

1. INTRODUCTION

1.1 Project Overview

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

In practice, effective retail inventory management results in lower costs and a better understanding of sales patterns. Retail inventory management tools and methods give retailers more information with which to run their businesses, including:

- Product locations.
- Quantities of each product type.
- Which stock sells well and which doesn't, by location and sales channel.
- Profit margin by style, model, product line or item.
- Ideal amount of inventory to have in back stock and storage.
- How many products to reorder and how often.
- When to discontinue a product.
- How changing seasons affect sales.

1.2 Purpose

The primary purpose of inventory management is to ensure there is enough goods or materials to meet demand without creating overstock, or excess inventory.

2. LITERATURE SURVEY

2.1 Existing problem

- Consumers are Choosing Multichannel Buying Experiences
- Customers Expect a Seamless Experience
- To Attract Customer Loyalty, Retailers Need an Experience Which Stands Out
- A Siloed Marketing Infrastructure Makes It Expensive and Unwieldy to get Your Message Across.
- So Many Technologies Exist to Drive Marketing and Sales, but They Don't Seem to Work Together

2.2 References

Altunışık R., Coşkun R., Bayraktaroğlu S., Yıldırım E., (2012), "Araştırma Yöntemleri SPSS Uygulamalı", Sakarya Yayıncılık, 7. Baskı.

Amit P. and Kameshvari B., (2012), "A Study on Consumer Behaviour of Organized and Unorganized Retail Outlets in Vadodara City" International Journal of Engineering and Management Sciences,

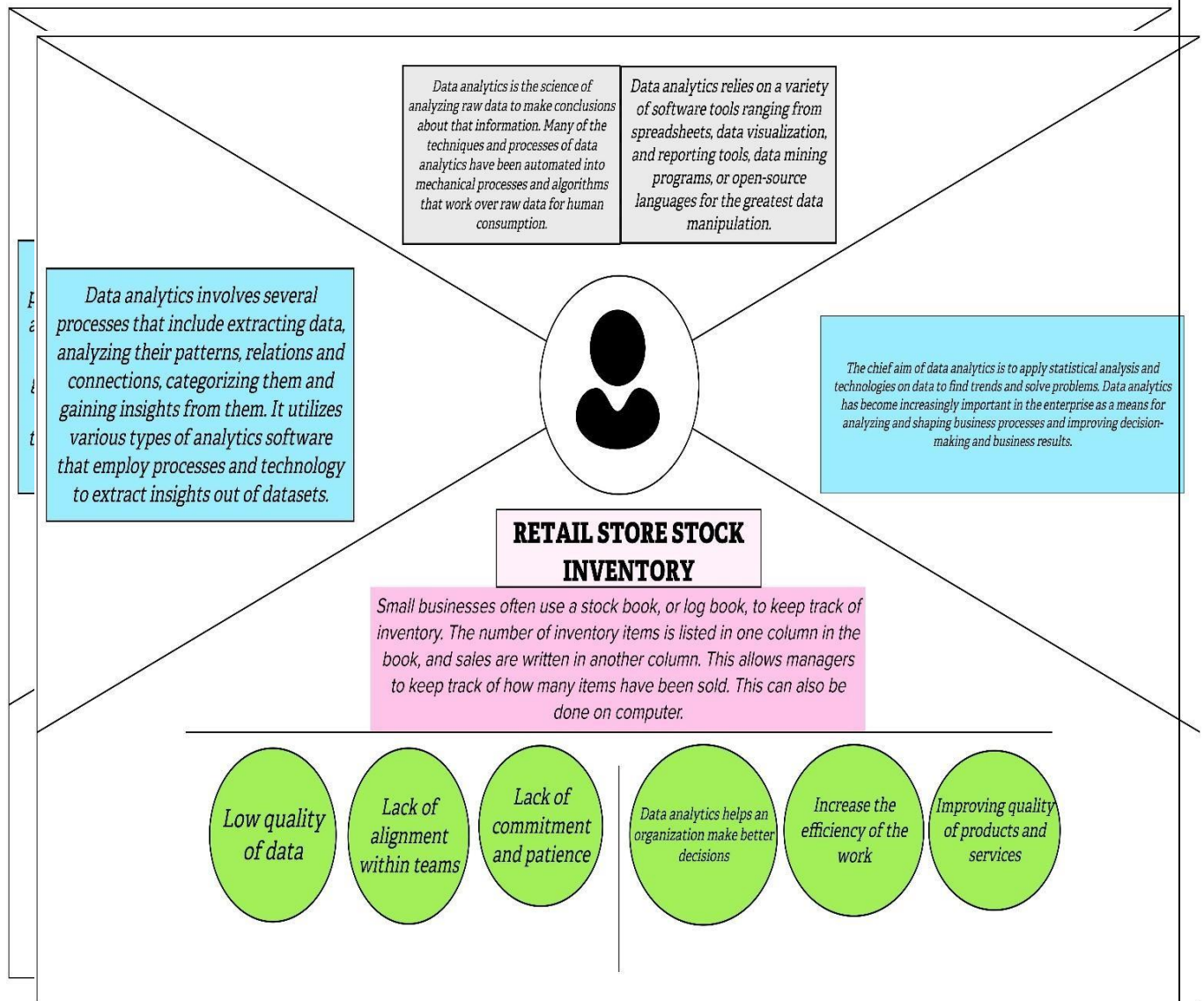
- Aydın K., (2013), “Perakende Yönetiminin Temelleri”, Nobel Yayını, 4. Basım.
- Chunawalla S.A., (2009), “Contours of Retailing Management”, Himalaya Publishing House
- Dhotre, M. (2010), “Channel Management and Retail Marketing”, Himalaya Publishing House, Revised Edition.
- Fernie S., Fernie J., Moore Ch., (2013), “Principles of Retailing”, Edinburgh Business School, Heriot-Watt University
- Kotler P. and Armstrong G., (2012), “Principles of Marketing”, Pearson Prentice Hall, 14th Edition.
- Kotler P. and Keller K.L., (2012), “Marketing Management”, Prentice Hall, 14th Edition.
- Milli Eğitim Bakanlığı (MEB), (2008), “Pazarlama ve Perakende – Perakendeciliğin Özellikleri”, Ankara: MEGEP (Mesleki Eğitim ve Öğretim Sisteminin Güçlendirmesi Projesi”.
- Mucuk I., (2009), “Pazarlama İlkeleri”, Türkmen Kitabevi, 17. Basım.
- Perakendecilik Noktaları [Online] . Available from:
<http://www.tml.web.tr/konu-19.php>, [Accessed 05 January 2016]
- Perreault W., Cannon J., McCarthy J., (2013), Pazarlamanın Temelleri: Biz Pazarlama Stratejisi Planlama Yaklaşımı”, Mc-Graw Hill, 13 Basımdan Çeviri (Nobel Yayını)
- Pradhan S., (2008), “Retail Management – Text and Cases”, McGraw-Hill Publishing Company, 2nd Edition.
- Rudrabasavaraj M. N., (2010), “Dynamic Global Retailing Management”, Himalaya Publishing House, 1st Edition.
- Sharma B., (2008), “Strategic Retail Management”, Book Enclave
- Tang A., Lim S., (2008), “Retail Operations – How to Run Your Own Store”, Pearson Prentice Hall, 2nd Edition.
- Tenekecioğlu B., Tokol T., Çalık N., Karalar R., Timur N., Öztürk S. (2005), “Pazarlama Yönetimi”, Anadolu Üniversitesi Yayını, No. 1478
- Tiwari R.S., (2009), “Retail Management, Retail Concepts and Practices”, Himalaya Publishing House, 1st Edition.

2.3 Problem Statement Definition

To Create an Analytical Dashboard for Sales Data that needs to show the stocks of a retail store that is to be analysed with the help of Visualizations, Data Modules, Dashboards, Explorations and Story. The Visualizations, Dashboards, Data Modules, Explorations and Story should be Created using IBM Cognos Analytics. The Data is to be stored in the IBM Cloud.

3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas



3.2 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

Template



Brainstorm & idea prioritization





Before you collaborate

A little bit of Discussion about the process of a retail store that how a retail store will buy stocks, store the stocks and sell the stocks based on the real time scenario.

 10 minutes

 **Team gathering**

We have gathered as all of our 4 members in our team.

 **Set the goal**

As Our Mincsel is very clean, We should produce a meaningful Application that will fulfill the customer needs to be satisfied.

 **Learn how to use the facilitation tools**

Use the Facilitation Superpowers to run a happy and productive session.

[Open article](#) 



Define your problem statement

As we defined our problem statement based on the clear and Simple Ideas that to create a meaningful dashboard which is to be understandable by all the users. Our Motto is to keep things simple and Normal and not to Hype.

 5 minutes

PROBLEM

To Create a Analytical Dashboard for Sales Data that needs to show the stocks of a retail store that is to be analyzed with the help of Visualizations, Data Modules, Dashboards, Explorations and Story. The Visualizations, Dashboards, Data Modules, Explorations and Story should be Created using IBM Cognos Analytics. The Data is to be stored in the IBM Cloud.



Key rules of brainstorming

To run an smooth and productive session

 Stay in topic.

 Encourage wild ideas.

 Defer judgment.

 Listen to others.

 Go for volume.

 If possible, be visual.

4

Step-2: Brainstorm, Idea Listing and Grouping

2

Brainstorm solo

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

🕒 10 minutes

Prasanah Kumar S

Details of stock should be safe	Visit the stores	How much inventory should i carry?
lack of inventory leads to lost sales	Create multiple analysis graphs	Detect the various type of stock needed

Nehru Prasanth

Collection of data	Average order value	By key metrics, you predict inventory needs
Examine gross margin trends	Examine Inventory/ Receivable Trends	Tabulate Tangible Book Value

Praveen Kumar L

Understand the dataset	How much your customers are spending per order	Short term forecasting
Examine gross margin trends	Identifying potential risks	Year wise stock using line graph

Kishore D S

Strength of e commerce sales	Unhappy customers and a damaged brand	Examine gross margin trends
How many items your customers is purchasing per order	Inventory turnover ratio	Live stock reports

Step-3: Idea Prioritization

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.

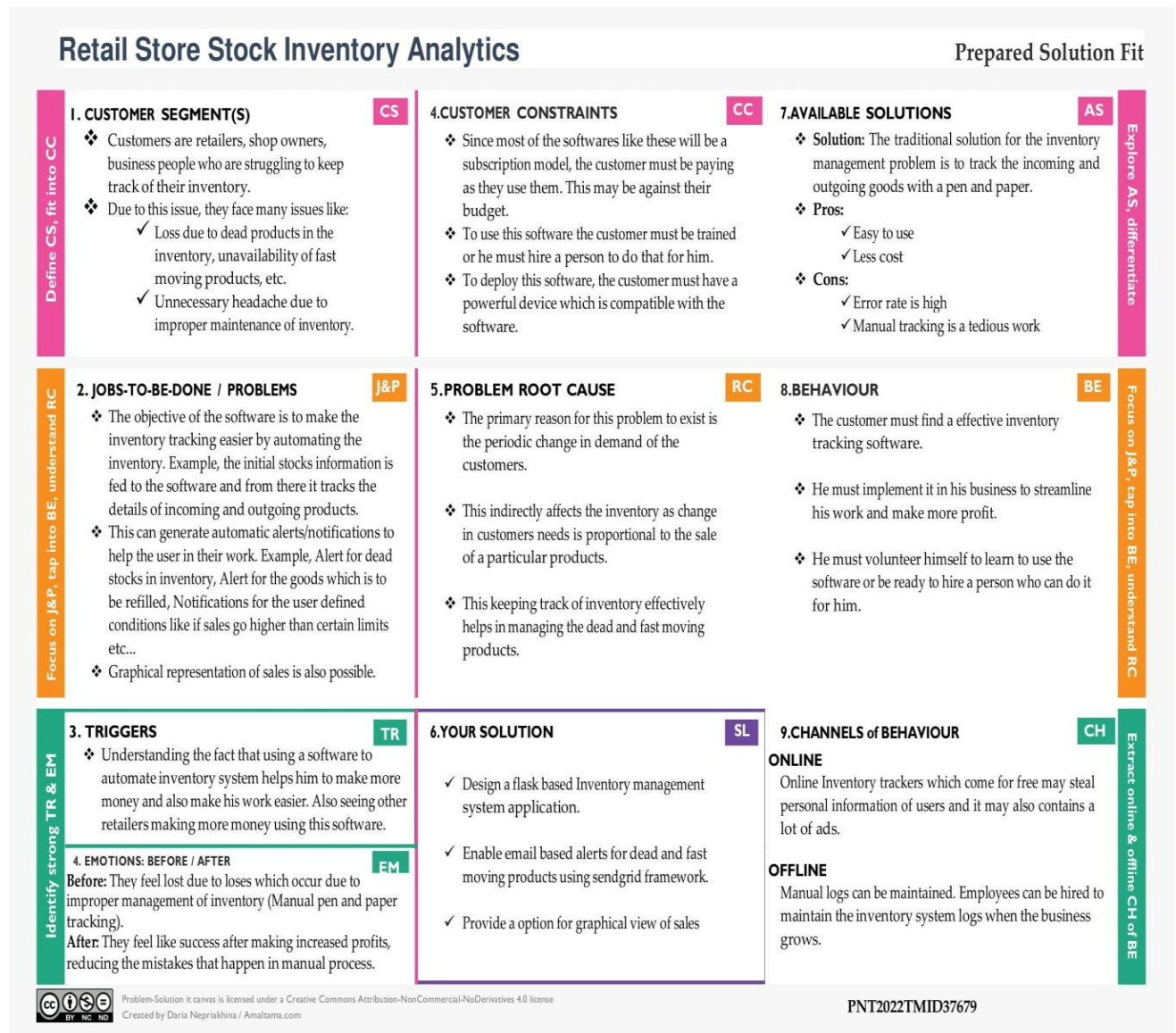
⌚ 20 minutes



3.3 Proposed Solution

S No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	To Create an Analytical Dashboard for Sales Data that needs to show the stocks of a retail store that is to be analyzed with the help of Visualizations, Data Modules, Dashboards, Explorations and Story. The Visualizations, Dashboards, Data Modules, Explorations and Story should be Created using IBM Cognos Analytics. The Data is to be stored in the IBM Cloud.
2.	Idea / Solution description	In Our Project, we are Planning to develop a dashboard using IBM Cognos Analytics Which express our talent as our Outcomes. We are Using Python Language for backend Database Connection. The Data will be stored in the IBM Cloud Platform.
3.	Novelty / Uniqueness	IBM Cognos Analytics is the platform that we are going to build the Dashboard is something Unique. All the Details Of our Profit and Loss ratio Will be Shared to the Customers and we will like to maintain some Transparency Should be maintained with the Customers.
4.	Social Impact / Customer Satisfaction	As the Customers come to know the exact details about the retail store stocks which gives the satisfaction
5.	Business Model (Revenue Model)	In the perspective of Income and Generating Profit is Such Difficult in Retail Stocks. So, Our Dashboard will make the customers to buy our products more and we can do some Advanced marketing Strategies to make Revenue.
6.	Scalability of the Solution	We will produce exact information to the customers through Our Project and the functionality of our project will be best in the market.

3.4 Problem Solution fit



4. REQUIREMENT ANALYSIS

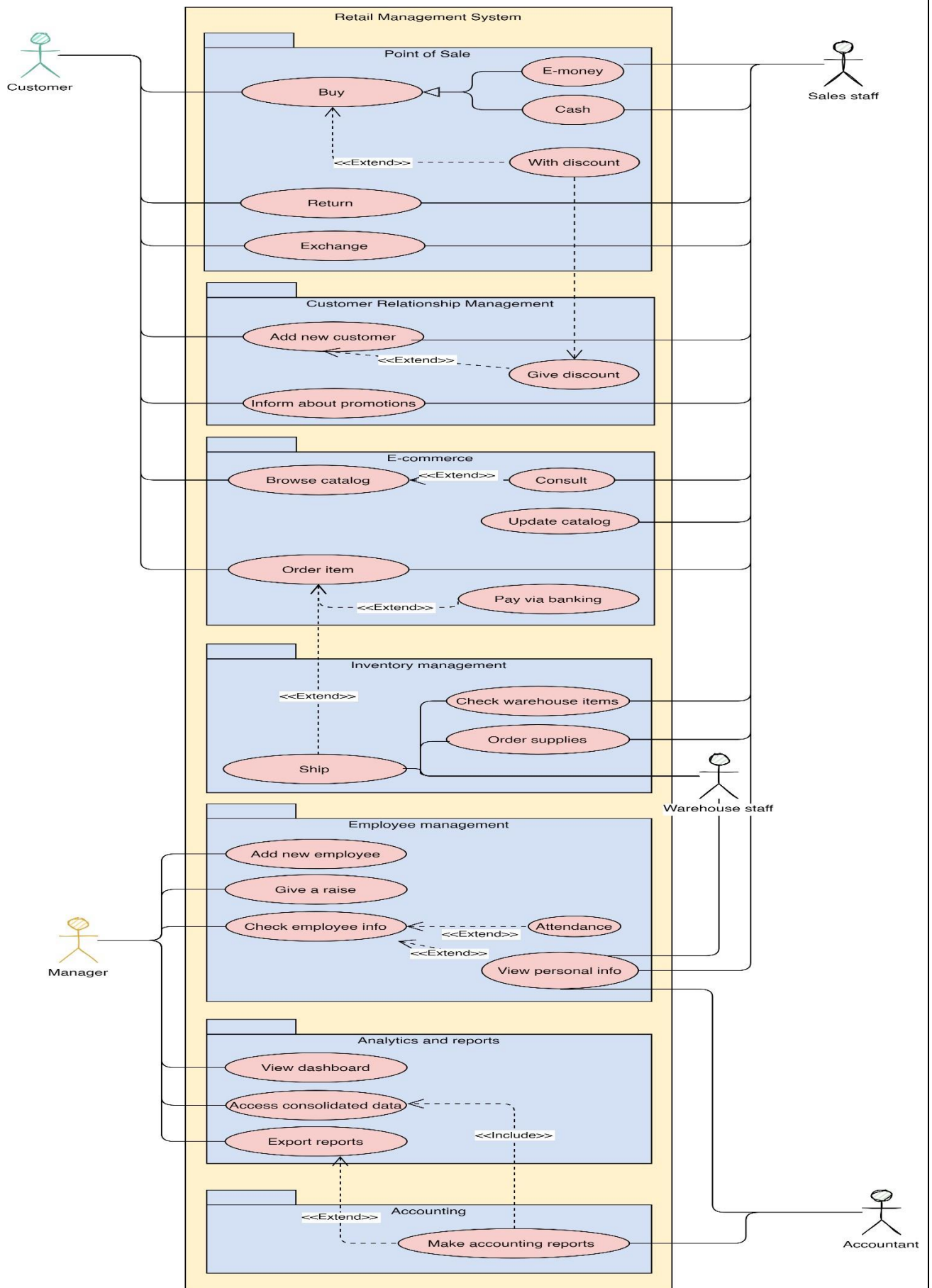
4.1 Functional requirement

Retail Management System has a modular nature, so it is logical to address requirements to different modules (some of the modules can be integrated into RMS as stand-alone solutions).

- **Point of sale** - manages sales in a retail store
 - Scan item barcode and load the item info from the system
 - Allow manual input and search of sale item
 - Record sale, return, and exchange to the database
 - Accept payment by cash, credit cards, I Pay
 - Print invoice

- **Customer Relationship Management** - consolidates information about customers
 - Save and store customer information (name, date of birth, contact info, client id, password)
 - Apply client's discount levels
 - Automate marketing and customer satisfaction analytics
- **E-commerce** - manages an online store
 - Add item to the catalogue (manually or from file)
 - Update item in the catalogue
 - Show sale items info
 - Record online orders
 - Allow online payment
 - Connect customer with the sales manager for consultation
- **Inventory management** - track goods throughout the supply chain
 - Remove or add items from the store
 - Remove or add items to the warehouse
 - Ship items between stores
 - Ship items to customers
 - Order supplies
- **Employee management** - automate HR tasks
 - Store information about employees
 - Track employees' attendance
 - Manage absence and leave requests
 - Collect productivity data
 - Manage payments
 - Manage employee accounts permissions
- **Accounting** - transfer data to the accounting department/software and import approved reports
 - Integrate with accounting software
- **Analytics and reports** - consolidate data from all active modules for reports and dashboard
 - Collect data from modules
 - Run informative dashboard
 - Provide templates for periodic reports
 - Allow custom reports creation

The use case diagram below summarizes the conceptual areas of application of the Retail Management System and interaction between customers, sales department, warehouse, accounting department, and the manager of the retail business.



4.2 Non-Functional requirements

Non-functional requirements define requirements to the system as a whole and should be considered with attention before approaching the architecture phase, as introducing changes in later stages will be difficult.

Usability

The Retail Management system is required to have a simple and user-friendly interface, and allow to customize the interface and dashboard for individual users. The system must fully support languages that business operates on and allow to import and export of data in .csv and spreadsheet file formats.

Interoperability

The System must allow integration and partial replacement of sub-modules.

Data integrity

All data about sales, orders, client information, and other must be accurate and consistent over the entire life cycle.

Environmental

The system must require minimum resource usage for its maintenance to comply with the principles of sustainability.

Security

The system restricts access to client data, analytics, and reports to only authorized users. The rights to add or correct data must be restricted for individual employees. Financial data must be secured with two-factor authentication. Data relative to the latest operational year must be duplicated on a reserve server.

Maintainability

System maintenance must run without shutting down or in automated mode during non-working hours.

Capacity

The system must be capable of handling 100 employee accounts and 10000 orders per day without affecting its performance.

Availability

The availability of the system must be not less than 99.999% during the retail working hours, and not less than 95% round-the-clock for the e-commerce module.

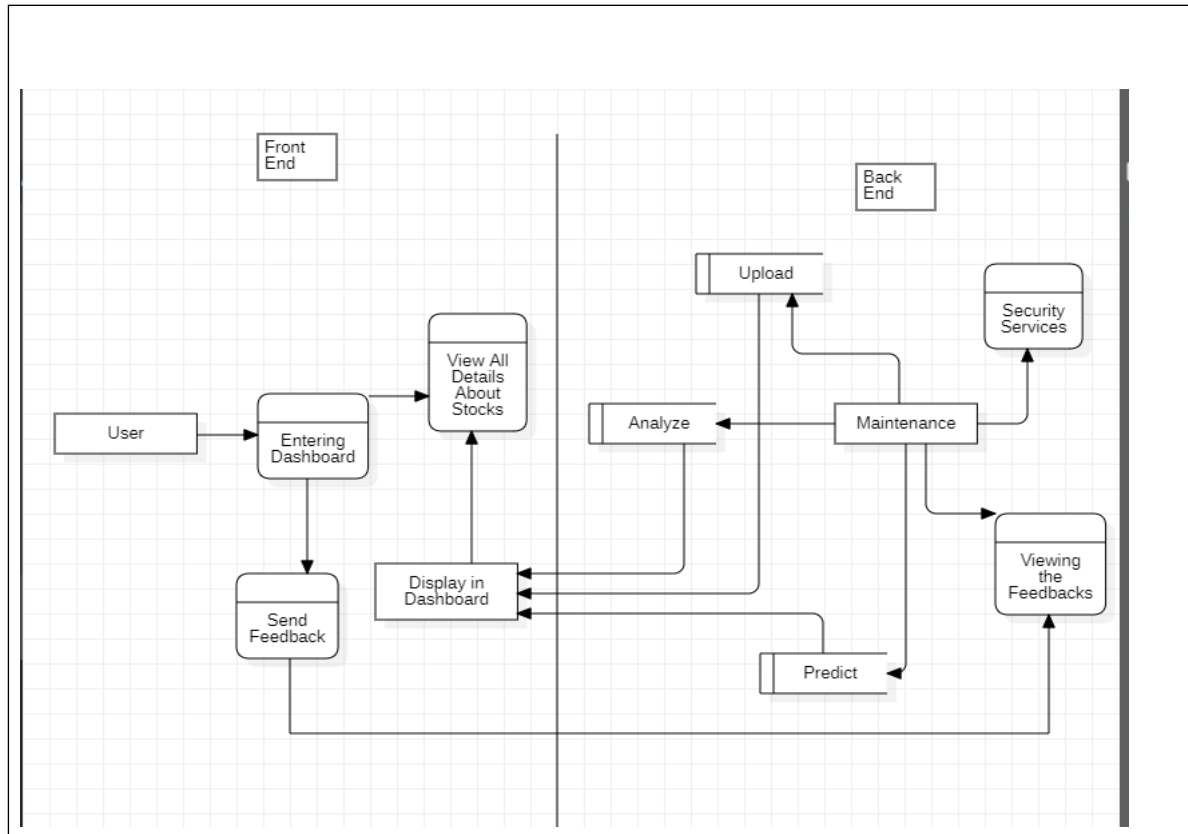
Scalability

The system must support implementing new features and modules without disrupting existing processes. The system must support horizontal scaling for launching new retail stores with multiple POS.

5. PROJECT DESIGN

5.1 Data Flow Diagrams

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.

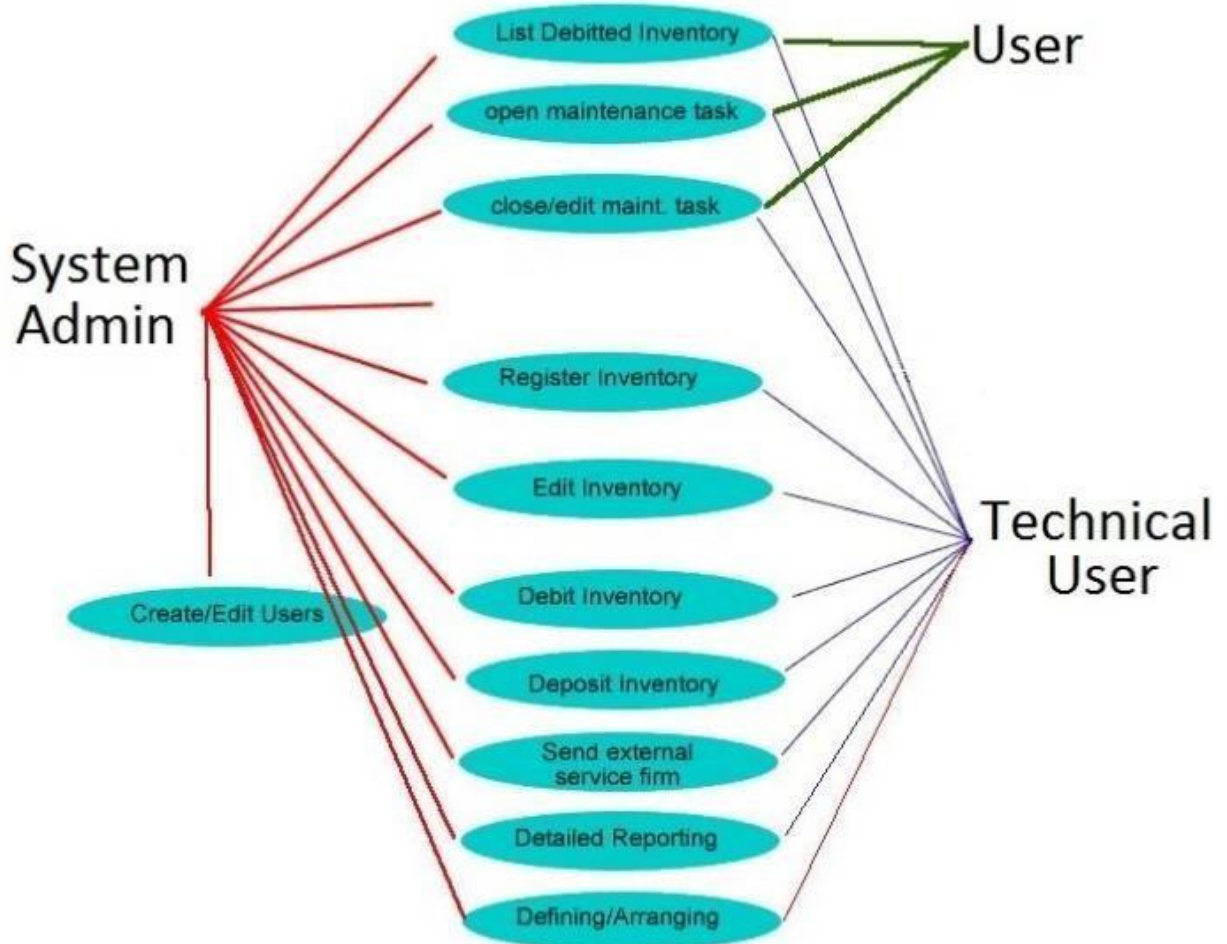


5.2 Solution & Technical Architecture

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.

Solution Architecture Diagram:



5.3 User Stories

CUSTOMER JOURNEY MAP



6. PROJECT PLANNING & SCHEDULING

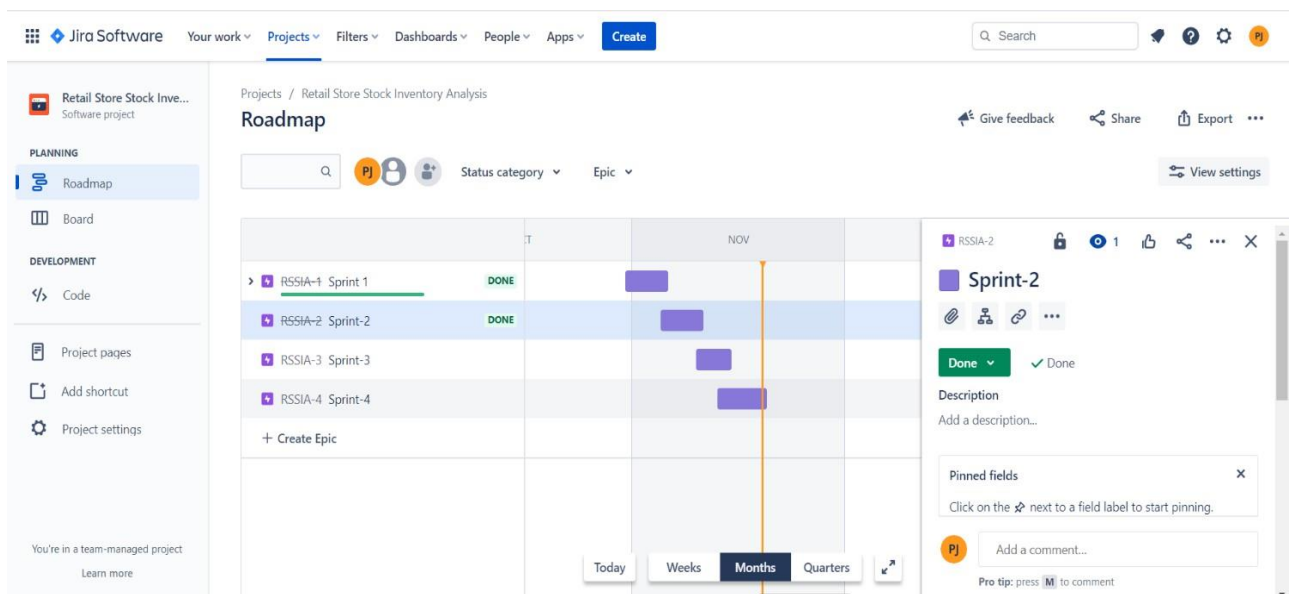
6.1 Sprint Planning & Estimation

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

6.2 Sprint Delivery Schedule

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date	Story Completed(as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	06 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	13 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

6.3 Reports from JIRA



7. CODING & UTIONING (Explain the features added in the project along with code)

7.1 Feature 1

File name - index.html

```
<!DOCTYPE html>
<html>
<head>
<title>Sales Inventory Login</title>
<meta name="viewport" content="width=device-width, initial-scale=1">
<style>
body {
  background-image: url('logi111.jpg');
}
/* Style all input fields */
input {
  width: 100%;
  padding: 12px;
  border: 1px solid #ccc;
  border-radius: 4px;
  box-sizing: border-box;
  margin-top: 6px;
  margin-bottom: 16px;
}

/* Style the submit button */
input[type=submit] {
  background-color: #04AA6D;
  color: white;
}

/* Style the container for inputs */
.container {
  background-color: #f1f1f1;
  padding: 20px;
}

/* The message box is shown when the user clicks on the password field */
#message {
  display:none;
  background: #f1f1f1;
  color: #000;
  position: relative;
  padding: 20px;
  margin-top: 10px;
```



```

}

#message p {
  padding: 10px 35px;
  font-size: 18px;
}

/* Add a green text color and a checkmark when the requirements are right */
.valid {
  color: green;
}

.valid:before {
  position: relative;
  left: -35px;
  content: "✓";
}

/* Add a red text color and an "x" when the requirements are wrong */
.invalid {
  color: red;
}

.invalid:before {
  position: relative;
  left: -35px;
  content: "✗";
}
</style>
</head>
<body>
<h1 align="center" style="color:#FFFFFF">Sales Login</h1>
<h3 align="center"></h3>
<div class="container">
  <form action="/action_page.php">
    <label for="username">Username</label>
    <input type="text" id="username" name="username" required>

    <label for="psw">Password</label>
    <input type="password" id="psw" name="psw" pattern="(?=.*\d)(?=.*[a-z])(?=.*[A-Z]).{8,}"
    title="Must contain at least one number and one uppercase and lowercase letter, and at least 8 or more
    characters" required>

    <center><a href="story1.html">Login</a></center>
  </form>
</div>

```

```
<div id="message">
  <h3>Password must contain the following:</h3>
  <p id="letter" class="invalid">A <b>lowercase</b> letter</p>
  <p id="capital" class="invalid">A <b>capital (uppercase)</b> letter</p>
  <p id="number" class="invalid">A <b>number</b></p>
  <p id="length" class="invalid">Minimum <b>8 characters</b></p>
</div>
```

```
<script>
var myInput = document.getElementById("psw");
var letter = document.getElementById("letter");
var capital = document.getElementById("capital");
var number = document.getElementById("number");
var length = document.getElementById("length");

// When the user clicks on the password field, show the message box
myInput.onfocus = function() {
document.getElementById("message").style.display = "block";
}

// When the user clicks outside of the password field, hide the message box
myInput.onblur = function() {
document.getElementById("message").style.display = "none";
}

// When the user starts to type something inside the password field
myInput.onkeyup = function() {
  // Validate lowercase letters
  var lowerCaseLetters = /[a-z]/g;
  if(myInput.value.match(lowerCaseLetters)) {
    letter.classList.remove("invalid");
    letter.classList.add("valid");
  } else {
    letter.classList.remove("valid");
    letter.classList.add("invalid");
  }

  // Validate capital letters
  var upperCaseLetters = /[A-Z]/g;
  if(myInput.value.match(upperCaseLetters)) {
    capital.classList.remove("invalid");
    capital.classList.add("valid");
  } else {
    capital.classList.remove("valid");
    capital.classList.add("invalid");
  }
}
```

```

// Validate numbers
var numbers = /[0-9]/g;
if(myInput.value.match(numbers)) {
    number.classList.remove("invalid");
    number.classList.add("valid");
} else {
    number.classList.remove("valid");
    number.classList.add("invalid");
}

// Validate length
if(myInput.value.length >= 8) {
    length.classList.remove("invalid");
    length.classList.add("valid");
} else {
    length.classList.remove("valid");
    length.classList.add("invalid");
}
}
</script>
</body>
</html>

```

File name - app.py

```

import ibm_db
import re
from flask import Flask, render_template, request, redirect, url_for, session, flash

app = Flask(__name__)
app.secret_key="i don't care"
myconn=ibm_db.connect('DATABASE=bludb;HOSTNAME=b1bc1829-6f45-4cd4-bef4-10cf081900bf.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=32304;SECURITY=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=ytl86182;PWD=IwLcHrD3CEMU2chS', "", "")

@app.route("/signup")
def signup():
    return render_template("signup.html")

@app.route('/register', methods=['GET', 'POST'])
def register():
    msg = ""
    if request.method == 'POST' :
        username = request.form['username']
        password = request.form['password']

        query = 'SELECT * FROM admin WHERE username =?;'

```

```

stmt=ibm_db.prepare(myconn,query)
ibm_db.bind_param(stmt,1,username)
ibm_db.execute(stmt)
account = ibm_db.fetch_assoc(stmt)
print(account)
if account:
    msg = 'Account already exists !'
elif not re.match(r'[a-z0-9]+', username):
    msg = 'name must contain only lowercase characters and numbers !'
else:
    query = "INSERT INTO ADMIN (username,password) VALUES (?,?)"
    stmt=ibm_db.prepare(myconn,query)
    ibm_db.bind_param(stmt,1,username)
    ibm_db.bind_param(stmt,2,password)
    ibm_db.execute(stmt)
    msg = 'You have successfully registered !'
    return render_template('login.html', msg = msg)

```

```

@app.route("/")
@app.route("/login",methods=['GET','POST'])
def login():
    if request.method=="POST":
        Username=request.form['Username']
        Password=request.form['Password']
        query="select * from admin where username=? and password=?;"
        stmt=ibm_db.prepare(myconn, query)
        ibm_db.bind_param(stmt, 1, Username)
        ibm_db.bind_param(stmt, 2, Password)
        ibm_db.execute(stmt)
        data=ibm_db.fetch_assoc(stmt)
        if data:
            session['loggedin']=True
            return render_template('cognos.html')

        else:
            flash("Incorrect Username or Password")
    return render_template("login.html")

```

```

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

```

File name - login.html

```
<!DOCTYPE html>
<html>

<head>
  <title>Login Form</title>
  <link rel="stylesheet" type="text/css" href="..\static\css\login.css">
  <link href="https://fonts.googleapis.com/css?family=Poppins:600&display=swap" rel="stylesheet">
  <script src="https://kit.fontawesome.com/a81368914c.js"></script>
  <meta name="viewport" content="width=device-width, initial-scale=1">

</head>

<body>

  <!--  -->
  <div class="container">

    <!-- <div class="img">
      <div id="png"><a href="/" title="HOME"></a></div>
      
    </div> -->

    <br>
    <br>
    <br>
    <div class="login-content">

      <form action="{ { url_for('login') } }" method="POST">
        <div class="msg">{ { msg } }</div>
        
        <h2 class="title">Welcome</h2>
        <div class="input-div one">
          <div class="i">
            <i class="fas fa-user"></i>
          </div>
          <div class="div">

            <input type="text" name="Username" placeholder="username" class="input" required>
          </div>
        </div>
        <div class="input-div pass">
          <div class="i">
```

```

        <i class="fas fa-lock"></i>
    </div>
    <div class="div">

        <input type="password" name="Password" placeholder="password" class="input" required>
    </div>
</div>
<div class="btn">
    <button type="login" class="btn btn-default">Login</button>
</div><br><br><br><br>

    <div class="app"><b>Don't have an account?</b><a id="app1" href="\signup">Signup Here
!</a></div>
</form>

</div>

</div>

<script type="text/javascript" src="..\static\js\login.js"></script>
</body>
</html>

```

7.2 Feature 2

File name - salesdashboard.html

```

<html>
<head>
<title>
Data Demo
</title>
<meta charset="utf-8">
<meta name="viewport" content="width=device.width, initial-scale=1">
<link                href="https://cdn.jsdelivr.net/npm/bootstrap@S.2.1/dist/css/bootstrap.min.css"
rel="stylesheet">
<script src="https://cdn.jsdelivr.net/npm/bootstrap@S.2.1/dist/js/bundle.min.js"></script>
</head>
<body>
<div class="container-fluid p-5 bg-primary text-white text-center">
    <center><h1>Sales Dashboard</h1></center>
</div>
<center><iframe
src="https://us1.ca.analytics.ibm.com/bi/?perspective=dashboard&amp;pathRef=.my_folders%2FInve
ntory%2BTotals%2Bdashboard&amp;closeWindowOnLastView=true&amp;ui_appbar=false&amp;ui

```

```
_navbar=false&shareMode=embedded&action=view&mode=dashboard&subView=
model00000183fadcd3b_00000000" width="768" height="1024" frameborder="0"
gesture="media" allow="encrypted-media" allowfullscreen=""></iframe></center>
</body>
</html>
```

File name - salesstory.html

```
<html>

<head>

<title>
Sales Story
</title>
<style>
body{
  background-image: url('homepage111.webp');
}
</style>

<meta charset="utf-8">
<meta name="viewport" content="width=device.width, initial-scale=1">
<link href="https://cdn.jsdelivr.net/npm/bootstrap@S.2.1/dist/css/bootstrap.min.css"
rel="stylesheet">

<script src="https://cdn.jsdelivr.net/npm/bootstrap@S.2.1/dist/js/bundle.min.js"></script>

</head>

<body>

<div class="container-fluid p-5 bg-primary text-white text-center">
  <center style="color:#FFFFFF"><h1>Sales Story</h1></center>
</div>

<center><iframe
src="https://us1.ca.analytics.ibm.com/bi/?perspective=story&pathRef=.my_folders%2FInventory
%2Btotal%2Bstory&closeWindowOnLastView=true&ui_appbar=false&ui_navbar=false&shareMode=embedded&action=view&sceneId=-1&sceneTime=0"
width="1080" height="1920" frameborder="0" gesture="media" allow="encrypted-media"
allowfullscreen=""></iframe></center>
</body>
</html>
```

7.3 Database Schema (if Applicable)

IBM Db2 on Cloud

Load Data Load History **Tables** Views Indexes Aliases MQTs Sequences Application objects

Find schemas or tables Refresh

Tables New table +

Name	Schema	Properties
ADMIN	YTL86182	...
RETAIL	YTL86182	...

Total: 2, selected: 0

Table definition

ADMIN
Approximate 4 rows (32.0 KB)
Updated on 2022-11-18 09:22:36

Name	Data type	Nullable	Length	Scale
ID	BIGINT	N		0
USERNAME	VARCHAR	Y	30	0
PASSWORD	VARCHAR	Y	255	0

View data

IBM Db2 on Cloud

Load Data Load History **Tables** Views Indexes Aliases MQTs Sequences Application objects

Find schemas or tables Refresh

Tables New table +

Name	Schema	Properties
ADMIN	YTL86182	...
RETAIL	YTL86182	...

Total: 2, selected: 0

Table definition

RETAIL
Approximate 369 rows (32.0 KB)
Updated on 2022-11-18 07:17:52

Name	Data type	Nullable	Length	Scale
DATA_YEAR_	DATE	Y	4	0
MONTH	SMALLINT	Y		0
VENDA_SALES_	SMALLINT	Y		0
ESTOQUE_STOCK_	SMALLINT	Y		0
PRECO_PRICE_	DECIMAL	Y	5	2
REVENUE	DECIMAL	Y	7	2

View data

8. TESTING

8.1 Test Cases

1	Test Cases ID	Retail_1	Test Case Description	Test the login functionality in sales DashBoard		
2	Created by	Prasanah S	Reviewed By	Vignesh	Version	1
3						
4						
5						
6	Tester's Name	Prasanah S	Date Tested	15-Nov-22	Test Case(Pass/Fail)	Pass
7						
8	S#	Prerequisites:		S#	Test Data	
9	1	Access to Chrome Browser			1 Userid = Prasanah	
10	2				2 Pass = 12345	
11	3				3	
12	4				4	
13						
14	Test Scenario	Verify on entering valid userid and Password, the customer can login				
15						
16	Step#	Step Details	Expected Results	Actual Results	Pass/Fail	
17		Navigate to http://127.0.0.1:5000	site should open	As Expected	Pass	
18		Enter Userid & Password	Credential can be entered	As Expected	Pass	
19		Click login	Customer is logged in	As Expected	Pass	

8.2 User Acceptance Testing

8.2.1.Purpose of Document

The purpose of this document is to briefly explain the test coverage and open issues of the Retail Store Stock Inventory Analytics project at the time of the release to User Acceptance Testing (UAT).

8.2.2. Defect Analysis

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

8.2.3. Test Case Analysis

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	3	0	0	0	3
Duplicate	1	0		0	1
External	2	0	0	0	2
Fixed	6	0	0		6
Not Reproduced	0	0	1	0	1
Skipped	0	0	1	1	2
Won't Fix	0	1	2	1	4
Totals	12	1	4	2	19

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

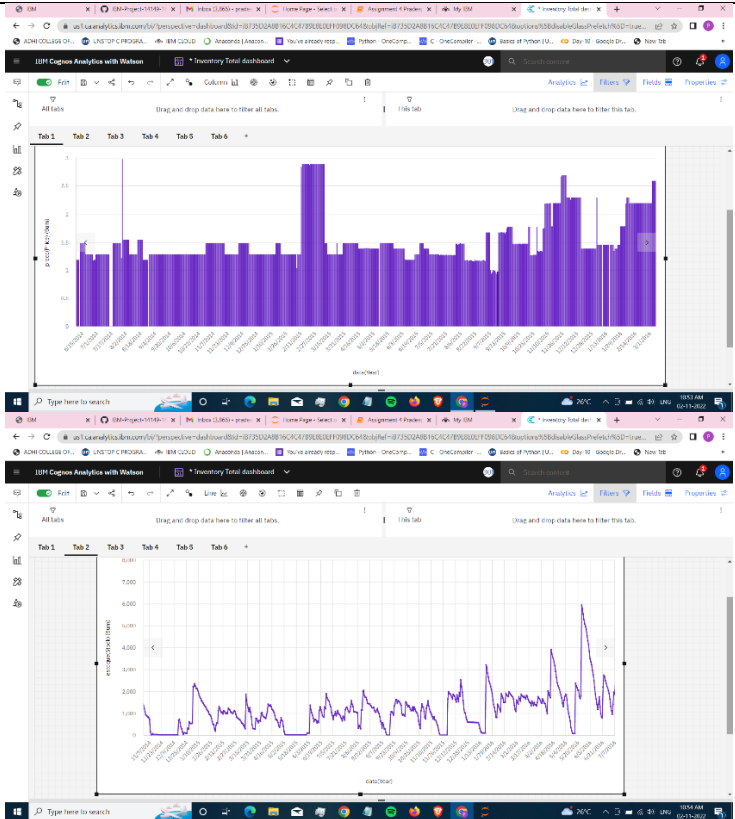
Section	Total Cases	Not Tested	Fail	Pass
Print Engine	7	0	0	7
Client Application	1	0	0	1
Security	1	0	0	1
Outsource Shipping	3	0	0	3
Exception Reporting	2	0	0	2
Final Report Output	4	0	0	4
Version Control	1	0	0	1

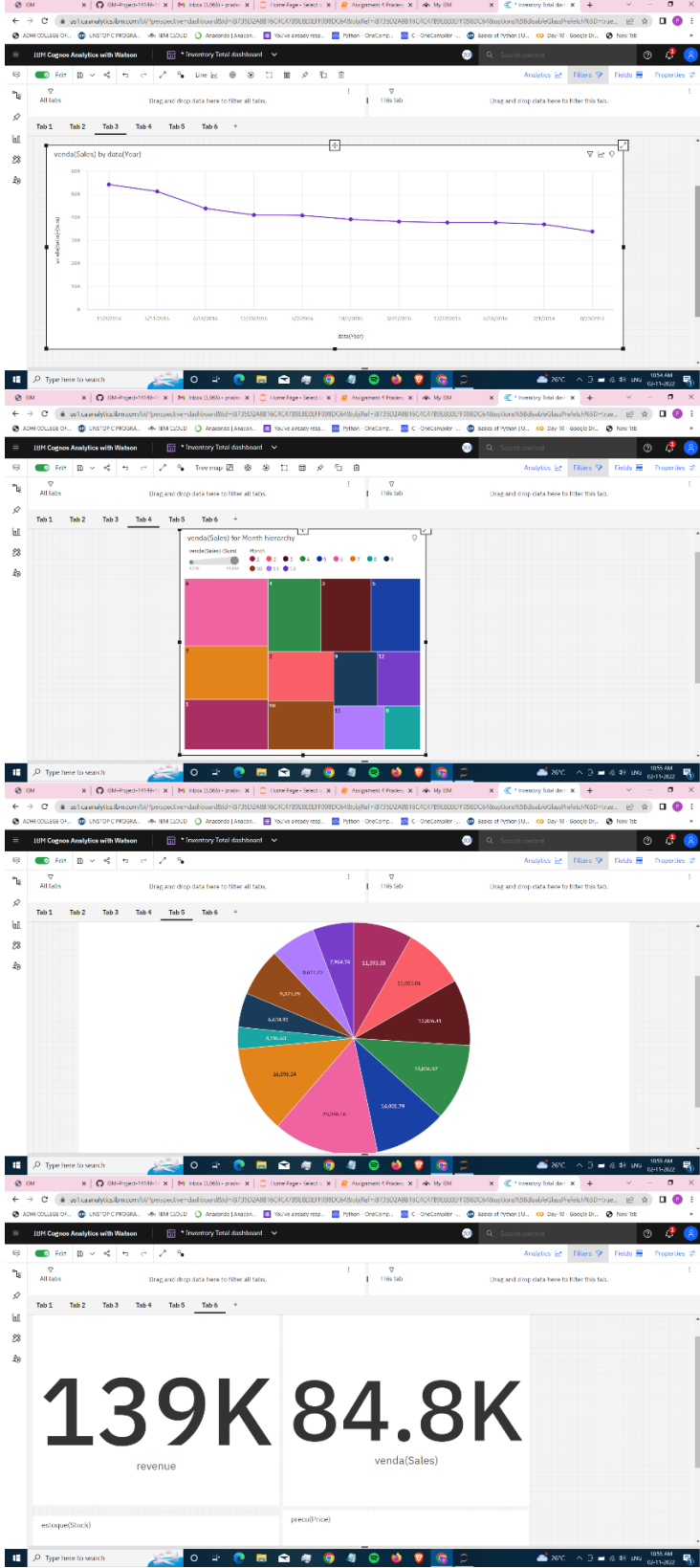
9. RESULTS


9.1 Performance Metrics

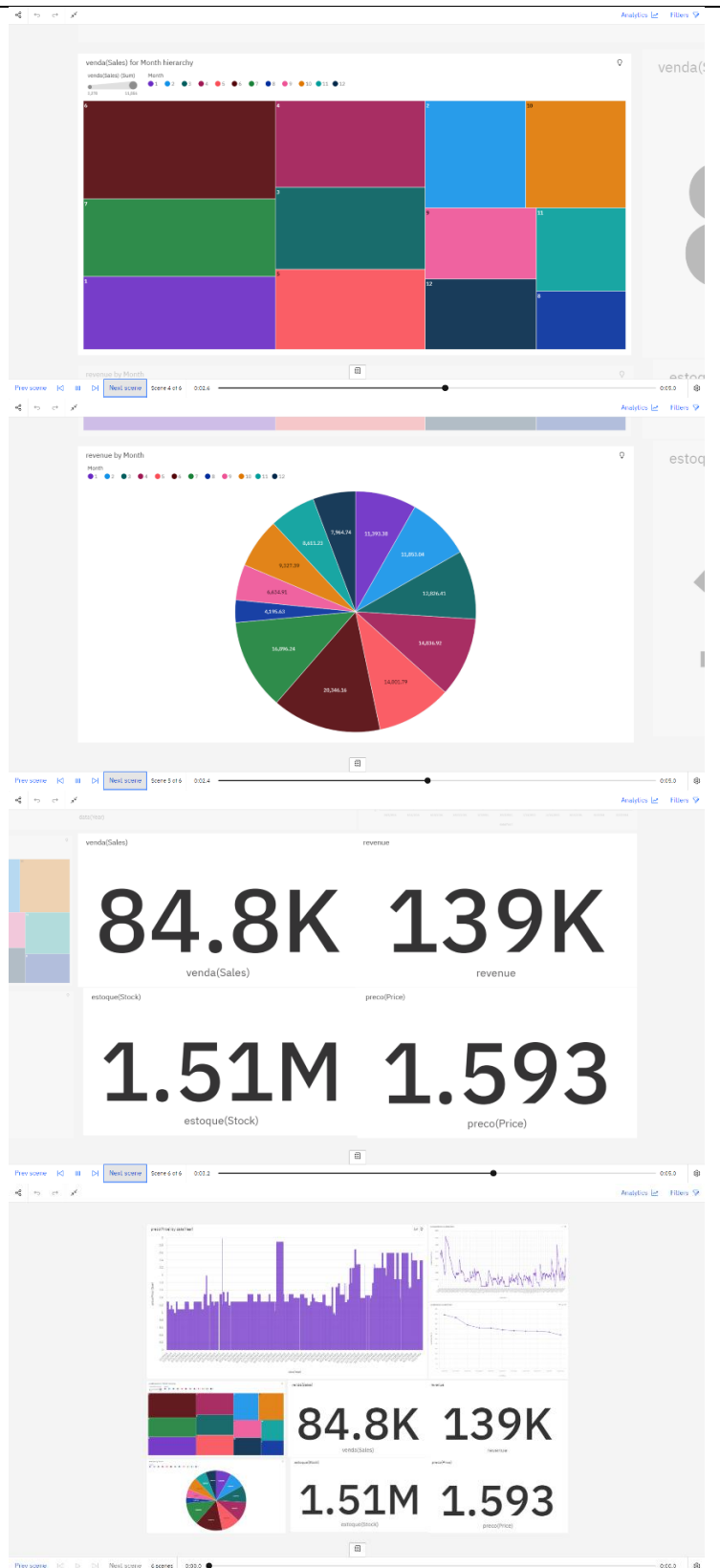
Model Performance Testing:

Project team shall fill the following information in model performance testing template.

S.No.	Parameter	Screenshot / Values
1.	Dashboard design	

		 <p>The figure displays four sequential screenshots of the IBM Cognos Analytics 'Inventory Total dashboard'. Each screenshot shows a different tab selected from a top navigation bar (Tab 1 to Tab 6).</p> <ul style="list-style-type: none"> Tab 1: A line chart titled 'venda(Sales) by data(Year)' showing a downward trend in sales from 2015 to 2018. The y-axis is labeled 'venda(Sales)' and ranges from 0 to 600. The x-axis is labeled 'data(Year)' and shows dates from 1/1/2015 to 12/31/2018. Tab 2: A treemap titled 'venda(Sales) by Month hierarchy'. It shows a hierarchical breakdown of sales by month, with colors representing different categories. The y-axis is labeled 'venda(Sales)' and ranges from 0 to 600. The x-axis is labeled 'data(Year)' and shows dates from 1/1/2015 to 12/31/2018. Tab 3: A pie chart titled 'venda(Sales) by Month hierarchy'. It shows the distribution of sales across different months. The chart is divided into 12 segments, each representing a month. The y-axis is labeled 'venda(Sales)' and ranges from 0 to 600. The x-axis is labeled 'data(Year)' and shows dates from 1/1/2015 to 12/31/2018. Tab 4: A KPI summary dashboard. It features two large KPI cards: '139K revenue' and '84.8K venda(Sales)'. Below these are two smaller KPI cards: 'estoque(Stock)' and 'preco(Price)'. The y-axis is labeled 'venda(Sales)' and ranges from 0 to 600. The x-axis is labeled 'data(Year)' and shows dates from 1/1/2015 to 12/31/2018.
2.	Data Responsiveness	Data is Very Responsive and Well Distributed.

3.	Amount Data to Rendered (DB2 Metrics)	Data in DB2 is about 500+ and it is rendered fastly.
4.	Utilization of Data Filters	Data Filters is used in our Data Sets as we used IBM cognos analytics and Exploratory Data Analysis.
5.	Effective User Story	 <p>The image displays three screenshots of an IBM Cognos Analytics dashboard, illustrating different data visualizations and filters. The top screenshot shows a bar chart titled 'precio(Price) by data(Year)' with a y-axis ranging from 0 to 3. The middle screenshot shows a line chart titled 'estoque(Stock) by data(Year)' with a y-axis ranging from 0 to 6,000. The bottom screenshot shows a line chart titled 'venda(Sales) by data(Year)' with a y-axis ranging from 0 to 400. Each chart includes a 'Next score' button and a 'Score 2 of 6' indicator. The dashboard also features a 'Filters' panel on the right side of each chart.</p>



6. Descriptive Reports 6 Visualizations used

10. 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. If you enter into a warehouse or a facility without proper inventory management, it is immediately apparent. Managers and business owners who do not implement an effective inventory management system usually have troubles keeping track of assets and executing work order so they just end up not being kept in the right place. It is not just an eyesore but it can also lead to a variety of safety hazards as well.
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 box of washers or elbows at no charge to the customer without raising any alarms. This is done for the sake of customer 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 material feel more 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 lead to 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 space so 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. The optimum level of inventory for a business could still be a lot of goods and they will need space to be stored in and in some cases additional operational costs to manage the inventory. This will in turn increase cost and impact negatively on the amount of profit the business makes.
5. **Complexity:** some methods and strategies of inventory management can be relatively complex and difficult to understand on the part of the staff. This may result in the need for employees to undergo training in order to grasp how the system works.
6. Some inventory management systems such as the fixed order period system compels a periodic review of all items. This itself makes the system a bit inefficient.
7. **High implementation costs:** some inventory management systems can come at a high price because the business needs to install specialized systems and software in order to use them. This can be problematic for large businesses which operate in difficult locations. Even after installing the costly system, it still needs to be maintained and upgraded on a regular basis, thus incurring more costs.
8. Even with an efficient inventory management method, you can control but not eliminate business risk.
9. The control of inventory is complex because of the many functions it performs. It should thus be viewed as a shared responsibility.
10. Holding inventory can result to a greater risk of loss to devaluation (changes in price).

11. CONCLUSION

To conclude, Inventory Management System is a simple desktop-based application basically suitable for small organization. It has every basic items which are used for the small organization. Our team is successful in making the application where we can update, insert and delete the item as per the requirement. This application also provides a simple report on daily basis to know the daily sales and purchase details. This application matches for small organization where there small limited go downs. Through it has some limitations, our team strongly believes that the implementation of this system will surely benefit the organization.

12. FUTURE SCOPE

1. The Fourth Industrial Revolution will continue to drive technological change that will impact the way that we manage inventories.
2. Successful companies will view inventory as a strategic asset, rather than an aggravating expense or an evil to be tolerated.
3. Collaboration with supply chain partners, coupled with a holistic approach to supply chain management, will be key to effective inventory management.
4. The nature of globalization will change, impacting inventory deployment decisions dramatically.
5. Increased focus on supply chain security, and concerns about the quality of inventory itself, will be primary motivators to changing supply chain and inventory strategy.

13. APPENDIX

Source code

File name - index.html

```
<!DOCTYPE html>
<html>
<head>
<title>Sales Inventory Login</title>
<meta name="viewport" content="width=device-width, initial-scale=1">
<style>
body {
  background-image: url('logi111.jpg');
}
/* Style all input fields */
input {
  width: 100%;
  padding: 12px;
  border: 1px solid #ccc;
  border-radius: 4px;
  box-sizing: border-box;
  margin-top: 6px;
  margin-bottom: 16px;
}

/* Style the submit button */
input[type=submit] {
  background-color: #04AA6D;
  color: white;
}

/* Style the container for inputs */
```

```

.container {
  background-color: #f1f1f1;
  padding: 20px;
}

/* The message box is shown when the user clicks on the password field */
#message {
  display:none;
  background: #f1f1f1;
  color: #000;
  position: relative;
  padding: 20px;
  margin-top: 10px;
}

#message p {
  padding: 10px 35px;
  font-size: 18px;
}

/* Add a green text color and a checkmark when the requirements are right */
.valid {
  color: green;
}

.valid:before {
  position: relative;
  left: -35px;
  content: "✓";
}

/* Add a red text color and an "x" when the requirements are wrong */
.invalid {
  color: red;
}

.invalid:before {
  position: relative;
  left: -35px;
  content: "✗";
}
</style>
</head>
<body>
<h1 align="center" style="color:#FFFFFF">Sales Login</h3>
<h3 align="center"></h3>

```

```

<div class="container">
  <form action="/action_page.php">
    <label for="username">Username</label>
    <input type="text" id="username" name="username" required>

    <label for="psw">Password</label>
    <input type="password" id="psw" name="psw" pattern="(?=.*\d)(?=.*[a-z])(?=.*[A-Z]).{8,}"
    title="Must contain at least one number and one uppercase and lowercase letter, and at least 8 or more
    characters" required>

    <center><a href="story1.html">Login</a></center>
  </form>
</div>

<div id="message">
  <h3>Password must contain the following:</h3>
  <p id="letter" class="invalid">A <b>lowercase</b> letter</p>
  <p id="capital" class="invalid">A <b>capital (uppercase)</b> letter</p>
  <p id="number" class="invalid">A <b>number</b></p>
  <p id="length" class="invalid">Minimum <b>8 characters</b></p>
</div>

<script>
var myInput = document.getElementById("psw");
var letter = document.getElementById("letter");
var capital = document.getElementById("capital");
var number = document.getElementById("number");
var length = document.getElementById("length");

// When the user clicks on the password field, show the message box
myInput.onfocus = function() {
document.getElementById("message").style.display = "block";
}

// When the user clicks outside of the password field, hide the message box
myInput.onblur = function() {
document.getElementById("message").style.display = "none";
}

// When the user starts to type something inside the password field
myInput.onkeyup = function() {
  // Validate lowercase letters
  var lowerCaseLetters = /[a-z]/g;
  if(myInput.value.match(lowerCaseLetters)) {
    letter.classList.remove("invalid");
    letter.classList.add("valid");
  } else {

```

```

letter.classList.remove("valid");
letter.classList.add("invalid");
}

// Validate capital letters
var upperCaseLetters = /[A-Z]/g;
if(myInput.value.match(upperCaseLetters)) {
capital.classList.remove("invalid");
capital.classList.add("valid");
} else {
capital.classList.remove("valid");
capital.classList.add("invalid");
}

// Validate numbers
var numbers = /[0-9]/g;
if(myInput.value.match(numbers)) {
number.classList.remove("invalid");
number.classList.add("valid");
} else {
number.classList.remove("valid");
number.classList.add("invalid");
}

// Validate length
if(myInput.value.length >= 8) {
length.classList.remove("invalid");
length.classList.add("valid");
} else {
length.classList.remove("valid");
length.classList.add("invalid");
}
}
</script>
</body>
</html>

```

File name - app.py

```

import ibm_db
import re
from flask import Flask, render_template, request, redirect, url_for, session, flash

app = Flask(__name__)
app.secret_key="i don't care"

```

```
myconn=ibm_db.connect('DATABASE=bludb;HOSTNAME=b1bc1829-6f45-4cd4-bef4-10cf081900bf.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=32304;SECURITY=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=ytl86182;PWD=IwLcHrD3CEMU2chS', ", ")
```

```
@app.route("/signup")
```

```
def signup():
```

```
    return render_template("signup.html")
```

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

```
def register():
```

```
    msg = "
```

```
    if request.method == 'POST' :
```

```
        username = request.form['username']
```

```
        password = request.form['password']
```

```
        query = 'SELECT * FROM admin WHERE username =?;'
```

```
        stmt=ibm_db.prepare(myconn,query)
```

```
        ibm_db.bind_param(stmt,1,username)
```

```
        ibm_db.execute(stmt)
```

```
        account = ibm_db.fetch_assoc(stmt)
```

```
        print(account)
```

```
        if account:
```

```
            msg = 'Account already exists !'
```

```
        elif not re.match(r'[a-z0-9]+', username):
```

```
            msg = 'name must contain only lowercase characters and numbers !'
```

```
        else:
```

```
            query = "INSERT INTO ADMIN (username,password) VALUES (?,?)"
```

```
            stmt=ibm_db.prepare(myconn,query)
```

```
            ibm_db.bind_param(stmt,1,username)
```

```
            ibm_db.bind_param(stmt,2,password)
```

```
            ibm_db.execute(stmt)
```

```
            msg = 'You have successfully registered !'
```

```
            return render_template('login.html', msg = msg)
```

```
@app.route("/")
```

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

```
def login():
```

```
    if request.method=="POST":
```

```
        Username=request.form['Username']
```

```
        Password=request.form['Password']
```

```
        query="select * from admin where username=? and password=?;"
```

```
        stmt=ibm_db.prepare(myconn, query)
```

```
        ibm_db.bind_param(stmt, 1, Username)
```

```
        ibm_db.bind_param(stmt, 2, Password)
```

```

        ibm_db.execute(stmt)
        data=ibm_db.fetch_assoc(stmt)
        if data:
            session['loggedin']=True
            return render_template('cognos.html')

        else:
            flash("Incorrect Username or Password")
    return render_template("login.html")

```

```

if _name_ == "_main_":
    app.run(debug=True)

```

File name - login.html

```

<!DOCTYPE html>
<html>

<head>
    <title>Login Form</title>
    <link rel="stylesheet" type="text/css" href="..\static\css\login.css">
    <link href="https://fonts.googleapis.com/css?family=Poppins:600&display=swap" rel="stylesheet">
    <script src="https://kit.fontawesome.com/a81368914c.js"></script>
    <meta name="viewport" content="width=device-width, initial-scale=1">

</head>

<body>

    <!--  -->
    <div class="container">

        <!-- <div class="img">
            <div id="png"><a href="/" title="HOME"></a></div>
                
            </div> -->

        <br>
        <br>
        <br>
        <div class="login-content">

```

```

<form action="{{ url_for('login') }}" method="POST">
  <div class="msg">{{ msg }}</div>
  
  <h2 class="title">Welcome</h2>
  <div class="input-div one">
    <div class="i">
      <i class="fas fa-user"></i>
    </div>
    <div class="div">

      <input type="text" name="Username" placeholder="username" class="input" required>
    </div>
  </div>
  <div class="input-div pass">
    <div class="i">
      <i class="fas fa-lock"></i>
    </div>
    <div class="div">

      <input type="password" name="Password" placeholder="password" class="input" required>
    </div>
  </div>
  <div class="btn">
    <button type="login" class="btn btn-default">Login</button>
  </div><br><br><br><br>

  <div class="app"><b>Don't have an account?</b><a id="app1" href="\signup">Signup Here
!</a></div>
</form>

</div>

</div>

<script type="text/javascript" src="..\static\js\login.js"></script>
</body>
</html>

```

File name - salesdashboard.html

```

<html>
<head>
<title>

```

Data Demo

```
</title>
<meta charset="utf-8">
<meta name="viewport" content="width=device.width, initial-scale=1">
<link                href="https://cdn.jsdelivr.net/npm/bootstrap@S.2.1/dist/css/bootstrap.min.css"
rel="stylesheet">
<script src="https://cdn.jsdelivr.net/npm/bootstrap@S.2.1/dist/js/bundle.min.js"></script>
</head>
<body>
<div class="container-fluid p-5 bg-primary text-white text-center">
  <center><h1>Sales Dashboard</h1></center>
</div>
<center><iframe
src="https://us1.ca.analytics.ibm.com/bi/?perspective=dashboard&amp;pathRef=.my_folders%2FInve
ntory%2BTotals%2Bdashboard&amp;closeWindowOnLastView=true&amp;ui_appbar=false&amp;ui
_navbar=false&amp;shareMode=embedded&amp;action=view&amp;mode=dashboard&amp;subVie
w=model00000183fadcd3b_00000000"      width="768"      height="1024"      frameborder="0"
gesture="media" allow="encrypted-media" allowfullscreen=""></iframe></center>
</body>
</html>
```

File name - salesstory.html

```
<html>
<head>

<title>
Sales Story
</title>
<style>
body{
  background-image: url('homepage111.webp');
}
</style>

<meta charset="utf-8">
<meta name="viewport" content="width=device.width, initial-scale=1">
<link                href="https://cdn.jsdelivr.net/npm/bootstrap@S.2.1/dist/css/bootstrap.min.css"
rel="stylesheet">

<script src="https://cdn.jsdelivr.net/npm/bootstrap@S.2.1/dist/js/bundle.min.js"></script>

</head>

<body>

<div class="container-fluid p-5 bg-primary text-white text-center">
  <center style="color:#FFFFFF"><h1>Sales Story</h1></center>
```


</div>

```
<center><iframe
src="https://us1.ca.analytics.ibm.com/bi/?perspective=story&pathRef=.my_folders%2FInventory
%2Btotal%2Bstory&closeWindowOnLastView=true&ui_appbar=false&ui_navbar=false&shareMode=embedded&action=view&sceneId=-1&sceneTime=0"
width="1080" height="1920" frameborder="0" gesture="media" allow="encrypted-media"
allowfullscreen=""></iframe></center>
</body>
</html>
```

GitHub & Project Demo Link

GitHub Link

<https://github.com/IBM-EPBL/IBM-Project-32691-1660211446>