Project Documentation

Date	19 November 2022
Team ID	PNT2022TMID03890
Project Name	Smart Fashion Recommender Application

1. **INTRODUCTION**

1.1 Project Overview

Fashion is perceived as a meaningful way of self-expressing that people use for different purposes Fashionable products are highly demanded, and consequently, fashion is perceived as a desirable and profitable industry. Although this massive demand for fashion products provides an excellent opportunity for companies to invest in fashion-related sectors, it also faces different challenges in answereing their customer needs. Fashion recommender systems have been introduced to address these needs. Fashion is perceived as a meaningful way of self-expressing that people use for different purposes Fashionable products are highly demanded, and consequently, fashion is perceived as a desirable and profitable industry. Although this massive demand for fashion products provides an excellent opportunity for companies to invest in fashion-related sectors, it also faces different challenges in answereing their customer needs. Fashion recommender systems have been introduced to address these needs.

1.2 Purpose

Physical stores are no longer the only way that shoppers are interacting with brands, especially fashion retailers. As shoppers increasingly expect mobile purchasing, one-day shipping and 24/7 customer support, fashion retailers are redesigning the entire shopping experience.

Luckily, the fashion industry is already a master of change, with seasonal trends coming and going before you can even take a breath. However, the fashion world’s digital transformation may be its greatest change yet.

From larger luxury brands to smaller names like Modern Vintage Boutique, retailers across the industry are tailoring their ecommerce strategies to attract customers and adapt to the evolving landscape.

1. **LITERATURE SURVEY**

1.1 Existing problem

People are choosing the internet to satiate their fashion needs, while the ecommerce fashion industry is trying its best to adjust to the changing consumer needs.

Despite the growing market and increasing demand for ecommerce fashion, brands within this space face unique challenges when it comes to optimising their websites and connecting effectively with customers. The four main challenges in ecommerce fashion:

- Keyword Targeting

- Competitive Differentiation
- Crawlability & Indexability
- Budgets & Resourcing

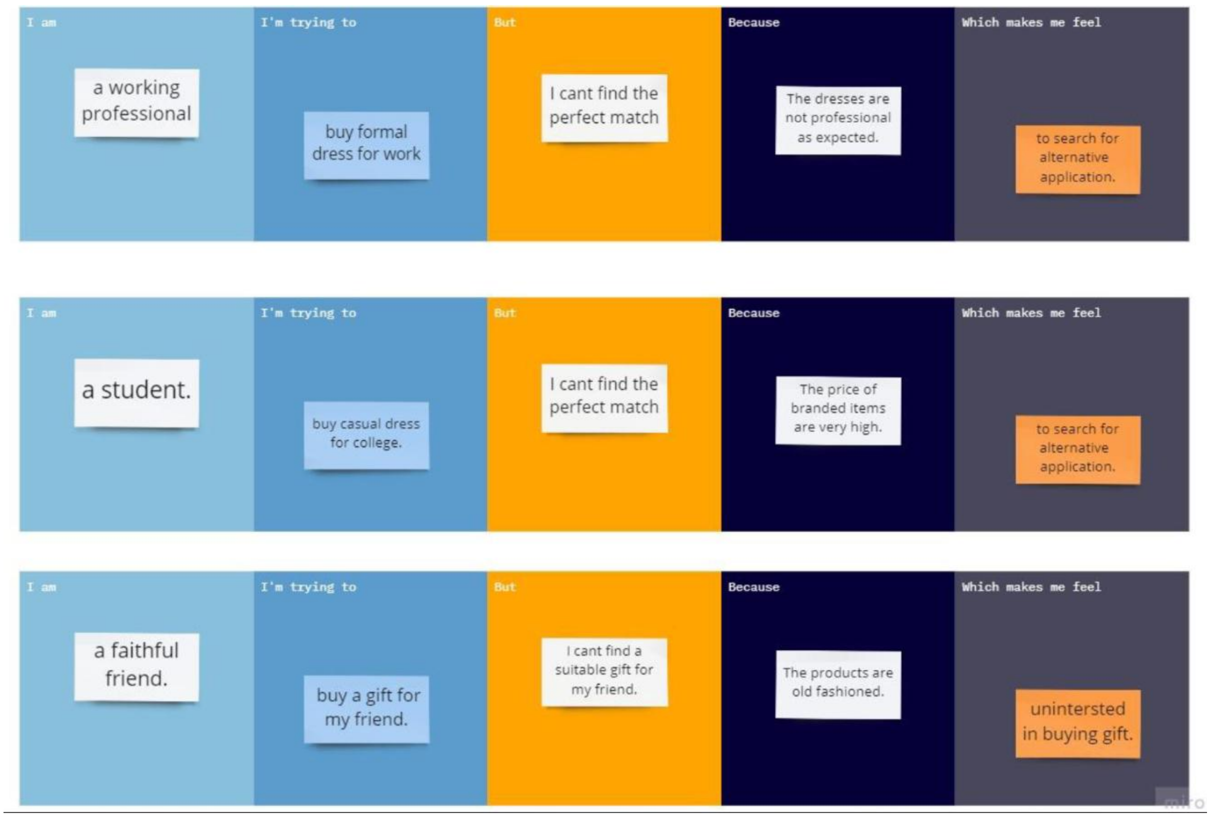
1.2 References

1. Paper Title: A COMPREHENSIVE REVIEW ON ONLINE FASHION RECOMMENDATION
Publication: December 2020 Author name: Samit Chakraborty
2. Paper Title: Image-based fashion recommender system.
Publication: Year (2021).
Author name: Shaghayegh Shirkhani.
3.Paper Title: Fashion Recommendation Systems
Author name: Samit Chakraborty , Md. Saiful Hoque, Naimur Rahman Jeem, Manik Chandra Biswas, Deepayan Bardhan and Edger Lobaton.
4.Paper Title: A Review onClothes Matching and Recommendation System Based on User
Attributes
Author name: Atharv Pandit , Kunal Goel , Manav Jain , Neha Katre
5.Paper Title: Individualized fashion recommender system
Year: 10 October 2020
Author name: M Sridevi, N ManikyaArun, MSheshikala and E Sudarshan

1.3 Problem Statement Definition

Physical stores are no longer the only way that shoppers are interacting with brands, especially fashion retailers. As shoppers increasingly expect mobile purchasing, one-day shipping and 24/7 customer support, fashion retailers are redesigning the entire shopping experience.

Fashion is perceived as a meaningful way of self-expressing that people use for different purposes Fashionable products are highly demanded, and consequently, fashion is perceived as a desirable and profitable industry. Although this massive demand for fashion products provides an excellent opportunity for companies to invest in fashion-related sectors, it also faces different challenges in answereing their customer needs. Fashion recommender systems have been introduced to address these needs.



I am	I'm trying to	But	Because	Which makes me feel
a working professional	buy formal dress for work	I cant find the perfect match	The dresses are not professional as expected.	to search for alternative application.

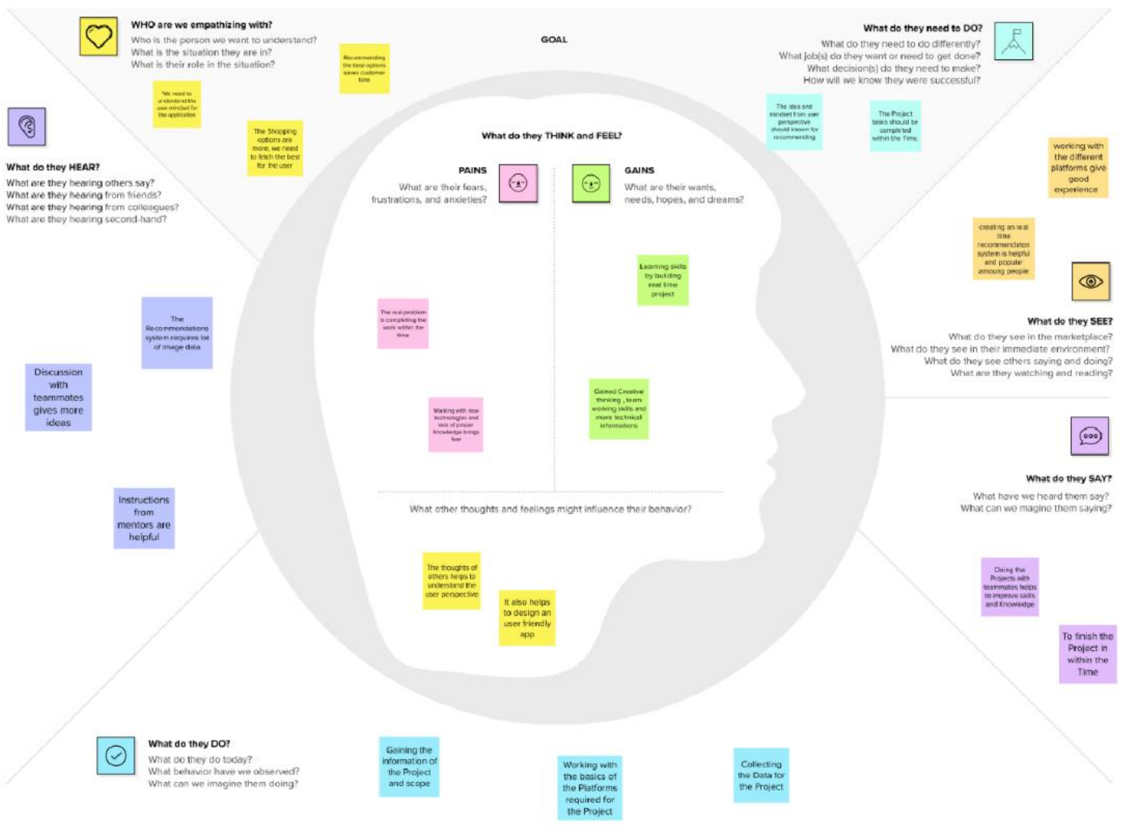
I am	I'm trying to	But	Because	Which makes me feel
a student.	buy casual dress for college.	I cant find the perfect match	The price of branded items are very high.	to search for alternative application.

I am	I'm trying to	But	Because	Which makes me feel
a faithful friend.	buy a gift for my friend.	I cant find a suitable gift for my friend.	The products are old fashioned.	unintersted in buying gift.

1. **IDEATION & PROPOSED SOLUTION**

1.1 Empathy Map Canvas

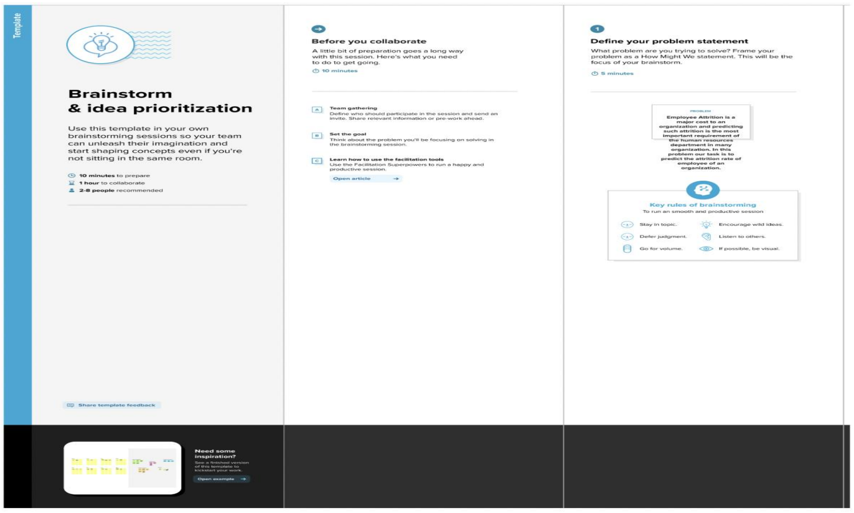
An empathy map is a simple, easy-to-digest visual that captures knowledge about a user's behaviours and attitudes. It is a useful tool to help teams better understand their users. Creating an effective solution requires understanding the true problem and the person who is experiencing it. The exercise of creating the map helps participants consider things from the user's perspective along with his or her goals and challenges.


WHO are we empathizing with?
Who is the person we want to understand?
What is the situation they are in?
What is their role in the situation?
GOAL
What do they need to DO?
What do they need to do differently?
What job(s) do they want or need to get done?
What decision(s) do they need to make?
How will we know they were successful?
We need to understand the user mindset for the application
Recommending the best options saves customer time
The Shopping options are more, we need to fetch the best for the user
The idea and mindset from user perspective should known for recommending
The Project tasks should be completed within the Time.
What do they HEAR?
What are they hearing others say?
What are they hearing from friends?
What are they hearing from colleagues?
What are they hearing second-hand?
What do they THINK and FEEL?
PAINS
What are their fears, frustrations, and anxieties?
GAINS
What are their wants, needs, hopes, and dreams?
working with the different platforms give good experience
creating an real time recommendation system is helpful and popular amoung people
Learning skills by building real time project
The real problem is completing the work within the time
The Recommendations system requires lot of image data
Discussion with teammates gives more ideas
What do they SEE?
What do they see in the marketplace?
What do they see in their immediate environment?
What do they see others saying and doing?
What are they watching and reading?
Working with new technologies and lack of proper Knowledge brings fear
Gained Creative thinking , team working skills and more technical informations
What do they SAY?
What have we heard them say?
What can we magine them saying?
Instructions from mentors are helpful
What other thoughts and feelings might influence their behavior?
The thoughts of others helps to understand the user perspective
It also helps to design an user friendly app
Doing the Projects with teammates helps to improve skills and Knowledge
To finish the Project in within the Time
What do they DO?
What do they do today?
What behavior have we observed?
What can we imagine them doing?
Gaining the information of the Project and scope
Working with the basics of the Platforms required for the Project
Collecting the Data for the Project

1.2 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. Prioritising 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. Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

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


Template

Brainstorm & idea prioritization

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

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

Share template feedback

Need some inspiration? See a finished version of this template to kickstart your work. Open example

Before you collaborate

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

10 minutes

A **Team gathering** Define who should participate in the session and send an invite. Share relevant information or pre-work ahead.

B **Set the goal** Think about the problem you'll be focusing on solving in the brainstorming session.

C **Learn how to use the facilitation tools** Use the Facilitation Superpowers to run a happy and productive session.

Open article

1 **Define your problem statement**

What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.

5 minutes

PROBLEM

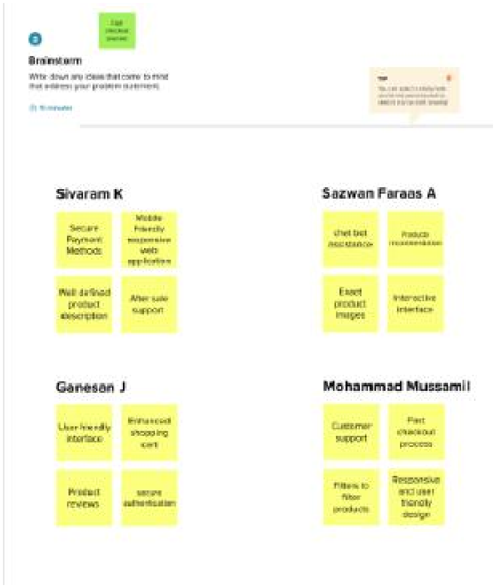
Employee Attrition is a major cost to an organization and predicting such attrition is the most important requirement of the human resources department in many organization. In this problem our task is to predict the attrition rate of employee of an organization.

Key rules of brainstorming

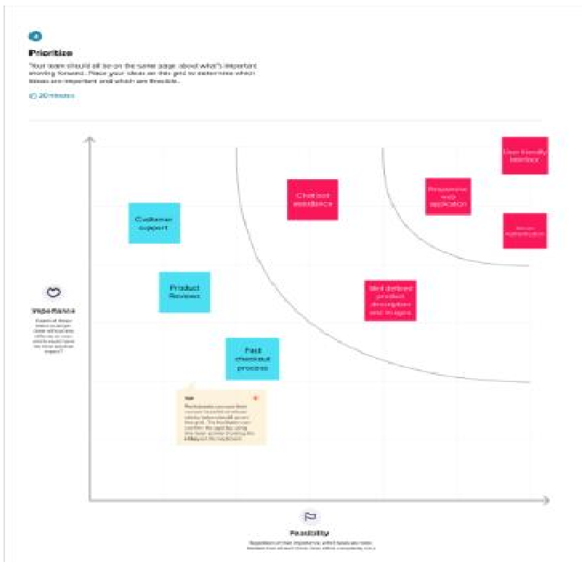
To run an smooth and productive session

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

Step-2: Brainstorm, Idea Listing and Grouping



Step-3: Idea Prioritisation

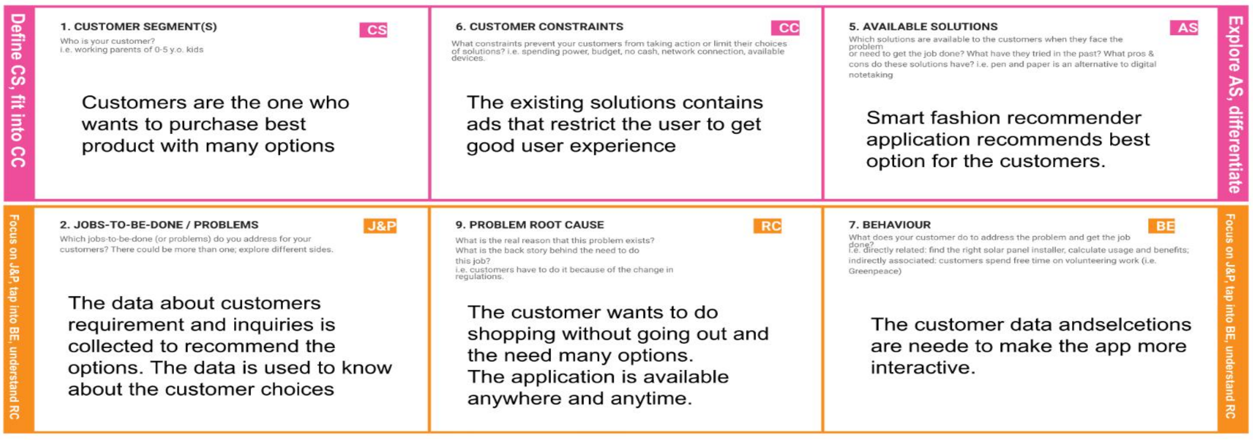


1.3 Proposed Solution

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	When users land on the application, they expect to find what they are looking for quickly and easily. Also, users are not sure about the brands or the actual products they want to purchase. They have a very broad idea about what they want to buy.
2.	Idea / Solution description	By using Smart fashion recommender application: Effective recommendation of products. Recommendation within a single page via chat-bot Collect feedback instantly. Reduce human error Proper guidance in accessing application.
3.	Novelty / Uniqueness	Chat-bot asks and learns from user preference which recommends appropriate products to the user without making them to search through various filters. Reduces time in choosing right product thus increases sales.
4.	Social Impact / Customer Satisfaction	Feedback from the user at the end of session or after placing order is one of the most important factor in deriving customer satisfaction and providing better services.
5.	Business Model (Revenue Model)	The application can be developed at minimum cost with high performance and interactive user interface.
6.	Scalability of the Solution	The solution can be made scalable by using micro service architecture provided that each server responsible for certain functionality of the application. Storing user preferences along with product in browser cookie will enable to provide response instantly and allows for fetching related products.

1.4 Problem Solution fit

Define CS, fit into CC



Define CS, fit into CC
Explore AS, differentiate
Focus on J&P, tap into BE, understand RC
CS
CC
AS
J&P
RC
BE

1. CUSTOMER SEGMENT(S)

Who is your customer?
i.e. working parents of 0-5 y.o. kids

Customers are the one who wants to purchase best product with many options

6. CUSTOMER CONSTRAINTS

What constraints prevent your customers from taking action or limit their choices of solutions? i.e. spending power, budget, no cash, network connection, available devices.

The existing solutions contains ads that restrict the user to get good user experience

5. AVAILABLE SOLUTIONS

Which solutions are available to the customers when they face the problem or need to get the job done? What have they tried in the past? What pros & cons do these solutions have? i.e. pen and paper is an alternative to digital notetaking

Smart fashion recommender application recommends best option for the customers.

2. JOBS-TO-BE-DONE / PROBLEMS

Which jobs-to-be-done (or problems) do you address for your customers? There could be more than one; explore different sides.

The data about customers requirement and inquiries is collected to recommend the options. The data is used to know about the customer choices

9. PROBLEM ROOT CAUSE

What is the real reason that this problem exists? What is the back story behind the need to do this job?
i.e. customers have to do it because of the change in regulations.

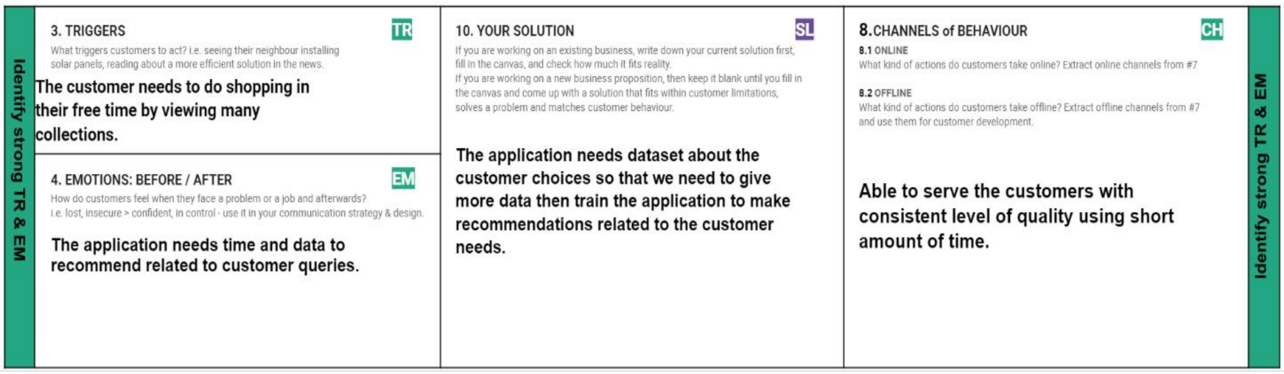
The customer wants to do shopping without going out and the need many options.
The application is available anywhere and anytime.

7. BEHAVIOUR

What does your customer do to address the problem and get the job done?
i.e. directly related: find the right solar panel installer, calculate usage and benefits; indirectly associated: customers spend free time on volunteering work (i.e. Greenpeace)

The customer data andselcetions are neede to make the app more interactive.

Identify Strong TR & EM


Identify strong TR & EM

3. TRIGGERS TR

What triggers customers to act? i.e. seeing their neighbour installing solar panels, reading about a more efficient solution in the news.

The customer needs to do shopping in their free time by viewing many collections.

4. EMOTIONS: BEFORE / AFTER EM

How do customers feel when they face a problem or a job and afterwards? i.e. lost, insecure > confident, in control - use it in your communication strategy & design.

The application needs time and data to recommend related to customer queries.

10. YOUR SOLUTION SL

If you are working on an existing business, write down your current solution first, fill in the canvas, and check how much it fits reality.
If you are working on a new business proposition, then keep it blank until you fill in the canvas and come up with a solution that fits within customer limitations, solves a problem and matches customer behaviour.

The application needs dataset about the customer choices so that we need to give more data then train the application to make recommendations related to the customer needs.

8.CHANNELS of BEHAVIOUR CH

8.1 ONLINE
What kind of actions do customers take online? Extract online channels from #7

8.2 OFFLINE
What kind of actions do customers take offline? Extract offline channels from #7 and use them for customer development.

Able to serve the customers with consistent level of quality using short amount of time.

Identify strong TR & EM

2. REQUIREMENT ANALYSIS

2.1 Functional requirement

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Gmail Id and Password Registration through form by entering details like Name, Mobile Number etc.
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	Searching the Product	Searching and filtering the product by filters provided in the provided in the App.
FR-4	Assistance by Chatbot	Chatbot Assistance is Provided for easy working with interface and searching for suitable product.
FR-5	Shopping the Product	This App contains the following options: 1. Adding the Product in Cart. 2. Remove from the Cart. 3. Adding to Wishlist. 4. Remove from Wishlist. 5. Cancel the Product. 6. Returning the Product.
FR-6	Make Payment	Payment Processing is done via UPI, Internet Banking etc.
FR-7	Confirmation	Confirmation of the product and tracking the product services is also provided.
FR-8	Return Policy	Return Policy is provide if the customer is not satisfied by the product.

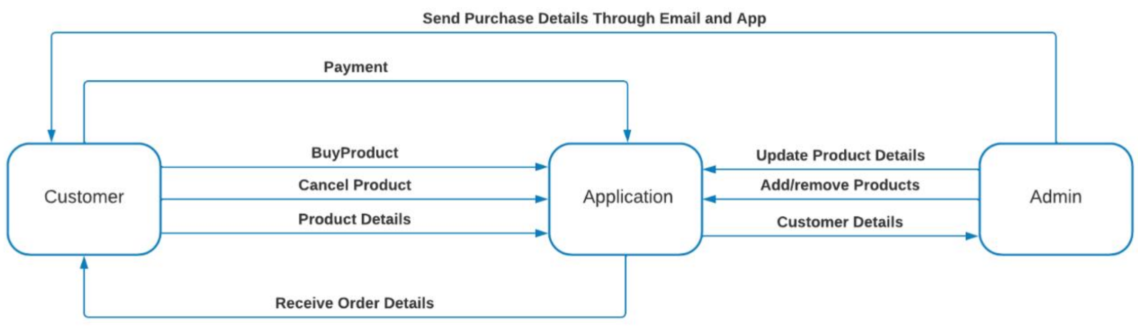
2.2 Non-Functional requirements

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	Usability of the app relies on the smart recommendation system and also the chatbot application for ease of use and to increase the interface of the app.
NFR-2	**Security**	Security is provided by the HTTPS encryption of the transferred data packets.
NFR-3	**Reliability**	Reliability is achieved by the cloud storage for high traffic and to improve efficiency of the app.
NFR-4	**Performance**	Cloud services provide the best performance and also high quality plugins are used.
NFR-5	**Availability**	The app provides services 24X7 and it is available anytime.
NFR-6	**Scalability**	The app is scalable extensive to all kinds of mobile phone users.

3. PROJECT DESIGN

3.1 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.



3.2 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.

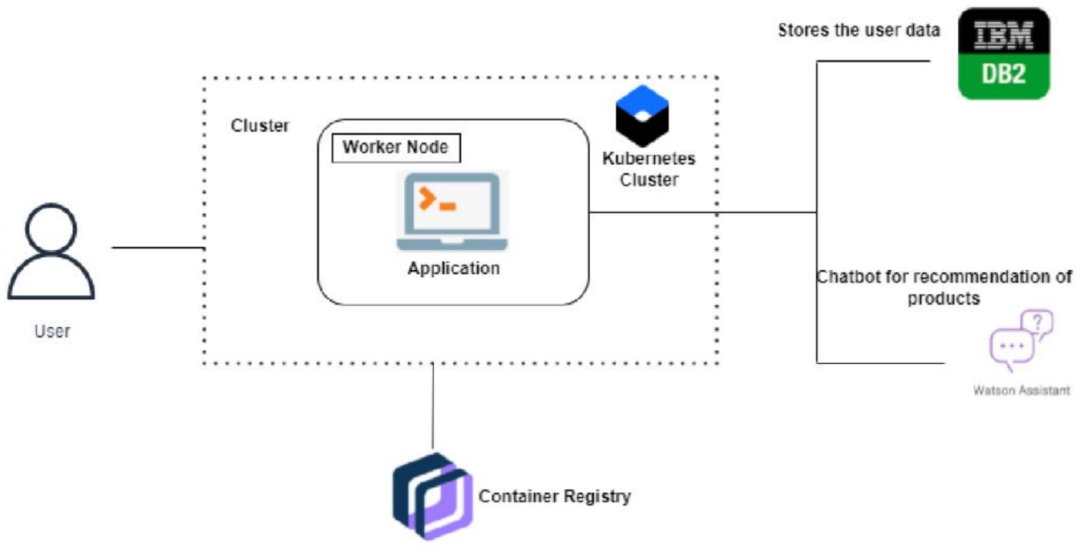


Fig. Technical Architecture Diagram

3.3 User Stories

A user story is an informal, general explanation of a software feature written from the perspective of the end user. Its purpose is to articulate how a software feature will provide value to the customer.

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
		USN-3	As a user, I can register for the application through Facebook	I can register & access the dashboard with Facebook Login	Low	Sprint-2
		USN-4	As a user, I can register for the application through Gmail	I can register and access the dashboard with Gmail login	Medium	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering email & password	Log into website using email and password	High	Sprint-1
	Dashboard	USN-6	As a user I can access the dashboard of App by logging in.	By logging in,can access the dashboard of the website	High	Sprint-1
Customer (Web user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
		USN-3	As a user, I can register for the application through Facebook	I can register & access the dashboard with Facebook Login	Low	Sprint-2
		USN-4	As a user, I can register for the application through Gmail	I can register and access the dashboard with Gmail login	Medium	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering email & password	Log into website using email and password	High	Sprint-1

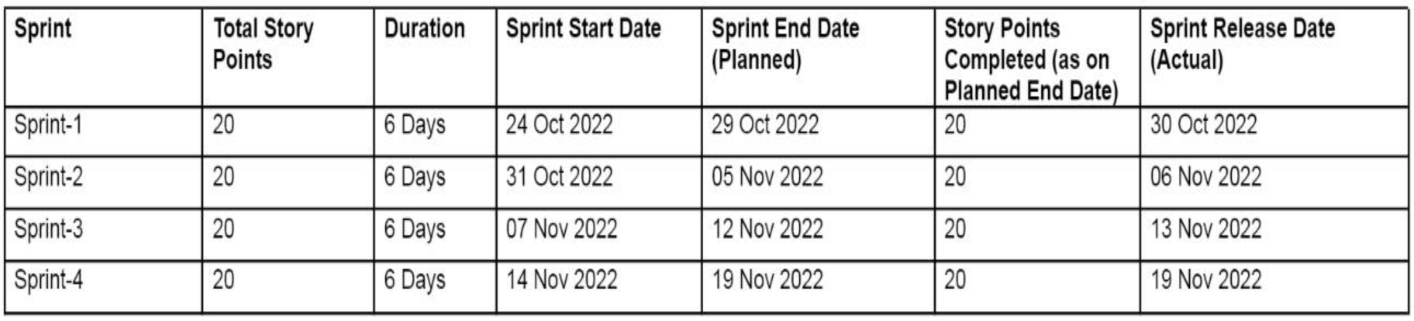
User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
	Dashboard	USN-6	As a user can access the dashboard after logging in and registration of the website.	As a user can access the dashboard after logging in and registration of the website.	High	Sprint-1
Customer Care Executive	Login	USN-1	As a Customer care executive,I will login to the website by entering my respective email and password.	I can login to the website by entering my executive email id and password.	Low	Sprint-2
	Dashboard	USN-2	As a Customer care executive,I can access the dashboard of the website by logging in.	I can access the dashboard of the website by logging in.	Low	Sprint-2
	Service	USN-3	As a Customer care executive,I can access the customer service page of the website by logging in.	I can access the customer service page of the website by logging in.	Low	Sprint-2
Administrator	Login	USN-1	As a administrator,I will login to the website by entering my respective email and password.	I can login to the website by entering my executive email id and password.	High	Sprint-1
	Dashboard	USN-2	As a Administrator ,I can access the dashboard of the website by logging in.	I can access the dashboard of the website by logging in.	High	Sprint-1
	Service	USN-3	As a administrator ,I can access the administration service page of the website by logging in.	I can access the administration service page of the website by logging in.	High	Sprint-1

4. PROJECT PLANNING & SCHEDULING

Sprint Planning & Estimation

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	2	High	Sivaram Mohamed Musammil Ganeshan
Sprint-1		USN-2	As a user, I will receive confirmation email once I have registered for the application	1	High	Sivaram Ganeshan Sazwan Faraas
Sprint-2		USN-3	As a user, I can register for the application through Facebook	2	Low	Ganeshan Sazwaan Faraas
Sprint-1		USN-4	As a user, I can register for the application through Gmail	2	Medium	Sivaram Mohamed Musammil Sazwaan Faraas
Sprint-1	Login	USN-5	As a user, I can log into the application by entering email & password	1	High	Mohamed Musammil Sivaram Ganeshan
	Dashboard	USN-6	As a user can see and filter the product using Chatbot	2	High	Mohamed Musammil Sazwan Faraas Sivaram Ganeshan
	Admin Side	USN-7	As an admin can access add/remove items	2	High	Mohamed Musammil Sazwan Faraas Sivaram
	Payment	USN-8	As a user can purchase and process the payment	1	Medium	Mohamed Musammil Sazwan Faraas Ganeshan

4.1 Sprint Delivery Schedule

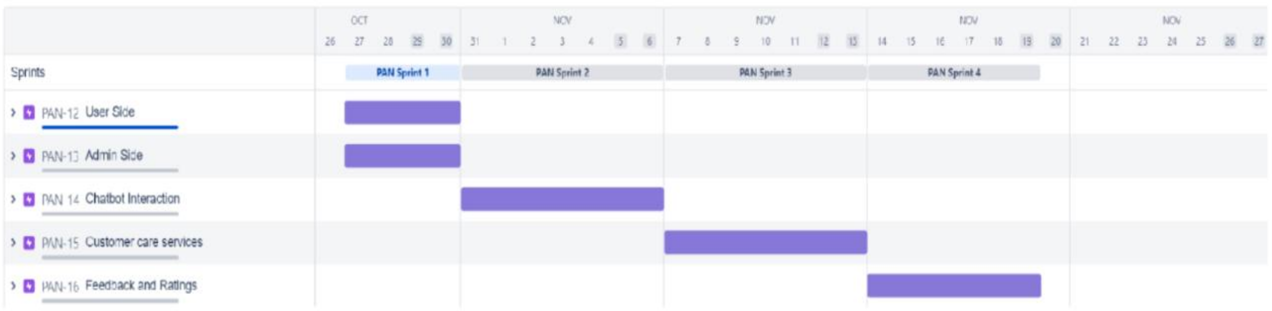


Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	30 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	06 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	13 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

Velocity:

Sprints	**Sprint Duration**	**Velocity**	**Actual Velocity**
Sprint-1	6	7	1.16
Sprint-2	6	13	2.16
Sprint-3	6	14	2.33
Sprint-4	6	15	2.5

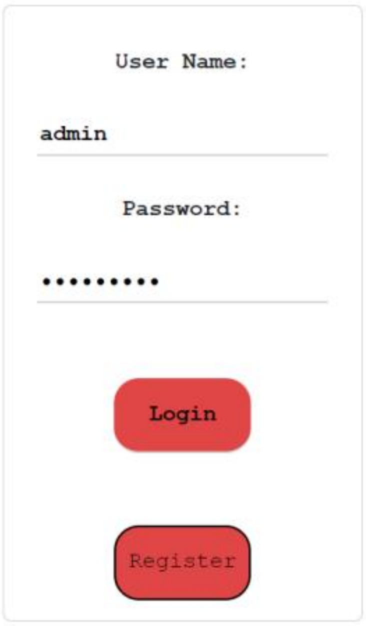
4.2 Reports from JIRA

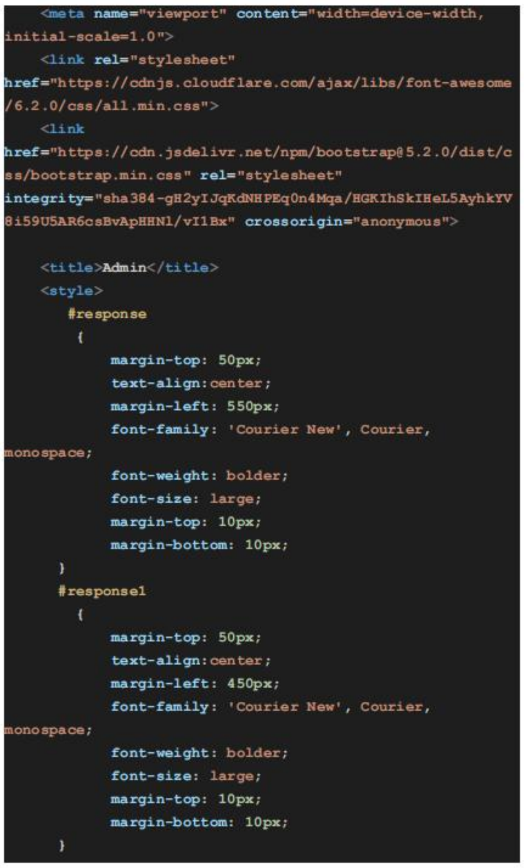


5. CODING & SOLUTIONING (Explain the features added in the project along with code)

5.1 Feature 1

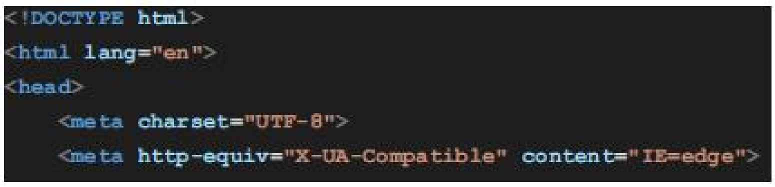
Login and Signup



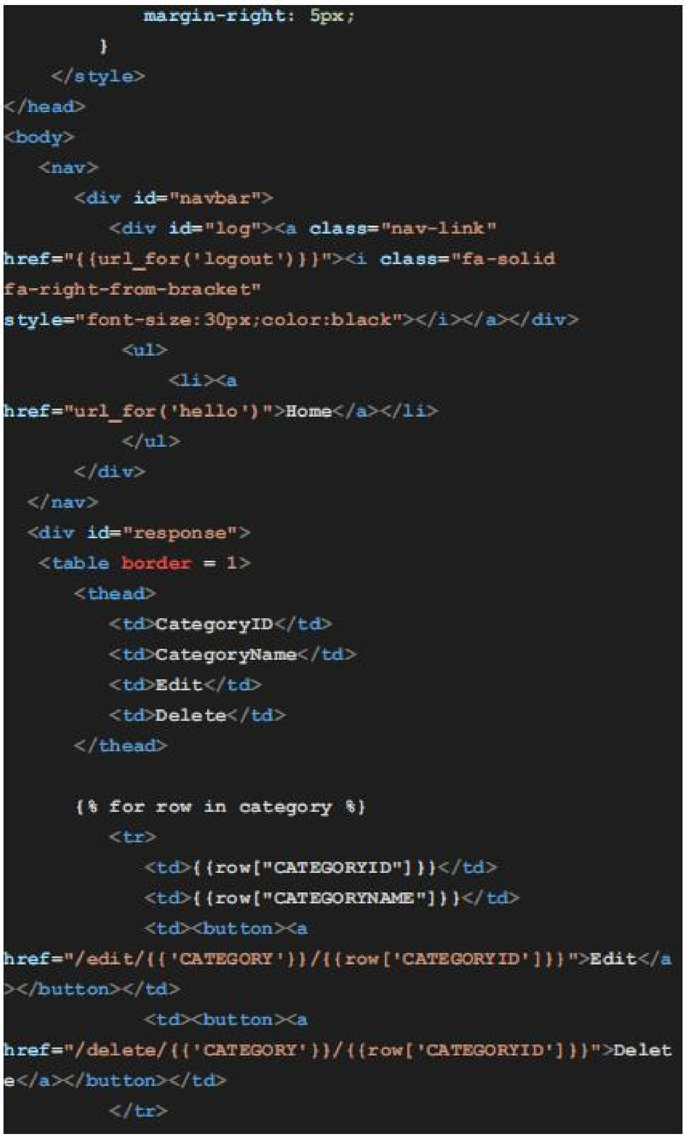


```
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome
/6.2.0/css/all.min.css">
    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.0/dist/c
ss/bootstrap.min.css" rel="stylesheet"
integrity="sha384-gH2yIJqKdNHPEq0n4Mqa/HGKIhSkIHeL5AyhkYV
8i59U5AR6csBvApHHNl/vI1Bx" crossorigin="anonymous">

    <title>Admin</title>
    <style>
       #response
        {
            margin-top: 50px;
            text-align:center;
            margin-left: 550px;
            font-family: 'Courier New', Courier,
monospace;
            font-weight: bolder;
            font-size: large;
            margin-top: 10px;
            margin-bottom: 10px;
      }
      #response1
        {
            margin-top: 50px;
            text-align:center;
            margin-left: 450px;
            font-family: 'Courier New', Courier,
monospace;
            font-weight: bolder;
            font-size: large;
            margin-top: 10px;
            margin-bottom: 10px;
      }
```



```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
```

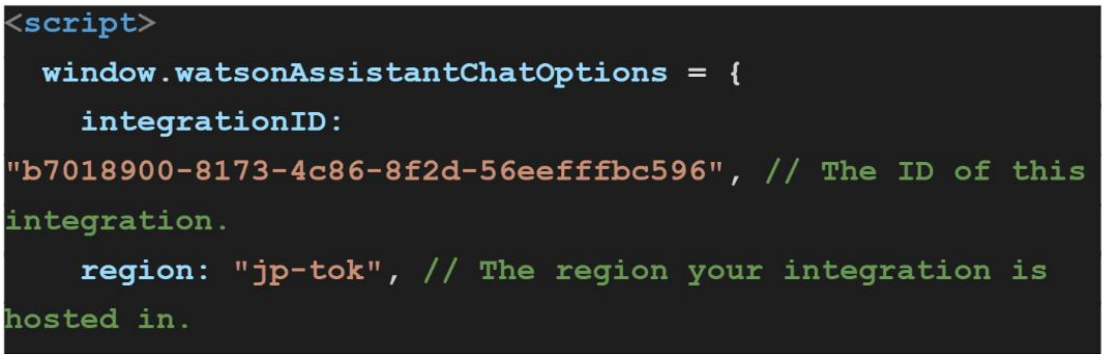


```
            margin-right: 5px;
        }
    </style>
</head>
<body>
   <nav>
      <div id="navbar">
         <div id="log"><a class="nav-link"
href="{{url_for('logout')}}"><i class="fa-solid
fa-right-from-bracket"
style="font-size:30px;color:black"></i></a></div>
          <ul>
              <li><a
href="url_for('hello')">Home</a></li>
          </ul>
      </div>
  </nav>
  <div id="response">
   <table border = 1>
      <thead>
         <td>CategoryID</td>
         <td>CategoryName</td>
         <td>Edit</td>
         <td>Delete</td>
      </thead>

      {% for row in category %}
         <tr>
            <td>{{row["CATEGORYID"]}}</td>
            <td>{{row["CATEGORYNAME"]}}</td>
            <td><button><a
href="/edit/{{'CATEGORY'}}/{{row['CATEGORYID']}}">Edit</a
></button></td>
            <td><button><a
href="/delete/{{'CATEGORY'}}/{{row['CATEGORYID']}}">Delet
e</a></button></td>
         </tr>
```

5.2 Feature 2

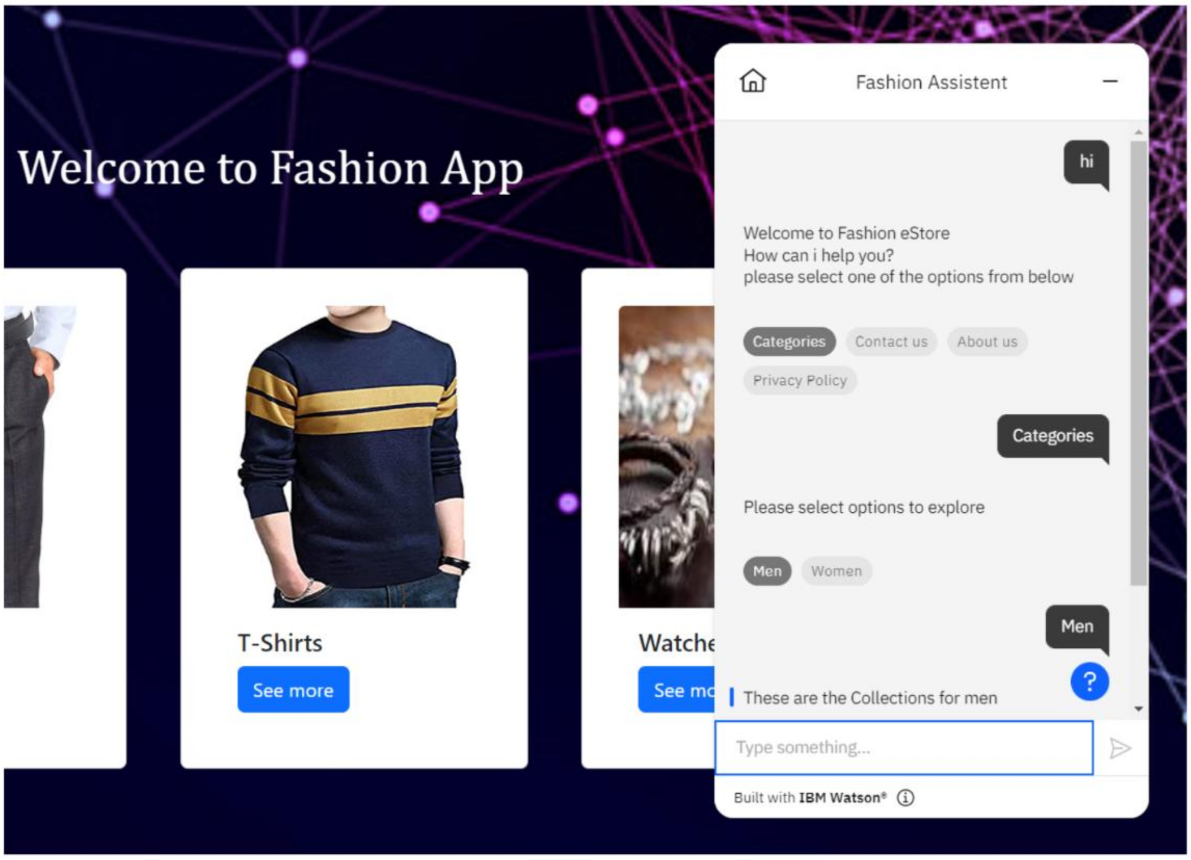
ChatBot



```
<script>
  window.watsonAssistantChatOptions = {
    integrationID:
"b7018900-8173-4c86-8f2d-56eefffbc596", // The ID of this
integration.
    region: "jp-tok", // The region your integration is
hosted in.
```

```
    serviceInstanceID:
"c38d61d0-0061-4f2a-8197-96810297fe55", // The ID of your
service instance.
    onLoad: function(instance) { instance.render(); }
  };
  setTimeout(function(){
    const t=document.createElement('script');

t.src="https://web-chat.global.assistant.watson.appdomain
.cloud/versions/" +
(window.watsonAssistantChatOptions.clientVersion ||
'latest') + "/WatsonAssistantChatEntry.js";
    document.head.appendChild(t);
  });
</script>
```



Welcome to Fashion App

Fashion Assistent

hi

Welcome to Fashion eStore
How can i help you?
please select one of the options from below

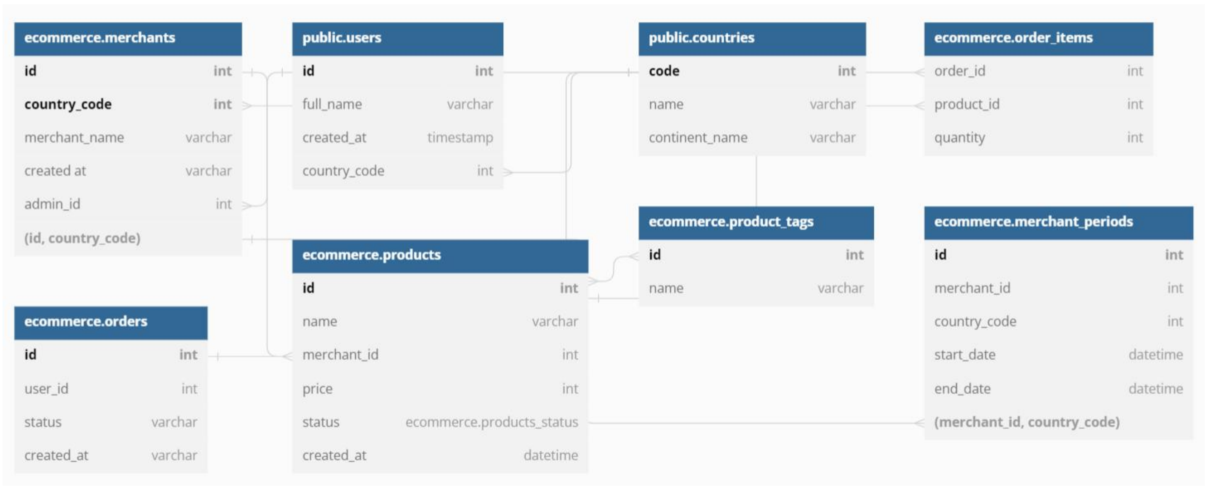
Categories

Please select options to explore

Men

These are the Collections for men

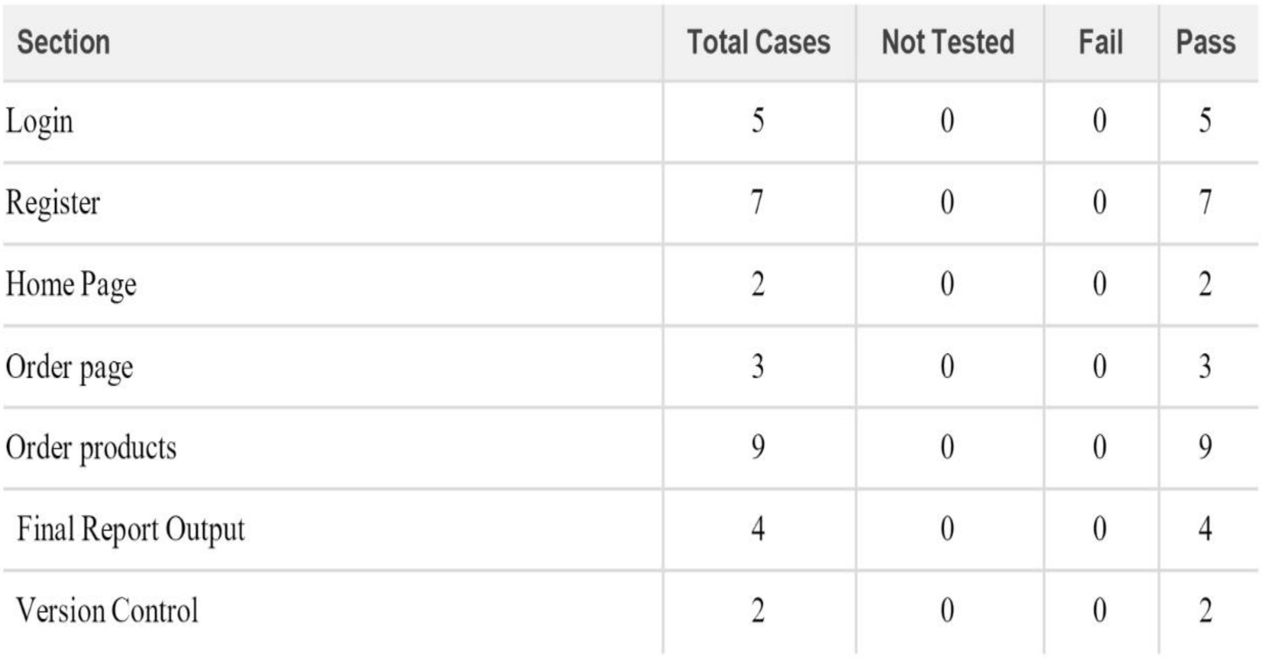
5.3 Database Schema (if Applicable)



6. TESTING

6.1 Test Cases

Test Case Analysis



Section	Total Cases	Not Tested	Fail	Pass
Login	5	0	0	5
Register	7	0	0	7
Home Page	2	0	0	2
Order page	3	0	0	3
Order products	9	0	0	9
Final Report Output	4	0	0	4
Version Control	2	0	0	2

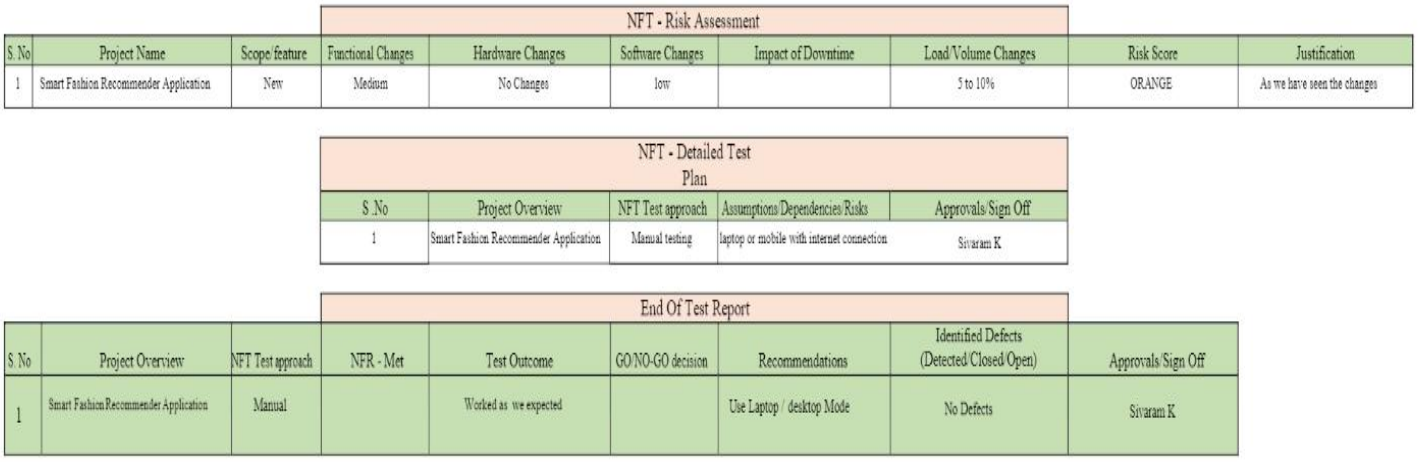
6.2 User Acceptance Testing

Defect Analysis

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	5	5	2	3	21
Duplicate	1	0	3	0	4
External	2	3	0	1	6
Fixed	11	2	4	20	37
Not Reproduced	0	0	1	0	1
Skipped	0	0	1	1	2
Won't Fix	0	5	2	1	8
Totals	24	14	13	26	77

7. **RESULTS**

7.1 Performance Metrics



			NFT - Risk Assessment						
S. No	Project Name	Scope/feature	Functional Changes	Hardware Changes	Software Changes	Impact of Downtime	Load/Volume Changes	Risk Score	Justification
1	Smart Fashion Recommender Application	New	Medium	No Changes	low		5 to 10%	ORANGE	As we have seen the changes

NFT - Detailed Test Plan				
S.No	Project Overview	NFT Test approach	Assumptions/Dependencies/Risks	Approvals/Sign Off
1	Smart Fashion Recommender Application	Manual testing	laptop or mobile with internet connection	Sivaram K

			End Of Test Report					
S. No	Project Overview	NFT Test approach	NFR - Met	Test Outcome	GO/NO-GO decision	Recommendations	Identified Defects (Detected/Closed/Open)	Approvals/Sign Off
1	Smart Fashion Recommender Application	Manual		Worked as we expected		Use Laptop / desktop Mode	No Defects	Sivaram K

8. **ADVANTAGES**

1) Easy recommendations make fewer searches and sometimes end up in good deals

2) User reviews will give accurate information, this is also an advantage if we purchase online as we can see other reviews too, most of the time honest

3) Speed up the process of decision and purchase based on the previous statistics

4) A recommendation engine can bring traffic to were sites. It accomplishes this with customized email messages and target blasts.

DISADVANTAGES

1) If the system recommends products with bias, then the customer will be landing on the wrong deals

2) Chances are that some websites may suggest products wrongly based on analysis of

little information gathered

3) Since the feature representations of the items are hand-engineered to some extent, this technique requires a lot of domain knowledge. Therefore, the model can only be as good as the hand-engineered features.

4) The model can only make recommendations based on the existing interests of the user. In other words, the model has limited ability to expand on the users' existing interests.

9. CONCLUSION

Recent advancements in cloud computing helping ease the fashion industrys transition from customer stores into modern online shops equipped with high-tech features such as virtual try-on and fashionsynthesis systems. This article sheds some light on different applications related to these systems,tracked the research progressthrough the years, and illustrated the field’s rapid growth. Althoughscientists have achieved significant milestones, still many unsolved matters remain. One main issue isthe systems’ performance compared to human abilities; another important factor is the applicability ofmethods regarding computational effort and energy efficiency. Another critical problem is the definitionof a well-structured and uniform objective metric to assess the results.

10. FUTURE SCOPE

Physical stores are no longer the only way that shoppers are interacting with brands, especially fashion retailers. As shoppers increasingly expect mobile purchasing, one-day shipping and 24/7 customer support, fashion retailers are redesigning the entire shopping experience.

Luckily, the fashion industry is already a master of change, with seasonal trends coming and going before you can even take a breath. However, the fashion world’s digital transformation may be its greatest change yet.

11. APPENDIX

Source Code

App.py

```
import sqlite3 as sql


from flask import (Flask, jsonify, redirect, render_template, request, session,
                   url_for,flash)
from flask_sqlalchemy import SQLAlchemy
from markupsafe import escape
```

```
import ibm_db

app= Flask(__name__)
app.config['DEBUG']=True
app.secret_key="mgyftuiu"
connection=ibm_db.connect("DATABASE=bludb;HOSTNAME=b1bc1829-6f45-4cd4-
    bef4-
    10cf081900bf.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=32304;SEC
    URITY=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=zpj04244;PWD
    =xB9fQaRJsweFh5pv","","")

@app.route('/home')
def hello():
    if 'loggedin' in session and 'name' in session:
      return render_template('home.html',msg="TRUE")
    return redirect(url_for('login'))

@app.route('/')
def index():
   return render_template('index.html')

@app.route('/hospital')
def hospital():
   return render_template('hospitalhome.html')

@app.route("/blog")
def blog():
   return "<h1>Hello World from blog page</h1>"

@app.route("/blog/<int:id>")
def blogId(id):
   return "<h1>Hello World from blog page" +str(id)+"</h1>"

@app.route('/about')
```

```
def about():
    if 'loggedin' in session and 'name' in session:
        return render_template('about.html')
    return redirect(url_for('login'))



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

@app.route('/login')
def login():
    return render_template('login.html')

@app.route("/profiles/<username>")
def user(username):
    return "<h2>Hello "+username+"</h2>"

@app.route('/adduser',methods= ['POST','GET'])
def adduser():
    message=""
    if request.method=='POST':
        try:
            name= request.form['username']
            mail= request.form['mail']
            mobile= request.form['mobile']
            password= request.form['password']

            # with sql.connect("database.db") as con:
            #     cur= con.cursor()
            #     cur.execute('insert into user values(?,?,?,?)',(name,mail,mobile,password))
            #     con.commit()
            #     message="User Registered"
            #     return redirect(url_for('login'))
            sql = "SELECT * FROM ACCOUNT WHERE USERNAME =?"
            stmt = ibm_db.prepare(connection, sql)
            ibm_db.bind_param(stmt,1,name)
            ibm_db.execute(stmt)
            account = ibm_db.fetch_assoc(stmt)

            if account:
                return render_template('list.html', msg="Account Already Exists")
            else:
                command="INSERT                    INTO                    ACCOUNT
        (USERNAME,EMAIL,PHONENO,PASSWORD) VALUES (?,?,?,?)"
                prep_stmt = ibm_db.prepare(connection, command)
                ibm_db.bind_param(prep_stmt, 1, name)
                ibm_db.bind_param(prep_stmt, 2, mail)
                ibm_db.bind_param(prep_stmt, 3, mobile)
                ibm_db.bind_param(prep_stmt, 4, password)
                ibm_db.execute(prep_stmt)
                return render_template('login.html', msg="TRUE")

        except:
            con.rollback()
            message="error"
            return render_template("signup.html",error="TRUE")
    # return jsonify({message})
    # return render_template('signup.html')

@app.route('/admin')
def admin():
    data=[]
    products=[]
    if 'loggedin' in session and 'name' in session:
        sql="SELECT * FROM CATEGORY"
        stmt = ibm_db.prepare(connection, sql)
        ibm_db.execute(stmt)
        resultSet = ibm_db.fetch_both(stmt)
        while resultSet !=False:
            data.append(resultSet)
            resultSet=ibm_db.fetch_both(stmt)
        command="SELECT * FROM PRODUCT"
        stmt1 = ibm_db.prepare(connection, command)
        ibm_db.execute(stmt1)
        results = ibm_db.fetch_both(stmt1)
        while results !=False:
            products.append(results)
            results=ibm_db.fetch_both(stmt1)
        return render_template('admin.html',category=data,prods=products)
    return render_template('login.html')

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

        account=""
        # con=sql.connect("database.db")
        # con.row_factory=sql.Row

        # cur =con.cursor()
        # cur.execute("SELECT * FROM user")

        # # rows= cur.fetchall()
        # cur.execute('SELECT * FROM user where name=? and password=?',
      (username,password))
        # account = cur.fetchone()
        try:
            if username.lower()=='admin' and password.lower()=='admin@123':
```

```
            session['loggedin'] = True
            session['name'] = username.lower()
            return redirect(url_for('admin'))

         else:
            command= "SELECT * FROM ACCOUNT WHERE USERNAME =? AND
      PASSWORD =?"
            stmt=ibm_db.prepare(connection,command)
            ibm_db.bind_param(stmt, 1, username)
            ibm_db.bind_param(stmt, 2, password)
            ibm_db.execute(stmt)
            result = ibm_db.fetch_assoc(stmt)
            while result !=False:
               account=result['USERNAME']
               id=result['USERID']
               result = ibm_db.fetch_assoc(stmt)
            if account:
               session['loggedin'] = True
               session['name'] = account
               session['id']= id
               return redirect(url_for('hello'))
            else:
               msg = "Incorrect username/password!"
               return render_template('login.html', msg=msg)
      except:
         return render_template('login.html', msg="Account not found")


      #return render_template("list.html",rows=rows)

@app.route('/logout')
def logout():
   session.pop('loggedin', None)
   session.pop('name', None)
```

```
  return redirect(url_for('login'))

@app.route('/addRecord/<tablename>',methods=['POST','GET'])
def addRecord(tablename):
    if request.method=='POST':
        print(tablename)
        if tablename=='CATEGORY':
            name=request.form['catname']
            sql=f"INSERT INTO CATEGORY (CATEGORYNAME) VALUES
    ('{escape(name)}')"
            stmt1 = ibm_db.prepare(connection, sql)
            ibm_db.execute(stmt1)
            return redirect(url_for('admin'))
        elif tablename=='PRODUCT':
            pname=request.form['prodname']
            catid=request.form['catid']
            unit=request.form['unit']
            price=request.form['price']
            sql=f"INSERT INTO PRODUCT
    (PRODUCTNAME,CATEGORYID,UNIT,PRICE) VALUES
    ('{escape(pname)}',{escape(catid)},{escape(unit)},{escape(price)})"
            stmt1 = ibm_db.prepare(connection, sql)
            ibm_db.execute(stmt1)
            return redirect(url_for('admin'))




@app.route('/delete/<tablename>/<int:Id>')
def delete(tablename,Id):
  if tablename=='CATEGORY':
    sql=f"SELECT * FROM {escape(tablename)} WHERE CATEGORYID = ?"
    stmt = ibm_db.prepare(connection, sql)
```

```
ibm_db.bind_param(stmt, 1, Id)
ibm_db.execute(stmt)
result = ibm_db.fetch_row(stmt)
print ("The table Name is : ",  result)
if result:
    sql1 = f"DELETE FROM {escape(tablename)} WHERE CATEGORYID = 
  {escape(Id)}"
    stmt = ibm_db.exec_immediate(connection, sql1)
    # data = []
    # sql = f"SELECT * FROM {escape(tablename)}"
    # stmt = ibm_db.exec_immediate(connection, sql)
    # dictionary = ibm_db.fetch_both(stmt)
    # while dictionary != False:
    #     data.append(dictionary)
    #     dictionary = ibm_db.fetch_both(stmt)
        # return render_template("admin.html", category = data, deletemsg="True")
    return redirect(url_for('admin'))
    # return "success..."
else:
    sql=f"SELECT * FROM PRODUCT WHERE PRODUCTID = {escape(str(Id))}"
    print('hi')
    print(tablename)
    stmt = ibm_db.prepare(connection, sql)
    # ibm_db.bind_param(stmt, 1, Id)
    ibm_db.execute(stmt)
    result = ibm_db.fetch_row(stmt)
    print ("The table Name is : ",  result)
    if result:
        sql1 = f"DELETE FROM {escape(tablename)} WHERE PRODUCTID = 
  {escape(Id)}"
        print('hello')
        stmt = ibm_db.exec_immediate(connection, sql1)
        return redirect(url_for('admin'))
    else:
            print("no value")
            return redirect(url_for('admin'))

@app.route('/deleteRec/<int:id>')
def deleteRec(id):
    sql=f"SELECT * FROM PRODUCT WHERE PRODUCTID = ?"
    print("hi")
    stmt = ibm_db.prepare(connection, sql)
    ibm_db.bind_param(stmt, 1, id)
    ibm_db.execute(stmt)
    result = ibm_db.fetch_row(stmt)
    print ("The table Name is : ",  result)
    if result:
        print("hello")
        sql1 = f"DELETE FROM PRODUCT WHERE PRODUCTID = {escape(id)}"
        stmt = ibm_db.exec_immediate(connection, sql1)
        return redirect(url_for('admin'))

@app.route('/edit/<int:ID>')
def edit(ID):
    return "<h2>Hello "+str(ID)+"</h2>"

@app.route('/pants')
def pants():
    return render_template('Pants.html')

@app.route('/watch')
def watch():
    return render_template('watch.html')

@app.route('/shoes')
def shoes():
    return render_template('Shoes.html')
```

```
@app.route('/tshirts')
def tshirts():
    return render_template('tshirts.html')

@app.route('/shirts')
def shirts():
    return render_template('Shirts.html')

@app.route('/order',methods=['POST','GET'])
def order():
    if request.method=='POST':
        user=request.form['uname']
        quan=request.form['quan']
        amount=request.form['amount']

        total=int(amount)*int(quan)
        uid=session['id']
        sql=f"INSERT INTO ORDER (USERID,AMOUNT) VALUES
      ({escape(uid)},{escape(total)})"
        stmt=ibm_db.exec_immediate(connection,sql)
        return redirect(url_for('hello'))


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

Bucket.py

```
import ibm_boto3
from ibm_botocore.client import Config, ClientError

COS_ENDPOINT="https://s3.jp-tok.cloud-object-storage.appdomain.cloud"
COS_API_KEY_ID="BRYojvVpcYt4K1NRQuKhi_QS-b7_UvD1KExgCM1xb6WS"
```

```
COS_INSTANCE_CRN="crn:v1:bluemix:public:cloud-object-
   storage:global:a/f573a30aa84f432a91a05468f825e886:e1113bc9-1b85-49b8-84fc-
   495e1c5f9311::"
COS_BUCKET_LOCATION="jp-tok-smart"


cos = ibm_boto3.resource("s3",
  ibm_api_key_id=COS_API_KEY_ID,
  ibm_service_instance_id=COS_INSTANCE_CRN,
  config=Config(signature_version="oauth"),
  endpoint_url=COS_ENDPOINT
)

def create_bucket(bucket_name):
  print("Creating new bucket: {0}".format(bucket_name))
  try:
    cos.Bucket(bucket_name).create(
      CreateBucketConfiguration={
        "LocationConstraint":COS_BUCKET_LOCATION
      }
    )
    print("Bucket: {0} created!".format(bucket_name))
  except ClientError as be:
    print("CLIENT ERROR: {0}\n".format(be))
  except Exception as e:
    print("Unable to create bucket: {0}".format(e))
```

Connect.py

```
import ibm_db

connection=ibm_db.connect("DATABASE=bludb;HOSTNAME=b1bc1829-6f45-4cd4-
   bef4-
   10cf081900bf.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=32304;SEC
```

```
    URITY=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=zpj04244;PWD
    =xB9fQaRJsweFh5pv","","")

print(connection)
print("connection Successfully")
```

Sqlite.py

```
import sqlite3

conn= sqlite3.connect('database.db')
print('opened')

conn.execute('CREATE TABLE user (name TEXT, email TEXT, phone TEXT,
    password TEXT)')
print("table created")
conn.close()
```

Home.html

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta http-equiv="X-UA-Compatible" content="IE=edge">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
    awesome/6.2.0/css/all.min.css">
   <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.0/dist/css/bootstrap.min.css"
    rel="stylesheet" integrity="sha384-
    gH2yIJqKdNHPEq0n4Mqa/HGKIhSkIHeL5AyhkYV8i59U5AR6csBvApHHNl/vI1B
    x" crossorigin="anonymous">

   <title>Home</title>

    <nav>
    <a class="navbar-brand" href="#">Smart Fashion Reccomender</a>
    <a class="navbar-brand" href="{{url_for('about')}}">About</a>
    <a class="navbar-brand" href="{{url_for('blog')}}">Blog</a>

    <div  id="log"><a  class="nav-link"  href="{{url_for('logout')}}"><i  class="fa-solid
  fa-right-from-bracket" style="font-size:30px;color:white"></i></a></div>
  </nav>

<style>
    #log
    {
        border: none;
        background-color: transparent;
        width: 100px;
        align: right;
        float: right;
        margin-right: 10px;
    }
  html,body{
        height: 100%;
        background-image: url('static/images/codetree.jpg');
        background-attachment: fixed;
        background-repeat: no-repeat;
        background-size:cover;
    }

    nav{
      color:white;
      font-size: large;
      margin-bottom: 50px;
      margin-top: 50px;
    }
    nav a{
      margin-left: 20px;
      margin-right: 20px;

    }
    a:hover{
      border:none;
      color: aqua;
    }
    a:visited{
      border:none;
      color:aquamarine;
    }
    #images{
      margin-top: 60px;
    }
    #shirt{
      display: inline-block;
      padding: 30px;
      margin-left: 20px;
      margin-right: 20px;
    }
    #pant{
      display: inline-block;
      width:25%;
      padding: 30px;
      margin-left: 20px;
      margin-right: 20px;
    }
    #tshirt{
      display: inline-block;
      width:25%;
      padding: 30px;
      margin-left: 20px;
      margin-right: 20px;
    }
```

```
        #watch{
          display: inline-block;
          width:25%;
          padding: 30px;
          margin-left: 20px;
          margin-right: 20px;
        }
        #shoes{
          display: inline-block;
          width:25%;
          padding: 30px;
          margin-top:30px;
          margin-left: 20px;
          margin-right: 20px;
        }
    </style>
</head>
<body onload="mypop()">
  <script>
    function mypop()
    {
        if("{{ msg }}"=="T")
        {
          alert("Login successfull")
        }
    }

</script>
<script>
  window.watsonAssistantChatOptions = {
    integrationID: "b7018900-8173-4c86-8f2d-56eefffbc596", // The ID of this integration.
    region: "jp-tok", // The region your integration is hosted in.
    serviceInstanceID: "c38d61d0-0061-4f2a-8197-96810297fe55", // The ID of your
      service instance.
    onLoad: function(instance) { instance.render(); }
  };
  setTimeout(function(){
    const t=document.createElement('script');
    t.src="https://web-chat.global.assistant.watson.appdomain.cloud/versions/"              +
      (window.watsonAssistantChatOptions.clientVersion           ||           'latest')           +
      "/WatsonAssistantChatEntry.js";
    document.head.appendChild(t);
  });
</script>


    <h1 style="text-align:center;color:white;font-family: Cambria, Cochin, Georgia, Times,
      'Times New Roman', serif;">Welcome to Fashion App</h1>
    <!-- <form align="right" id="logout">
        <button    type="submit"    id="log"><i    class="fa-solid    fa-right-from-bracket"
      style="font-size:30px;color:black"></i></button>
    </form> -->
    <div id="images">
      <div id="shirt" class="card" style="width: 18rem;">
        <img    class="card-img-top"    style="height:250px;"    src="https://smart-fashion-
      app.s3.jp-tok.cloud-object-storage.appdomain.cloud/shirt.jpg" alt="Shirt">
        <div class="card-body">
          <h5 class="card-title">Shirts</h5>
          <a href="/shirts" class="btn btn-primary">See more</a>
        </div>
      </div>

        <div id="pant" class="card" style="width: 18rem;">
          <img   class="card-img-top"   style="height:250px;"     src="https://smart-fashion-
      app.s3.jp-tok.cloud-object-storage.appdomain.cloud/pants.jpg" alt="Pants">
          <div class="card-body">
            <h5 class="card-title">Pants</h5>
            <a href="/pants" class="btn btn-primary">See more</a>
          </div>
        </div>

          <div id="tshirt" class="card" style="width: 18rem;">
            <img class="card-img-top" style="height:250px;" src="https://smart-fashion-
      app.s3.jp-tok.cloud-object-storage.appdomain.cloud/tshirt.jpg" alt="T-Shirts">
            <div class="card-body">
              <h5 class="card-title">T-Shirts</h5>
              <a href="/tshirts" class="btn btn-primary">See more</a>
            </div>
          </div>

           <div id="watch" class="card" style="width: 18rem;">
              <img class="card-img-top" style="height:250px;" src="https://smart-fashion-
      app.s3.jp-tok.cloud-object-storage.appdomain.cloud/watch.jfif" alt="Watches">
              <div class="card-body">
                <h5 class="card-title">Watches</h5>

                <a href="/watch" class="btn btn-primary">See more</a>
              </div>
            </div>

            <div id="shoes" class="card" style="width: 18rem;">
                <img class="card-img-top" style="height:250px;" src="https://smart-fashion-
      app.s3.jp-tok.cloud-object-storage.appdomain.cloud/shoes.jfif" alt="Shoes">
                <div class="card-body">
                  <h5 class="card-title">Shoes</h5>

                  <a href="/shoes" class="btn btn-primary">See more</a>
                </div>
            </div>
      </div>
```

```
    </div>
    <script
      src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.11.5/dist/umd/popper.min.js"
      integrity="sha384-
      Xe+8cL9oJa6tN/veChSP7q+mnSPaj5Bcu9mPX5F5xIGE0DVittaqT5lorf0EI7Vk"
      crossorigin="anonymous"></script>
    <script          src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.0/dist/js/bootstrap.min.js"
      integrity="sha384-
      ODmDIVzN+pFdexxHEHFBQH3/9/vQ9uori45z4JjnFsRydbmQbmL5t1tQ0culUzyK
      " crossorigin="anonymous"></script>

</body>
</html>
```

Index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link      href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.0/dist/css/bootstrap.min.css"
      rel="stylesheet"                                              integrity="sha384-
      gH2yIJqKdNHPEq0n4Mqa/HGKIhSkIHeL5AyhkYV8i59U5AR6csBvApHHNl/vI1B
      x" crossorigin="anonymous">

    <title>Home</title>
    <style>
        body{
            background-image: linear-gradient(to bottom right, #02AABD,#00CDAC);
            background-repeat: no-repeat;
            background-position: center;
            background-size: cover;
    background-attachment: fixed;
  }
  /* #outer{
    text-align: center;
    margin: auto;
    width: 50%;
    margin-top: 230px;
    margin-left: 450px;
  } */
  #cards{
    text-align: center;
  }
  #admin{
    display: inline-block;
    width: 50%;
    margin-top: 230px;
    margin-left: 450px;
    box-shadow: 10px 10px 5px lightblue;
    position: relative;
    top: 0;
    transition: top ease 0.5s;

  }
  #login
 {
  align:right;
  margin-right: 20px;
 }
  #admin:hover{
    top:-30px;
  }
  #user{
    display: inline-block;
    width: 50%;
```

```
            margin-top: 230px;
            margin-right: 450px;
            margin-left: 20px;
            box-shadow: 10px 10px 5px lightblue;
            position: relative;
            top: 0;
            transition: top ease 0.5s;
        }
        #user:hover{
            top:-30px;
        }
    </style>
</head>
<body>
    <nav class="navbar navbar-expand-lg bg-light">
        <div class="container-fluid">

          <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-
      target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-
      expanded="false" aria-label="Toggle navigation">
            <span class="navbar-toggler-icon"></span>
          </button>
          <div class="collapse navbar-collapse" id="navbarSupportedContent">
            <ul class="navbar-nav me-auto mb-2 mb-lg-0">
              <li class="nav-item">
                <a class="nav-link active" aria-current="page" href="/home">Home</a>
              </li>
              <li class="nav-item">
                <a class="nav-link" href="/blog">Blog</a>
              </li>
              <li class="nav-item">
                <a class="nav-link" href="/about">About</a>
              </li>
              <li class="nav-item">
          <a class="nav-link" href="/signup">SignUp</a>
        </li>
      </ul>
    </div>
  </div>

  <div id="login">
      <a class="nav-link" type="button" href="/login">Login</a>
  </div>
</nav>
<h1 style="font-family: Cambria, Cochin, Georgia, Times, 'Times New Roman',
  serif;color:white;text-align: center;">Welcome to E-Fashion Website</h1>
<div id ="outer" style="text-align:center;">
  <div id="cards">
      <div class="card" id="admin" style="width: 18rem;">
      <img class="card-img-top"  src="static/images/adminad.jfif" alt="Card image
  cap">
      <div style="text-align:center;margin-bottom: 10px;">
          <a href="{{url_for('login')}}" style="width: 140px;padding:5px;" class="btn
  btn-primary">Admin</a>
      </div>
      </div>
      <div class="card" id="user" style="width: 18rem;">
          <img class="card-img-top" src="static/images/adminad.jfif" alt="Card image
  cap">
          <div style="text-align:center;margin-bottom: 10px;">
              <a href="{{url_for('login')}}" style="width: 140px;padding:5px;" class="btn
  btn-primary">User</a>
          </div>
      </div>
  </div>

</div>
```

```
  <script
   src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.11.5/dist/umd/popper.min.js"
   integrity="sha384-
   Xe+8cL9oJa6tN/veChSP7q+mnSPaj5Bcu9mPX5F5xIGE0DVittaqT5lorf0EI7Vk"
   crossorigin="anonymous"></script>
  <script     src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.0/dist/js/bootstrap.min.js"
   integrity="sha384-
   ODmDIVzN+pFdexxHEHFBQH3/9/vQ9uori45z4JjnFsRydbmQbmL5t1tQ0culUzyK
   " crossorigin="anonymous"></script>

</body>
</html>
```

Login.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Login Page</title>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Login</title>
    <link  href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.0/dist/css/bootstrap.min.css"
   rel="stylesheet"                                    integrity="sha384-
   gH2yIJqKdNHPEq0n4Mqa/HGKIhSkIHeL5AyhkYV8i59U5AR6csBvApHHNl/vI1B
   x" crossorigin="anonymous">
    <link rel="stylesheet" href="{{url_for('static',filename='css/login.css')}}">
    <style>
           body{
      background-image: url('static/images/gradient-cactus.jpg');
      background-repeat: no-repeat;
      background-position: center;
      background-size: cover;
      background-attachment: fixed;
    }
    </style>

</head>
<body onload="mypop()">
  <script>
    function mypop()
    {
      m='{{msg}}'
      if(m=="T")
        alert("Student Data saved successfuly")
    }
  </script>
  <div class="card" style="width: 18rem;">
    <div class="card-body">
    <form id="form" method="POST" action="{{url_for('check')}}"
   autocomplete="off">
      <p><strong style="color:red">{{msg}}</strong></p>
      <label>User Name:</label><br>
      <input type="text"      name="username" placeholder="Enter the name"><br>
      <label>Password:</label><br>
      <input type="password" name="password" placeholder="enter password"><br>
      <input type="submit" id="sign" value="Login"><br><br>
    </form>
    <a type="button" href='{{url_for( "signup" )}}'><button
   id="sign">Register</button></a>
    </div>

  </div>
  <script
   src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.11.5/dist/umd/popper.min.js"
   integrity="sha384-
   Xe+8cL9oJa6tN/veChSP7q+mnSPaj5Bcu9mPX5F5xIGE0DVittaqT5lorf0EI7Vk"
   crossorigin="anonymous"></script>
```

```
    <script         src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.0/dist/js/bootstrap.min.js"
      integrity="sha384-
      ODmDIVzN+pFdexxHEHFBQH3/9/vQ9uori45z4JjnFsRydbmQbmL5t1tQ0culUzyK
      " crossorigin="anonymous"></script>
</body>
</html>
```

Signup.html

```
{% extends "base.html" %}
<!DOCTYPE html>
<html lang="en">
<head>
    {% block head %}
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link      href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.0/dist/css/bootstrap.min.css"
      rel="stylesheet"                                                   integrity="sha384-
      gH2yIJqKdNHPEq0n4Mqa/HGKIhSkIHeL5AyhkYV8i59U5AR6csBvApHHNl/vI1B
      x" crossorigin="anonymous">
    <link rel="stylesheet" href="{{url_for('static',filename='css/signup.css')}}">
    <title>Sign Up</title>
    <script>
    function Validate() {
        var password = document.getElementsByName("password")[0].value;
        var confirmPassword = document.getElementsByName("re-password")[0].value;
        if (password != confirmPassword) {
            alert("Passwords do not match.");
            return false;
        }
        return true;
    }

    </script>
    {% endblock %}
```

```
</head>
<body onload="mypop()">
    {% block body %}
    <script>
        function mypop()
        {
            if("{{ error }}"=="T")
            {
              alert("Signup error")
            }
        }
    </script>
    <div id="outer">

        <div id="signup">
            <div class="card" style="width: 600px;">
                <div class="card-body">
            <form id="form" action="{{ url_for('adduser') }}" method="POST"
      autocomplete="off">
                <div class="msg">{{ msg }}</div>
                <label>User Name:</label><br>
                <input type="text"       name="username" placeholder="Enter the name"><br>
                <label>Email:</label><br>
                <input type="email" name="mail" placeholder="Enter the email id"><br>
                <label>Phone no:</label><br>
                <input type="tele"      name="mobile"   placeholder="Enter   the   mobile
      no"><br>
                <label>Password:</label><br>
                <input           type="password"    name="password"    placeholder="enter
      password"><br>
                <label>Retype Password:</label>
                <input    type="password"  name="re-password"  placeholder="retype  the
      password"><br>
```

```
          <input type="submit" id="sign" onclick="return Validate()"
    value="SignUp"><br><br>

          <p>Already a user? <a href="{{url_for('login')}}">Log in</a></p>
        </form>
      </div></div>
      </div>
      {% if error %}
        <p class="error"><strong>Error:</strong> {{ error }}
      {% endif %}

    </div>
    <script
      src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.11.5/dist/umd/popper.min.js"
      integrity="sha384-
      Xe+8cL9oJa6tN/veChSP7q+mnSPaj5Bcu9mPX5F5xIGE0DVittaqT5lorf0EI7Vk"
      crossorigin="anonymous"></script>
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.0/dist/js/bootstrap.min.js"
      integrity="sha384-
      ODmDIVzN+pFdexxHEHFBQH3/9/vQ9uori45z4JjnFsRydbmQbmL5t1tQ0culUzyK
      " crossorigin="anonymous"></script>
    <script>

    </script>
    {% endblock %}
</body>
</html>
```

GitHub & Project Demo Link

Github link - https://github.com/IBM-EPBL/IBM-Project-17555-1659673361

Project Demo Link - https://www.kapwing.com/videos/6378f08a1941c70288b16ffb

.