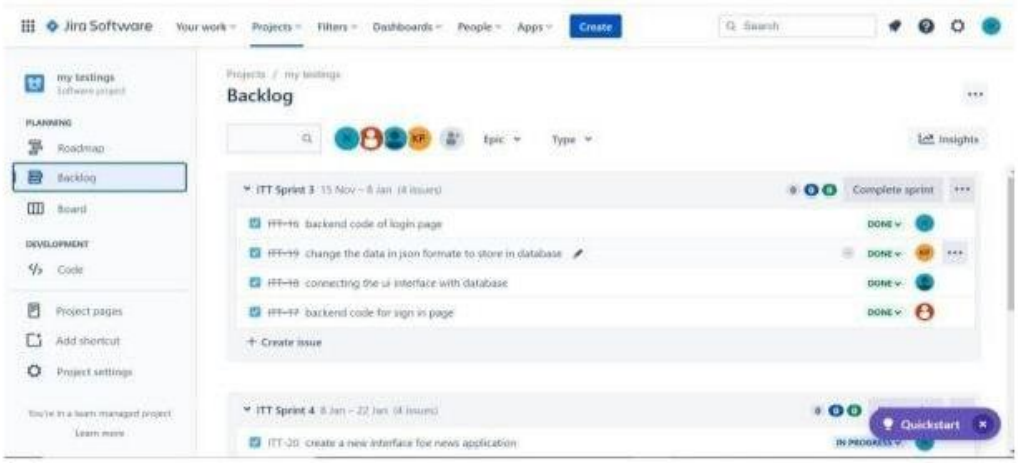
Sprint-1



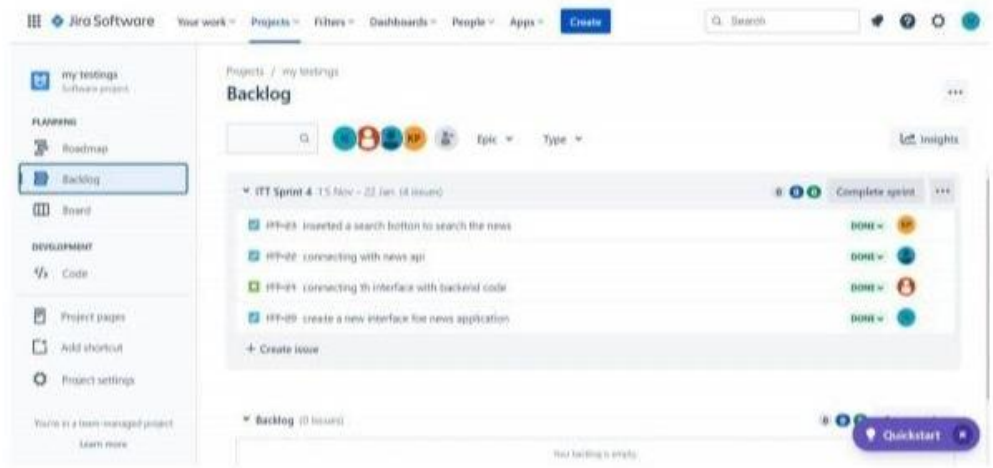
Sprint-2



Sprint-3

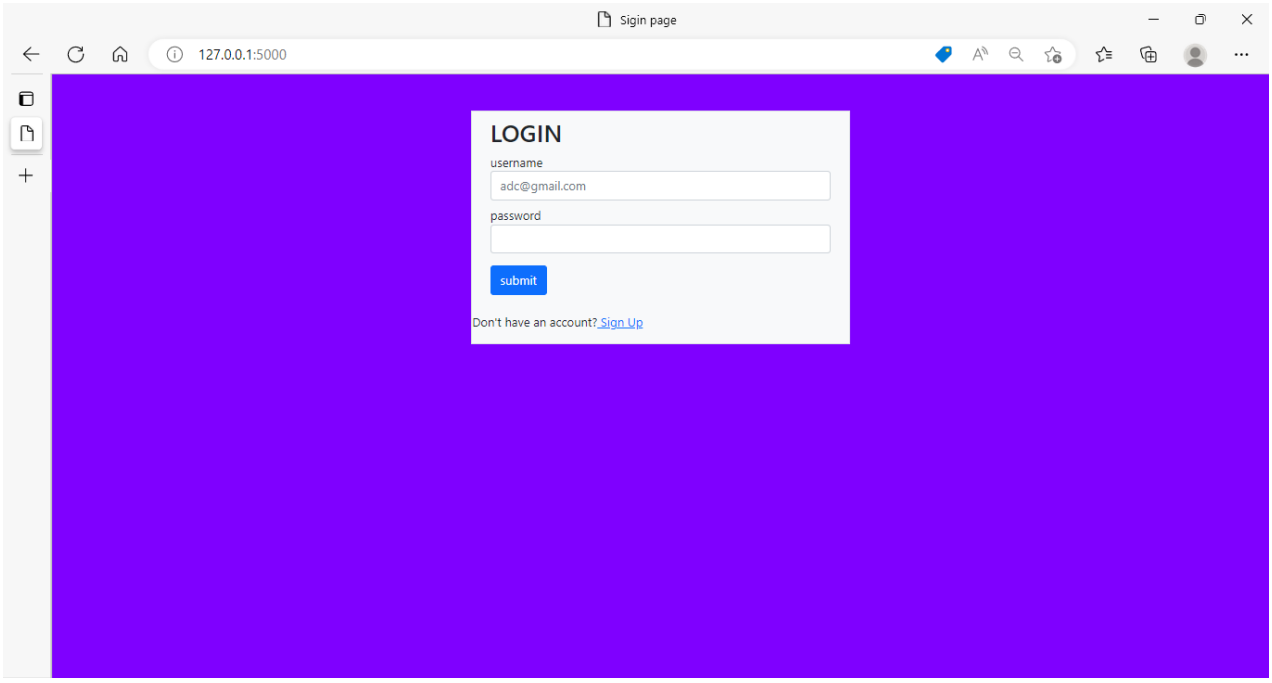


Sprint-4

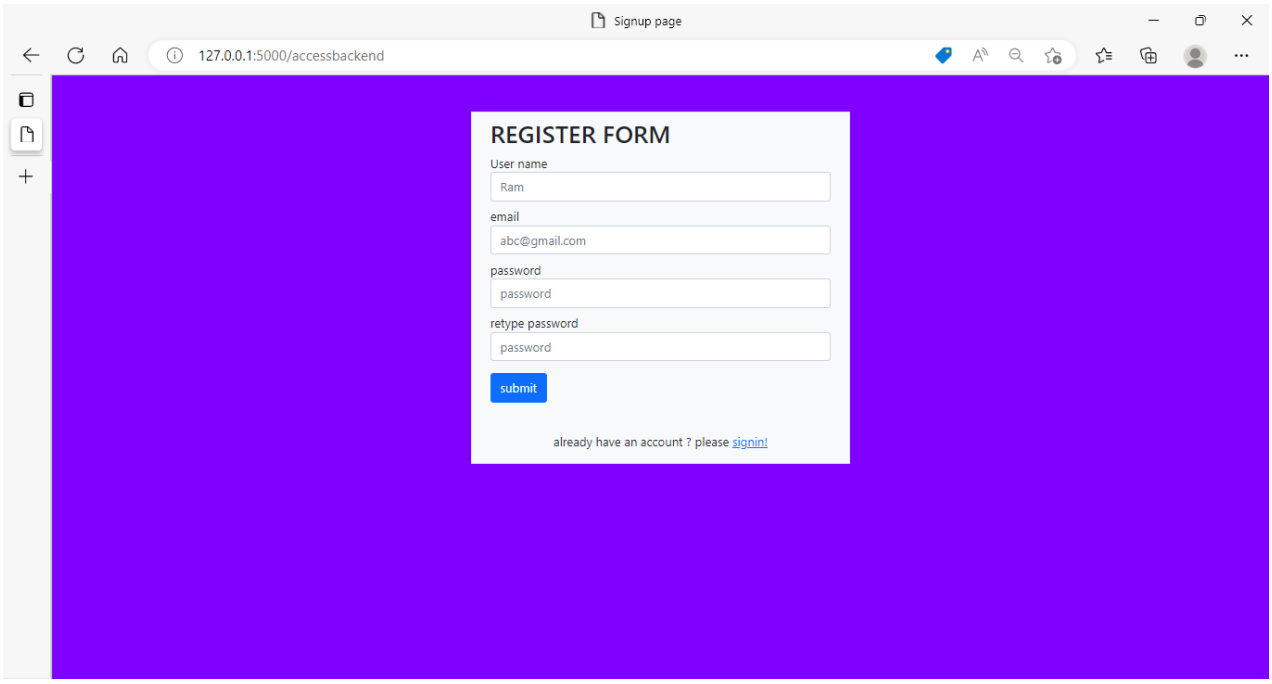


7.CODING & SOLUTIONING

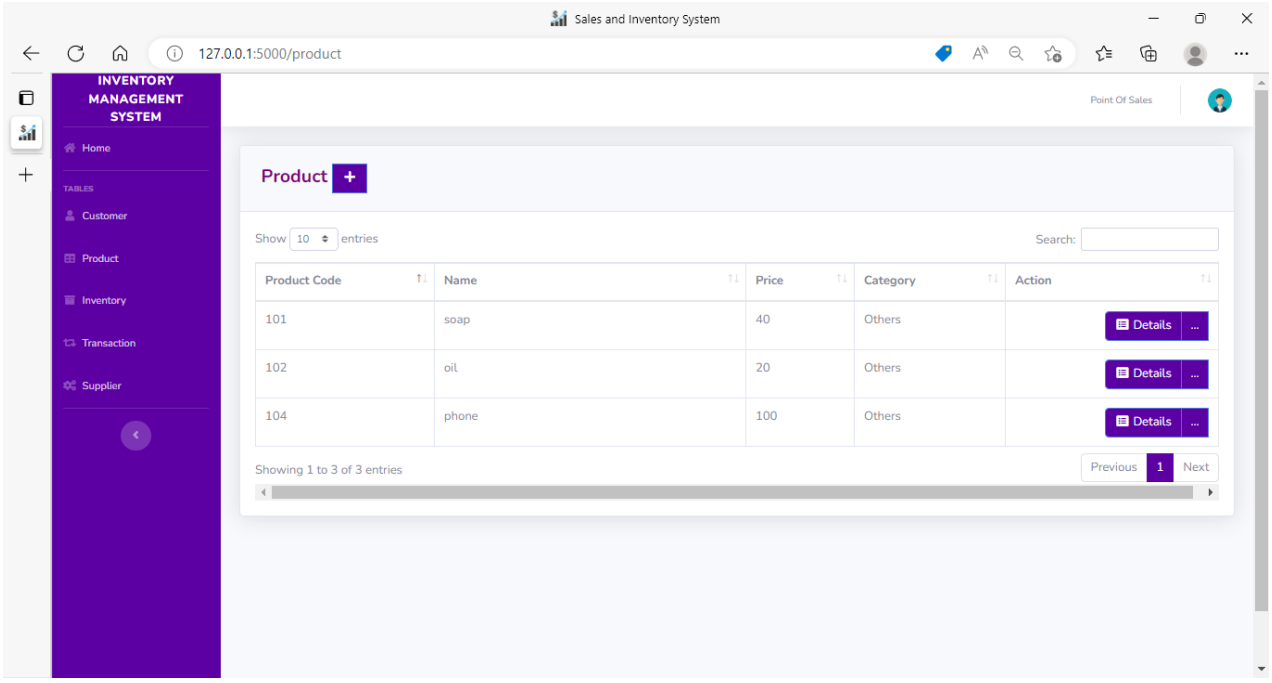
7.1. FEATURE1

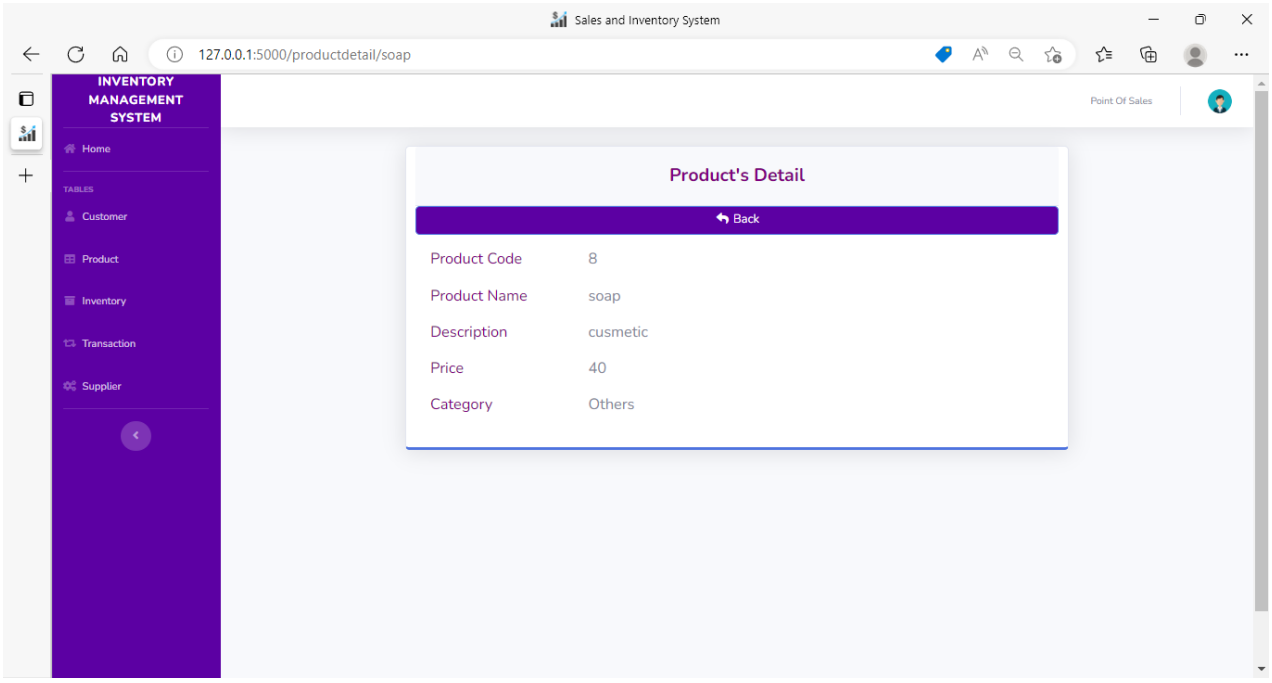


7.2. FEATURE 2



7.3. DATABASE SCHEMA





7. TESTING

8.1. TEST CASE

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

8.2. USER ACCEPTANCE TESTING

Test Case: Testing the Add Recipe Interface and its functioning

Case 1: Testing the Quantity input field.Case 2: Testing the Recipe Name field.
Case 3: Testing the Ingredients in recipe list and Quantity of ingredient list.
Case 4: Testing the available ingredients list.
Case 5: Testing the all the above cases together and checking if the entriesare updated to the tables in database.

Test Case: Check Threshold Interface

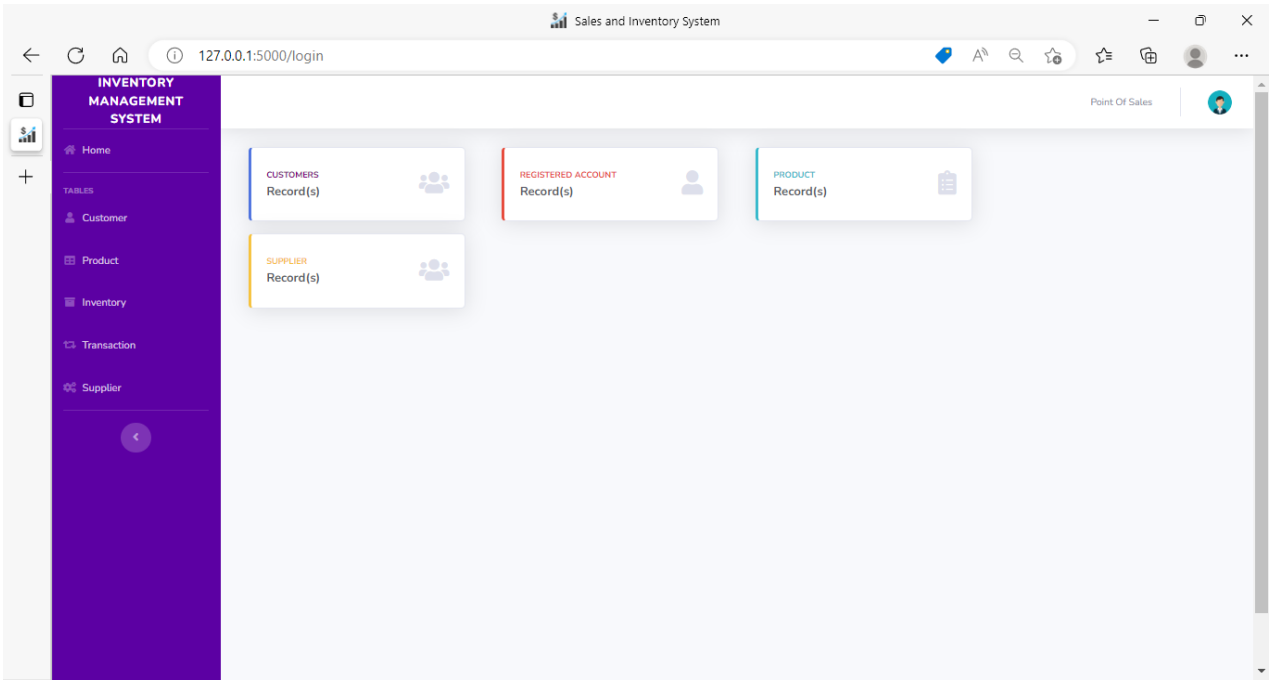
Case 1: Check if the Ingredients under the threshold values are shown in the Ingredients below threshold list.
Case 2: Check if the Create order button asks the user to enter values for all the ingredients listed under the ingredients below threshold list.
Case 3: Check if pressing the Process Order button creates a file with the order details in it.

Test Case : Testing the Update after sales interface

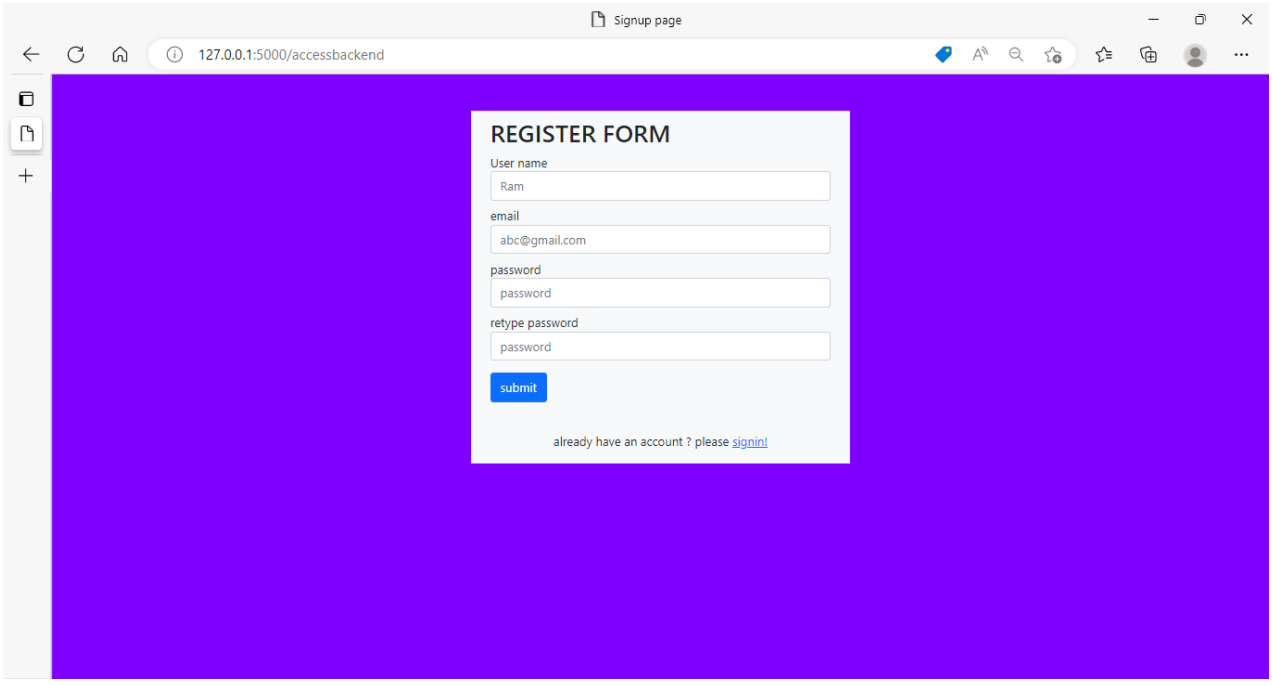
Case 1: Test the Recipe list box.
Case 2: Test the quantity text field..
Case 3: Test the recipe sold list box quantity sold list box.
Case 4: Test if the details are updated to the database when requested.

8. RESULTS

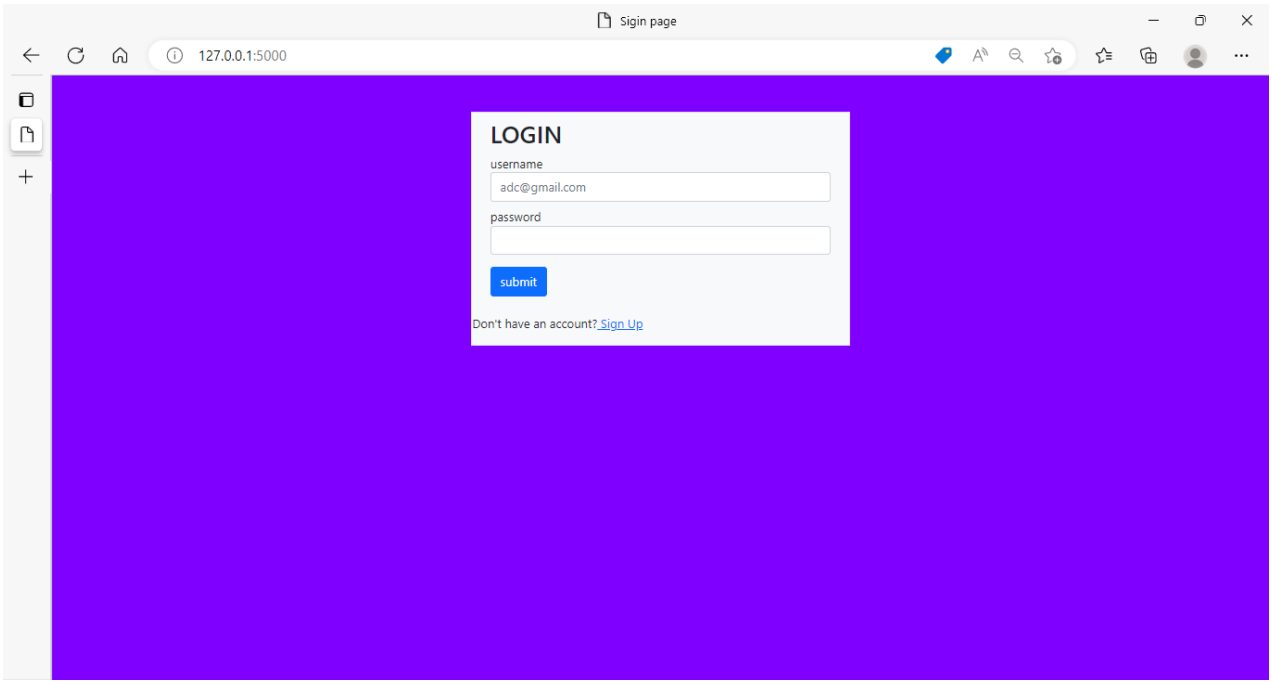
HOME PAGE



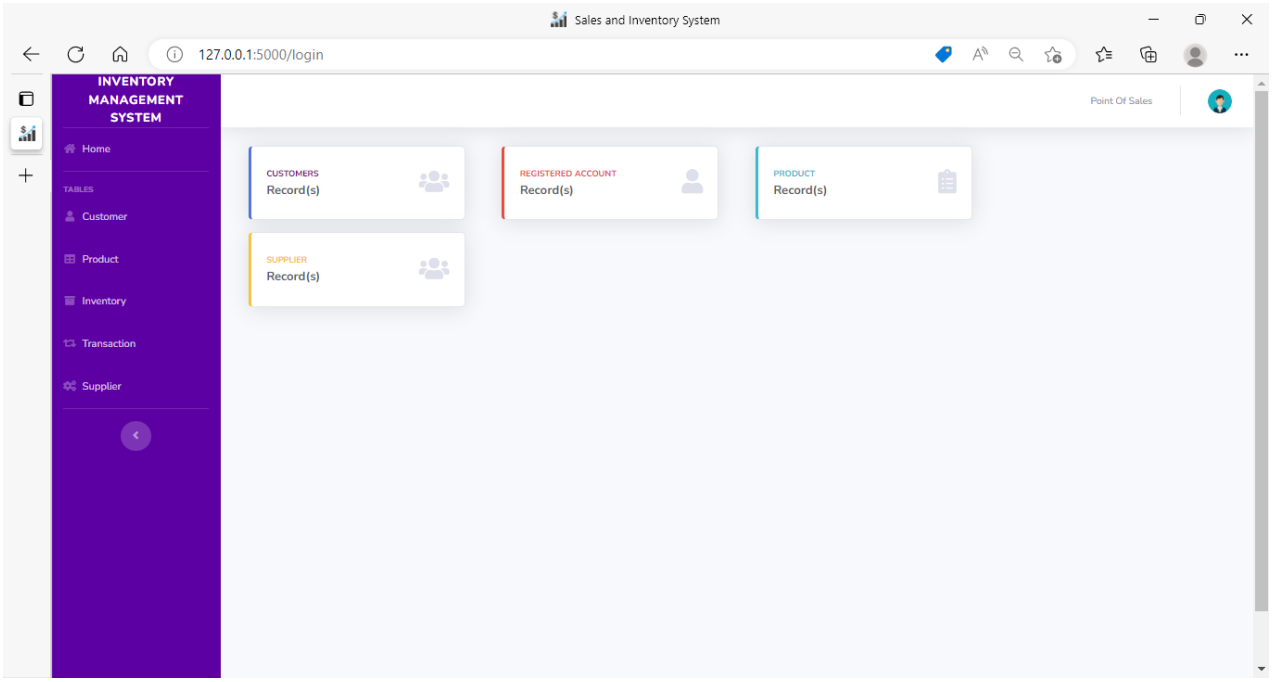
REGISTRATION FORM



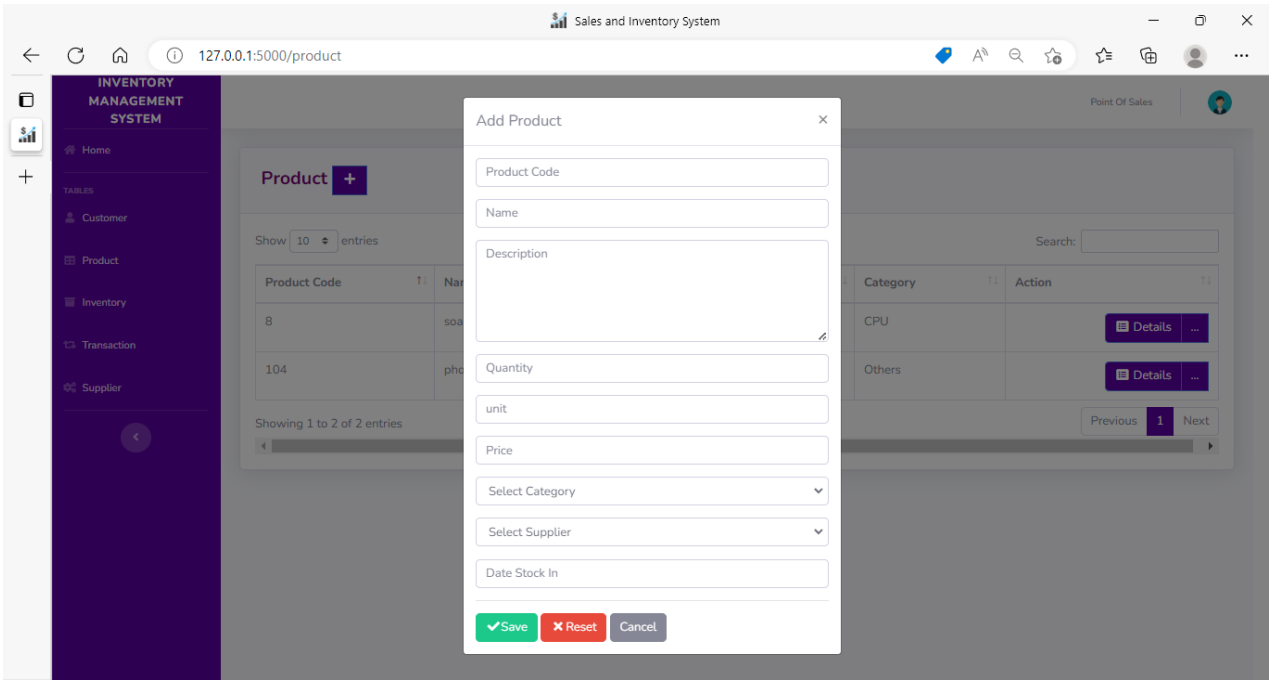
LOGIN FORM



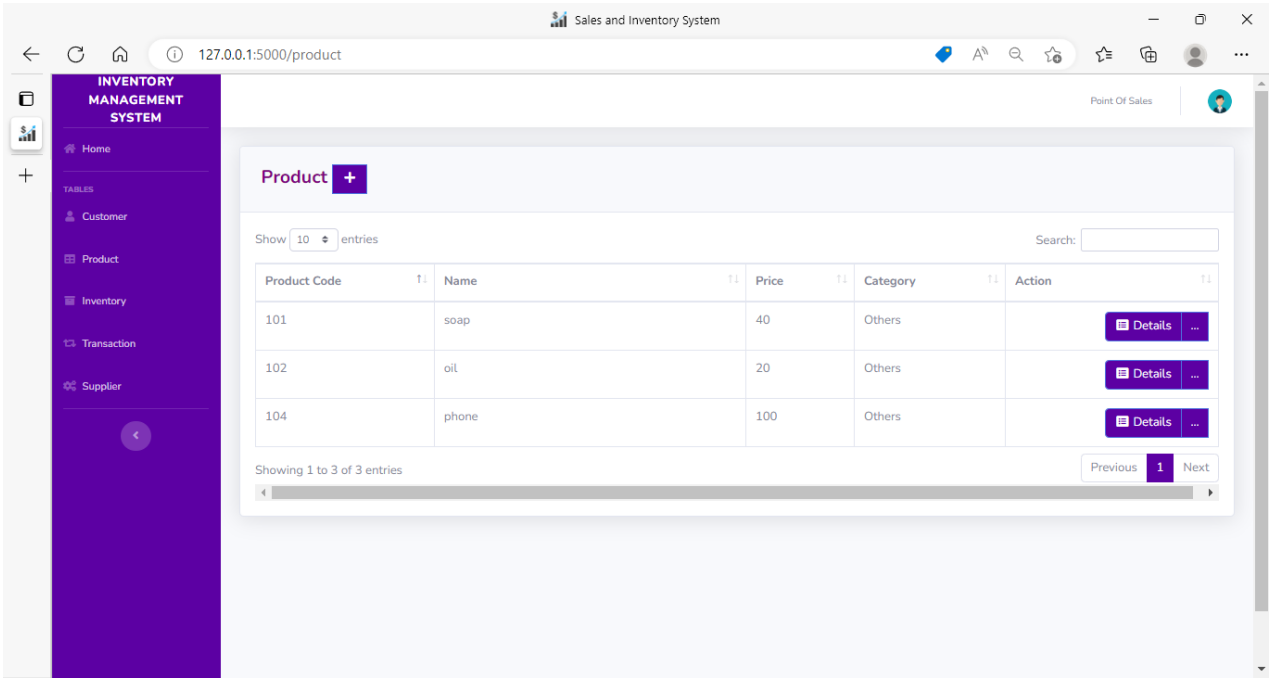
DASHBOARD



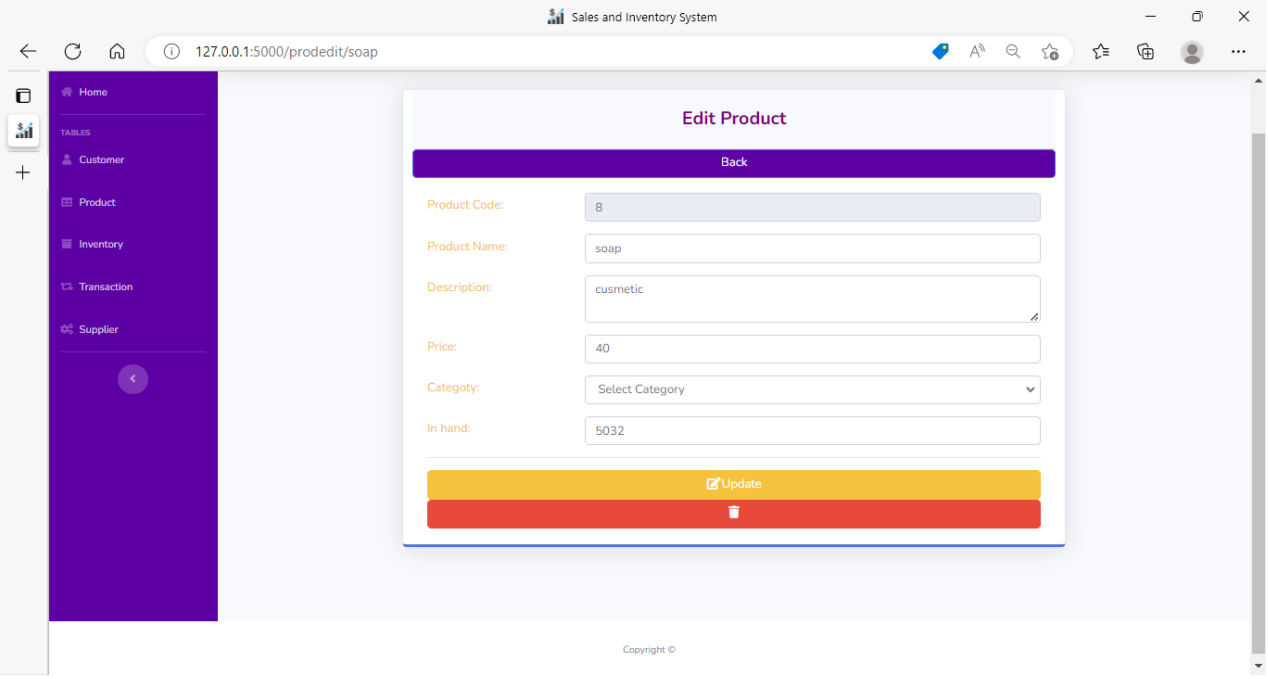
ADDSTOCK



VIEW STOCK



UPDATE AND DELETE STOCK



9.1. PERFORMANCE METRICS

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

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

• **Inventory Turns**: The number of times inventory is replaced in a year.

9. ADVANTAGES & DISADVANTAGES

ADVANTAGES

1. **It helps to maintain the right amount of stocks:** contrary to the belief that is held by some people, inventory management does not seek to reduce the amount of inventory that you have in stock, however, it seeks to maintain an equilibrium point where your inventory is working at a maximum efficiency and you do not have to have many stocks or too few stocks at hand at any particular point in time. The goal is to find that zone where you are never losing money in your inventory in either direction. With the aid of an efficient inventory management strategy, it is easy to improve the accuracy of inventory order.

2. **It leads to a more organized warehouse:** with the aid of a good inventory management system, you can easily organize your warehouse. If your warehouse is not organized, you will find it very difficult to manage your inventory. A lot of businesses choose to optimize their warehouse by putting the items that have the highest sales together in a place that is easy to access in the warehouse. This ultimately helps to speed up order fulfilment and keeps clients happy.

3. **It saves time and money:** an effective inventory management system can translate to time and money saved on the part of the business. By keeping track of the product that you already have at hand, you can save yourself the hassles of having to do an inventory recount in order to ensure your records are accurate. It also allows you to save cash that would have otherwise been spent on slow moving products.

4. **Improves efficiency and productivity:** inventory management devices like bar code scanners and inventory management software can help to greatly increase the efficiency and productivity of a business. They do this by eliminating the manual way of doing things thus allowing employees to do other more important

things for the business.

5. **A well-structured** inventory management system leads to improved customer retention: for customers to keep patronizing you, you will need to always have the goods they want, at the amount they want, and at the time they want it. Inventory management helps you to meet up this demand by allowing you to have the right products all the times so that you and your customers are never stranded.
6. **Avoid lawsuits and regulatory fines**: like mentioned previously, inventory management allows you to keep your warehouse or facility in order. If it is not kept in order, it can result in lawsuits, injury and fines associated with not following regulatory guidelines and rules. In addition, proper inventory management (including keeping records of your staff activities) helps document your actions in the event of an undesirable situation.
7. **Schedule maintenance:** once you get hold of a new appliance, you can begin to schedule routine and preventative maintenance, issue work order to your staff and track that the maintenance was actually carried out. This will help to elongate the life span of that particular asset.
8. **Reduction in holding costs**: yet another benefit of an efficient management system is that it helps to save on inventory cost. These types of cost can be large and can be detrimental to a healthy profit margin. These types of costs are financing costs, warehouse rent, warehouse staff salaries, electricity bills, security et al. The key to keeping these costs in check is to have only the amount of inventory that you need at a particular time. With an inventory management program that assists you to make good forecasts, you can avoid over stocking and thus over pay on holding costs. Furthermore, having confidence in your forecast will mean that you will not have to hold a lot of "safety stock".
9. **Flexibility:** a good inventory management strategy will allow the manager to be flexible and adapt to situations as they arise. The business world is dynamic and

often unpredictable, and the same can also be said for inventory management. There are a plethora of problems that could come up such as incorrect shipments, warehouse accidents, manufacturing issues, theft et al. It is usually not possible to foresee or predict with certainty when they could happen, but if they happen, the best case scenario will be for the manager to know at once so that he or she can rectify the issue.

10. **Increased information transparency:** a good inventory management helps to keep the flow of information transparent. This information includes when items were received, picked, packed, shipped, manufactured et al. You also get to know when you need to order more of any good, when you have too much stock or too little stock.

DISADVANTAGES

1. **Bureaucracy:** even though inventory management allows employees at every level of the company to read and manipulate company stock and product inventory, the infrastructure required to build such a system adds a layer of bureaucracy to the whole process and the business in general. In instances where inventory control is in-house, this includes the number of new hires that are not present to regulate the warehouse and facilitate transactions. In instances where the inventory management is in the hands of a third party, the cost is a subscription price and a dependence on another separate company to manage its infrastructure. No matter the choice you go for, it translates to a higher overhead cost and more layers of management between the owner and the customer. From the view point of the customer, a problem that requires senior management to handle will take a longer period of time before it will be trashed out.
2. **Impersonal touch:** another disadvantage of inventory management is a lack of personal touch. Large supply chain management systems make products more accessible across the globe and most provide customer service support in case of difficulty, but the increase in infrastructure can often mean a decrease in the personal touch that helps a company to stand out above the rest. For instance, the sales manager of a small manufacturing company that sells plumbing supplies to local plumbers can throw in an extra boxof washers or elbows at no charge to the customer without raising any alarms. This is done for the sake ofcustomer relations and often makes the customer feel like he is special. While free materials can also be provided under inventory management, processing time and paper work make obtaining the materialfeelmore like a chore for the customer or even an entitlement.
3. **Production problem:** even though inventory management can reveal to you the amount of stock you have at hand and the amount that you have sold off, it can also hide production problems that could leadto customer service disasters. Since

the management places almost all of its focus on inventory management to the detriment of quality control, broken or incorrect items that would normally be discarded are shipped along with wholesome items.

4. **Increased space is need to hold the inventory:** in order to hold inventory, you will need to have spaceso unless the goods you deal in are really small in size, then you will need a warehouse to store it. In addition, you will also need to buy shelves and racks to store your goods, forklifts to move around the stock and of course staff.

11.CONCLUSION

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

.

12.FUTURE SCOPE

• The Fourth Industrial Revolution will continue to drive technological change that will impact the way that we manage inventories.

• Successful companies will view inventory as a strategic asset, rather than an aggravating expense or an evil to be tolerated.

• Collaboration with supply chain partners, coupled with a holistic approach to supply chain management, will be key to effective inventory management.

13..APPENDIX

App.py

```
from turtle import st
from flask import Flask, render_template, request, redirect, url_for, session
import ibm_db
from array import *
from markupsafe import escape
conn =ibm_db.connect("DATABASE=bludb;HOSTNAME=0c77d6f2-5da9-
48a9-81f8-
86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud;PORT=311
98;SECURITY=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=
hgt43191;PWD=pG1XZqjaI1xcDGA8",",")

app = Flask(__name__)

def fetch(name):
    pd = []
    sql = f"SELECT * FROM product WHERE name='{escape(name)}'"
    stmt = ibm_db.exec_immediate(conn, sql)
    dict = ibm_db.fetch_both(stmt)

    while dict != False:
        pd.append(dict)
        dict = ibm_db.fetch_both(stmt)
    return(pd)

def invtcheck(name,quantity):
    pd = []
    sql = f"SELECT * FROM product WHERE EXISTS(SELECT * FROM
product WHERE NAME='{escape(name)}') "
    stmt = ibm_db.exec_immediate(conn, sql)
    dict = ibm_db.fetch_both(stmt)

    while dict != False:
        pd.append(dict)
        dict = ibm_db.fetch_both(stmt)

    if pd:
```

```
        for row in pd:
            if row['NAME']==name:
                quantity+=int(row['QTY_STOCK'])
                sql = f"UPDATE product SET QTY_STOCK ={quantity} WHERE
name = '{escape(name)}';"
                stmt = ibm_db.exec_immediate(conn, sql)
                return False

    return True

def select():
    prod3=[]
    sql1="select * from addsale"
    stmt1 = ibm_db.exec_immediate(conn, sql1)

    dictionary3 = ibm_db.fetch_both(stmt1)
    while dictionary3 != False:
        prod3.append(dictionary3)
        dictionary3 = ibm_db.fetch_both(stmt1)

    return fetchinfo(prod3)

def fetchinfo(prod3):
    pd=[]
    for row in prod3:
        pd.append(fetch(row['NAME']))
    return(pd)









@app.route("/")
def index():
    return render_template("blog/home.html")
```

```
@app.route("/home")
def home():
    return render_template("blog/home.html")


@app.route("/customer")
def customer():
    prod = []

    sql = "SELECT * FROM customer"


    stmt = ibm_db.exec_immediate(conn, sql)
    dictionary = ibm_db.fetch_both(stmt)


    while dictionary != False:
        prod.append(dictionary)
        dictionary = ibm_db.fetch_both(stmt)

    if prod:
        dictionary=[]
        return  render_template('blog/customer.html', cus= prod)
    else:
        return  render_template('blog/customer.html')

@app.route("/customer-detaile/<string:name>")
def custdetail(name):
    pd = []
    sql = f"SELECT * FROM customer WHERE
FIRST_NAME='{escape(name)}'"
    stmt = ibm_db.exec_immediate(conn, sql)
    dict = ibm_db.fetch_both(stmt)

    while dict != False:
      pd.append(dict)
      dict = ibm_db.fetch_both(stmt)
```

```
    if pd:
        dict=[]
        return render_template("blog/customer-detaile.html",prd=pd)
    else:
        dict=[]
        return  render_template('blog/customer-detaile.html')


@app.route("/product")
def product():
    prod = []
    prod2 = []

    sql = "SELECT * FROM product"
    sql2 = "SELECT DISTINCT CATEGORY_ID FROM product;"

    stmt = ibm_db.exec_immediate(conn, sql)
    stmt2 = ibm_db.exec_immediate(conn, sql2)
    dictionary = ibm_db.fetch_both(stmt)
    dictionary2 = ibm_db.fetch_both(stmt2)


    while dictionary != False:
        prod.append(dictionary)
        dictionary = ibm_db.fetch_both(stmt)

    while dictionary2 != False:
        prod2.append(dictionary2)
        dictionary2 = ibm_db.fetch_both(stmt2)

    if prod:
        dictionary=[]
        return  render_template('blog/product.html', product1=
prod,product2=prod2)
    else:
        return  render_template('blog/product.html')
```

```
@app.route("/productdetail/<string:name>")
def productdetail(name):
    pd = []
    sql = f"SELECT * FROM product WHERE name='{escape(name)}'"
    stmt = ibm_db.exec_immediate(conn, sql)
    dict = ibm_db.fetch_both(stmt)

    while dict != False:
      pd.append(dict)
      dict = ibm_db.fetch_both(stmt)


    if pd:
        dict=[]
        return render_template("blog/product-detial.html",prd=pd)
    else:
        dict=[]
        return  render_template('blog/product-detial.html')

@app.route("/produp/<string:name>",methods = ['POST', 'GET'])
def produp(name):
    if request.method == 'POST':
        prodcode= request.form['prodcode']
        name1= request.form['name']
        description = request.form['description']
        price =request.form['price']
        category =request.form['category']
        inhand =request.form['inhand']
        sql=f"UPDATE product SET product_code ='{prodcode}', name =
'{escape(name1)}', description='{escape(description)}',
price='{escape(price)}',
category_id='{escape(category)}',qty_stock='{inhand}'  WHERE name =
'{escape(name)}';"
        stmt = ibm_db.exec_immediate(conn, sql)
        return product()

    else:
        return product()
```

```
@app.route("/viewinvet")
def view():
    return render_template('blog/inventory-view.html')
```

```
@app.route("/prodedit/<string:name>")
def prodedit(name):
    pd=fetch(name)

    if pd:
        return  render_template('blog/prodedit.html',prd=pd)
    else:
        return  render_template('blog/prodedit.html')
```

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

```
def addcustomer():
    if request.method == 'POST':
        fn= request.form['firstname']
        ln= request.form['lastname']
        ph = request.form['phonenumber']
        insert_sql='INSERT INTO
customer(CUST_ID,FIRST_NAME,LAST_NAME,PHONE_NUMBER)VAL
UES(?,?,?,?)'
        prep_stmt=ibm_db.prepare(conn, insert_sql)
        ibm_db.bind_param(prep_stmt, 1, 1)
        ibm_db.bind_param(prep_stmt, 2, fn)
        ibm_db.bind_param(prep_stmt, 3, ln)
        ibm_db.bind_param(prep_stmt, 4, ph)
        ibm_db.execute(prep_stmt)

    return customer()

@app.route('/addproduct',methods = ['POST', 'GET'])
def addproduct():
    if request.method == 'POST':
        prodcode= request.form['prodcode']
        name= request.form['name']
        description = request.form['description']
        quantity = request.form['quantity']
        onhand = request.form['onhand']
        price =request.form['price']
        category =request.form['category']
        supplier =request.form['supplier']
        datestock =request.form['datestock']
        if invtcheck(name,int(quantity)):
            insert='INSERT INTO product (PRODUCT_CODE, NAME,
DESCRIPTION, QTY_STOCK, ON_HAND, PRICE, CATEGORY_ID,
SUPPLIER_ID, DATE_STOCK_IN)VALUES(?,?,?,?,?,?,?,?,?)'
            prep_stm=ibm_db.prepare(conn, insert)
            ibm_db.bind_param(prep_stm, 1, prodcode)
            ibm_db.bind_param(prep_stm, 2, name)
            ibm_db.bind_param(prep_stm, 3, description)
            ibm_db.bind_param(prep_stm, 4, quantity)
            ibm_db.bind_param(prep_stm, 5, onhand)
            ibm_db.bind_param(prep_stm, 6, price)
            ibm_db.bind_param(prep_stm, 7, category)
```

```
        ibm_db.bind_param(prep_stm, 8, supplier)
        ibm_db.bind_param(prep_stm, 9, datestock)
        ibm_db.execute(prep_stm)



    return product()



@app.route("/inventory")
def inventrory():
    invet = []
    sql = "SELECT * FROM product;"

    stmt = ibm_db.exec_immediate(conn, sql)
    dictionary = ibm_db.fetch_both(stmt)
    while dictionary != False:
        invet.append(dictionary)
        dictionary = ibm_db.fetch_both(stmt)



    if invet:
        dictionary=[]
        return  render_template('blog/inventory.html', invet0= invet,)
    else:
        return  render_template('blog/inventory.html')

@app.route("/inventory/<string:name>")
def invetshow(name):
    pd= fetch(name)
    if pd:
        return  render_template('blog/inventory-view.html',prd=pd,name1=name)
    else:
        return  render_template('blog/inventory-view.html')



@app.route("/transaction")
def transaction():
```

```
    return  render_template('blog/transaction.html')

@app.route("/supplierup/<string:name>",methods = ['POST', 'GET'])
def supplierup(name):
    if request.method == 'POST':
        sname= request.form['name']
        scity= request.form['city']
        sphno = request.form['phone']
        insert_sql=f"UPDATE SUPPLIER SET
COMPANY_NAME='{escape(sname)}', CITY= '{escape(scity)}',
PHONE_NUMBER='{escape(sphno)}'  WHERE COMPANY_NAME =
'{escape(name)}';"
        prep_stmt=ibm_db.prepare(conn, insert_sql)

        ibm_db.execute(prep_stmt)
    return supplier()

@app.route("/custup/<string:name>",methods = ['POST', 'GET'])
def custup(name):
    if request.method == 'POST':
        fn= request.form['firstname']
        ln= request.form['lastname']
        ph = request.form['phone']
        insert_sql=f"UPDATE customer SET FIRST_NAME='{escape(fn)}',
LAST_NAME= '{escape(ln)}', PHONE_NUMBER='{escape(ph)}'  WHERE
FIRST_NAME = '{escape(name)}';"
        prep_stmt=ibm_db.prepare(conn, insert_sql)

        ibm_db.execute(prep_stmt)
    return customer()


@app.route("/custdel/<string:name>")
def cdel(name):
    if request.method == 'POST':
        insert_sql=f"Delete from customer SET FIRST_NAME='{escape(fn)}',
LAST_NAME= '{escape(ln)}', PHONE_NUMBER='{escape(ph)}'  WHERE
FIRST_NAME = '{escape(name)}';"
        prep_stmt=ibm_db.prepare(conn, insert_sql)

        ibm_db.execute(prep_stmt)
```

```
    return customer()


@app.route("/supplier")


def supplier():

    supp = []
    sql = "SELECT * FROM SUPPLIER;"

    stmt = ibm_db.exec_immediate(conn, sql)
    dictionary = ibm_db.fetch_both(stmt)
    while dictionary != False:
        supp.append(dictionary)
        dictionary = ibm_db.fetch_both(stmt)
    return  render_template('blog/supplier.html',supply=supp)


@app.route("/supplieredit/<string:name>")
def supplieredit(name):
    pd=fetch(name)
    sql = f"SELECT * FROM SUPPLIER WHERE
COMPANY_NAME='{escape(name)}'"
    stmt = ibm_db.exec_immediate(conn, sql)
    dict = ibm_db.fetch_both(stmt)

    while dict != False:
        pd.append(dict)
        dict = ibm_db.fetch_both(stmt)

    if pd:
        return  render_template('blog/suppileredit.html',prd=pd)
    else:
        return  render_template('blog/suppileredit.html')



@app.route("/suplierdetails/<string:name>")
def supplydetail(name):
```

```
    pd = []

    sql = f"SELECT * FROM supplier WHERE
COMPANY_NAME='{escape(name)}'"

    stmt = ibm_db.exec_immediate(conn, sql)
    dict = ibm_db.fetch_both(stmt)




    while dict != False:
      pd.append(dict)
      dict = ibm_db.fetch_both(stmt)




    if pd:
        dict=[]
        return render_template("blog/suplierdetails.html",prd=pd)
    else:
        dict=[]
        return  render_template('blog/suplierdetails.html')


@app.route("/account")
def account():
    return  render_template('blog/account.html')

@app.route("/sales")
def sales():
    prod2 = []
    prod1 = []


    sql = "SELECT DISTINCT name FROM product;"
    stmt = ibm_db.exec_immediate(conn, sql)

    dictionary = ibm_db.fetch_both(stmt)
```

```
    while dictionary != False:
        prod1.append(dictionary)
        dictionary = ibm_db.fetch_both(stmt)



    sql1= "SELECT DISTINCT FIRST_NAME FROM CUSTOMER;"
    stmt1 = ibm_db.exec_immediate(conn, sql1)
    dictionary1 = ibm_db.fetch_both(stmt1)
    while dictionary1 != False:
        prod2.append(dictionary1)
        dictionary1 = ibm_db.fetch_both(stmt1)



    return
render_template('blog/sales.html',cname=prod2,detail=prod1,table=select())

@app.route("/addsale",methods = ['POST', 'GET'])
def addsale():
    if request.method == 'POST':
        prod3=[]
        pname= request.form['name']
        quan= request.form['quantity']
        sql3=f"INSERT INTO addsale(NAME, QUNATITY) VALUES(?,?)"
        prep_stm=ibm_db.prepare(conn, sql3)
        ibm_db.bind_param(prep_stm, 1, pname)
        ibm_db.bind_param(prep_stm, 2, quan)
        ibm_db.execute(prep_stm)
        return sales()

@app.route("/sales/<string:name>",methods = ['POST', 'GET'])
def bill(name):
    # prod2 = []
    # prod1 = []
    # sql2 = "SELECT * FROM CATEGORY;"
    # stmt2 = ibm_db.exec_immediate(conn, sql2)
    # dictionary2 = ibm_db.fetch_both(stmt2)

    # while dictionary2 != False:
    #     prod2.append(dictionary2)
```

```
    #     dictionary2 = ibm_db.fetch_both(stmt2)

    # sql = "SELECT * FROM product;"
    # stmt = ibm_db.exec_immediate(conn, sql)
    # dictionary = ibm_db.fetch_both(stmt)

    # while dictionary != False:
    #     prod1.append(dictionary)
    #     dictionary = ibm_db.fetch_both(stmt)

    pd= fetch(name)
    if pd:
        return
render_template('blog/sales.html',bill=pd,invet1=prod2,detail=prod1)
    else:
        return render_template('blog/sales.html')


if __name__ == '__main__':
    app.run(host='0.0.0.0', port=5000, debug=True)
```

GITHUB:

https://github.com/IBM-EPBL/IBM-Project-52068-1660988777

DEMOLINK:

https://vimeo.com/772997008