RETAIL STORE STOCK INVENTORY ANALYTICS

NALAIYA THIRAN PROJECT REPORT

IBM-Project-15021-1659593378

TEAM ID : PNT2022TMID08352

Submitted by

PRAVEENKUMAR A	**(810419104088)**
NAVANEETHAKRISHNAN S	**(810419104068)**
RAHUL G	**(810419104092)**
SARAVANAN G	**(810419104099)**

in partial fulfillment for the award of the degree

of

BATCHELOR OF ENGINNERING

IN

COMPUTER SCIENCE AND ENGINEERING

DHANALAKSHMI SRINIVASAN ENGINEERING COLLEGE
(AUTONOMOUS)
PERAMBALUR
621212

TABLE OF CONTENTS

1. INTRODUCTION

Project Overview:

Project is based on Retail Store Stock Inventory analytics which isused to supply the stocks for shops based on their needs .By managing inventory, retailers meet customer demand without running out of stock orcarrying excess supply.

Inventory management is vital for retailers because the practice helps themincrease profits. They are more likely to have enough inventory to capture every possible sale while avoiding overstock and minimizing expenses.

Purpose:

Purpose of retail store stock analysis is to find the necessary stockrequired for to supply customer when there are in need of , the shop holder view the stock,price and sale in form dashboard,report and story in webpage which helps them to track regularly the status of their stock availability.

It helps in managing the current stock levels, ordered items and products as well as ones already sold. It provides a constant supply of products to fulfill customer demand. It allows customer retention. Customers convertinto loyal customers by handling stock levels.

2. LITERATURE SURVEY

Existing problem:

Existing system consist of methods using ABC Analysis and Min-Max Analysis. In the Data Mart, the search for goods classes per sub category iscarried out using the ABC Analysis calculation method. Furthermore, in the Data Mart, the search for maximum and minimum stock values is based onthe Min-Max Analysis calculation method. The resulting maximum and minimum grade and stock values are then implemented into the goods

data table in the retail management information system database. The laststage is to arrange the order amount that is allowed in the order module in the retail management information system. Rules that are made based onthe class of goods along with the minimum and maximum stock values.

References:

The following are the references used:

[1] H S Sugiarto and H T Saksono 2016 Scheduling System on Goods OrderAt PT XYZ Using Economic Order Quantity Method The Third InternationalConference on Entrepreneurship.

[2] K E Fu and P Apichotwasurat 2013 Application of Economic Order Quantity on Production Scheduling and Control System for a Small Company. Proceedings of the Institute of Industrial Engineers Asian Conference 2013.

[3] M Rusănescu 2014 Abc Analysis , Model for Classifying Inventory HIDRAULICA.

[4] D Dhoka and Y L Choudary 2013 ABC Classification for Inventory Optimization IOSR J Bus Manage.

[5] Funaki, K.., "Strategies safety stock placement in supply chain design with due-date based demand," International Journal of Production Economics, vol. 135, pp 4-13, 2012. [6] Grewal, CS, Enns, ST, and Rogers, P., "Dynamic reorder point replenishment strategies for a capacitated supply chain with seasonal demand," Computer, and industrial engineering, vol.
.80, pp 97-110, 2015.

[7] Indrajit, RE, and Djokopranoto, R., "General merchandise and inventory management of spare parts for maintenance, repair and operation", Yogyakarta: Grasindo, 2014.

[8] Mebarki, N. and Shahzad, A., "Correlation among tardiness based

measures for priority scheduling using dispatching rules" Month, pp 1- 14, 2012.

Problem Statement Definition:

The shop holder sell the product as requested by the consumer at the time when the stock is not available it will reduce customer satisfaction and it isalso a loss for the seller.

In order to overcome this drawback we using visualization to represent stocks availability and sale.Therefore quality and quantity of product can be delivered without any demand.

Problem Statement (PS)	**I am (Customer)**	**I'm trying to**	**But**	**Because**	**Which makes me feel**
PS-1	A Shopaholic	Buy my favourite products	Most of the products are not available in store	But the demand for the product is high	Unsatisfied
PS-2	A Foodie	Buy my favourite chocolate	It is not available in any stores	It is imported chocolates	Frustrated

Problem Statement-1:

Problem Statement-2:

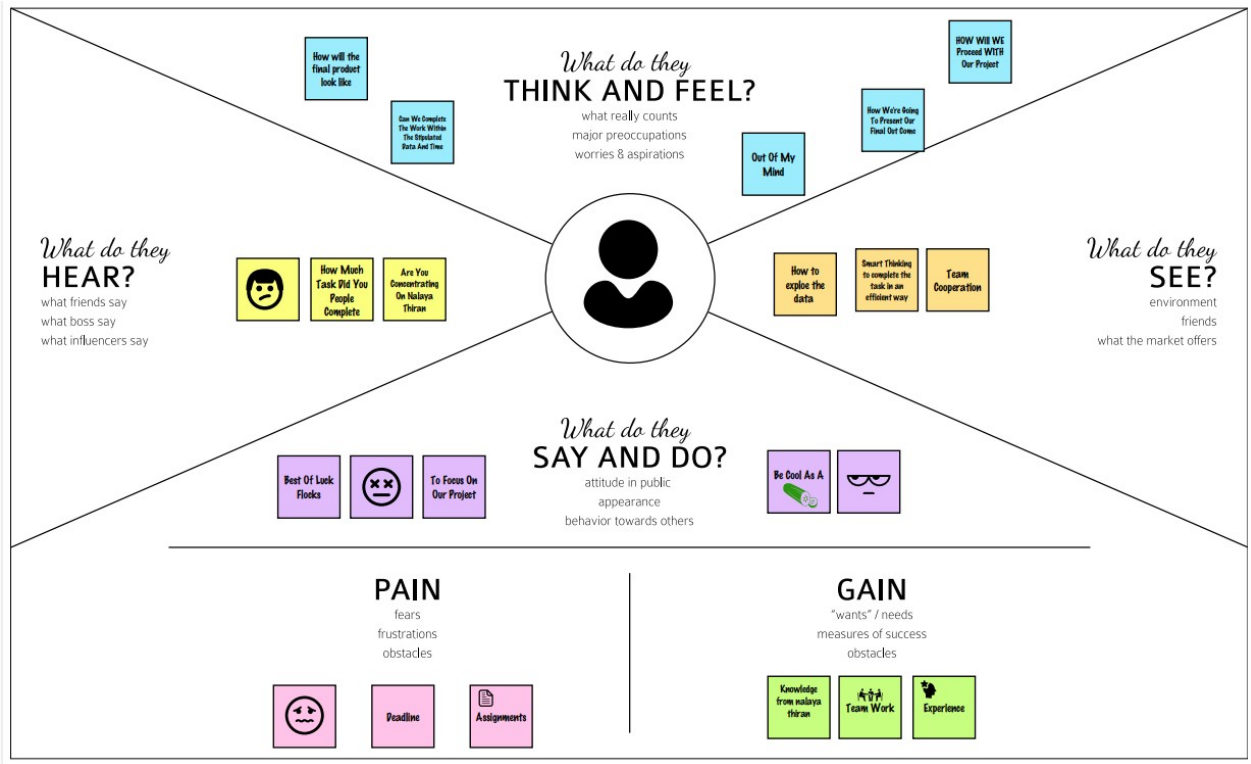
3. IDEATION & PROPOSED SOLUTION

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.

An Empathy Map consists of four quadrants. The four quadrants reflect four key traits, which the user demonstrated/possessed during the observation/research stage. The four quadrants refer to what the user:

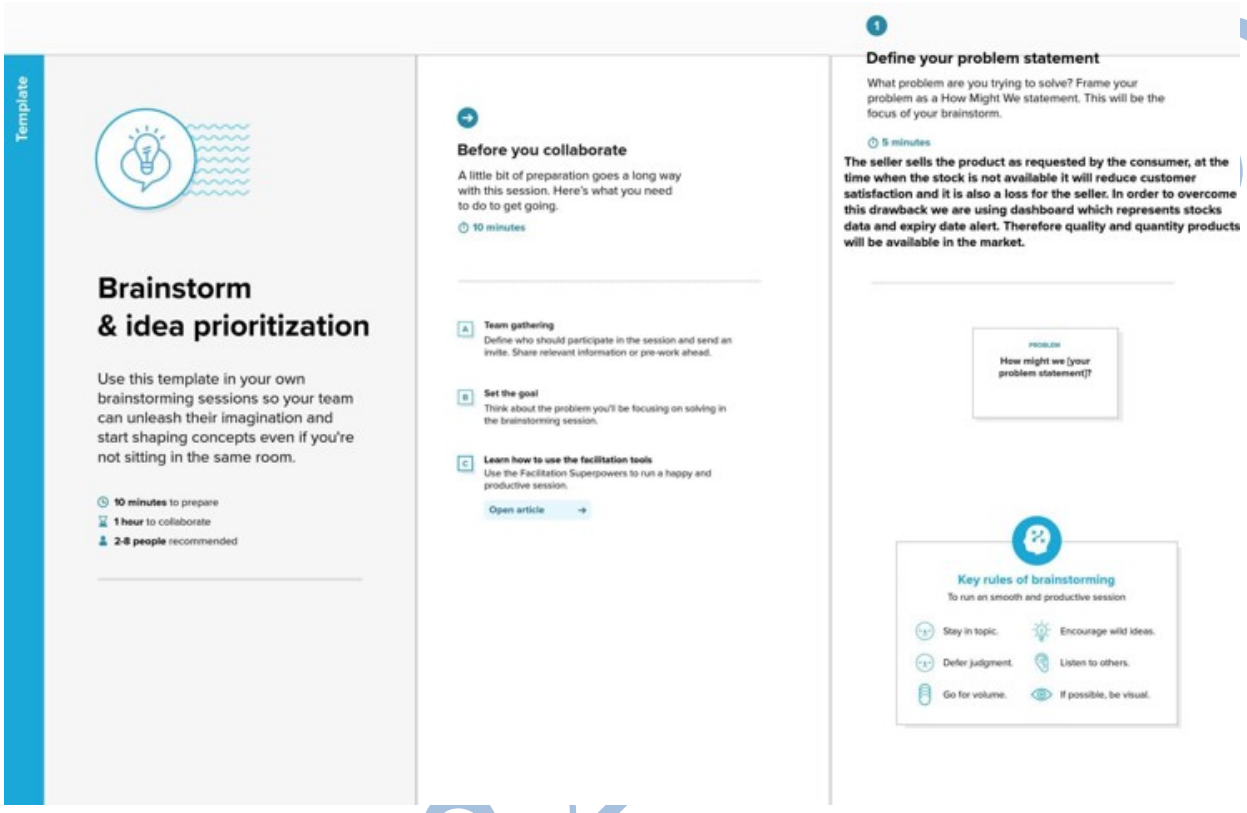
Said, Did, Thought, and Felt. It's fairly easy to determine what the user saidand did.



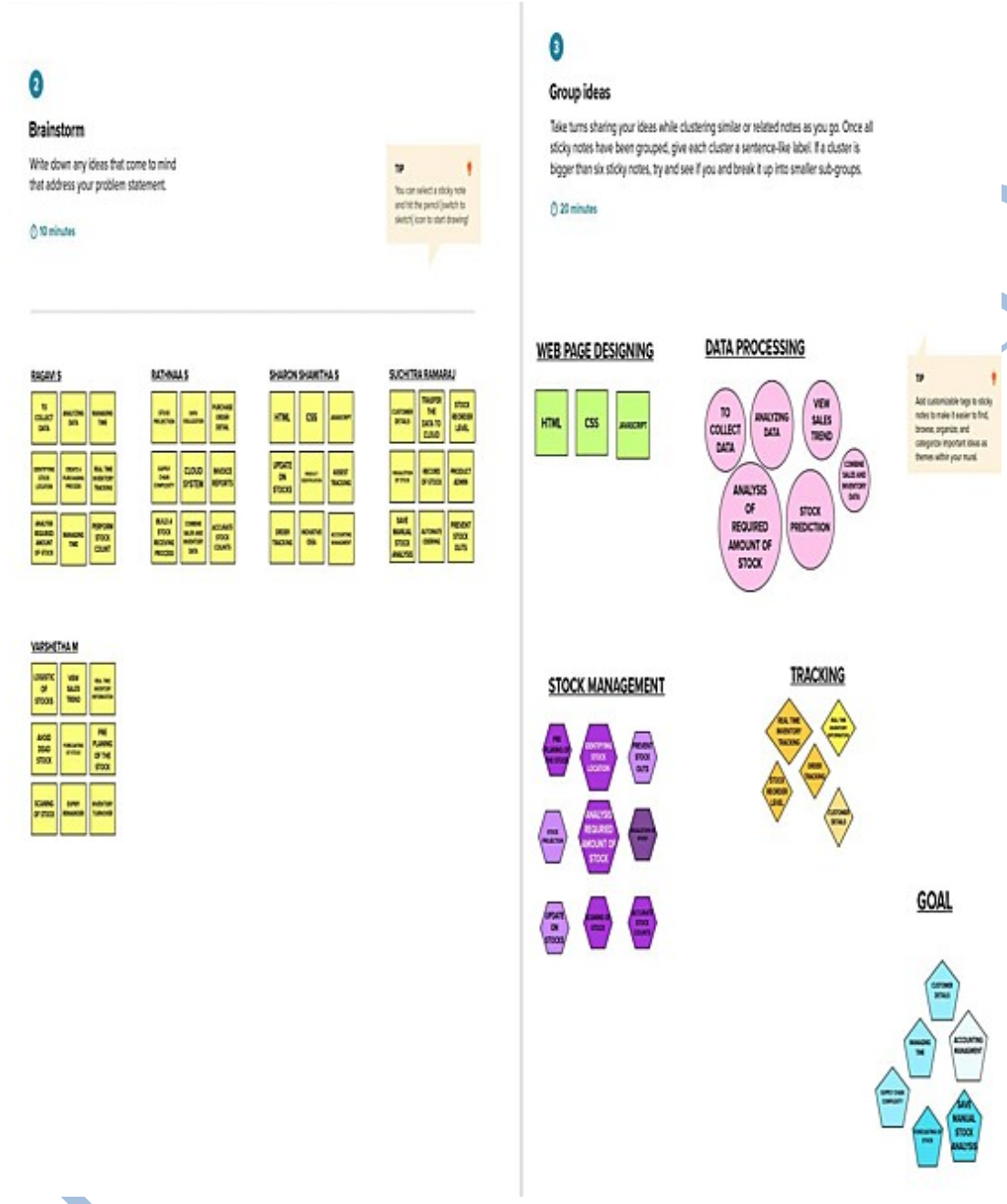
Ideation & Brainstorming:

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 amount of creative solutions.

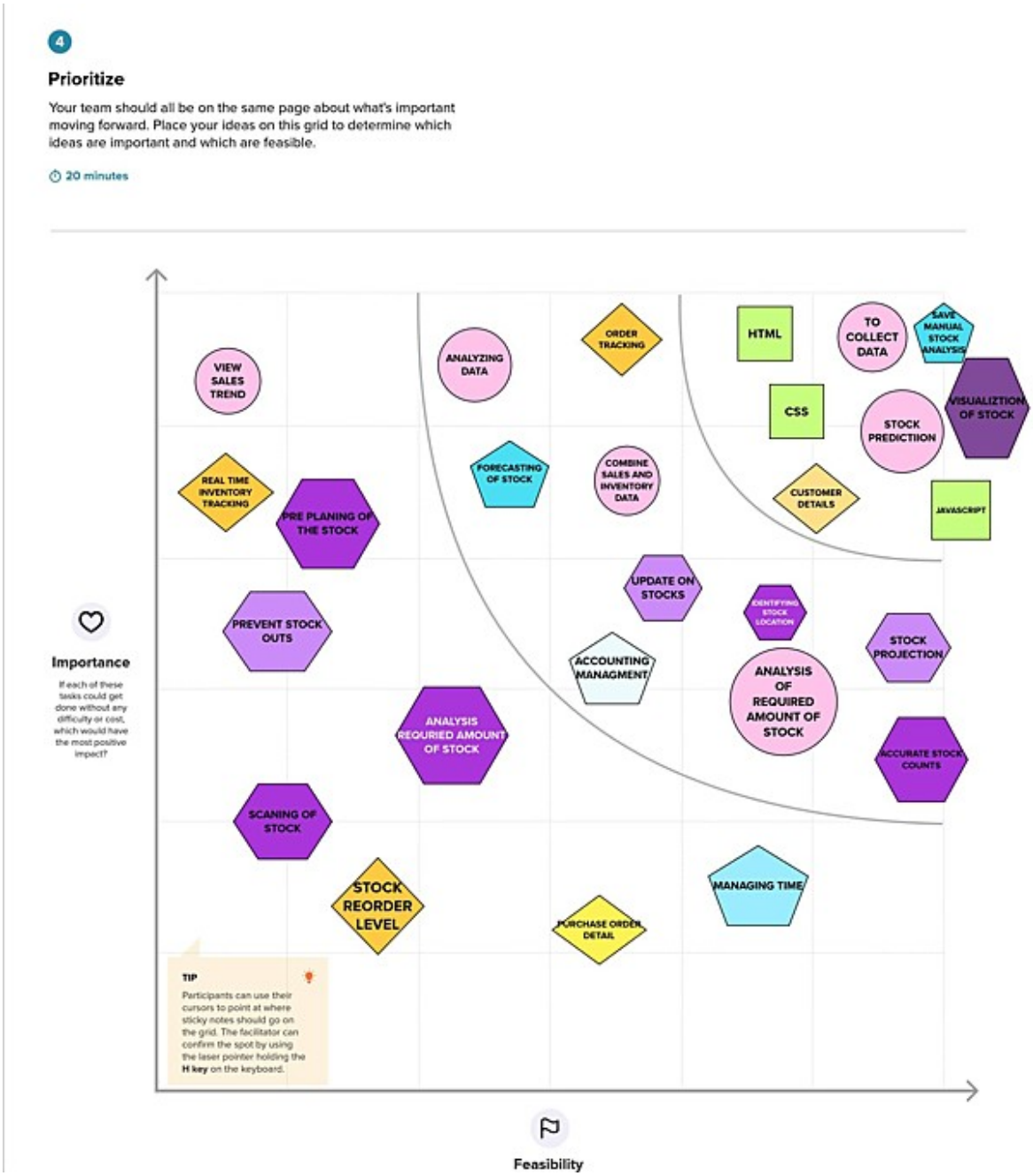
Step-1: Team Gathering, Collaboration and Select the Problem Statement



Step-2: Brainstorm, Idea Listing and Grouping



Step-3: Idea Prioritization



Proposed Solution:

Proposed Solution means the technical solution to be provided by the Implementation agency in response to the requirements and the objectives of the Project.

The main goal of presenting a business proposal is to provide solution to a problem faced by a potential buyer. This section should be as comprehensive as possible, and able to address all the needs that you havepointed in the first section.

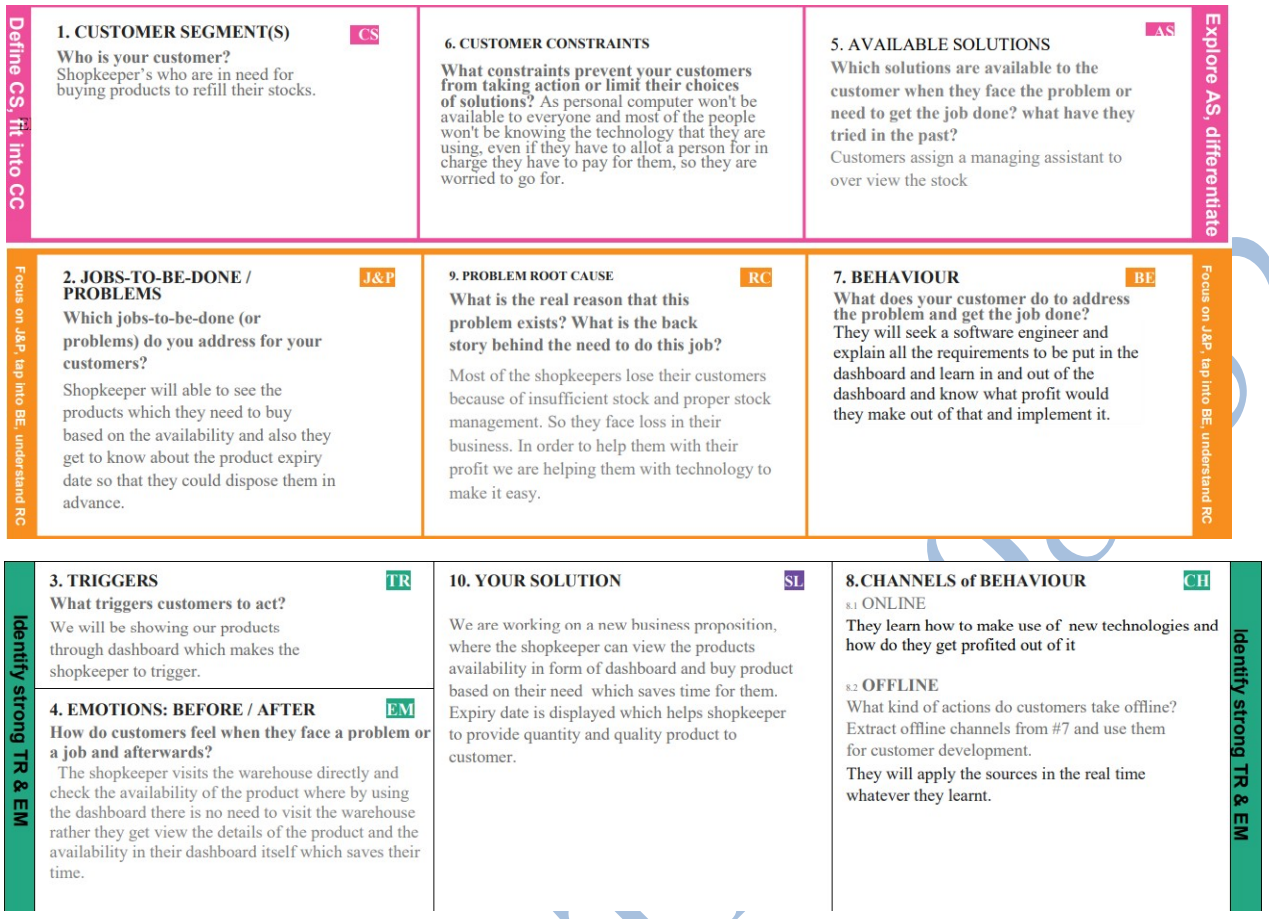
S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Customers gets disappointed when the store does not meets the satisfaction level of them.
2.	Idea / Solution description	Using dashboard it would become easy for the store to keep a track on their stock, sothat they can meet customer's satisfaction level.
3.	Novelty / Uniqueness	Expiry alert of the product will be given.
4.	Social Impact / Customer Satisfaction	Quality and Quantity of the product can maintained to the best, and customer's will have a heart full feeling while leaving the store.
5.	Business Model (Revenue Model)	Using this method the companywill have reputed customers and stocks will be delivered on time,so there is no need of last minute hassle.

6.	Scalability of the Solution	When your inventory is hard to identify or locate in the warehouse, it leads to incomplete, inaccurate or delayed shipments. Receiving and finding the right stock is vital to efficient warehouse operations and provides a positive customer experiences.

Problem Solution fit:

Problem-Solution Fit simply means that you have found a problem with your customer and that the solution you have realized for it actually solves the customer's problem.The Problem-Solution Fit is an important step towards the Product-Market Fit, but often an underestimated one.

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.



4. REQUIREMENT ANALYSIS

Functional requirement:

Functional requirements may involve calculations, technical details, data manipulation and processing, and other specific functionality that define what a system is supposed to accomplish. Behavioral requirements describe all the cases where the system uses the functional requirements, these are captured in use cases.

Functional requirements drive the application architecture of a system, while non-functional requirements drive the technical architecture of a system.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through webpage
FR-2	User Login	Login through webpage
FR-3	User Stock List	View in the webpage
FR-4	Sales List	View in the webpage
FR-5	Revenue Detail	View in the webpage

Non-Functional requirements:

Non-functional requirements are often mistakenly called the "quality attributes" of a system, however there is a distinction between the two. Non-functional requirements are the criteria for evaluating how a softwaresystem should perform and a software system must have certain quality attributes in order to meet non-functional requirements.

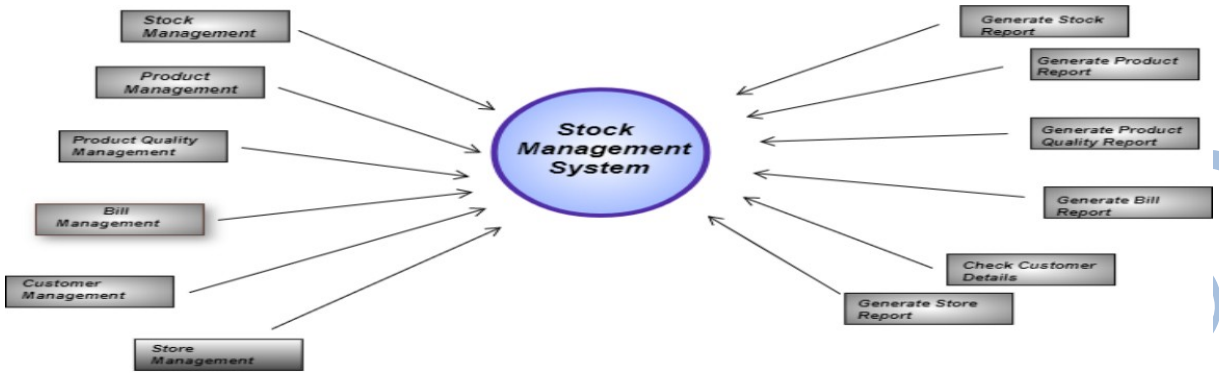
FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	Clear view about Stock Details which provides benefit to the users.
NFR-2	**Security**	Only authorized users can log in toview that provide security to the users.

NFR-3	**Reliability**	Information about one store cannot be viewed by the other store users.
NFR-4	**Performance**	Available and required amount of stock can be viewed in dashboard by visualization hence the user can make decision according to it.
NFR-5	**Availability**	Visualization shows the stock availability and the products which need to be refilled can be viewed that prevent user from last minute shortage.
NFR-6	**Scalability**	Product expiry date can be viewed which helps the user to sell those product in prior that provides benefits to the shop owner.

5. PROJECT DESIGN

Data Flow Diagrams:

A data-flow diagram is a way of representing a flow of data through a process or a system. The DFD also provides information about the outputs and inputs of each entity and the process itself. A data-flow diagram has no control flow — there are no decision rules and no loops.



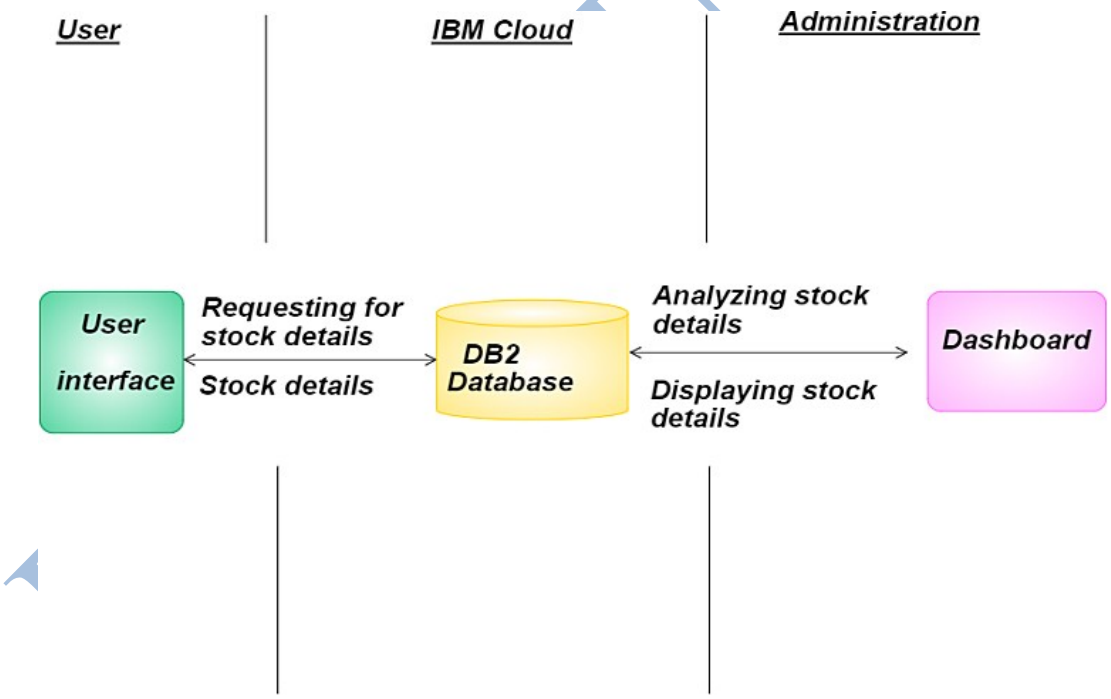
FIRST LEVEL DATA FLOW DIAGRAM

Solution & Technical Architecture:

Solution architecture is a complex process – with many sub-processes – that bridges the gap between business problems and technology solutions. Its goals are to:

- Find the best tech solution to solve existing business problems.
- Describe the structure, characteristics, behavior, and other aspects of the software to project stakeholders.
- Define features, development phases, and solution requirements.
- Provide specifications according to which the solution is defined, managed, and delivered.

Technical Architecture (TA) is a form of IT architecture that is used to design computer systems. It involves the development of a technical blueprint with regard to the arrangement, interaction, and interdependenceof all elements so that system-relevant requirements are met.



User Stories:

A user story is an informal, general explanation of a software featurewritten 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 valueback to the customer. In software development and product management, a user story is an informal, natural language description of features of a software system.

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	Medium	Suchitra Ramaraj, Ragavi.S
Sprint-1	Login	USN-2	As a user, I can log into the application by entering username& password.	2	Medium	Varshetha.M, Sharon Shamitha.S
Sprint-2	Data Upload	USN-3	As a user, I can upload my data so that I can have a visual representation of it.	1	Low	Ragavi.S
Sprint-2	Dashboard Creation	USN-4	As a user, I can view the visual representation Of my data in dashboard.	3	High	Rathnaa.S, Suchitra Ramaraj
Sprint-3	Report Creation	USN-5	As a user, I can view the visual representation Of my data in report.	3	High	Varshetha.M, Sharon Shamitha.S
Sprint-4	Story Creation	USN-6	As a user, I can view the visual representation Of my data in story.	3	High	Ragavi.S, Rathnaa.S

6. PROJECT PLANNING & SCHEDULING

Sprint Planning & Estimation:

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

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	30 Oct 2022	20	30 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	06 Nov 2022	20	06 Nov 2022
Sprint-3	20	6 Days	06 Nov 2022	13 Nov 2022	20	13 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

Sprint Delivery Schedule:

In Scrum project sprint delivery schedule is used to estimate when sprint has started and delivery date of the sprint.Due to estimation of the sprint delivery schedule it helps the developer to complete their project within theestimated time.

Reports from JIRA:

The reports in jira has been denoted below:

BACKLOG:

Backlog is usually a list of issues describing what your team is going to do on a project. It's a convenient place for creating, storing, and managing several kinds of issues: issues that you're currently working on (you can also see them on the board and in the current sprint if you're using a Scrumproject).

7. CODING &SCREENSHOT

Code:

```
from flask import Flask, render_template, flash, request, session,send_file
from flask import render_template, redirect, url_for, request
import datetime
import mysql.connector
import sys
app = Flask(__name__)
app.config['DEBUG']
app.config['SECRET_KEY'] = '7d441f27d441f27567d441f2b6176a'
@app.route("/")
def homepage():
    import os, shutil
    folder = 'static/plott'
    for filename in os.listdir(folder):
        file_path = os.path.join(folder, filename)
        try:
            if os.path.isfile(file_path) or os.path.islink(file_path):
                os.unlink(file_path)
            elif os.path.isdir(file_path):
                shutil.rmtree(file_path)
```

```
        except Exception as e:
            print('Failed to delete %s. Reason: %s' % (file_path, e))

    return render_template('index.html')
@app.route("/ViewData")
def ViewData():
    conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1Medicalddb')
    cur = conn.cursor()
    cur.execute("SELECT * FROM salestb ")
    data = cur.fetchall()
   return render_template('ViewData.html',data=data)
@app.route("/excelpost", methods=['GET', 'POST'])
def uploadassign():
    if request.method == 'POST':
        conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1Medicalddb')
        cursor = conn.cursor()
        cursor.execute("truncate table salestb ")
        conn.commit()
        conn.close()
        file = request.files['fileupload']
        file_extension = file.filename.split('.')[1]
        print(file_extension)
        #file.save("static/upload/" + secure_filename(file.filename))
        import pandas as pd
        import matplotlib.pyplot as plt
        df = ''
        if file_extension == 'xlsx':
            df = pd.read_excel(file.read(), engine='openpyxl')
        elif file_extension == 'xls':
            df = pd.read_excel(file.read())
        elif file_extension == 'csv':
            df = pd.read_csv(file)
        print(df)
        print("Preprocessing Completed")
        print(df)
        conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1Medicalddb')
        cursor = conn.cursor()
        for row in df.itertuples():
```

```
        cursor.execute(" INSERT INTO salestb VALUES ('"+ row.Month +"','"+
row.Customer+"','"+ row.Period +"','"+row.Product +"','"+ row.Location +"','"+
row.SalesRep +"','"+ row.Supplier+"','"+ row.WarehouseLocations +"','"+ str(row.Actual) +
"','"+str(row.CSales)+"','"+ str(row.InventoryStock)+"','"+ str(row.LSales)+"','"+
str(row.MSales) +"','"+str(row.NumberofRecords) + "','"+str(row.ReceivedInventory) +"','"+
str(row.RepSales) +"','"+str(row.Target) +"' )")
    conn.commit()
    conn.close()
    conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1Medicalddb')
    cur = conn.cursor()
    cur.execute("SELECT * FROM salestb ")
    data = cur.fetchall()
    return render_template('ViewData.html', data=data)
@app.route("/Customer")
def Customer():
  conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1Medicalddb')
  # cursor = conn.cursor()
  cur = conn.cursor()
  cur.execute(
    "SELECT  distinct Customer  FROM salestb ")
  customer = cur.fetchall()
  #print(coorname)

  return render_template('Customer.html', customer=customer)



@app.route("/csearch", methods=['GET', 'POST'])
def csearch():
  if request.method == 'POST':
    cname = request.form['Customer']
    import matplotlib.pyplot as plt
    import matplotlib
    matplotlib.use('Agg')
    conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1Medicalddb')
    mycursor = conn.cursor()
    # Fecthing Data From mysql to my python progame
    mycursor.execute("select Month, sum(CSales) as CSales from salestb where
```

```
Customer='"+ cname +"' group by Month")
        result = mycursor.fetchall
        Month = []
        CSales = []
        Month.clear()
        CSales.clear()
        for i in mycursor:
          Month.append(i[0])
            CSales.append(i[1])
        print("Month = ", Month)
        print("Total Sales = ", CSales)
        # Visulizing Data using Matplotlib
        plt.figure(figsize=(12, 10))
        plt.bar(Month, CSales, color=['black', 'red', 'green', 'blue', 'cyan'])
        #plt.ylim(0, 5)
        ax = plt.gca()
        plt.draw()
        ax.tick_params(axis='x', rotation=70)
        plt.xlabel("Month",fontsize=5)
        plt.ylabel("Total Sales")
        plt.title("Customer Sales")
        import random
        n = random.randint(1111, 9999)
        plt.savefig('static/plott/' + str(n) + '.jpg')
        iimg = 'static/plott/' +str(n)+ '.jpg'


    conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1Medicalddb')
        cur = conn.cursor()
        cur.execute("SELECT * FROM salestb where Customer='"+ cname +"' ")
        data = cur.fetchall()
        conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1Medicalddb')
        # cursor = conn.cursor()
        cur = conn.cursor()
        cur.execute(
            "SELECT  distinct Customer  FROM salestb ")
        customer = cur.fetchall()
        return render_template('Customer.html', data=data,dataimg=iimg,customer=customer)
@app.route("/Location")
```

```
def Location():
  conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1Medicalddb')
   # cursor = conn.cursor()
   cur = conn.cursor()
   cur.execute(
      "SELECT  distinct Location  FROM salestb ")
   location = cur.fetchall()
   #print(coorname)
   return render_template('Location.html', locat=location)
@app.route("/lsearch", methods=['GET', 'POST'])
def lsearch():
   if request.method == 'POST':
      lllocation = request.form['loc']
      import matplotlib.pyplot as plt
      import matplotlib
      matplotlib.use('Agg')
      conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1Medicalddb')
      mycursor = conn.cursor()
      mycursor.execute("select Month, sum(MSales) as MSales from salestb where
Location='"+ lllocation +"' group by Month")
      result = mycursor.fetchall
      Month = []
      MSales = []
      Month.clear()
      MSales.clear()
      for i in mycursor:
         Month.append(i[0])
         MSales.append(i[1])
      print("Month = ", Month)
      print("Total Sales = ", MSales)
      # Visulizing Data using Matplotlib
      plt.figure(figsize=(12, 10))
      plt.bar(Month, MSales, color=['yellow', 'red', 'green', 'blue', 'cyan'])
      #plt.ylim(0, 5)
      ax = plt.gca()
      plt.draw()
      ax.tick_params(axis='x', rotation=70)
      plt.xlabel("Month")
      plt.ylabel("Total Sales")
```

```
    plt.title("Sales By Location")
    import random
    n = random.randint(1111, 9999)
    plt.savefig('static/plott/'+str(n)+'.jpg')
    iimg = 'static/plott/'+str(n)+'.jpg'
    conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1Medicalddb')
    cur = conn.cursor()
    cur.execute("SELECT * FROM salestb where Location='"+ lllocation +"' ")
    data = cur.fetchall()
    conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1Medicalddb')
    # cursor = conn.cursor()
    cur = conn.cursor()
    cur.execute(
        "SELECT  distinct location  FROM salestb ")
    locati = cur.fetchall()
    return render_template('Location.html', data=data, dataimg=iimg, locat=locati)
@app.route("/Sales")
def Sales():
  conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1Medicalddb')
  # cursor = conn.cursor()
  cur = conn.cursor()
  cur.execute(
    "SELECT  distinct Month  FROM salestb ")
  location = cur.fetchall()
  #print(coorname)
  return render_template('Sales.html', mon=location)
@app.route("/salsearch", methods=['GET', 'POST'])
def salsearch():
  if request.method == 'POST':
    month = request.form['loc']
    import matplotlib.pyplot as plt
    import matplotlib
    matplotlib.use('Agg')
    conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1Medicalddb')
    mycursor = conn.cursor()
    mycursor.execute("select Product, sum(RepSales) as MSales from salestb group by
Product")
```

```
    result = mycursor.fetchall
    Month = []
    MSales = []
    Month.clear()
    MSales.clear()
    for i in mycursor:
        Month.append(i[0])
        MSales.append(i[1])
    print("Month = ", Month)
    print("Total Sales = ", MSales)
    # Visulizing Data using Matplotlib
    plt.figure(figsize=(12, 10))
    plt.bar(Month, MSales, color=['yellow', 'red', 'green', 'blue', 'cyan'])
    #plt.ylim(0, 5)
    ax = plt.gca()
    plt.draw()
    ax.tick_params(axis='x', rotation=70)
    plt.xlabel("Product")
    plt.ylabel("Total Sales")
    plt.title("Sales By Product")
    import random
    n = random.randint(1111, 9999)
    plt.savefig('static/plott/'+str(n)+'.jpg')
    iimg = 'static/plott/'+str(n)+'.jpg'
   conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1Medicalddb')
    cur = conn.cursor()
    cur.execute("SELECT * FROM salestb where Month='"+ month +"' ")
    data = cur.fetchall()
    conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1Medicalddb')
    # cursor = conn.cursor()
    cur = conn.cursor()
    cur.execute(
      "SELECT  distinct Month  FROM salestb ")
    locati = cur.fetchall()
   return render_template('Sales.html', data=data, dataimg=iimg, mon=locati)
@app.route("/SupplierInventory")
def SupplierInventory():
  conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1Medicalddb')
```

```
    # cursor = conn.cursor()
    cur = conn.cursor()
    cur.execute(
        "SELECT  distinct Supplier  FROM salestb ")
    customer = cur.fetchall()
    #print(coorname)
    return render_template('SupplierInventory.html', sup=customer)
@app.route("/supsearch", methods=['GET', 'POST'])
def supsearch():
    if request.method == 'POST':
        cname = request.form['sup']
       import matplotlib.pyplot as plt
        import matplotlib
        matplotlib.use('Agg')
        conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1Medicalddb')
        mycursor = conn.cursor()
        # Fecthing Data From mysql to my python progame
        mycursor.execute("select Month, sum(InventoryStock) as InventoryStock from salestb
where Supplier='"+ cname +"' group by Month")
        result = mycursor.fetchall
       Month = []
        CSales = []
        Month.clear()
        CSales.clear()

        for i in mycursor:
            Month.append(i[0])
            CSales.append(i[1])
        print("Month = ", Month)
        print("Total Sales = ", CSales)
        # Visulizing Data using Matplotlib
        plt.figure(figsize=(12, 10))
        plt.bar(Month, CSales, color=['black', 'red', 'green', 'blue', 'cyan'])
        #plt.ylim(0, 5)
        ax = plt.gca()
        plt.draw()
        ax.tick_params(axis='x', rotation=70)
        plt.xlabel("Month")
        plt.ylabel("Inventory Stock")
        plt.title("Inventory")
```

```
        import random
        n = random.randint(1111, 9999)
        plt.savefig('static/plott/' + str(n) + '.jpg')
        iimg = 'static/plott/' +str(n)+ '.jpg'
        conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1Medicalddb')
        cur = conn.cursor()
        cur.execute("SELECT * FROM salestb where Supplier='"+ cname +"' ")
        data = cur.fetchall()
        conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1Medicalddb')
        # cursor = conn.cursor()
        cur = conn.cursor()
        cur.execute(
            "SELECT  distinct Supplier  FROM salestb ")
        customer = cur.fetchall()
        return render_template('SupplierInventory.html',
data=data,dataimg=iimg,sup=customer)
@app.route("/Inventory")
def Inventory():
    conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1Medicalddb')
    # cursor = conn.cursor()
    cur = conn.cursor()
    cur.execute(
        "SELECT  distinct Month  FROM salestb ")
    location = cur.fetchall()
    #print(coorname)
    return render_template('Inventory.html', mon=location)
@app.route("/insearch", methods=['GET', 'POST'])
def insearch():
    if request.method == 'POST':
        month = request.form['loc']
        import matplotlib.pyplot as plt
       import matplotlib
        matplotlib.use('Agg')
        conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1Medicalddb')
        mycursor = conn.cursor()
        mycursor.execute("select Product, sum(InventoryStock) as InventoryStock from salestb
where Month='"+ month +"' group by Product")
```

```
    result = mycursor.fetchall
    Month = []
    MSales = []
    Month.clear()
    MSales.clear()
    for i in mycursor:
        Month.append(i[0])
        MSales.append(i[1])
    print("Month = ", Month)
    print("Total Sales = ", MSales)
    # Visulizing Data using Matplotlib
    plt.figure(figsize=(12, 10))
    plt.bar(Month, MSales, color=['yellow', 'red', 'green', 'blue', 'cyan'])
    #plt.ylim(0, 5)
    ax = plt.gca()
    plt.draw()
    ax.tick_params(axis='x', rotation=70)
    plt.xlabel("Product")
    plt.ylabel("Inventory Stock")
    plt.title(" Inventory")
    import random
    n = random.randint(1111, 9999)
    plt.savefig('static/plott/'+str(n)+'.jpg')
   iimg = 'static/plott/'+str(n)+'.jpg'
    conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1Medicalddb')
    cur = conn.cursor()
    cur.execute("SELECT * FROM salestb where Month='"+ month +"' ")
    data = cur.fetchall()
    conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1Medicalddb')
    # cursor = conn.cursor()
    cur = conn.cursor()
    cur.execute(
      "SELECT  distinct Month  FROM salestb ")
    locati = cur.fetchall()
   return render_template('Inventory.html', data=data, dataimg=iimg, mon=locati)
@app.route("/SalesTrend")
def SalesTrend():
  conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1Medicalddb')
```

```
    # cursor = conn.cursor()
    cur = conn.cursor()
    cur.execute(
        "SELECT  distinct Month  FROM salestb ")
    location = cur.fetchall()
    #print(coorname)
    return render_template('SalesTrend.html', mon=location)
@app.route("/stsearch", methods=['GET', 'POST'])
def stsearch():
    if request.method == 'POST':
        month = request.form['loc']
        import matplotlib.pyplot as plt
        import matplotlib
        matplotlib.use('Agg')
        conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1Medicalddb')
        mycursor = conn.cursor()
        mycursor.execute("select Product, sum(Actual) as Actual from salestb where
Month='"+ month +"' group by Product order by Actual  DESC")
        result = mycursor.fetchall
        Month = []
        MSales = []
        Month.clear()
        MSales.clear()
        for i in mycursor:
            Month.append(i[0])
            MSales.append(i[1])
        print("Month = ", Month)
        print("Total Sales = ", MSales)
        # Visulizing Data using Matplotlib
        plt.figure(figsize=(12, 10))
        plt.bar(Month, MSales, color=['yellow', 'red', 'green', 'blue', 'cyan'])
        #plt.ylim(0, 5)
        ax = plt.gca()
        plt.draw()
        ax.tick_params(axis='x', rotation=70)
        plt.xlabel("Product")
        plt.ylabel("Total Sales")
        plt.title("Sales Trend")
        import random
        n = random.randint(1111, 9999)
```

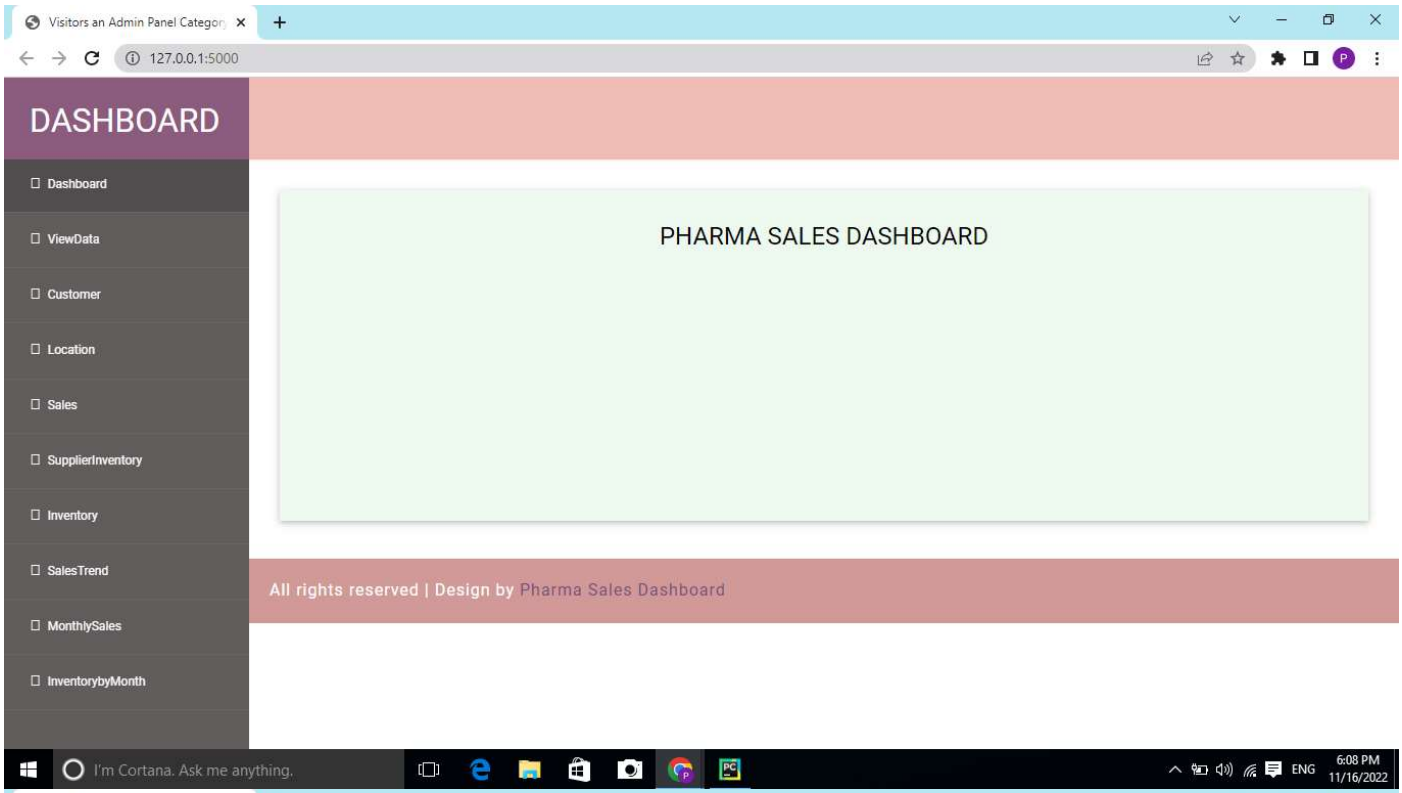
```
        plt.savefig('static/plott/'+str(n)+'.jpg')
        iimg = 'static/plott/'+str(n)+'.jpg'
        conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1Medicalddb')
        cur = conn.cursor()
        cur.execute("SELECT * FROM salestb where Month='"+ month +"' ")
        data = cur.fetchall()
        conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1Medicalddb')
        # cursor = conn.cursor()
        cur = conn.cursor()
        cur.execute(
            "SELECT  distinct Month  FROM salestb ")
        locati = cur.fetchall()
        return render_template('SalesTrend.html', data=data, dataimg=iimg, mon=locati)
@app.route("/MonthlySales")
def MonthlySales():
    conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1Medicalddb')
    # cursor = conn.cursor()
    cur = conn.cursor()
    cur.execute(
        "SELECT  distinct Month  FROM salestb ")
    location = cur.fetchall()
    #print(coorname)

    return render_template('MonthlySales.html', mon=location)
@app.route("/msearch", methods=['GET', 'POST'])
def msearch():
    if request.method == 'POST':
        month = request.form['loc']
        import matplotlib.pyplot as plt
        import matplotlib
        matplotlib.use('Agg')
        conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1Medicalddb')
        mycursor = conn.cursor()
        mycursor.execute("select Product, sum(MSales) as MSales from salestb where
Month='"+ month +"' group by Product ")
        result = mycursor.fetchall
        Month = []
```

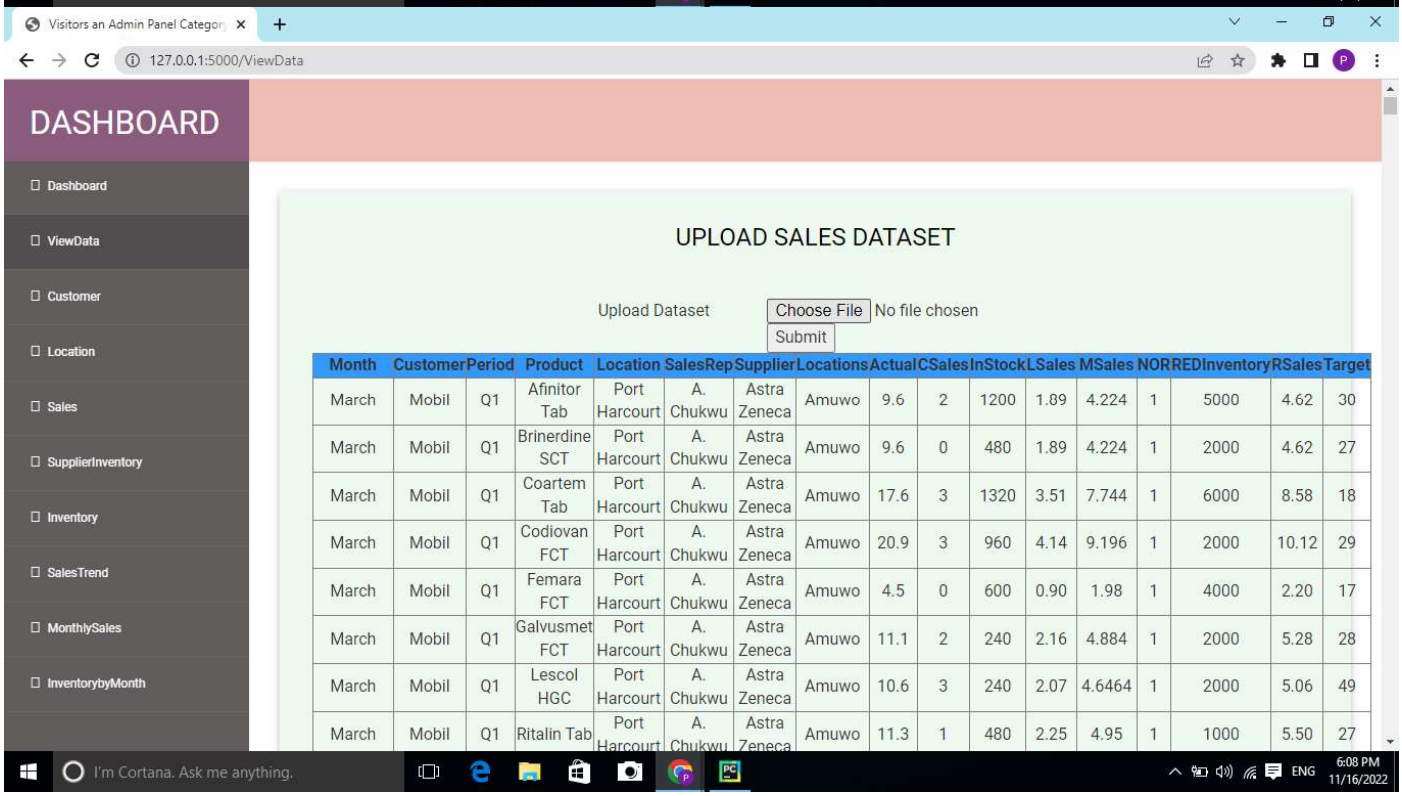
```
    MSales = []
    Month.clear()
    MSales.clear()
    for i in mycursor:
        Month.append(i[0])
        MSales.append(i[1])
    print("Month = ", Month)
    print("Total Sales = ", MSales)
    # Visulizing Data using Matplotlib
    plt.figure(figsize=(12, 10))
    plt.bar(Month, MSales, color=['yellow', 'red', 'green', 'blue', 'cyan'])
    #plt.ylim(0, 5)
    ax = plt.gca()
    plt.draw()
    ax.tick_params(axis='x', rotation=70)
    plt.xlabel("Product")
    plt.ylabel("Total Sales")
    plt.title("Monthly Sales")
    import random
    n = random.randint(1111, 9999)
    plt.savefig('static/plott/'+str(n)+'.jpg')
    iimg = 'static/plott/'+str(n)+'.jpg'
    conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1Medicalddb')
    cur = conn.cursor()
    cur.execute("SELECT * FROM salestb where Month='"+ month +"' ")
    data = cur.fetchall()
    conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1Medicalddb')
    # cursor = conn.cursor()
    cur = conn.cursor()
    cur.execute(
        "SELECT  distinct Month  FROM salestb ")
    locati = cur.fetchall()
    return render_template('MonthlySales.html', data=data, dataimg=iimg, mon=locati)
@app.route("/InventorybyMonth")
def InventorybyMonth():
  conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1Medicalddb')
  # cursor = conn.cursor()
  cur = conn.cursor()
```

```
    cur.execute(
        "SELECT  distinct Month  FROM salestb ")
    location = cur.fetchall()
    #print(coorname)
    return render_template('InventorybyMonth.html', mon=location)
@app.route("/insalsearch", methods=['GET', 'POST'])
def insalsearch():
    if request.method == 'POST':
       month = request.form['loc']
        import matplotlib.pyplot as plt
        import matplotlib
        import  numpy as np
        matplotlib.use('Agg')
        conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1Medicalddb')
        mycursor = conn.cursor()
        mycursor.execute("select Product, sum(Actual) as Actual, sum(ReceivedInventory) as
ReceivedInventory from salestb where Month='"+ month +"' group by Product ")
        result = mycursor.fetchall
        Month = []
        Actual = []
        inven = []
        Month.clear()
        Actual.clear()
        inven.clear()
        for i in mycursor:
            Month.append(i[0])
            Actual.append(i[1])
            inven.append(i[2])
            # Visulizing Data using Matplotlib
        #labels = ['G1', 'G2', 'G3', 'G4', 'G5']
        #men_means = [20, 34, 30, 35, 27]
        #women_means = [25, 32, 34, 20, 25]
        plt.figure(figsize=(12, 10))
        x = np.arange(len(Month))  # the label locations
        width = 0.35  # the width of the bars
        fig, ax = plt.subplots()
        rects1 = ax.bar(x - width / 2, Actual, width, label='Actual')
        rects2 = ax.bar(x + width / 2, inven, width, label='inven')
        # Add some text for labels, title and custom x-axis tick labels, etc.
       ax.set_ylabel('Count')
```

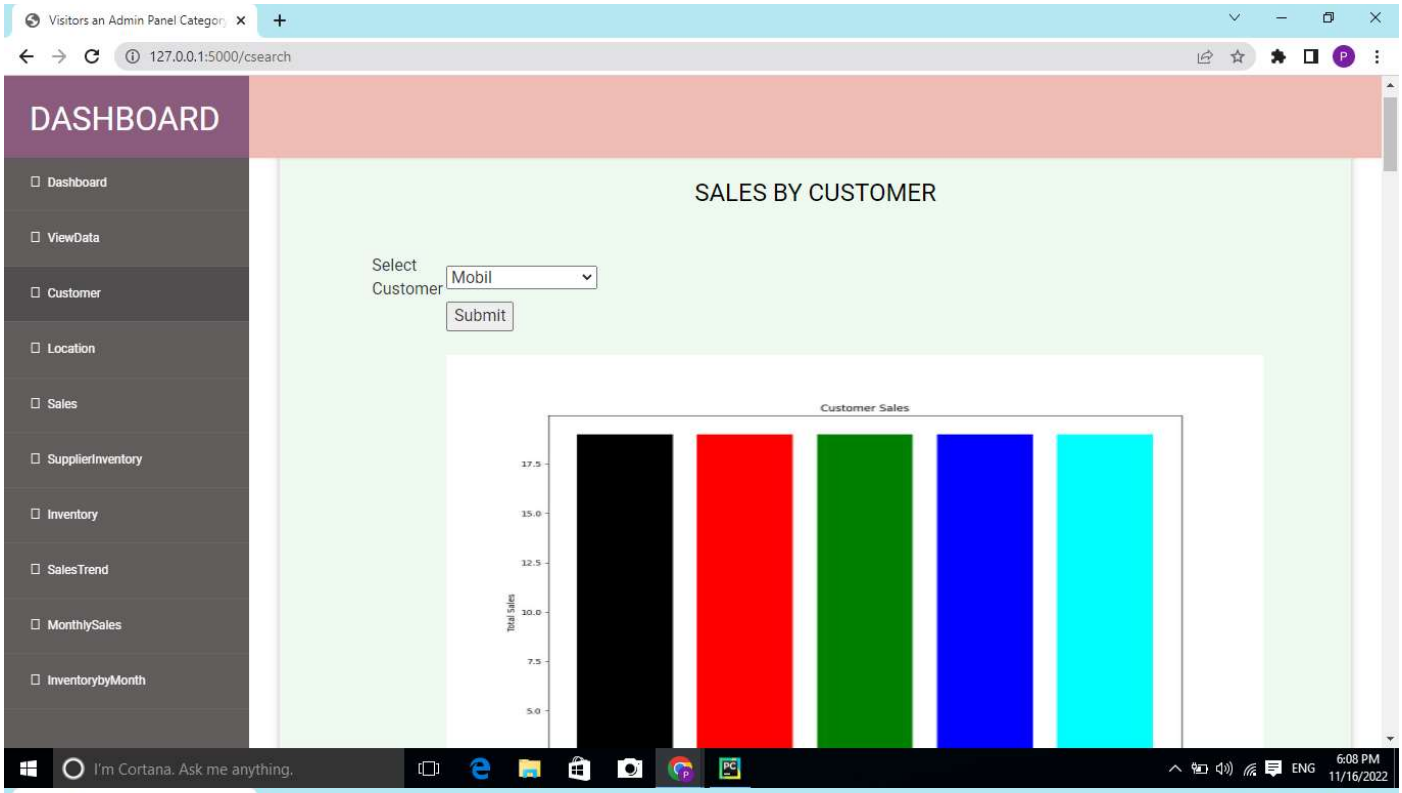
```
ax.set_title(' Actual and Received Inventory by Month')
ax.set_xticks(x, Month,fontsize=8)
ax.tick_params(axis='x', rotation=70)
ax.legend()
ax.bar_label(rects1, padding=3)
ax.bar_label(rects2, padding=3)
fig.tight_layout()
#@plt.show()
import random
n = random.randint(1111, 9999)
plt.savefig('static/plott/'+str(n)+'.jpg')
iimg = 'static/plott/'+str(n)+'.jpg'
conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1Medicalddb')
cur = conn.cursor()
cur.execute("SELECT * FROM salestb where Month='"+ month +"' ")
data = cur.fetchall()
conn = mysql.connector.connect(user='root', password='', host='localhost',
database='1Medicalddb')
# cursor = conn.cursor()
cur = conn.cursor()
```

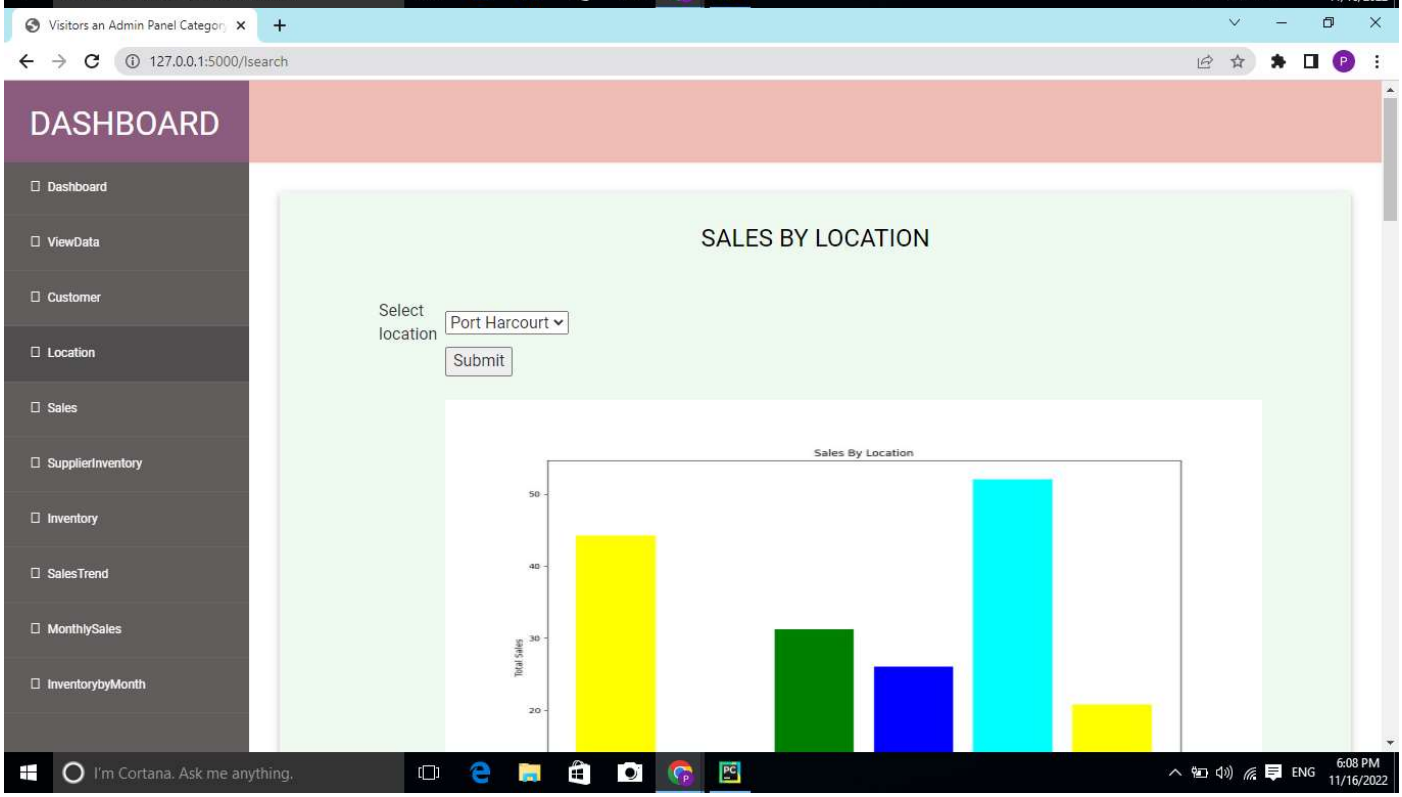
SCREENSHOTS

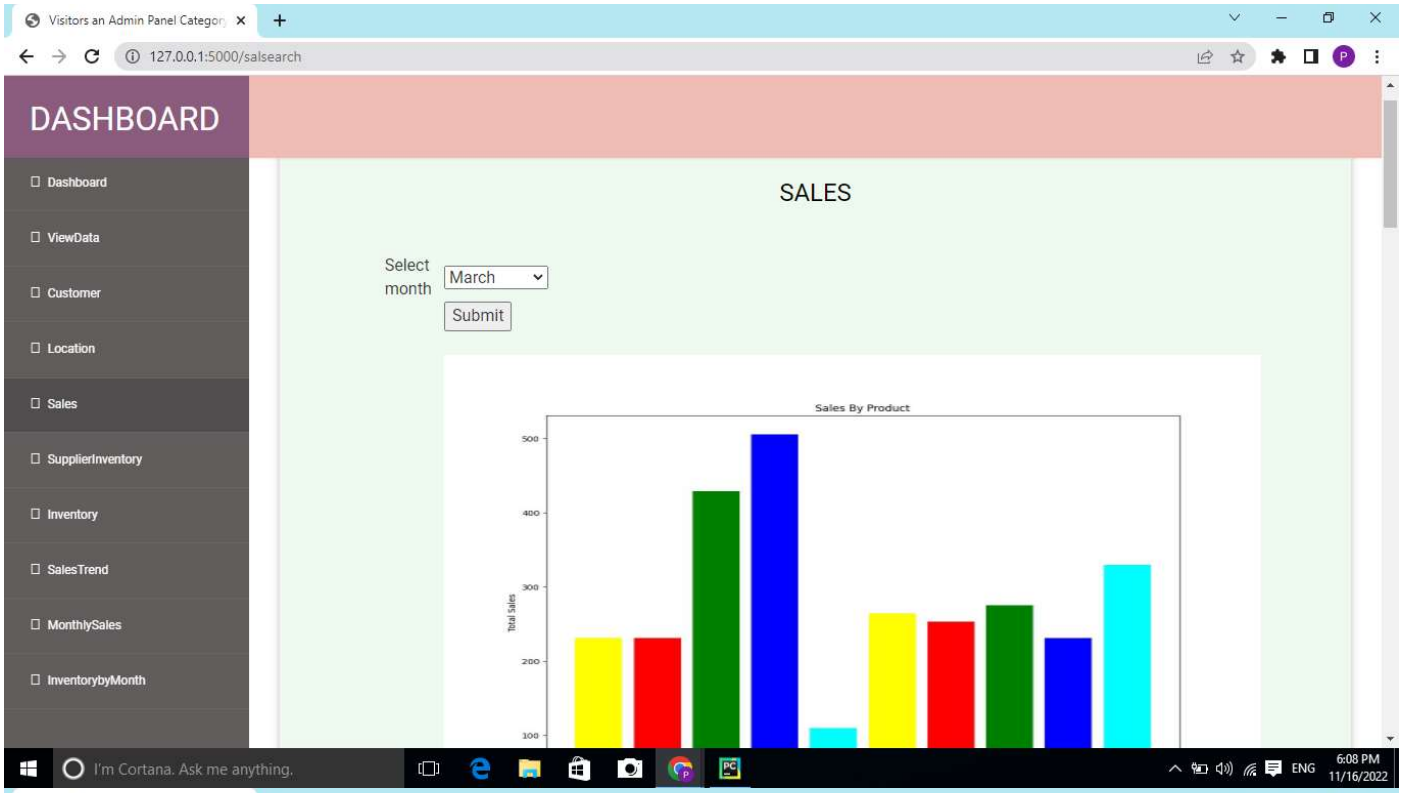


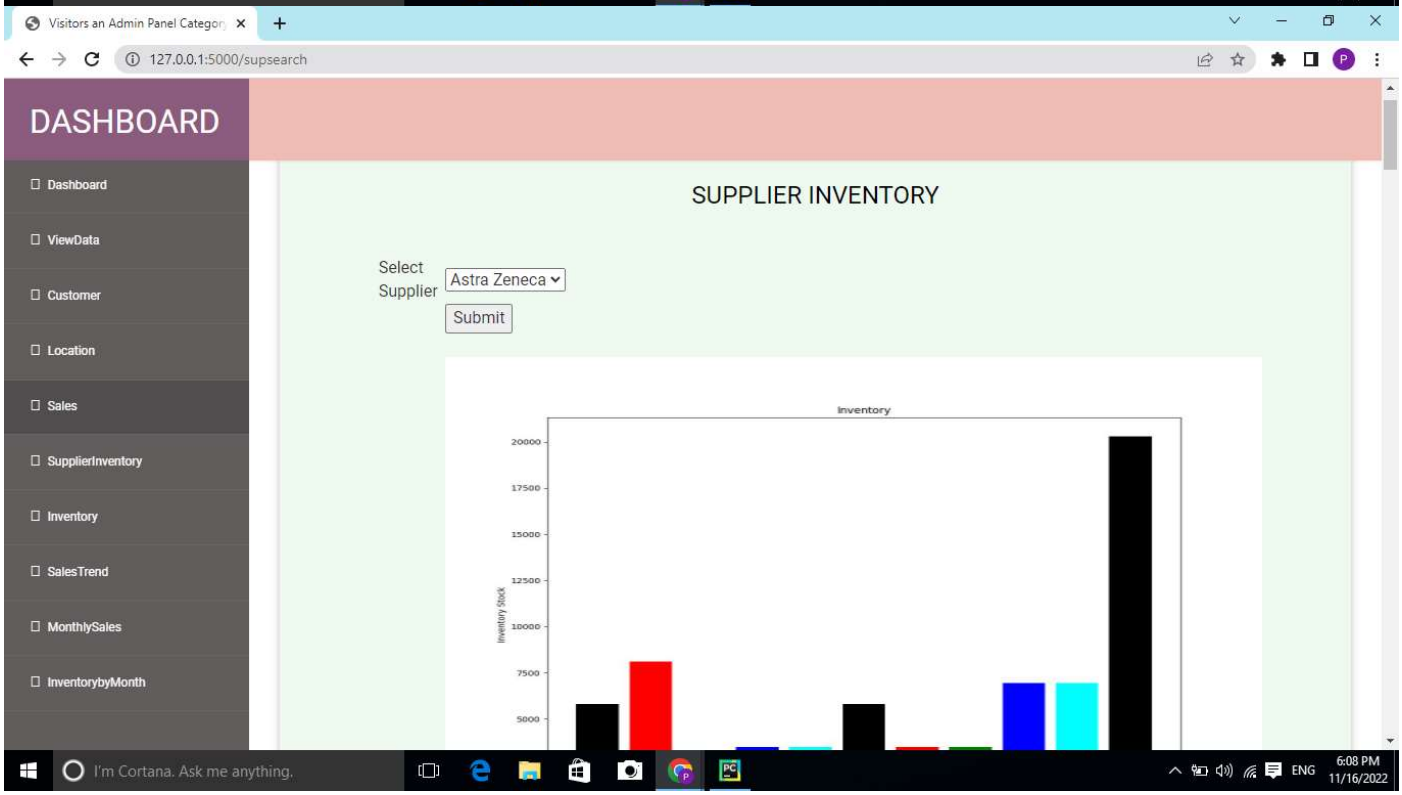


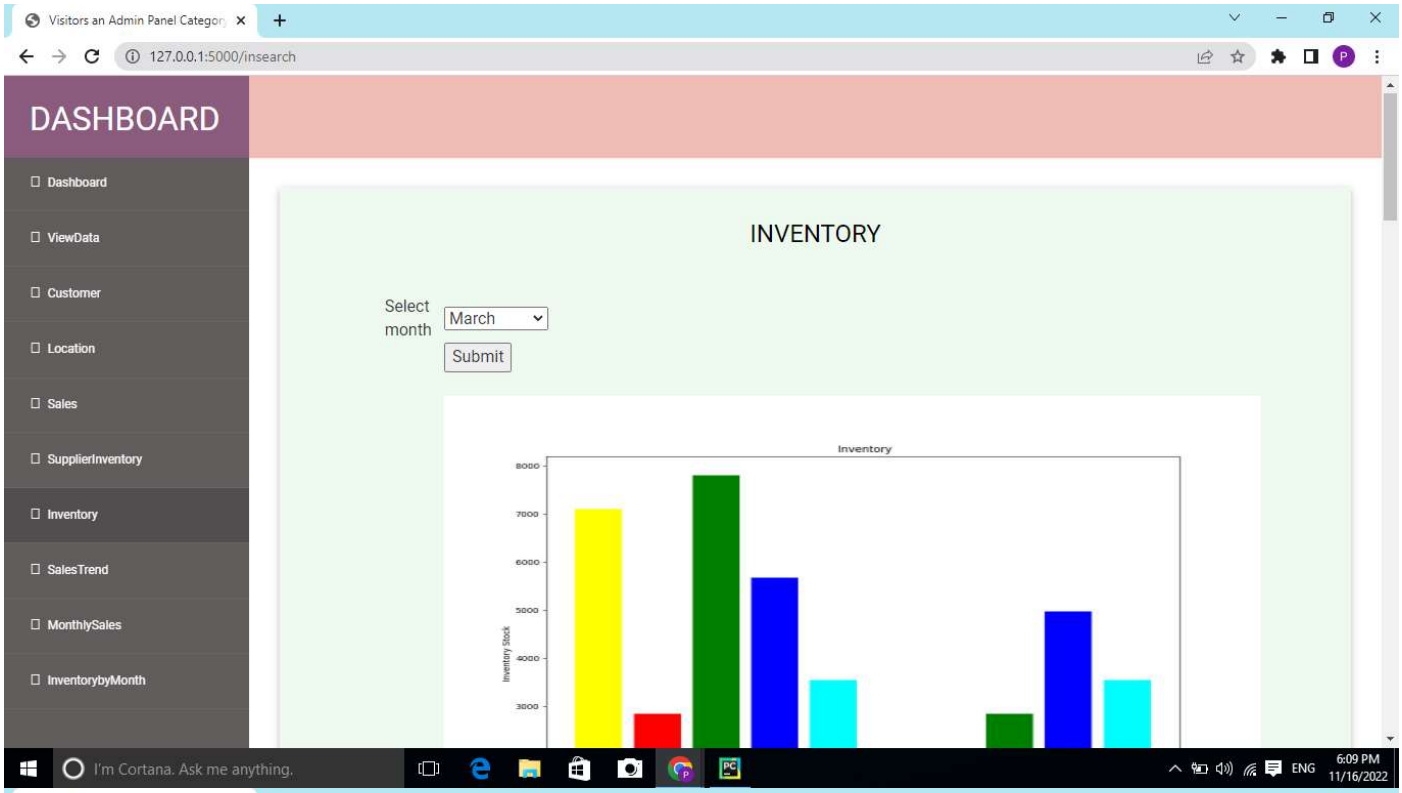
Month	Customer	Period	Product	Location	SalesRep	Supplier	Locations	Actual	CSales	InStock	LSales	MSales	NORR	EDInventory	RSales	Target
March	Mobil	Q1	Afinitor Tab	Port Harcourt	A. Chukwu	Astra Zeneca	Amuwo	9.6	2	1200	1.89	4.224	1	5000	4.62	30
March	Mobil	Q1	Brinerdine SCT	Port Harcourt	A. Chukwu	Astra Zeneca	Amuwo	9.6	0	480	1.89	4.224	1	2000	4.62	27
March	Mobil	Q1	Coartem Tab	Port Harcourt	A. Chukwu	Astra Zeneca	Amuwo	17.6	3	1320	3.51	7.744	1	6000	8.58	18
March	Mobil	Q1	Codiovan FCT	Port Harcourt	A. Chukwu	Astra Zeneca	Amuwo	20.9	3	960	4.14	9.196	1	2000	10.12	29
March	Mobil	Q1	Femara FCT	Port Harcourt	A. Chukwu	Astra Zeneca	Amuwo	4.5	0	600	0.90	1.98	1	4000	2.20	17
March	Mobil	Q1	Galvusmet FCT	Port Harcourt	A. Chukwu	Astra Zeneca	Amuwo	11.1	2	240	2.16	4.884	1	2000	5.28	28
March	Mobil	Q1	Lescol HGC	Port Harcourt	A. Chukwu	Astra Zeneca	Amuwo	10.6	3	240	2.07	4.6464	1	2000	5.06	49
March	Mobil	Q1	Ritalin Tab	Port Harcourt	A. Chukwu	Astra Zeneca	Amuwo	11.3	1	480	2.25	4.95	1	1000	5.50	27

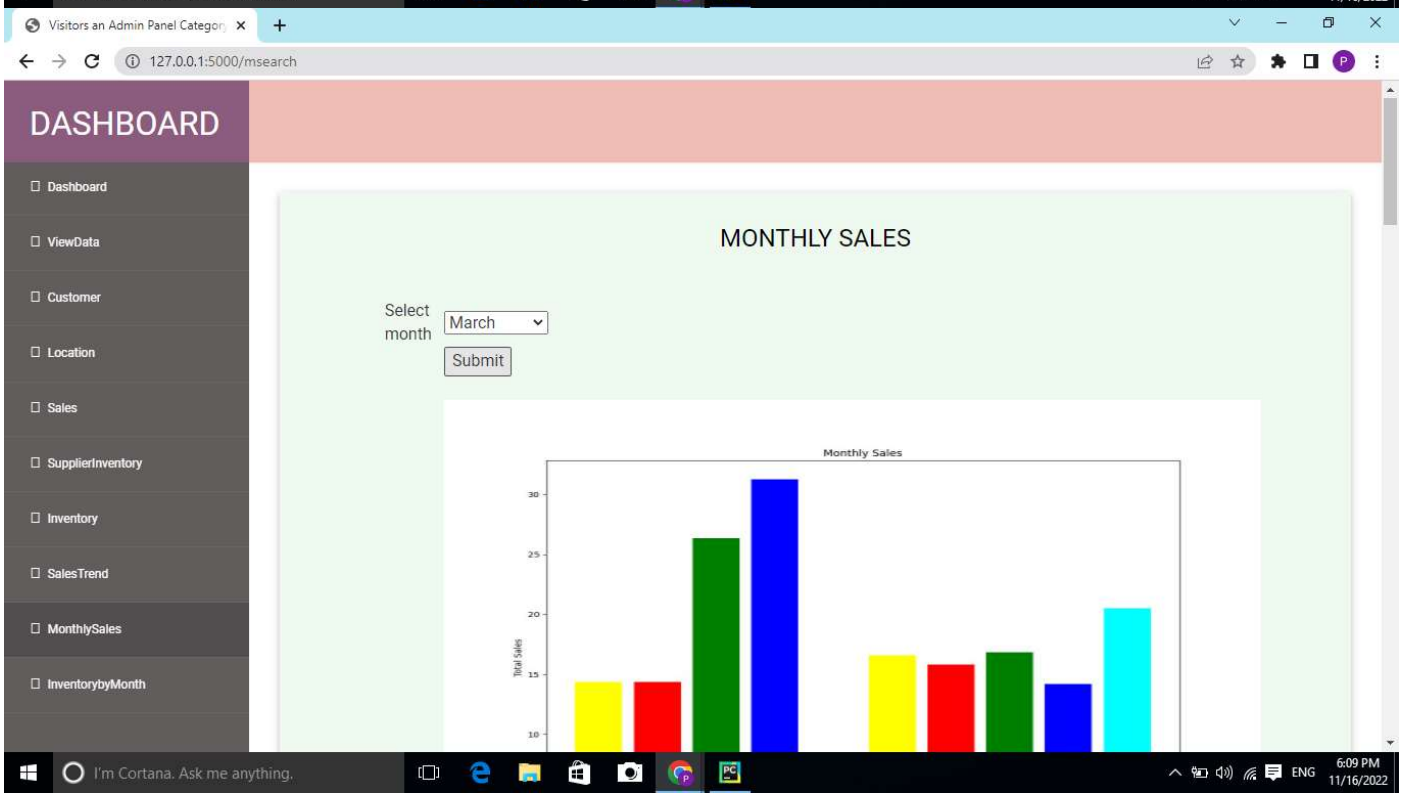

Visitors an Admin Panel Categor
127.0.0.1:5000/csearch
DASHBOARD
Dashboard
ViewData
Customer
Location
Sales
SupplierInventory
Inventory
SalesTrend
MonthlySales
InventorybyMonth
SALES BY CUSTOMER
Select Customer
Mobil
Submit
Customer Sales
Total Sales
17.5
15.0
12.5
10.0
7.5
5.0
I'm Cortana. Ask me anything.
ENG
6:08 PM
11/16/2022


Visitors an Admin Panel Categor
127.0.0.1:5000/lsearch
DASHBOARD
Dashboard
ViewData
Customer
Location
Sales
SupplierInventory
Inventory
SalesTrend
MonthlySales
InventorybyMonth
SALES BY LOCATION
Select location
Port Harcourt
Submit
Sales By Location
Total Sales
50
40
30
20
I'm Cortana. Ask me anything.
ENG
6:08 PM
11/16/2022


Visitors an Admin Panel Categor
127.0.0.1:5000/salsearch
DASHBOARD
Dashboard
ViewData
Customer
Location
Sales
SupplierInventory
Inventory
SalesTrend
MonthlySales
InventorybyMonth
SALES
Select month
March
Submit
Sales By Product
Total Sales
500
400
300
200
100
I'm Cortana. Ask me anything.
ENG
6:08 PM
11/16/2022


Visitors an Admin Panel Categor
127.0.0.1:5000/supsearch
DASHBOARD
Dashboard
ViewData
Customer
Location
Sales
SupplierInventory
Inventory
SalesTrend
MonthlySales
InventorybyMonth
SUPPLIER INVENTORY
Select Supplier
Astra Zeneca
Submit
Inventory
Inventory Stock
20000
17500
15000
12500
10000
7500
5000
I'm Cortana. Ask me anything.
ENG
6:08 PM
11/16/2022


Visitors an Admin Panel Categor
127.0.0.1:5000/insearch
DASHBOARD
Dashboard
ViewData
Customer
Location
Sales
SupplierInventory
Inventory
SalesTrend
MonthlySales
InventorybyMonth
INVENTORY
Select month
March
Submit
Inventory
Inventory Stock
8000
7000
6000
5000
4000
3000
I'm Cortana. Ask me anything.
ENG
6:09 PM
11/16/2022


Visitors an Admin Panel Categor
127.0.0.1:5000/msearch
DASHBOARD
Dashboard
ViewData
Customer
Location
Sales
SupplierInventory
Inventory
SalesTrend
MonthlySales
InventorybyMonth
MONTHLY SALES
Select month
March
Submit
Monthly Sales
Total Sales
30
25
20
15
10
I'm Cortana. Ask me anything.
ENG
6:09 PM
11/16/2022

8. TESTING

Test Case:

- Verifies whether the user can login if he/she was an registered user.
- Verifies whether an unregistered user cannot proceed with the login.
- Verifies whether an unregistered user can successfully register as anuser.
- Verifies whether an register user cannot register them self as an newuser.
- Verifies whether an alert message popsup when an unregistered user tries to login .
- Verifies whether an alert message popsup when an registered user tries to register again.
- Verifies whether an alert message popsup when an registered user enters his/her username or password incorrect.
- Verifies whether an alert message popsup when an new user registers.
- Verifies whether all UI button(signup,login now,logout,report,story,user dashboard) works efficiently.
- Verifies whether username popsup on the welcome note.

Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Comments	TC for Automation(Y/N	BUG ID	Executed By
Testcase_1	Functional	Login Page	Verifies whether the user can login if he/she was an registered user	Checks whether the logged in username is registered in backend.	1.Enter your username 2.Enter your password 3.click signin button	username: Nalayia Thrian passwoed:abcd	Homepage should display	Working as expected	Pass				user
Testcase_2	Functional	Login Page	Verifies whether an unregistered user cannot proceed with the login.	Checks whether the logged in username is not registered in backend.	1.Enter your username 2.Enter your password 3.click signin button	username:nalayathrian password:1234	Homepage will not display	Working as expected	pass				user
testcase_3	Functional	register page	Verifies whether an unregistered user can successfully register as an user.	The details given by the user is stored in backend	1.Enter your username 2.Enter your email 3.Enter your password 4.Enter your confirm password 5.Click on signup button	Enter your data	User will be able to access to login page	working as expected	pass				user
testcase_4	Functional	Register page	Verifies whether an register user cannot register themself as an new user.	checks whether the user name is present in the database.	1.Enter your username 2.Enter your email 3.Enter your password 4.Enter your confirm password 5.Click on signup button	username:Nalayia Thrian password:abcd	User will not be able to access to login page	working as expected	pass				user
Testcase_5	Functional	Login page	Verifies whether an alert message popsup when an unregistered user tries to login.	checks whether the user name is present in the database.	1.Enter your username 2.Enter your password 3.click signin button	username:roja password:abcd	message should display	Working as expected	Pass				user

Testcase_6	Functional	Register page	Verifies whether an alert message popsup when an registered user tries to register again.	checks whether the user name is present in the database.	1.Enter your username 2.Enter your email 3.Enter your password 4.Enter your confirm password 5.Click on signup button	Username:Nalayia Thrian password: abcd	.message should display	Working as expected	Pass				user
Testcase_7	Functional	Login page	Verifies whether an alert message popsup when an registered user enters his/her username or password incorrect.	checks the username given by the user against the username and password in the database.	1.Enter your username 2.Enter your password 3.click signin button	Username:Nalayia Thrian password: 1234	user will not be able to access the home page	working as expected	Pass				user
Testcase_8	Functional	Register page	Verifies whether an alert message popsup when an new user registers.	checks whether the username and password is iploaded in the database.	1.Enter your username 2.Enter your email 3.Enter your password 4.Enter your confirm password 5.Click on signup button	Username:Nalayia Thrian password: 1234	message should display	Working as expected	Pass				user
Testcase_9	UI	Home page	Verifies whether all UI button(signup,loginnow,logout,report,story,user dashboard) works efficiently.	checks whether all the button are working efficiently	1.Enter username, password to login. 2.Click on report, story,user dashboard and logout button		button should work	Working as expected	Pass				user
Testcase_10	Functional	Home page	Verifies whether username popsup on the welcome note.	check whether the message display	1.Enter your username 2.Enter your password 3.click signin button		welcome note should display	Working as expected	Pass				user

User Acceptance Testing:

The purpose of this is to briefly explain the test coverage and open issues of the retail store stock analytics project at the time of the release to User Acceptance Testing (UAT).

Defect Analysis:

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

Resolution	Severity1	Severity2	Severity3	Severity4	Subtotal
By Design	8	4	2	1	15
Duplicate	0	0	0	0	0
External	3	2	0	1	6
Fixed	4	0	1	0	5
Not Reproduced	0	0	1	0	1
Skipped	0	0	0	1	1
Won't Fix	0	0	1	0	1
Totals	15	6	5	3	29

Test Case Analysis:

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

Section	Total Cases	Not Tested	Fail	Pass
Print Engine	5	0	0	5
Client Application	30	0	0	30
Security	2	0	0	2
Outsource Shipping	4	0	0	4
Exception Reporting	8	0	0	8
Final Report Output	6	0	0	6
Version Control	2	0	0	2

9. RESULTS

Performance Metrics:

Performance metrics are defined as figures and data representative of an organization's actions, abilities, and overall quality.

S.No.	Parameter	Screenshot / Values
1.	Dashboard design	Dashboard consist of 8graph in 8 different tabs.
2.	Data Responsiveness	Data was responsive for creating dashboard, story and report.
3.	Amount Data to Rendered (DB2 Metrics)	Inventory management dataset which consist of 938 datas in it.
4.	Utilization of Data Filters	Data filters was used to find the top most of the data in form of visualization.
5.	Effective User Story	Story consist of 4 scenes and 5 graphs.
6.	Descriptive Reports	Created 2 reports with 7 graphs.

10. ADVANTAGES & DISADVANTAGES:

Advantage:

- An advantage of the retail inventory method is that it does not require a physical inventory.
- The retail inventory method only requires an organization to record the retail prices of inventory items.

Cost-Effective:

Manual inventory control would increase your labor and process costs.

Saves Time:

Paper-based retail inventory management can take a lot of time and effort.

Process Efficiency:

Inventory management is one of the crucial retail processes.

Disadvantage:

- Overstocking on products runs the risk of the product becoming obsolete.
- Higher storage and insurance costs.
- Certain goods might perish.
- Stock may become obsolete before it is used.
- Your capital is tied up

11. CONCLUSION:

Hence in Retail store stock analysis it helps shop holder to manage stock, sale and price and maintain the necessary stock without reaching to demand,by maintaining the stock it gains the trust for the customer to buyproduct on a regular basis which also provide gain to to shop holder by increasing the profit.

12. FUTURE SCOPE:

Inventory management systems have become more real-time, giving retailers more data about demographics, spending habits, shopping preferences, etc.. Stock control for omni channel retailing. Stores doing omni channel retailing

are at the top of their game; they attract the 90% of consumers who switch between at least three applications per day to complete specific tasks.Inventories that power experiential retail.

13. APPENDIX :

GitHub Link : https://github.com/IBM-EPBL/IBM-Project-15021-1659593378

Project Demo Link : https://drive.google.com/file/d/1t3eq_a-2x5T2FCwMmjYGEYOVA1N8Zbno/view?usp=drivesdk