SMART FASHION RECOMMENDER APPLICATION

Team ID: PNT2022TMID48338

SI No	Register Number	Name
1	913319104009	Balamurugan G
2	913319104026	Kalaivani T V
3	913319104012	Catherine S
4	913319104029	Kumaragurumoorthy R

1.INTRODUCTION

1.1 Project Overview

Clothing is a kind of symbol that represents people's internal perceptions through their outer appearance. It conveys information about their choices, faith, personality, profession, social status, and attitude towards life. Therefore, clothing is believed to be a nonverbal way of communicating and a major part of people's outer appearance Recent technological advancements have enabled consumers to track current fashion trends around the globe, which influence their choices .The fashion choices of consumers depend on many factors, such as demographics, geographic location, individual preferences, interpersonal influences, age, gender, season, and culture therefore, e-commerce has become the predominant channel for shopping in recent years. The ability of recommendation systems to provide personalized recommendations and respond quickly to the consumer's choices has contributed significantly to the expansion of e-commerce sales. An effective recommendation system is a crucial tool for successfully conducting an e-commerce business.Fashion recommendation systems (FRSs) generally provide specific recommendations to the consumer based on their queries

1.2 Purpose

In recent years, the textile and fashion industries have witnessed an enormous amount of growth in fast fashion. On e-commerce platforms, where numerous choices are available, an efficient recommendation system is required to sort, order, and efficiently convey relevant product content or information to users instead of navigating to several screens for booking products online

2. LITERATURE SURVEY

2.1 Existing problem

Abstract:

Fashion is perceived as a meaningful way of self-expressing that people use for different purposes. It seems to be an integral part of every person in modern societies, from everyday life to exceptional events and occasions. 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 answering their customer needs. Fashion recommender systems have been introduced to address these needs.

SURVEY: 1

Book Title: Fashion Recommendation Systems, Models and Methods

Book Author: Samit Chakraborty

Year of Publication: July 2021

Abstract: In recent years, the textile and fashion industries have witnessed an enormous amount of growth in fast fashion. On e-commerce platforms, where numerous choices are available, an efficient recommendation system is required to sort, order, and efficiently convey relevant product content or information to users. Image-based fashion recommendation systems (FRSs) have attracted a huge amount of attention from fast fashion retailers as they provide a personalized shopping experience to consumers.

SURVEY: 2

Book Title: Development of Novel Big Data Analytics Framework for Smart Clothing

Book Author: Siew Teay Hon

Year of Publication: August 2020

Abstract: In this paper, the prospects from smart clothing such as wearable devices in generating Big Data are critically analyzed with a focus on applications related to healthcare, sports and fashion.The proposed novel framework identifies and discusses

sources of Big Data from the human body, data collection, communication, data storage, data analytics and decision making using artificial intelligence (AI) algorithms. The paper concludes by identifying challenges facing the integration of Big Data analytics with smart clothing. Recommendation for further development opportunities and directions for future work are also suggested.

SURVEY: 3

Book Title: Intelligent Fashion Recommender System: Fuzzy Logic in Personalized Garment Design

Book Author: L.C.Wang,Y.Chen

Year of Publication: November 2014

Abstract: This paper proposes a new intelligent fashion recommender system to select the most relevant garment design scheme for a specific consumer in order to deliver new personalized garment products. This system integrates emotional fashion themes and human perception on personalized body shapes and professional designers' knowledge. The corresponding perceptual data are systematically collected from professional using sensory evaluation techniques.

SURVEY: 4

Book Title: Differentiated Fashion Recommendation Using Knowledge Graph and Data Augmentation

Book Author: Cairong Yan

Year of Publication: July 2019 Abstract: E-commerce recommender systems (RSs) can help users quickly find what they need or new products they might be interested in. The fashion e-commerce websites can collect the attributes of items and users as well as the user purchase behaviors, but lack the fine-grained classification of the items and the implicit relationship between items and users. This paper focuses on Amazon fashion dataset, one of the most widely used datasets in the fashion field. A differentiated recommendation framework is proposed that provides different recommendation paths for active and inactive users to improve the overall recommendation quality.

SURVEY: 5

Book Title: Smart Closet: Statistical-based apparel recommendation system

Book Author: Duangkamol Na Nakorn

Year of Publication: 2014

Abstract: Managing closet has long been a problem especially in today's world where people are always in hurry and most of the time, end up choosing to wear the same dressing styles or the same piece of clothes. In addition, we also notice that people tend to stick with one or two dressing style and buy new clothes that are very similar to the ones they already have. This usually results in a huge waste of time and money. Our team sees such the problems and proposes to develop the Smart Closet Application to help people efficiently manage their closet, easily keep track the clothes in the closet, and effectively utilize their closet inventories. Smart Closet Application allows you to store your closet directly in your smart phone. User can add his/her own clothes and accessories into the Smart Closet system.

2.2 References

1. https://www.verfacto.com/blog/ecommerce/customer-journey/

2. https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=8770225

3. https://mdpi-res.com/d_attachment/informatics/informatics-08-00049/article_deploy/informatics-08-00049.pdf

2.3 Problem Statement Definition

Create a problem statement to understand your customer's point of view. The Customer Problem Statement template helps you focus on what matters to create experiences people will love. A well-articulated customer problem statement allows you and your team to find the ideal solution for the challenges your customers face. Throughout the process, you'll also be able to empathize with your customers, which helps you better understand how they perceive your product or service.

Problem Statement (PS)	**I am (Customer)**	**I'm trying to**	**But**	**Because**	**Which makes me feel**
PS-1	Social Media Influencer	purchase a latest fashion outfit	It doesn't suggests current trends and only show few latest outfits	she doesn't want to navigate around old fashioned products	Disappointed
PS-2	Busy Manager	To purchase a formal outfit for an important meeting with my higher officials	It takes more time to search for the right product	He doesn't want to waste time on searching casuals or any other themed outfits	Frustrated

3.IDEATION & PROPOSED SOLUTION

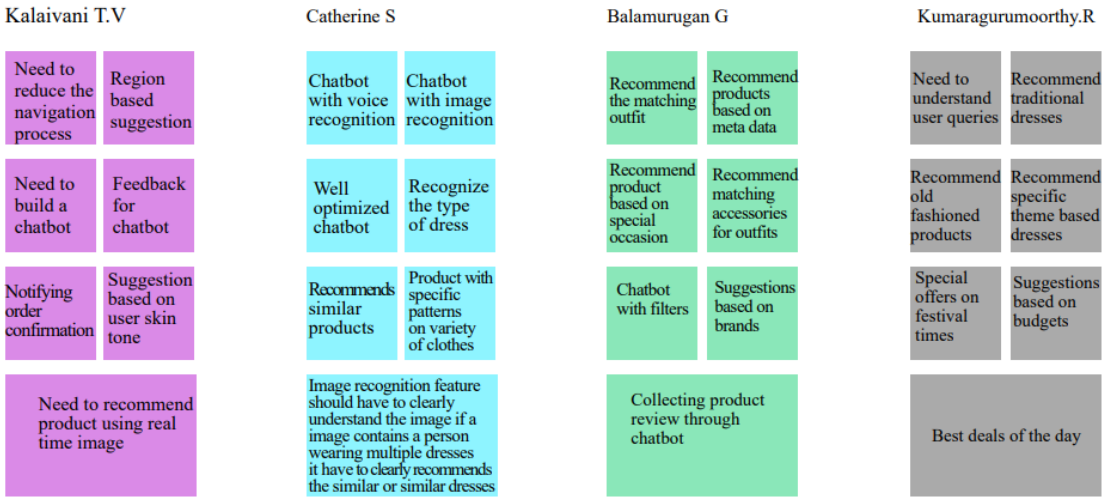
3.1 Empathy Map Canvas

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 helps 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

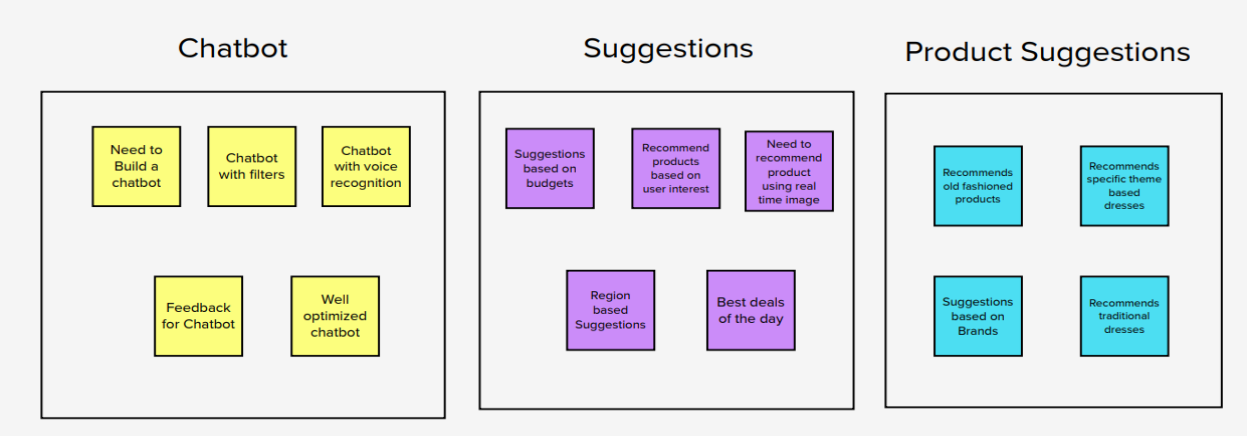
.

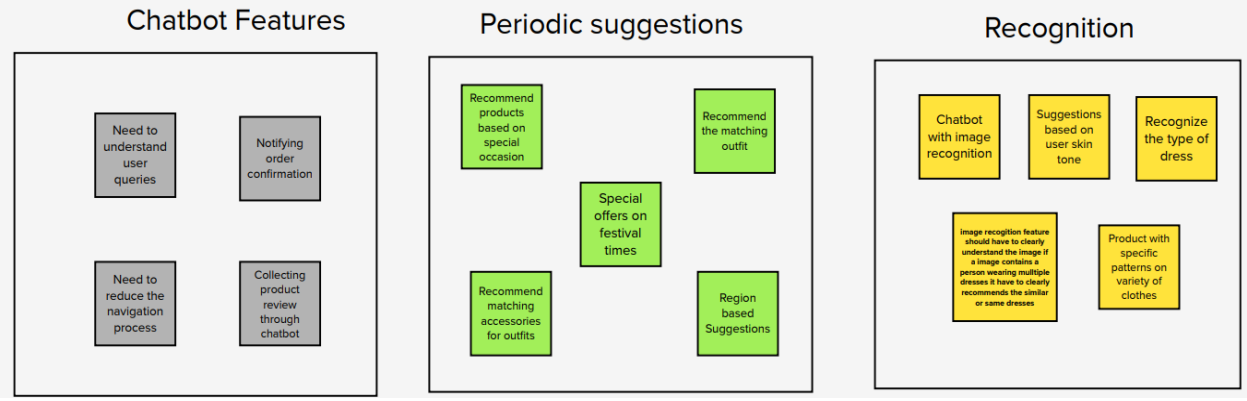
3.2 Ideation & Brainstorming

3.2.1 Brainstorm:



3.2.2 Group Ideas:





Idea Prioritization:

Proposed Solution

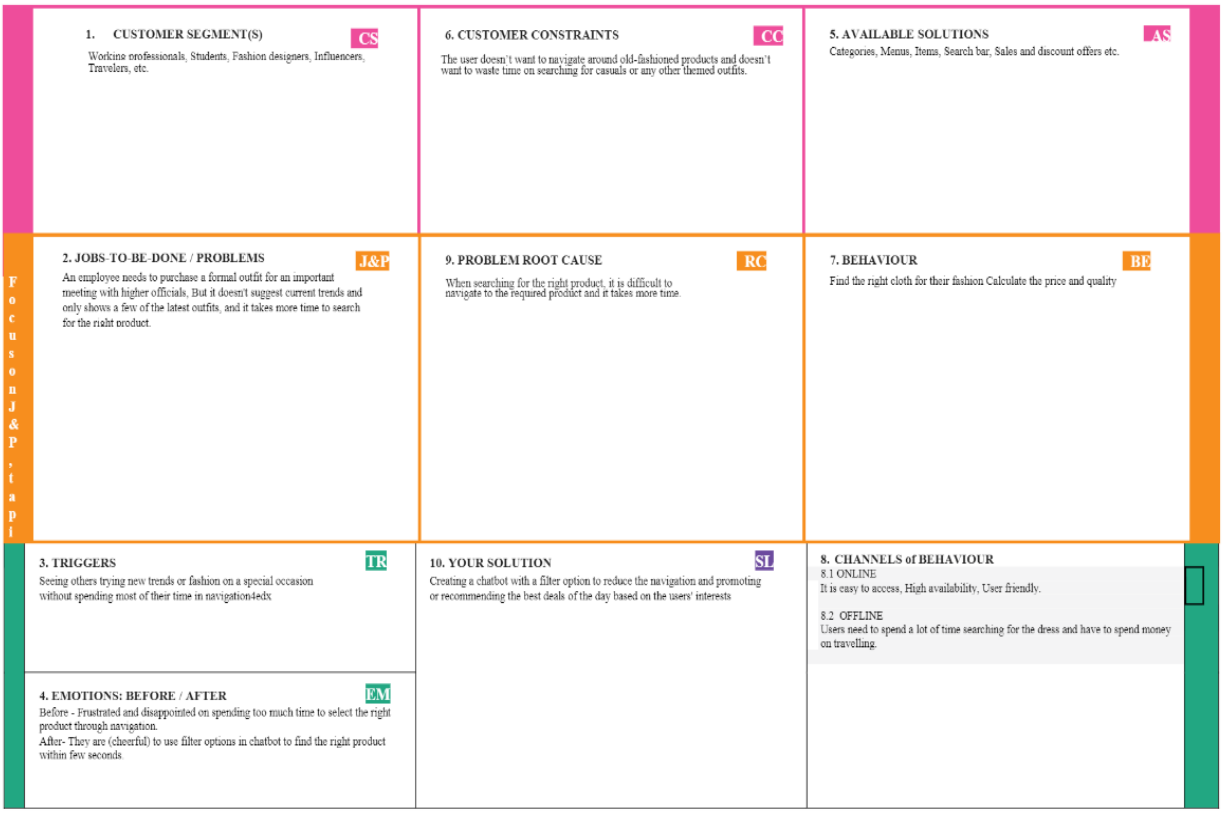
S.No.	Parameter	Description
1	Problem Statement (Problem to be solved)	In general, the user's interests and preferences are different from each other, and it depends on many cases. It could be based on the mood of the user and their personality. It takes more time to navigate to the required product on an e-commerce website. If the issue isn't fixed, then the organization's growth will be affected and the user will get confused while searching for the required product.
2	Idea / Solution description	Creating a Chatbot with filter option to reduce the navigation, and promote or recommends the best deals of the day based on the users interest.
3.	Novelty / Uniqueness	This web application provide filter options through a Chatbot to reduce the navigation.
4.	Social Impact / Customer Satisfaction	The Chatbot recommends suitable products based on the user interest and it also saves a lot of time.
5.	Business Model (Revenue Model)	A Chatbot can be used to sell the products to the customer. So that it can generate revenue.
6.	Scalability of the Solution	A Chatbot efficiently scales horizontally to handle millions of users and interactions per day. Chatbots can increase engagement by up to 90% and sales by 67%

3.4 Problem Solution fit

The Problem-Solution Fit simply means that you have found a problem with your customer and that the solution you have realized for it actually solves the customer's problem. It helps entrepreneurs, marketers and corporate innovators identify behavioral patterns and recognize what would work and why

Purpose:

- Solve complex problems in a way that fits the state of your customers.
- Succeed faster and increase your solution adoption by tapping into existing mediums and channels of behavior.
- Sharpen your communication and marketing strategy with the right triggers and messaging.
- Increase touch-points with your company by finding the right problem-behavior fit and building trust by solving frequent annoyances, or urgent or costly problems.
- Understand the existing situation in order to improve it for your target group



4REQUIREMENT ANALYSIS

4.1 Functional requirement

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form Registration through Gmail
FR-2	User Confirmation	Confirmation via Email Confirmation and also via Chatbot
FR-3.	Watson Assistant Chatbot	Inorder to reduce the navigation of searching the required products, Chatbot is used to recommend the suitable product based on the user queries and get the perfect recommendation within minutes
FR-4	Payment	Authentication, Transaction corrections, adjustments and cancellations.

4.2Non-Functional requirement

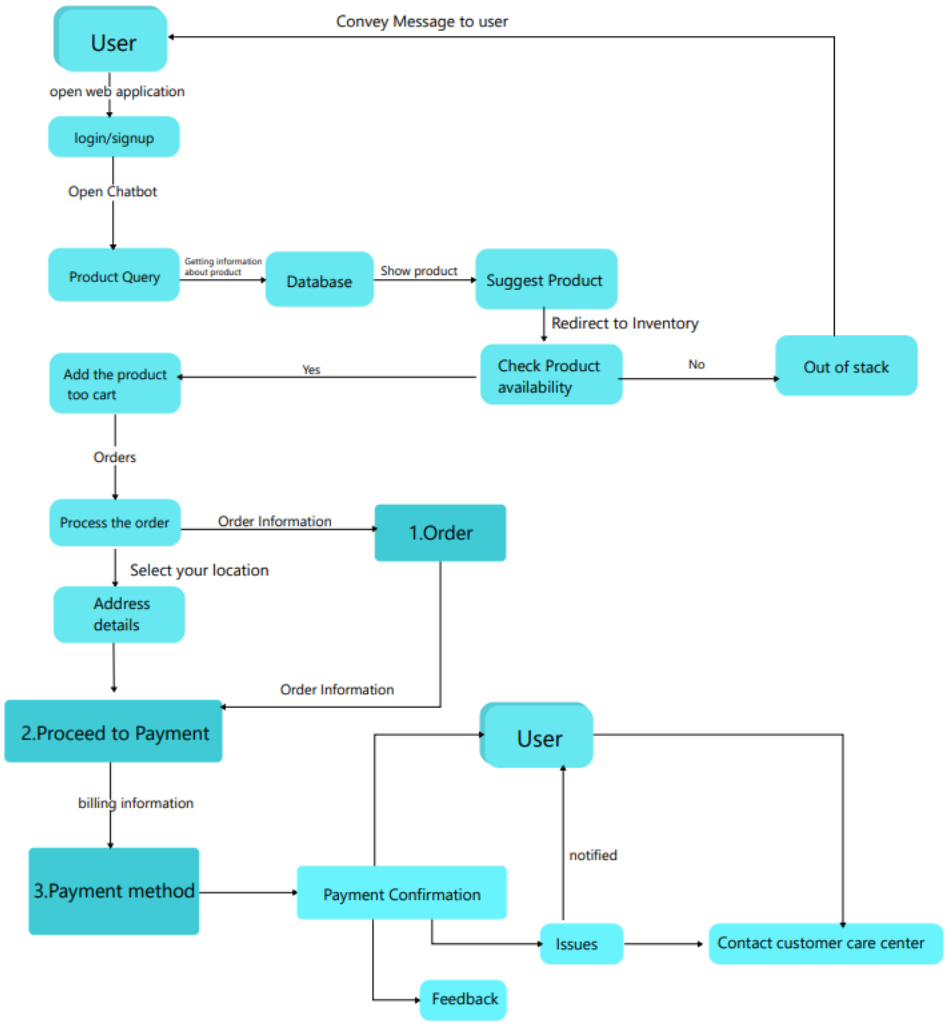
FR No	Non-Functional Requirement	Description
NFR-1	Usability	A chatbot is used to process the information and perform filtering to predict the products that users will like and accordingly rate the products based on users' preferences. A recommender system easily highlights the most relevant products to the users and ensures faster conversion.
NFR-2	Security	The user's personal information, ratings, and the storage and generation of recommendations are protected against malware attacks and information theft

NFR-3	Reliability	Main task of chatbot is to provide recommendations to the users, with the goals like obtaining enough reliable information about the user's preferences to support their users' decisionmaking process or to help them find relevant information.
NFR-4	Performance	The common way to assess the performance of a recommender system would be through standard metrics such as Accuracy,Precision or Recall. However,these metrics require ground truth knowledge about which recommendations are correct, which is hard to obtain at a large scale in our specific problem setting.In the absence of sufficient amounts of ground truth data, alternative metrics need to be used.
NFR-5	Availability	The Product Availability action lets the chatbot look for specific products in the Shopify store and display matched items as personalized recommendations in the chat. Bots can also boost sales, because of their 24/7 availability and fast responses rate. Service availability is simply the measure of the service being available and accessible to the customers during the time you promised to keep the service available. It's usually calculated as a percentage.
NFR-6	Scalability	A Chatbot efficiently scales horizontally to handle millions of users and interactions per day. Chatbots can increase engagement by up to 90% and sales by 67%

5.PROJECT DESIGN

5.1Data 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:

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

Its goals are to:

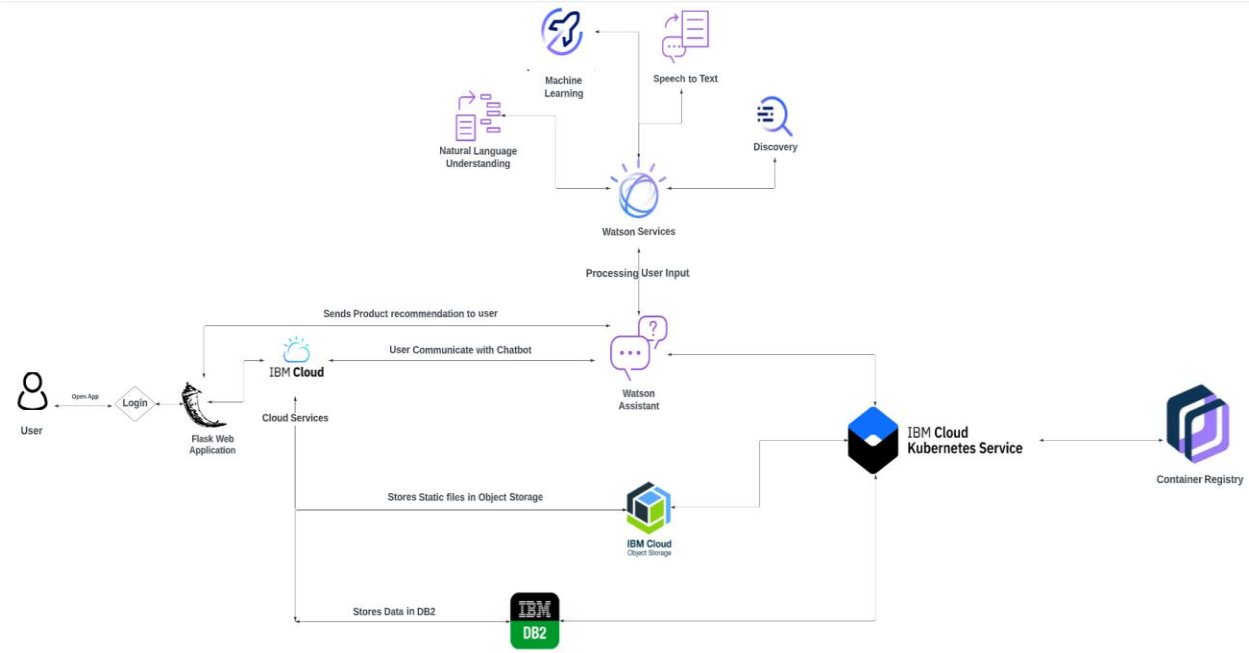
 Find the best tech solution to solve existing business problems.

 Describe the structure, characteristics, behaviour, and other aspects of the software to project stakeholders.

 Define features, development phases, and solution requirements.

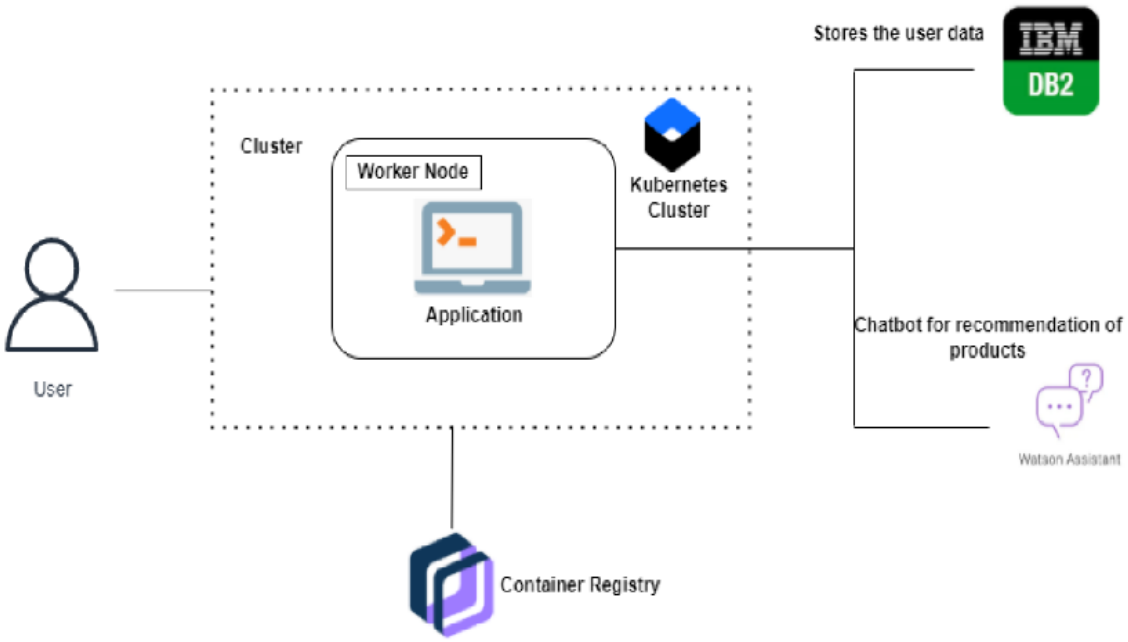
 Provide specifications according to which the solution is defined, managed, and delivered.

Solution Architecture Diagram:



Technical Architecture:

The Deliverable shall include the architectural diagram as below and the information as per the table1 & table 2



Components and Technologies

S.No	Component	Description	Technology
1	User Interface	User interacts with application through Web UI	HTML, CSS, Bootstrap, JavaScript, React Js
2	User login to the e-commerce website.	User will login to the e-commerce website to search and order the required products.	Python, Flask, HTML, CSS, JavaScript, IBM DB2, SendGrid
3	Communicates with Chatbot	The user communicates with the chatbot in either textual form or by speaking directly to the chatbot.	IBM Watson STT service
4	Chatbot recommends products	Chatbot processes the user's input and recommends products based on their needs and interests.	IBM Watson Assistant
5	Database	Stores the user's name, e-mail, address & his interests information in the database. Data types: Integer, String, Float, Varchar, etc.,	MySQL,SQLite, POSTgre SQL.
6	Cloud Database	Once the application is deployed in cloud, The information will be stored at cloud	IBM DB2, IBM Cloudant.
7	File Storage	Necessary files for the application should be maintained at easily accessible place	IBM Object Storage, IBM Block Storage or Local Filesystem

8	External API-1	To send emails to the user	SendGrid
9	Infrastructure (Server / Cloud)	Application Deployment on Local System/ Cloud Local Server Configuration: Cloud Server Configuration	Local, Cloud Foundry, Kubernetes.

Application and Characteristics

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	List the open-source frameworks used	Flask, Bootstrap(CSS framework), React Js
2.	Security Implementations	Each user is provided with unique ID & password. Assures all the data inside the system will be protected against malware attacks or unauthorized access	Encryptions, IAM Controls, SendGrid.
3.	Scalable Architecture	Justify the scalability of architecture(3–tier, Micro-services)	By using Docker and Kubernetes we make our application scalable without making major changes to it
4.	Availability	Justify the availability of application (e.g. use of load balancers, distributed servers etc.)	Usage of Cloud environment like object storage, Watson Assistant & DB2 and clustering using Docker & Kubernetes makes the available all time
5..	Performance	Design consideration for the performance of the application (number of requests per sec, use of Cache, use of CDN's) etc	IBM Cloud, Watson Assistant, Kubernetes cluster, Container Registry

5.2 User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user,I can register for the application by entering my email, password,& 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 Gmail	I can access my account / dashboard	Medium	Sprint-1
	Login	USN-1	As a user, I can log into the application by entering email & password	I can login into Application and Click Login	High	Sprint-1
	Dashboard	USN-1	As a user, I can see multiple products in the Homepage and can see the chatbot icon on the bottom right	I can see latest products and can open chatbot by clicking the Icon	Medium	Sprint-2
		USN-2	As a user, I can open the chatbot and can ask queries	I can ask queries to Chatbot	High	Sprint-2
	Chatbot	USN-1	Chatbot analyze user's request & gives recommendation based on queries	Chatbot recommend required product	High	Sprint-2
	Order	USN-1	As a user, I can add the product items to cart or I can directly order the product	I can order product by clicking add to cart	High	Sprint-3

	Payment	USN-1	As a user, I can proceed to payment via Cash on delivery or Online Payment	I can confirm my order by clicking Pay	Medium	Sprint-4
	Review	USN-1	As a user, I can share my shopping experience and give product review	I can give review for the product	High	Sprint-4
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 Gmail	I can access my account / dashboard	Medium	Sprint-1
	Login	USN-1	As a user, I can log into the application by entering email & password	I can login into Application and Click Login	High	Sprint-1
	Dashboard	USN-1	As a user, I can see multiple products in the Homepage and can see the chatbot icon on the bottom right	I can see latest products and can open chatbot by clicking the Icon	Medium	Sprint-2
		USN-2	As a user, I can open the chatbot and can ask queries	I can ask queries to Chatbot	High	Sprint-2
	Chatbot	USN-1	Chatbot analyze user's request and gives	Chatbot recommend required product	High	Sprint-2

			recommendation based on queries			
	Order	USN-1	As a user, I can add the product items to cart or I can directly order the product	I can order product by clicking add to cart	High	Sprint-3
	Payment	USN-1	As a user, I can proceed to payment via Cash on delivery or Online Payment	I can confirm my order by clicking Pay	Medium	Sprint-4
	Review	USN-1	As a user,I can share my shopping experience and give product review	I can give review for the product	High	Sprint-4
Administrator	Audit Tracking	USN-12	As a Admin, I can Confirm user order and I have to check availability of product	I can receive order confirmation through email	High	Sprint-3
		USN-13	As a Admin, I have to Check the accuracy of order and issuing invoices, maintaining sales records and compiling monthly sales report	I can manage all the functionalities	High	Sprint-3

6.PROJECT PLANNING & SCHEDULING

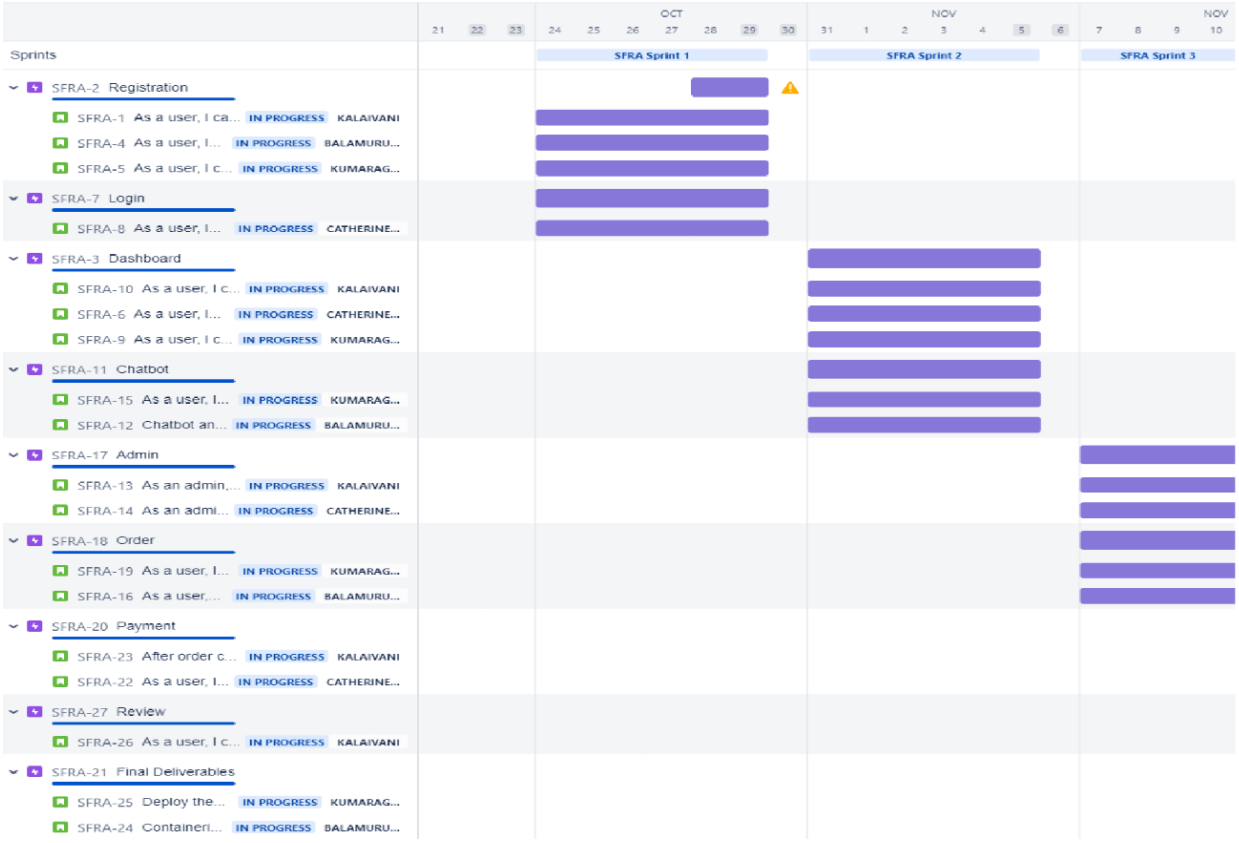
Sprint Planning

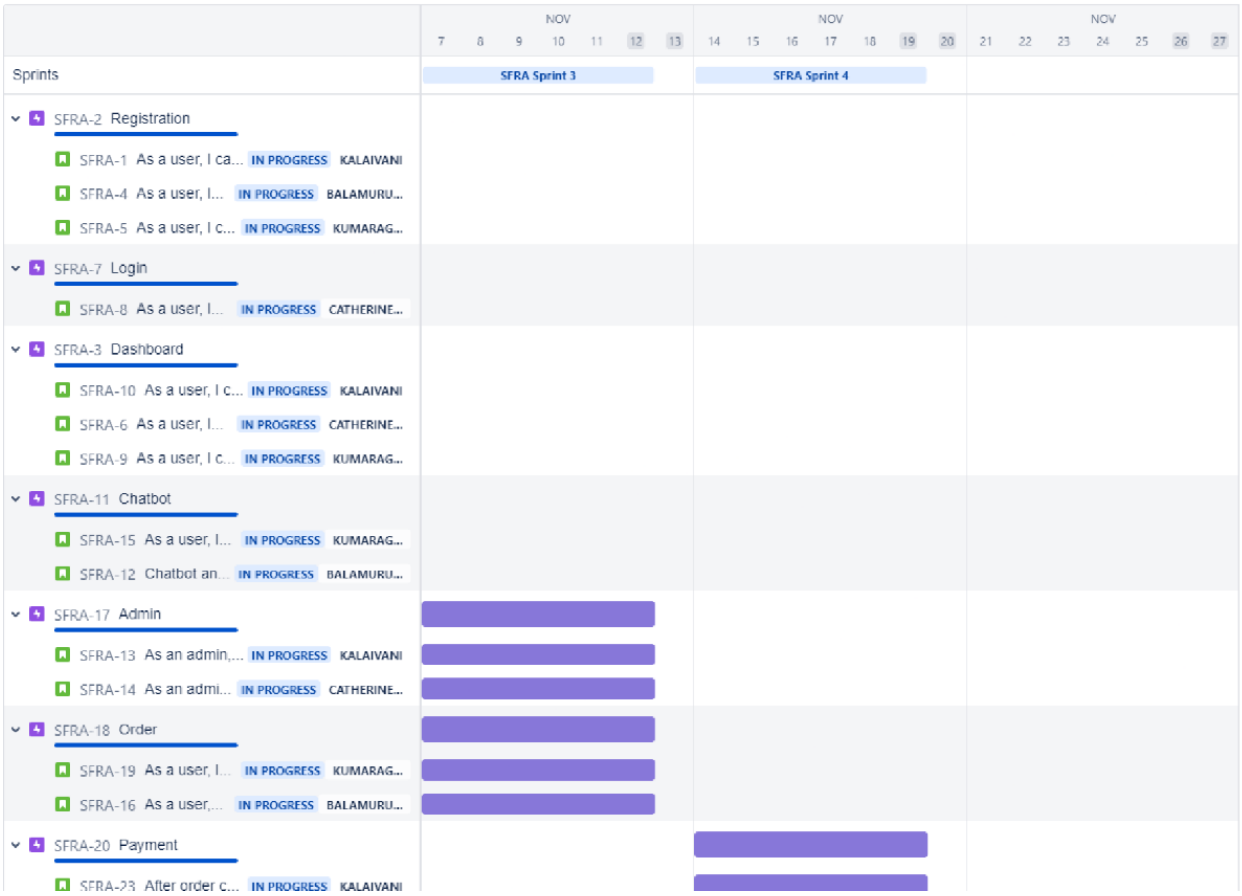
Sprint Planning:

Sprint	Functional Requirement (Epic)	USN	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,& confirming my password.	8	High	Kalaivani TV
Sprint-1		USN-2	As a user, I will receive confirmation email once I have registered for the application	4	High	Balamurugan
Sprint-1		USN-3	As a user, I can register for the application through Gmail	2	Medium	Kumaragurumoorthy
Sprint-1	Login	USN-1	As a user, I can log into the application by entering email & password	6	High	Catherine S
Sprint-2	Dashboard	USN-1	As a user, I can see multiple products in the Homepage	3	Medium	Catherine S
Sprint-2		USN-2	As a user, I can see the chatbot icon on the bottom right	1	High	Kumaragurumoorthy
Sprint-2		USN-3	As a user, I can open the chatbot and can ask queries	1	Medium	Kalaivani T V
Sprint-2	Chatbot	USN-1	Chatbot analyze user's request	8	High	Balamurugan G
		USN-2	As a User , I can receive product recommendation based on the queries from the chatbot	7	High	Kumaragurumoorthy

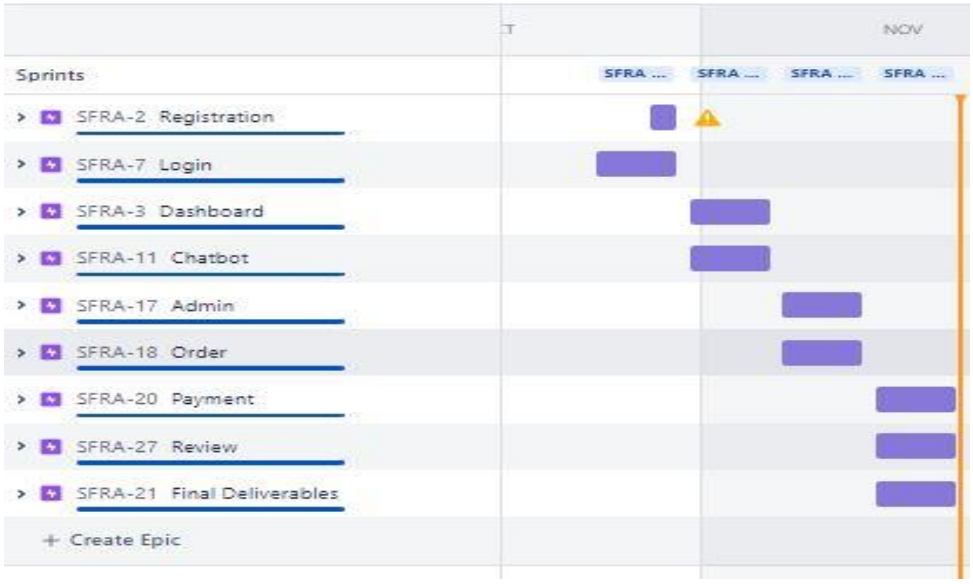
Sprint-3	Admin	USN-1	As an admin, I can check the availability of the selected product	6	High	Kalaivani TV
Sprint-3		USN-2	As an admin, I can track of all the things that the users are purchasing	4	High	Catherine S
Sprint-3	Order	USN-1	As a user, I can directly order the product via chatbot	6	High	Balamurugan G
Sprint-3		USN-2	As a user, I can add the available product items to cart	4	Medium	Kumaragurumoorthy
Sprint-4	Payment	USN-1	As a user, I can proceed to payment via Cash on delivery or Online Payment	4	High	Catherine S
Sprint-4		USN-2	After order confirmation, a notifiatication will be sent to the user by the chatbot	3	High	Kalaivani TV
Sprint-4	Review	USN-1	As a user, I can share my shopping experience and give product review	3	High	Kalaivani TV
Sprint-4	Final deliverables	USN-1	Containerize the application by using Docker and Kubernetes	7	High	Balamurugan G
Sprint-4		USN-2	Deploy the application in the IBM Cloud	3	High	Kumaragurumoorthy

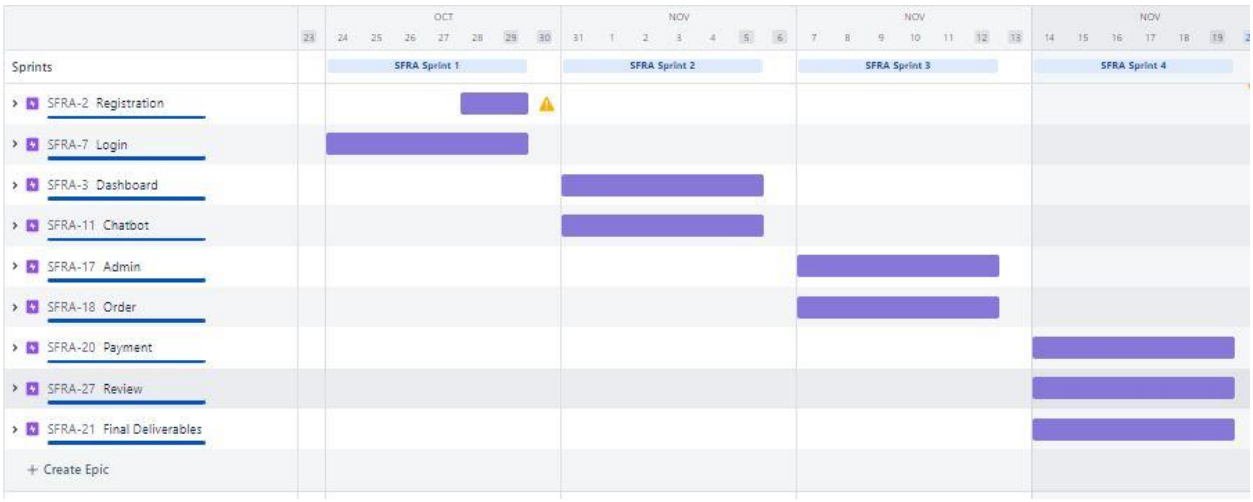
6.2 Sprint Delivery Schedule





6.3 Reports from JIRA





7.CODING & SOLUTIONING

(Explain the features added in the project along with code)

a) Feature 1

Base.html

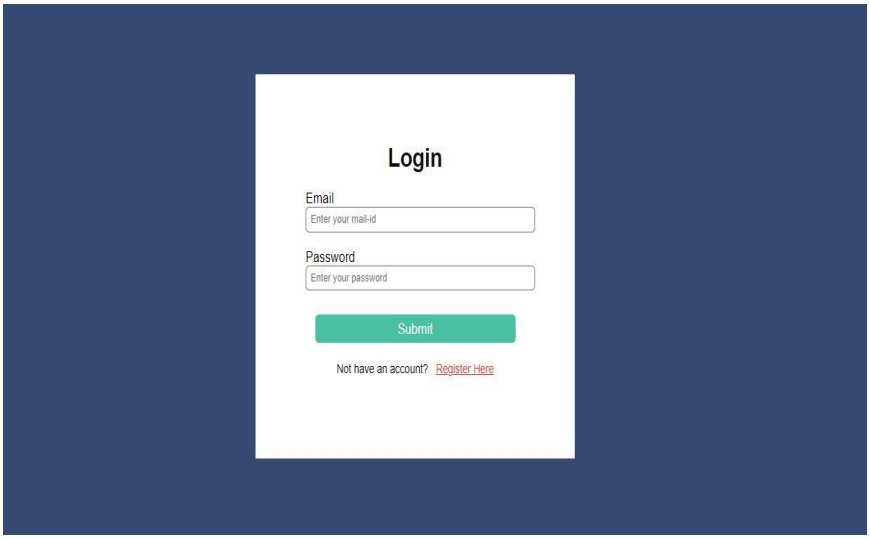
```
<!DOCTYPE html>

<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">

    <!-- CSS only -->

    <!-- <link

href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css"

rel="stylesheet" integrity="sha384-

Zenh87qX5JnK2Jl0vWa8Ck2rdkQ2Bzep5IDxbcnCeuOxjzrPF/et3URy9Bv1WTRi"

crossorigin="anonymous"> -->

    <!-- JavaScript Bundle with Popper -->
```

```
    <!-- <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/js/bootstrap.bundle.min.js
" integrity="sha384-
OERcA2EqjJCMA+/3y+gxIOqMEjwtxJY7qPCqsdltbNJuaOe923+mo//f6V8Qbsw3"
crossorigin="anonymous"></script> -->
    <link href="{{ url_for('static', filename='style.css') }}" rel="stylesheet">
    <link rel="stylesheet" href="{{ url_for('static', filename='dashstyle.css')
}}">
    <!-- <link rel="stylesheet" href="{{ url_for('static',
filename='checkoutstyle.css') }}"> -->

    <!-- JavaScript Bundle with Popper -->
    <!-- <script src="{{url_for('static', filename='alert.js')}}"></script> -->
    <!-- jquery -->
    <script
src="https://ajax.googleapis.com/ajax/libs/jquery/2.1.1/jquery.min.js"></script>
    <title>SFRA</title>
    {%block head%}
    {%endblock%}

</head>
<body>
    {%block body%}
    {%endblock%}
</body>
</html>
```

Login.html

```
{%extends "base.html"%}


{%block head%}


<form action="{{url_for('login')}}" method="POST">
        {%with messages = get_flashed_messages(with_categories=true)%}
            {%if messages%}
                {%for category,message in messages%}
                    <div class="alert alert-{{category}}">{{message}}</div>
                {%endfor%}
            {%endif%}
        {%endwith%}
        <div class="login-box">
            <h1>Login</h1>


              <label>Email</label>
              <input type="email" placeholder="Enter your mail-id" name="mailid"
/>

              <label>Password</label>
              <input type="password" placeholder="Enter your password"
name="pswd"/>
              <p class="para-3">


              <input type="submit" value="Submit" ></p>
```

```
            <p class="para-2">
              Not have an account?  
               <a href="{{url_for('registration')}}">Register Here</a>


            </p>


          </div>
</form>


{%endblock%}
```



Registration.html

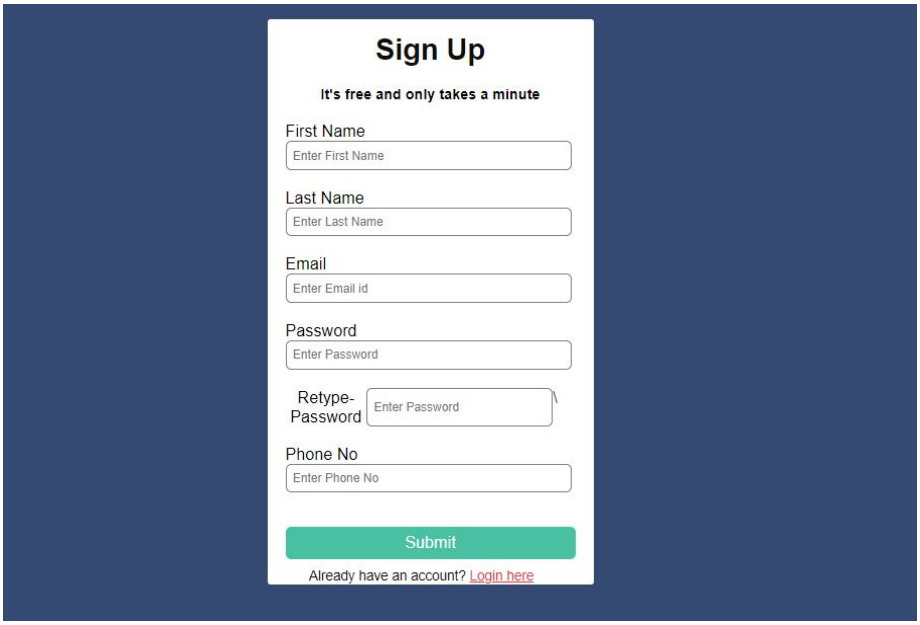
```
{% extends "base.html" %}
```

```
{%block head%}



  <body>
    {%with messages = get_flashed_messages(with_categories=true)%}
            {%if messages%}
                {%for category,message in messages%}
                    <div class="alert-error">{{message}}</div>
                {%endfor%}
            {%endif%}
    {%endwith%}
    <div class="signup-box">
      <h1>Sign Up</h1>
      <h4>It's free and only takes a minute</h4>
      <form action="{{url_for('addrec')}}" method="post">
        <label>First Name</label>
        <input type="text" placeholder="Enter First Name"name="User_fname" />
        <label>Last Name</label>
        <input type="text" placeholder="Enter Last Name" name="User_lname"/>
        <label>Email</label>
        <input type="email" placeholder="Enter Email id" name="User_mailid" />
        <label>Password</label>
        <input type="password" placeholder="Enter Password" name="User_pswd"
id="User_pswd"  /></label>
        <label>Retype-Password
```

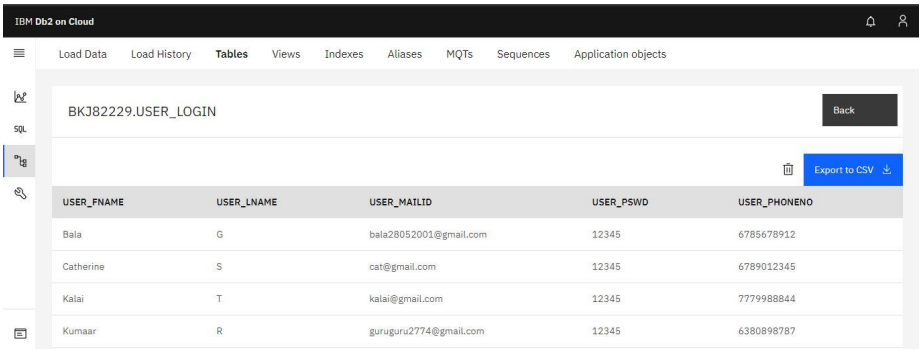
```
        <input type="password" placeholder="Enter Password" name="User_repswd"
id="User_repswd"  />\
        <span id='message'></span></label>
        <label>Phone No</label>
        <input type="phone" placeholder="Enter Phone No" name="User_phoneno"
/><br><br>
        <input type="submit" value="Submit" />


        <!-- <p>
          By clicking the Sign Up button,you agree to our <br />
          <a href="#">Terms and Condition</a> and <a href="#">Policy Privacy</a>
        </p> -->
        <p class="para-2">
        Already have an account? <a href="{{url_for('login')}}">Login here</a>
        </p>
      </form>
    </div>
  </body>


<!-- <form action="{{url_for('addrec')}}" method="post">
    <div class="mb-3">
      <label class="form-label">Enter First Name</label>
      <input type="text" class="form-control" name="User_fname" aria-
describedby="emailHelp">


    </div>
```

```
    <div class="mb-3">
      <label class="form-label">Enter Last Name</label>
      <input type="text" class="form-control" name="User_lname" aria-
describedby="emailHelp">

    </div>
    <div class="mb-3">
      <label class="form-label">Email address</label>
      <input type="email" class="form-control" name="User_mailid" aria-
describedby="emailHelp">

    </div>
    <div class="mb-3">
      <label class="form-label">Enter Password</label>
      <input type="password" class="form-control" name="User_pswd" aria-
describedby="emailHelp">
    </div>
    <div class="mb-3">
      <label class="form-label">Retype Password</label>
      <input type="password" class="form-control" name="User_repswd" aria-
describedby="emailHelp">
    </div>
    <div class="mb-3">
      <label class="form-label">Enter Phone Number</label>
      <input type="number" class="form-control" name="User_phoneno" aria-
describedby="emailHelp">
```

```
    </div>
    <button type="submit" class="btn btn-success btn-block">Register</button>
    <a href="{{url_for('login')}}" class="btn btn-primary"> Back to login </a>
  </form> -->
{%endblock%}
```



After registration the data about the user will be stored in the Database



b.Feature 2
Dashboard.html

```
<!-- <html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Dashboard</title>
</head>
<body>
    <p>Hello </p>
    <hr>
    <a href="{{url_for('logout')}}" class="btn btn-primary btn">Logout</a>

</body>

</html> -->

{%extends "base.html"%}

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

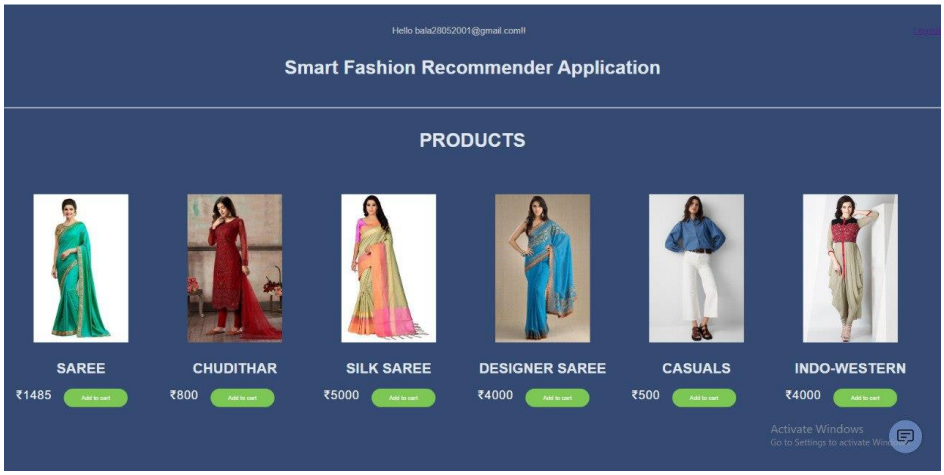
```
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <link rel="stylesheet" href="{{ url_for('static',
filename='dashstyle.css')}}">


    <!-- <link rel="stylesheet" href="Styles.css"> -->
    <script>
      window.watsonAssistantChatOptions = {
        integrationID: "a97c462c-ad3d-44dd-b2af-1fa5c113384e", // The ID of this
integration.
        region: "jp-tok", // The region your integration is hosted in.
        serviceInstanceID: "8984d694-4acd-415b-8d00-a4936a93261f", // 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>
</head>
<body>
```

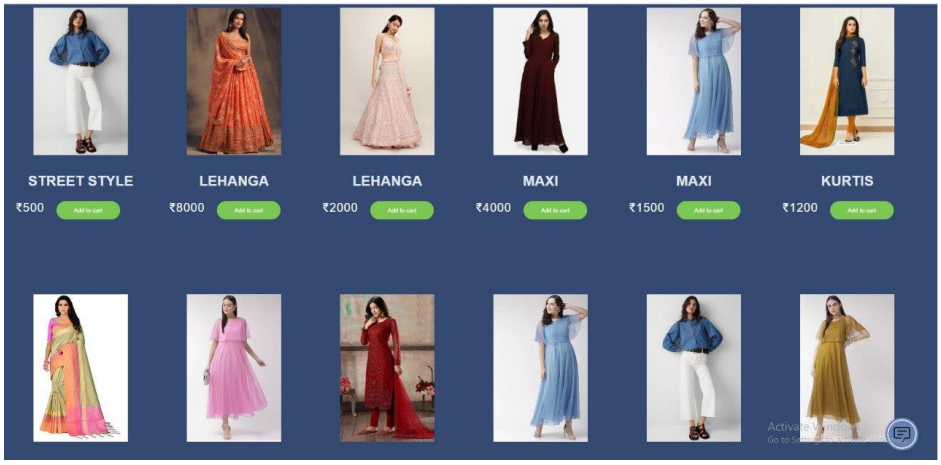
```
    <p >Hello {{session["mailid"]}}!!




    <a href="{{url_for('logout')}}" style="float:right; " >Logout</a>
  </p><h1>Smart Fashion Recommender Application</h1><br>
    <hr>
    <b>
    <h1>PRODUCTS</h1></b>
    {%for row in products%}
    <main class="container">
      <!-- Product Description -->
      <div class="responsive">
        <div class="gallery">
          <div class="product-description">
            <!-- <img src= "{{ url_for('static', filename='cat2.jpeg',
width="12", height="25")}}" > -->
            <img src='https://123sfra.s3.jp-tok.cloud-object-
storage.appdomain.cloud/static/{{row["PIMAGE"]}}'', width="12", height="25" >
            <h2>{{row["PNAME"]}}</h2>
            <!-- Product Pricing -->
            <div class="product-price">
              <span>₹{{row["PPRICE"]}}</span>
              <a href="{{url_for('checkout')}}" class="cart-btn">Add to cart</a>
            </div><br><br>
          </div>
        </div>
```

```
      </div>
    </main>
    {%endfor%}



    <h3>MENS SECTION</h3>
    {%for row in MEN%}
    <main class="container">
      <!-- Product Description -->
      <div class="responsive">
        <div class="gallery">
          <div class="product-description">
            <!-- <img src= "{{ url_for('static', filename='cat2.jpeg',
width="12", height="25")}}" > -->
            <img src='https://123sfra.s3.jp-tok.cloud-object-
storage.appdomain.cloud/static/{{row["PIMAGE"]}}'', width="12", height="25" >
            <h2>{{row["PNAME"]}}</h2>
            <!-- Product Pricing -->
            <div class="product-price">
              <span>₹{{row["PPRICE"]}}</span>
              <a href="{{url_for('checkout')}}" class="cart-btn">Add to cart</a>
            </div><br><br>
          </div>
        </div>
      </div>
    </main>
```

```
    {%endfor%}


    <h3>WOMENS SECTION</h3></b>


    {%for row in WOMEN%}
    <main class="container">
      <!-- Product Description -->
      <div class="responsive">
        <div class="gallery">
          <div class="product-description">
            <!-- <img src= "{{ url_for('static', filename='cat2.jpeg',
width="12", height="25")}}" > -->
            <img src='https://123sfra.s3.jp-tok.cloud-object-
storage.appdomain.cloud/static/{{row["PIMAGE"]}}'', width="12", height="25" >
            <h2>{{row["PNAME"]}}</h2>
            <!-- Product Pricing -->
            <div class="product-price">
              <span>₹{{row["PPRICE"]}}</span>
              <a href="{{url_for('checkout')}}" class="cart-btn">Add to cart</a>
            </div><br><br>
          </div>
        </div>
      </div>
    </main>
    {%endfor%}
```

```
    <h3>KIDS SECTION</h3></b>


    {%for row in KIDS%}
    <main class="container">
      <!-- Product Description -->
      <div class="responsive">
        <div class="gallery">
          <div class="product-description">
            <!-- <img src= "{{ url_for('static', filename='cat2.jpeg',
width="12", height="25")}}" > -->
            <img src='https://123sfra.s3.jp-tok.cloud-object-
storage.appdomain.cloud/static/{{row["PIMAGE"]}}'', width="12", height="25" >
            <h2>{{row["PNAME"]}}</h2>
            <!-- Product Pricing -->
            <div class="product-price">
              <span>₹{{row["PPRICE"]}}</span>
              <a href="{{url_for('checkout')}}" class="cart-btn">Add to cart</a>
            </div><br><br>
          </div>
        </div>
      </div>
    </main>
    {%endfor%}
</body>
</html>
{%endblock%}
```


Hello bala28052001@gmail.com!!
Smart Fashion Recommender Application
PRODUCTS
SAREE
₹1485
CHUDITHAR
₹800
SILK SAREE
₹5000
DESIGNER SAREE
₹4000
CASUALS
₹500
INDO-WESTERN
₹4000
Activate Windows


STREET STYLE
₹500
LEHANGA
₹8000
LEHANGA
₹2000
MAXI
₹4000
MAXI
₹1500
KURTIS
₹1200

KIDS SECTION

CHOLI

₹420 Add to cart

GOWN

₹500 Add to cart

GOWN

₹550 Add to cart

CHOLI

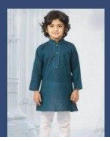
₹700 Add to cart

HOODIE

₹800 Add to cart

STREET STYLE

₹900 Add to cart



Activate Windows
Go to Settings to activate Windows.

WOMENS SECTION

SAREE

₹1485 Add to cart

CHUDITHAR

₹800 Add to cart

SILK SAREE

₹5000 Add to cart

DESIGNER SAREE

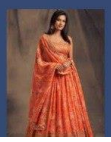
₹4000 Add to cart

CASUALS

₹500 Add to cart

INDO-WESTERN

₹4000 Add to cart



Activate Windows
Go to Settings to activate Windows.

MENS SECTION

JACKET

₹800 Add to cart

BLAZER

₹400 Add to cart

BLAZER

₹1000 Add to cart

HOODIE

₹1400 Add to cart

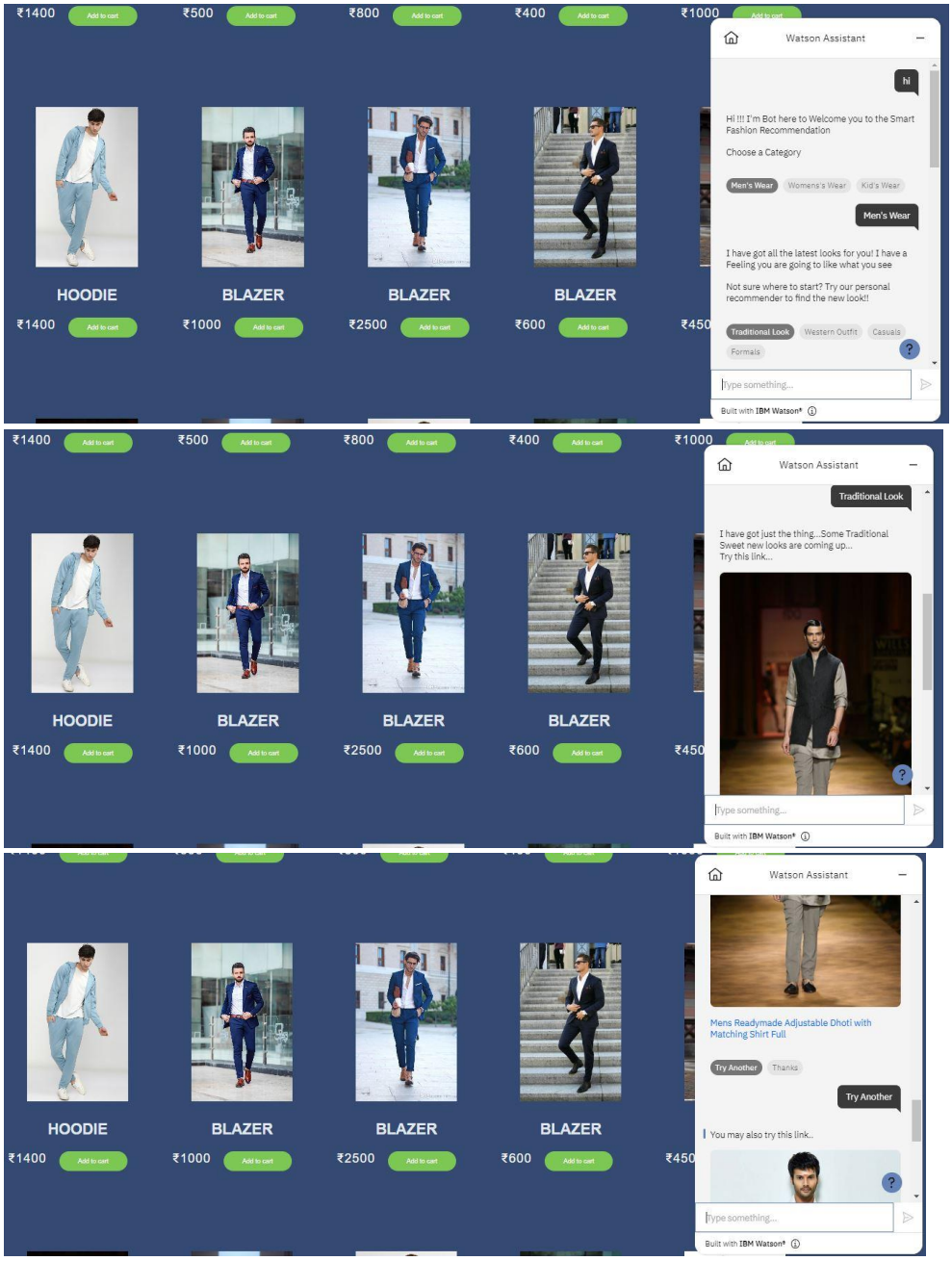
BLAZER

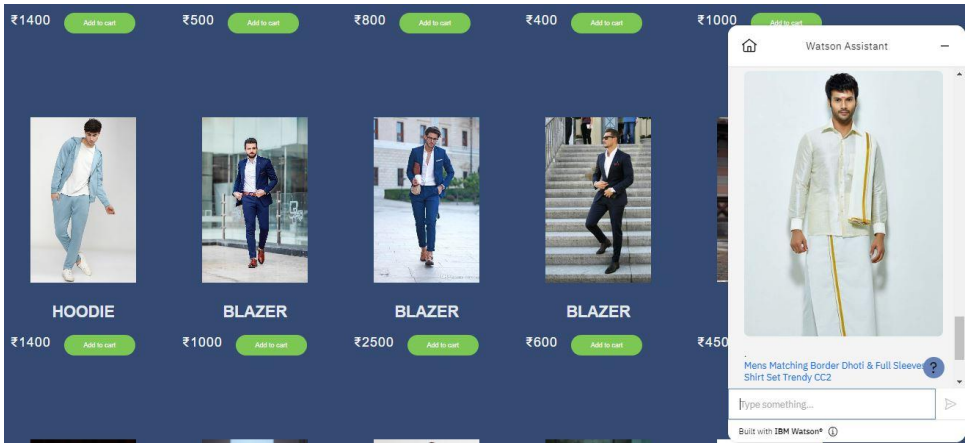
₹1000 Add to cart

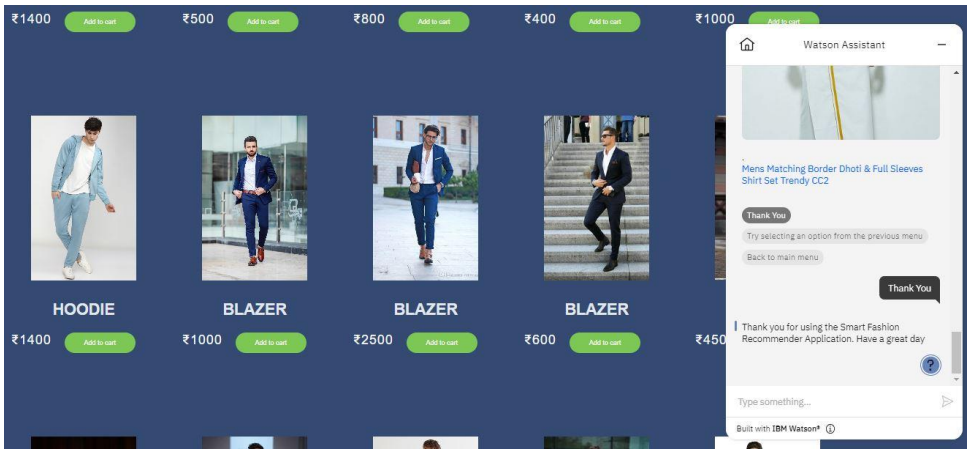
BLAZER

₹2500 Add to cart

Activate Windows
Go to Settings to activate Windows.


₹1400
Add to cart
₹500
Add to cart
₹800
Add to cart
₹400
Add to cart
₹1000
Add to cart
HOODIE
BLAZER
BLAZER
BLAZER
₹1400
Add to cart
₹1000
Add to cart
₹2500
Add to cart
₹600
Add to cart
₹450
Watson Assistant
hi
Hi !!! I'm Bot here to Welcome you to the Smart Fashion Recommendation
Choose a Category
Men's Wear
Womens's Wear
Kid's Wear
Men's Wear
I have got all the latest looks for you! I have a Feeling you are going to like what you see
Not sure where to start? Try our personal recommender to find the new look!!
Traditional Look
Western Outfit
Casuals
Formals
Type something...
Built with IBM Watson*
Watson Assistant
Traditional Look
I have got just the thing...Some Traditional Sweet new looks are coming up...
Try this link...
Type something...
Built with IBM Watson*
Watson Assistant
Mens Readymade Adjustable Dhoti with Matching Shirt Full
Try Another
Thanks
Try Another
You may also try this link..
Type something...
Built with IBM Watson*


₹1400
Add to cart
₹500
Add to cart
₹800
Add to cart
₹400
Add to cart
₹1000
Add to cart
HOODIE
BLAZER
BLAZER
BLAZER
₹1400
Add to cart
₹1000
Add to cart
₹2500
Add to cart
₹600
Add to cart
Watson Assistant
Mens Matching Border Dhoti & Full Sleeves Shirt Set Trendy CC2
Type something...
Built with IBM Watson®


₹1400
Add to cart
₹500
Add to cart
₹800
Add to cart
₹400
Add to cart
₹1000
Add to cart
HOODIE
BLAZER
BLAZER
BLAZER
₹1400
Add to cart
₹1000
Add to cart
₹2500
Add to cart
₹600
Add to cart
Watson Assistant
Mens Matching Border Dhoti & Full Sleeves Shirt Set Trendy CC2
Thank You
Try selecting an option from the previous menu
Back to main menu
Thank You
Thank you for using the Smart Fashion Recommender Application. Have a great day
Type something...
Built with IBM Watson®

Watson Assistant

hi

Hi !!! I'm Bot here to Welcome you to the Smart Fashion Recommendation

Choose a Category

Men's Wear | Womens's Wear | Kid's Wear

Womens's Wear

Not sure where to start? Try our personal recommender to find the new look!!

Saree | Chudithar | Lehanga | Western wear

Saree

Type something...

Built with IBM Watson®

Watson Assistant

Saree

What kind of Sarees?

Organza saree | Silk saree | Cotton saree | Designer saree

Organza saree

Shop from a massive variety of **Organza Saree** for women in India. Check out Organza Saree..

Type something...

Built with IBM Watson®

Watson Assistant

.Sangria Black Belted Organza Saree

Thanks for recommendation

Try selecting an option from the previous menu

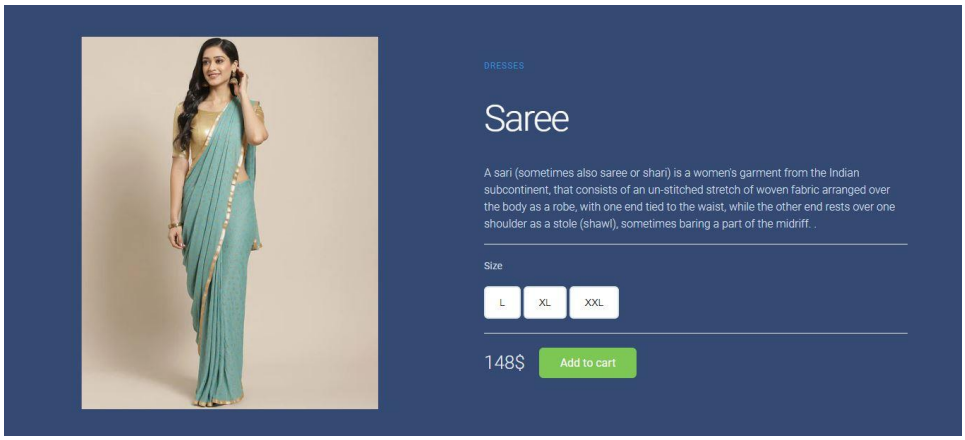
Back to main menu

Thanks for recommendation

Thank you for using the Smart Fashion Recommender Application. Have a great day

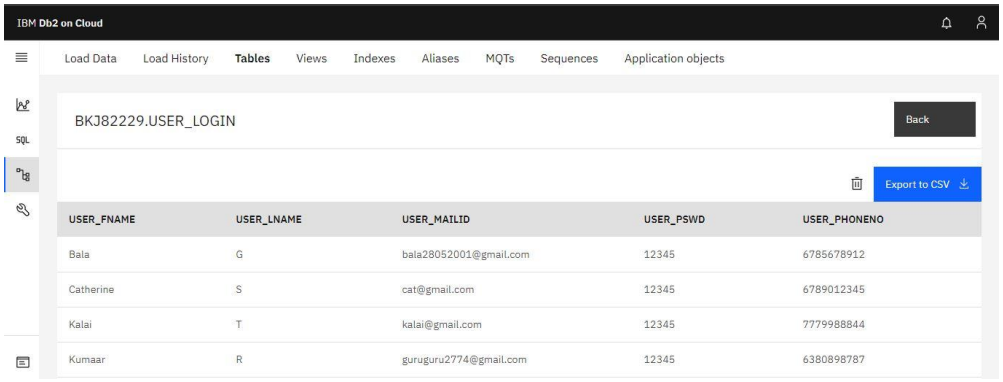
Type something...

Built with IBM Watson®

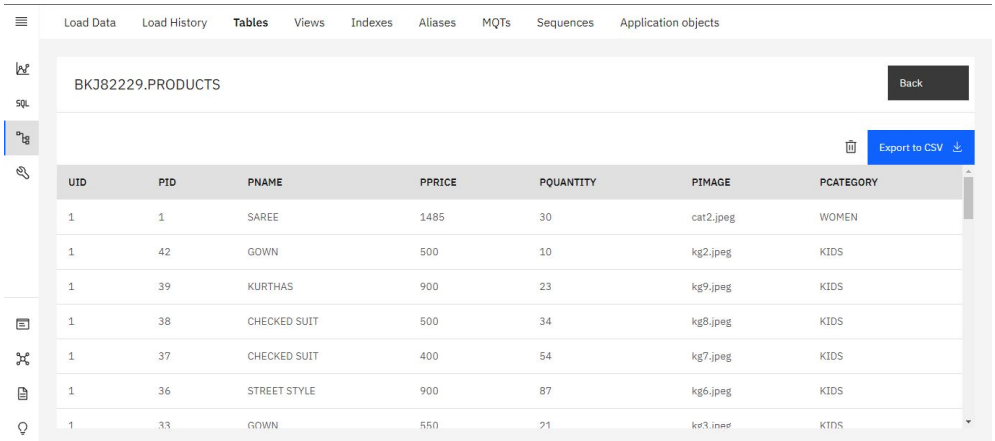


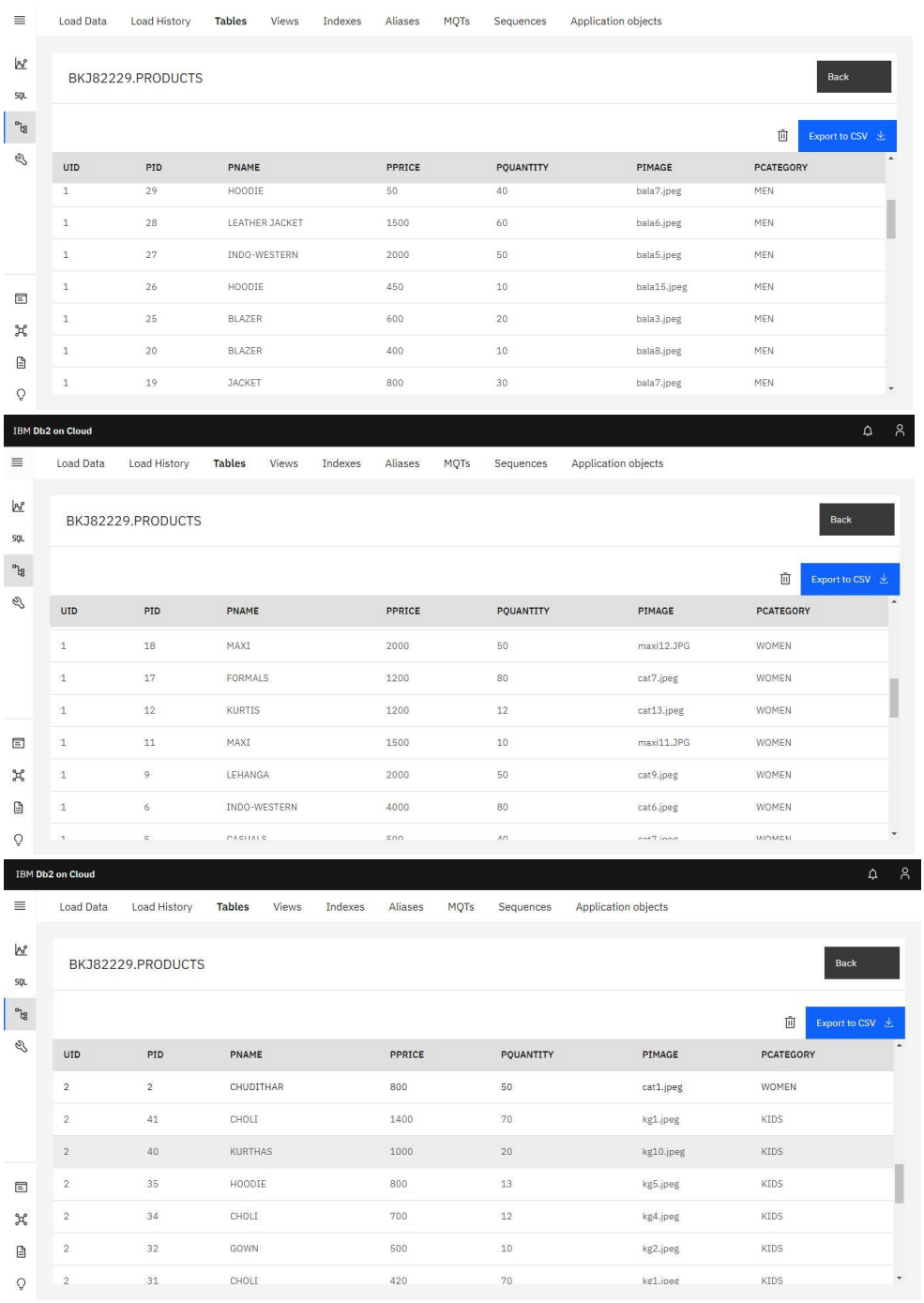
Database Schema (if Applicable)

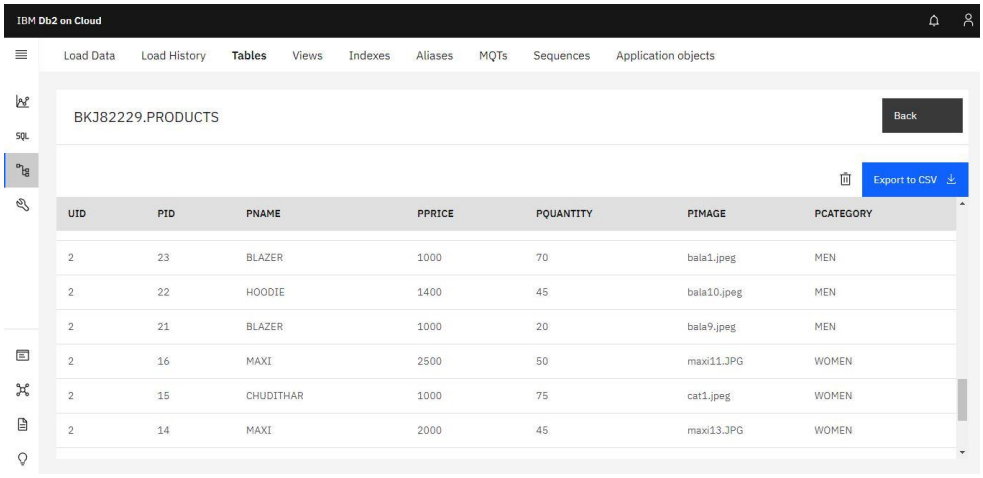
Database connectivity for registration

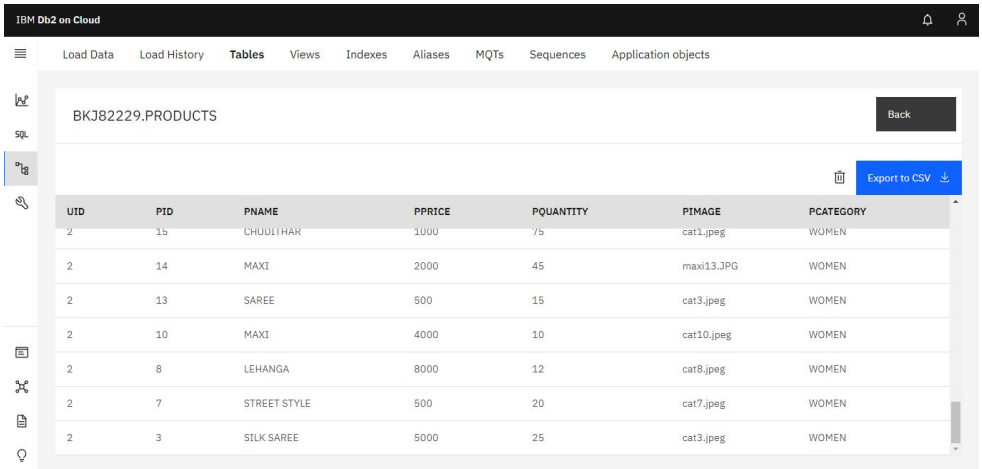


Database connectivity for dashboard




Load Data
Load History
Tables
Views
Indexes
Aliases
MQTs
Sequences
Application objects
BKJ82229.PRODUCTS
Back
Export to CSV
UID PID PNAME PPRICE PQUANTITY PIMAGE PCATEGORY
1 29 HOODIE 50 40 bala7.jpeg MEN
1 28 LEATHER JACKET 1500 60 bala6.jpeg MEN
1 27 INDO-WESTERN 2000 50 bala5.jpeg MEN
1 26 HOODIE 450 10 bala15.jpeg MEN
1 25 BLAZER 600 20 bala3.jpeg MEN
1 20 BLAZER 400 10 bala8.jpeg MEN
1 19 JACKET 800 30 bala7.jpeg MEN
IBM Db2 on Cloud
Load Data
Load History
Tables
Views
Indexes
Aliases
MQTs
Sequences
Application objects
BKJ82229.PRODUCTS
Back
Export to CSV
UID PID PNAME PPRICE PQUANTITY PIMAGE PCATEGORY
1 18 MAXI 2000 50 maxi12.JPG WOMEN
1 17 FORMALS 1200 80 cat7.jpeg WOMEN
1 12 KURTIS 1200 12 cat13.jpeg WOMEN
1 11 MAXI 1500 10 maxi11.JPG WOMEN
1 9 LEHANGA 2000 50 cat9.jpeg WOMEN
1 6 INDO-WESTERN 4000 80 cat6.jpeg WOMEN
IBM Db2 on Cloud
Load Data
Load History
Tables
Views
Indexes
Aliases
MQTs
Sequences
Application objects
BKJ82229.PRODUCTS
Back
Export to CSV
UID PID PNAME PPRICE PQUANTITY PIMAGE PCATEGORY
2 2 CHUDITHAR 800 50 cat1.jpeg WOMEN
2 41 CHOLI 1400 70 kg1.jpeg KIDS
2 40 KURTHAS 1000 20 kg10.jpeg KIDS
2 35 HOODIE 800 13 kg5.jpeg KIDS
2 34 CHOLI 700 12 kg4.jpeg KIDS
2 32 GOWN 500 10 kg2.jpeg KIDS
2 31 CHOLI 420 70 kg1.jpeg KIDS



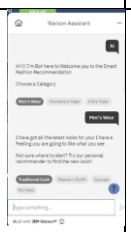


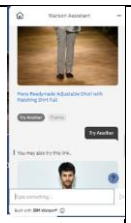
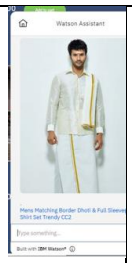
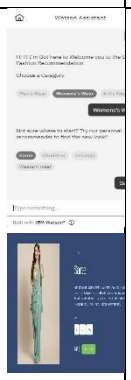
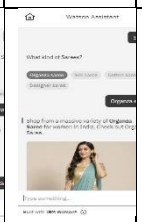
8.TESTING

8.1Test Cases

Tc Id	Feature type	Component	Test Scenario	Pre-Requisites	Step to execute	Test Data	Expected Result	Actual Result	Status	TC for automation (Y/N)	Executed by
Tc 01	Functional	Login Page	Verify user is able to log into application with valid credentials	HTML, CSS, JS, Python flask	1.click the url 2.Enter valid email in email text box 3.Enter Valid password in Password Text box 4.Click on login button	User name: Catherine Password: 12345	Application should show login successful	Working as expected	Pass	Yes	Catherine.S

Tc 02	Functional	Login Page	Verify user is able to log into application with invalid credentials	HTML, CSS, JS, Python flask	1.click the url 2.Enter valid email in email text box 3.Enter Valid password in Password Text box 4.Click on login button	User name: Kumar Password: kumar@gmail.com	Application should show login unsuccessful	Working as expected	Fail	Yes	Kalaivani T.V
Tc 03	Functional	Register Page	User enters his/her valid details to register into the platform for further accesing	HTML, CSS, JS, Python flask	1.click the url 2.click on register here 3.create valid user name 4.enter a valid email id 5.Enter valid password in password text box 6.Click on register button	User name: Kalai Email: kalai@gmail.com Password: 12345	Application should show register succesful	Working as expected	Pass	Yes	Balamurugan M
Tc 04	Functional	Register Page	User enters his/her invalid details to register into the platform for further accesing	HTML, CSS, JS, Python flask	1.click the url 2.click on register here 3.create invalid user name/ email id /password 4..Click on register button	User name: Kalai Email: cat@gmail.com Password: 12345	Application show invalid data register unsuccesful	Working as expected	Fail	Yes	Kumaraguru moorthy
Tc 05	UI	Home page	Helps the user to know well about the platform	HTML, CSS, JS, Python flask	1.enter url 2.verify login / signup popup	User name: Catherine Password: 12345	user should enter valid details to go into the homepage	Navigation passed	Pass	N	Catherine

					Displayed or not						
Tc 06	Functional	chatbot	Provides recommendation based on user queries	HTML, CSS, JS, Python flask, Watson assistant	1.enter url 2.complete the login activity 3.After the successful login it goes to the dashboard 4.Click the assistant icon on the bottom right corner	Hi	Chatbot get started	Working as expected	Pass		**Kalaivani**
Tc 07	Functional	chatbot	Provides recommendation based on user queries	HTML, CSS, JS, Python flask, Watson assistant	1.On the dashboard, Click the assistant icon on the bottom right corner 2.Enter hi to get started 3.Choose a category that chatbot provided			Working as expected	pass	N	Balamurugan M
Tc 08	Functional	chatbot	Provides recommendation based on user queries	HTML, CSS, JS, Python flask, Watson assistant	1.On the dashboard, Click the assistant icon on the bottom right corner 2.Enter hi to get started 3.Choose a category that chatbot provided			Working as expected	pass	y	kumaraguru moorthy

Tc 09	Functional	chatbot	Provides recommendation based on user queries	HTML, CSS, JS, Python flask, Watson assistant	1.On the dashboard, Click the assistant icon on the bottom right corner 2.Enter hi to get started 3.Choose a category that chatbot provided 4.choose the recommended product else try new			Working as expected	pass	N	Balamurugan M
Tc 10	Functional	chatbot	Provides recommendation based on user queries	HTML, CSS, JS, Python flask, Watson assistant	1.On the dashboard, Click the assistant icon on the bottom right corner 2.Enter hi to get started 3.Choose a category that chatbot provided and can add te product to cart			Working as expected	pass	N	Kalaivani

8.2User Acceptance Testing

1. Purpose of Document

The purpose of this document is to briefly explain the test coverage and open issues of the Smart Fashion Recommendation Application project at the time of the release to User Acceptance Testing (UAT).

2. Defect Analysis

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

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	3	0	2	0	5
Duplicate	1	0	0	1	2
External	2	2	1	1	6
Fixed	9	10	12	14	45
Not Reproduced	0	0	1	1	2
Skipped	1	0	1	1	3
Won't Fix	1	0	1	0	2
Total	17	12	18	18	65

3. Test Case Analysis

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

Section	Total Cases	Not Tested	Fail	Pass
Print Engine	8	0	0	8
Client Application	10	0	0	10
Security	10	0	0	10
Outsource Shipping	2	0	0	2
Execption Reporting	10	0	0	10
Final Report Output	9	0	0	9
Version Control	8	0	0	8

9.RESULTS

a. Performance Metrics

S. No	Project Overview	NFT test approach	Assumptions/Dependencies/Risks	Approvals/Signoff
1	Recommend Fashionable products based on user Queries	Stress	Appcrash/Development team/Network issues	Approved
2	Recommend Fashionable products based on user Queries	Load	Server crash/Site down	Approved

End of the Report

NFR-Met	Test Outcome	GO/NOGO Decision	Recommendations	Identified Defects (Detected/Closed/Open)
Performance	CPU-01	GO	High performence	Closed
Database Information	Storage	NO-GO	IBM DB2	Closed

10.ADVANTAGES & DISADVANTAGES

10.1 Advantage

Recommender systems are becoming an essential part of modern life. Recommender systems basically work in one of two ways: suggesting items similar to

the ones a person likes or suggesting items liked by people who are similar to the user. They might look at all the items that a user has rated and then look for items that are similar to the things the user likes. For customers, smart fashion recommender systems can help them find items which they are interested in. For enterprises, It can improve the loyalty of their customers by enhancing the UX and further recommend more browser to users

10.2 Disadvantage

E-commerce has dramatically affected consumer choice. Unconstrained by physical limitations of the brick-and-mortar model, businesses can offer virtually unlimited selections of products online, giving consumers access not only to popular items but to obscure, niche ones as well. The availability of abundant data is what a recommendation system needs. It can only recommend a product to a user when it has enough information about the user if it has not an enough information

9 CONCLUSION

The Smart Fashion Recommender application helps to shorten the distance between the customers need and satisfaction. Not only do they help find the search product; they also discover the needs that customers are not even aware they haveThese system use information filtering techniques to process information and provide the user with potentially more relevant items.

12FUTURE SCOPE

Currently, Our smart fashion recommendation system helps to reduce navigation instead of navigating to multiple screens or pages and it recommends suitable products based on user queries and filtering options. In future we enhance it with voice recognition

10 APPENDIX

Source Code

Base.html

```
<!DOCTYPE html>

<html lang="en">
```

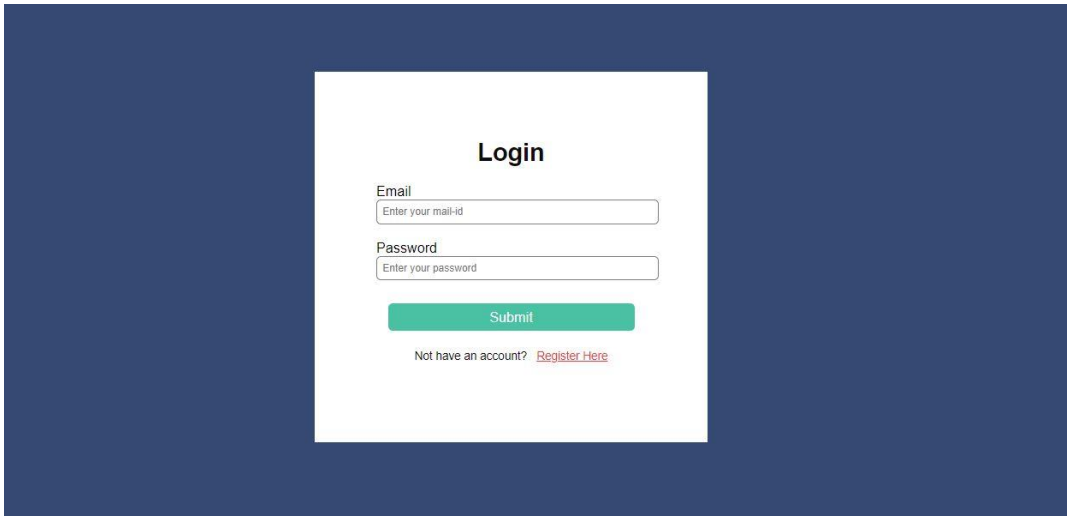
```
<head>


    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <!-- CSS only -->
    <!-- <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
Zenh87qX5JnK2Jl0vWa8Ck2rdkQ2Bzep5IDxbcnCeuOxjzrPF/et3URy9Bv1WTRi"
crossorigin="anonymous"> -->
    <!-- JavaScript Bundle with Popper -->
    <!-- <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/js/bootstrap.bundle.min.js
" integrity="sha384-
OERcA2EqjJCMA+/3y+gxIOqMEjwtxJY7qPCqsdltbNJuaOe923+mo//f6V8Qbsw3"
crossorigin="anonymous"></script> -->
    <link href="{{ url_for('static', filename='style.css') }}" rel="stylesheet">
    <link rel="stylesheet" href="{{ url_for('static', filename='dashstyle.css')
}}">
    <!-- <link rel="stylesheet" href="{{ url_for('static',
filename='checkoutstyle.css') }}"> -->


    <!-- JavaScript Bundle with Popper -->
    <!-- <script src="{{url_for('static', filename='alert.js')}}"></script> -->
    <!-- jquery -->
```

```
    <script
src="https://ajax.googleapis.com/ajax/libs/jquery/2.1.1/jquery.min.js"></script>
    <title>SFRA</title>
    {%block head%}
    {%endblock%}


</head>
<body>
    {%block body%}
    {%endblock%}
</body>
</html>
```

Login.html

```
{%extends "base.html"%}


{%block head%}


<form action="{{url_for('login')}}" method="POST">
        {%with messages = get_flashed_messages(with_categories=true)%}
            {%if messages%}
                {%for category,message in messages%}
                    <div class="alert alert-{{category}}">{{message}}</div>
                {%endfor%}
            {%endif%}
        {%endwith%}
```

```
        <div class="login-box">
            <h1>Login</h1>

              <label>Email</label>
              <input type="email" placeholder="Enter your mail-id" name="mailid"
/>
              <label>Password</label>
              <input type="password" placeholder="Enter your password"
name="pswd"/>
              <p class="para-3">

              <input type="submit" value="Submit" ></p>

              <p class="para-2">
                Not have an account?  
                 <a href="{{url_for('registration')}}">Register Here</a>

              </p>

          </div>
</form>

{%endblock%}
```



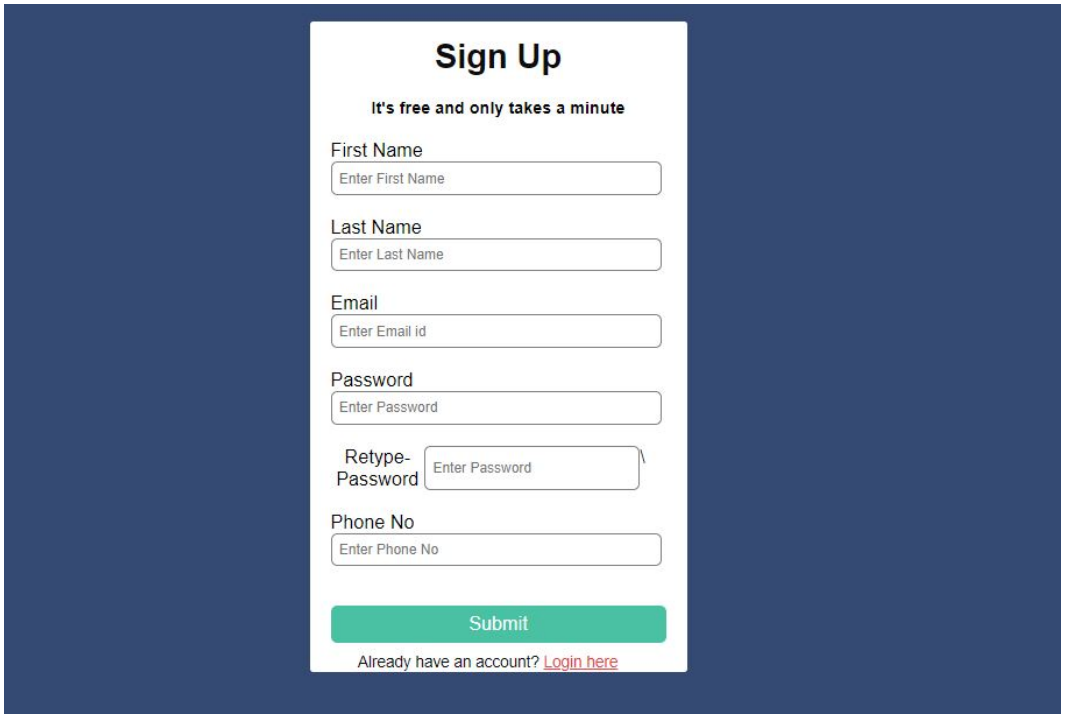
Registration.html

```
{% extends "base.html" %}


{%block head%}



  <body>
    {%with messages = get_flashed_messages(with_categories=true)%}
            {%if messages%}
                {%for category,message in messages%}
                    <div class="alert-error">{{message}}</div>
                {%endfor%}
            {%endif%}
    {%endwith%}
    <div class="signup-box">
      <h1>Sign Up</h1>
      <h4>It's free and only takes a minute</h4>
      <form action="{{url_for('addrec')}}" method="post">
```

```
        <label>First Name</label>
        <input type="text" placeholder="Enter First Name"name="User_fname" />
        <label>Last Name</label>
        <input type="text" placeholder="Enter Last Name" name="User_lname"/>
        <label>Email</label>
        <input type="email" placeholder="Enter Email id" name="User_mailid" />
        <label>Password</label>
        <input type="password" placeholder="Enter Password" name="User_pswd"
id="User_pswd"  /></label>
        <label>Retype-Password
        <input type="password" placeholder="Enter Password" name="User_repswd"
id="User_repswd"  />\
        <span id='message'></span></label>
        <label>Phone No</label>
        <input type="phone" placeholder="Enter Phone No" name="User_phoneno"
/><br><br>
        <input type="submit" value="Submit" />


        <!-- <p>
          By clicking the Sign Up button,you agree to our <br />
          <a href="#">Terms and Condition</a> and <a href="#">Policy Privacy</a>
        </p> -->
        <p class="para-2">
        Already have an account? <a href="{{url_for('login')}}">Login here</a>
        </p>
      </form>
```

```
    </div>
  </body>


<!-- <form action="{{url_for('addrec')}}" method="post">
    <div class="mb-3">
      <label class="form-label">Enter First Name</label>
      <input type="text" class="form-control" name="User_fname" aria-
describedby="emailHelp">

    </div>
    <div class="mb-3">
      <label class="form-label">Enter Last Name</label>
      <input type="text" class="form-control" name="User_lname" aria-
describedby="emailHelp">

    </div>
    <div class="mb-3">
      <label class="form-label">Email address</label>
      <input type="email" class="form-control" name="User_mailid" aria-
describedby="emailHelp">

    </div>
    <div class="mb-3">
      <label class="form-label">Enter Password</label>
      <input type="password" class="form-control" name="User_pswd" aria-
describedby="emailHelp">
```

```
    </div>
    <div class="mb-3">
      <label class="form-label">Retype Password</label>
      <input type="password" class="form-control" name="User_repswd" aria-
describedby="emailHelp">
    </div>
    <div class="mb-3">
      <label class="form-label">Enter Phone Number</label>
      <input type="number" class="form-control" name="User_phoneno" aria-
describedby="emailHelp">
    </div>
    <button type="submit" class="btn btn-success btn-block">Register</button>
    <a href="{{url_for('login')}}" class="btn btn-primary"> Back to login </a>
  </form> -->
{%endblock%}
```



Database creativity

Db.py

```
import ibm_db;

con=ibm_db.connect("DATABASE=bludb;HOSTNAME=2d46b6b4-cbf6-40eb-bbce-

6251e6ba0300.bs2io90l08kqb1od8lcg.databases.appdomain.cloud;PORT=32328;SECURITY=S

SL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=bkj82229;PWD=KCwhio0Cb0XQmB5

H",'','')


sql_command="""CREATE TABLE user_login

            (

            User_fname VARCHAR(20),

            User_lname VARCHAR(20),

            User_mailid VARCHAR(25),

            User_pswd VARCHAR(20),

            User_phoneno VARCHAR(13)
```

```
            );"""


stmt= ibm_db.prepare(con,sql_command)

ibm_db.execute(stmt)
```

dashboard.html

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

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

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

    <title>Dashboard</title>

</head>

<body>

    <p>Hello </p>

    <hr>

    <a href="{{url_for('logout')}}" class="btn btn-primary btn">Logout</a>


</body>


</html> -->


{%extends "base.html"%}
```

```
{%block head%}

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

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

    <title>Document</title>

    <link rel="stylesheet" href="{{ url_for('static',

filename='dashstyle.css')}}">


    <!-- <link rel="stylesheet" href="Styles.css"> -->

    <script>

      window.watsonAssistantChatOptions = {

        integrationID: "a97c462c-ad3d-44dd-b2af-1fa5c113384e", // The ID of this

integration.

        region: "jp-tok", // The region your integration is hosted in.

        serviceInstanceID: "8984d694-4acd-415b-8d00-a4936a93261f", // 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/" +
```

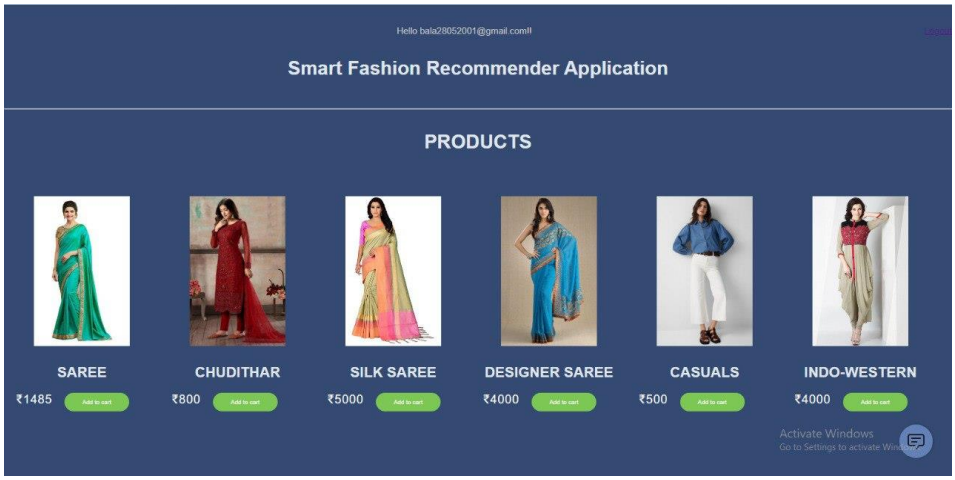
```
(window.watsonAssistantChatOptions.clientVersion || 'latest') +
"/WatsonAssistantChatEntry.js";
        document.head.appendChild(t);
      });
    </script>
</head>
<body>
    <p >Hello {{session["mailid"]}}!!



    <a href="{{url_for('logout')}}" style="float:right; " >Logout</a>
  </p><h1>Smart Fashion Recommender Application</h1><br>
    <hr>
    <b>
    <h1>PRODUCTS</h1></b>
    {%for row in products%}
    <main class="container">
      <!-- Product Description -->
      <div class="responsive">
        <div class="gallery">
          <div class="product-description">
            <!-- <img src= "{{ url_for('static', filename='cat2.jpeg',
width="12", height="25")}}" > -->
            <img src='https://123sfra.s3.jp-tok.cloud-object-
storage.appdomain.cloud/static/{{row["PIMAGE"]}}'', width="12", height="25" >
            <h2>{{row["PNAME"]}}</h2>
```

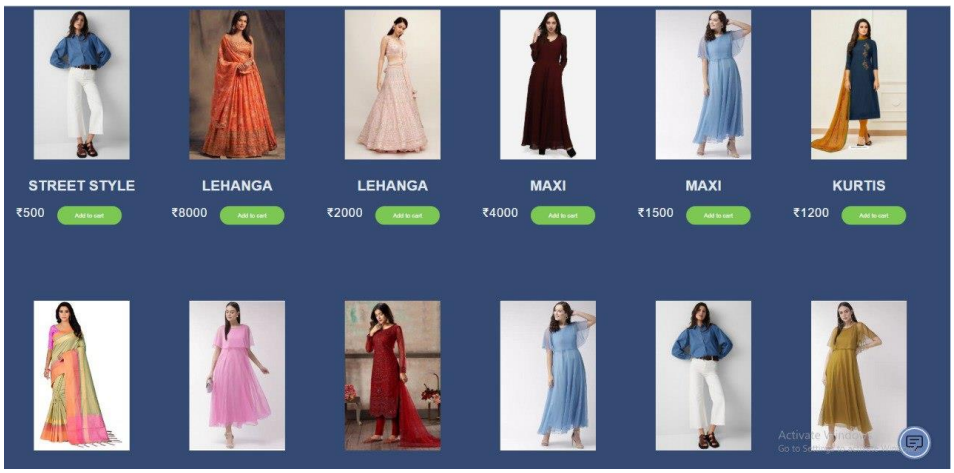
```
            <!-- Product Pricing -->
            <div class="product-price">
              <span>₹{{row["PPRICE"]}}</span>
              <a href="{{url_for('checkout')}}" class="cart-btn">Add to cart</a>
            </div><br><br>
          </div>
        </div>
      </div>
    </main>
    {%endfor%}



    <h3>MENS SECTION</h3>
    {%for row in MEN%}
    <main class="container">
      <!-- Product Description -->
      <div class="responsive">
        <div class="gallery">
          <div class="product-description">
            <!-- <img src= "{{ url_for('static', filename='cat2.jpeg',
width="12", height="25")}}" > -->
            <img src='https://123sfra.s3.jp-tok.cloud-object-
storage.appdomain.cloud/static/{{row["PIMAGE"]}}'', width="12", height="25" >
            <h2>{{row["PNAME"]}}</h2>
            <!-- Product Pricing -->
            <div class="product-price">
```

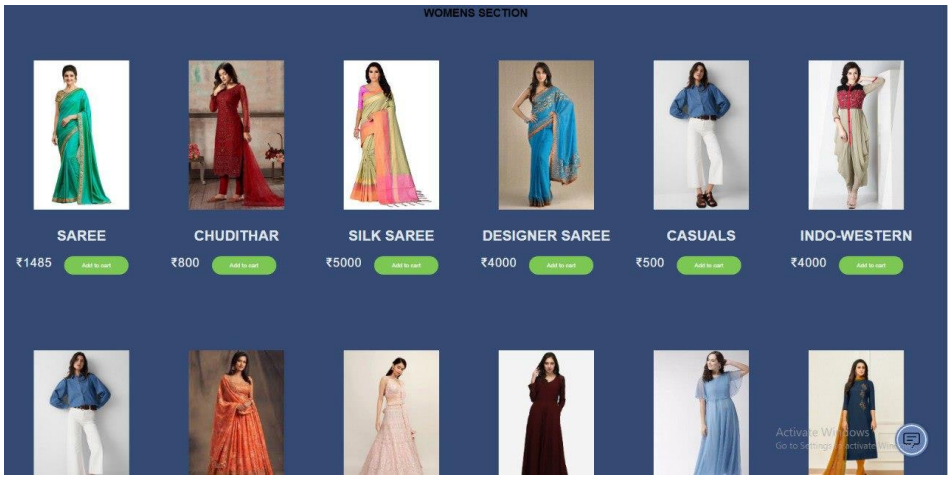
```
              <span>₹{{row["PPRICE"]}}</span>
              <a href="{{url_for('checkout')}}" class="cart-btn">Add to cart</a>
            </div><br><br>
          </div>
        </div>
      </div>
    </main>
    {%endfor%}

    <h3>WOMENS SECTION</h3></b>

    {%for row in WOMEN%}
    <main class="container">
      <!-- Product Description -->
      <div class="responsive">
        <div class="gallery">
          <div class="product-description">
            <!-- <img src= "{{ url_for('static', filename='cat2.jpeg',
width="12", height="25")}}" > -->
            <img src='https://123sfra.s3.jp-tok.cloud-object-
storage.appdomain.cloud/static/{{row["PIMAGE"]}}'', width="12", height="25" >
            <h2>{{row["PNAME"]}}</h2>
            <!-- Product Pricing -->
            <div class="product-price">
              <span>₹{{row["PPRICE"]}}</span>
              <a href="{{url_for('checkout')}}" class="cart-btn">Add to cart</a>
```

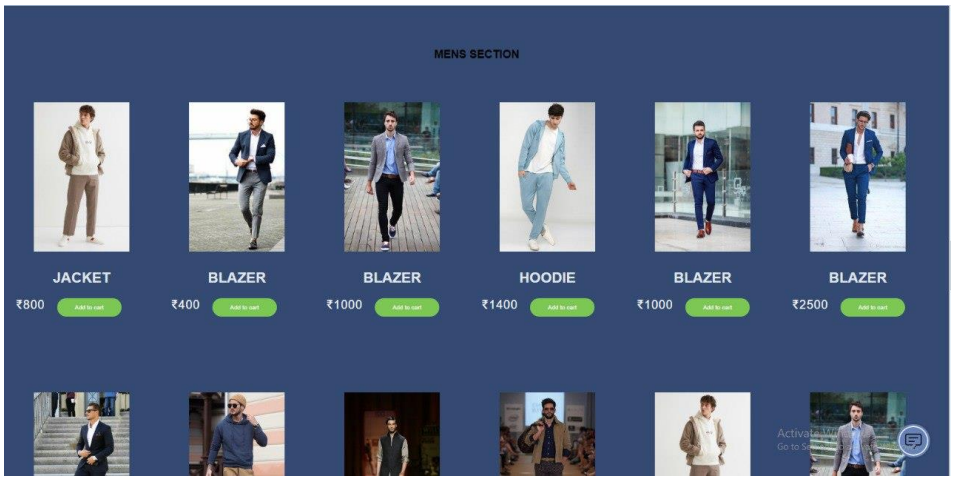
```
            </div><br><br>
          </div>
        </div>
      </div>
    </main>
    {%endfor%}


    <h3>KIDS SECTION</h3></b>


    {%for row in KIDS%}
    <main class="container">
      <!-- Product Description -->
      <div class="responsive">
        <div class="gallery">
          <div class="product-description">
            <!-- <img src= "{{ url_for('static', filename='cat2.jpeg',
width="12", height="25")}}" > -->
            <img src='https://123sfra.s3.jp-tok.cloud-object-
storage.appdomain.cloud/static/{{row["PIMAGE"]}}'', width="12", height="25" >
            <h2>{{row["PNAME"]}}</h2>
            <!-- Product Pricing -->
            <div class="product-price">
              <span>₹{{row["PPRICE"]}}</span>
              <a href="{{url_for('checkout')}}" class="cart-btn">Add to cart</a>
            </div><br><br>
          </div>
```

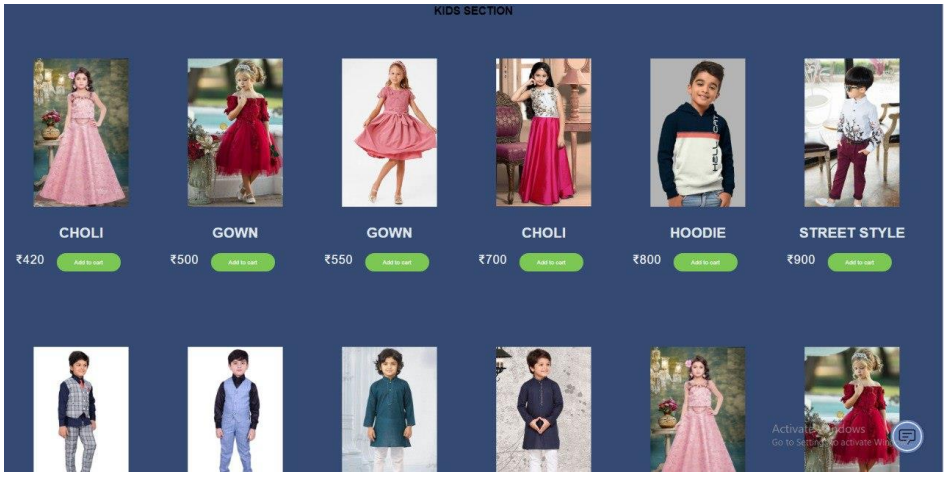
```
        </div>
      </div>
    </main>
    {%endfor%}
</body>
</html>
{%endblock%}
```



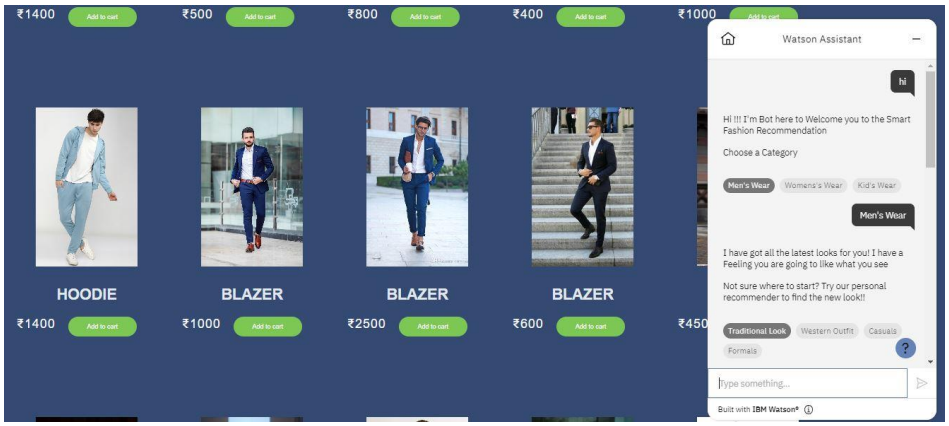


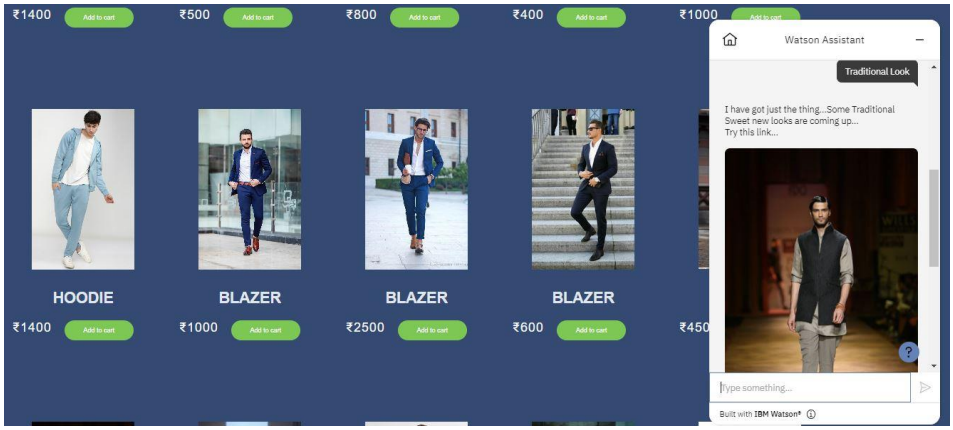


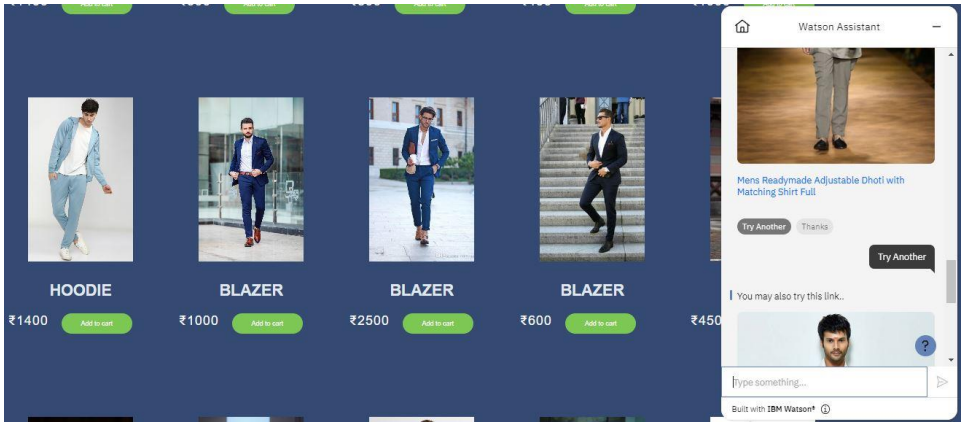


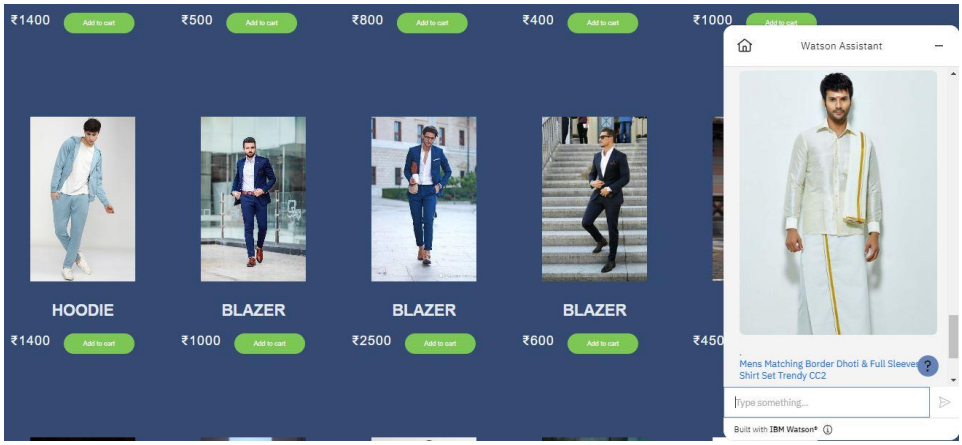


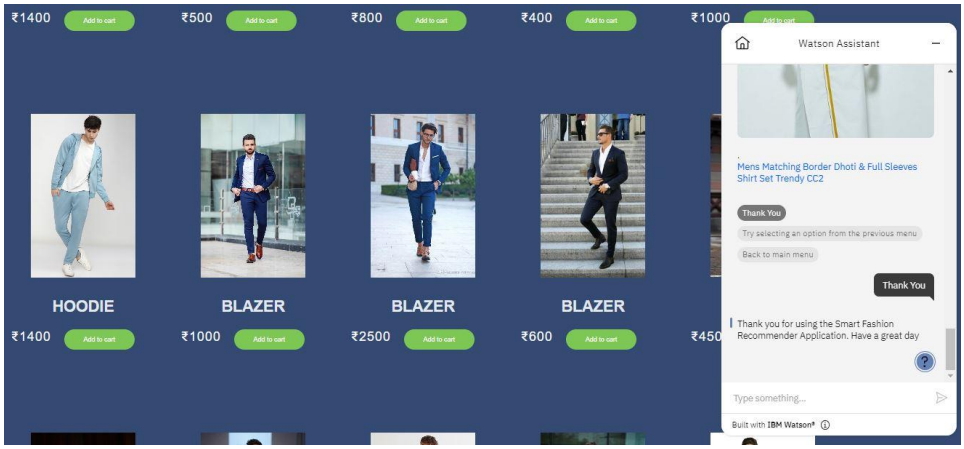
Watson Assistant

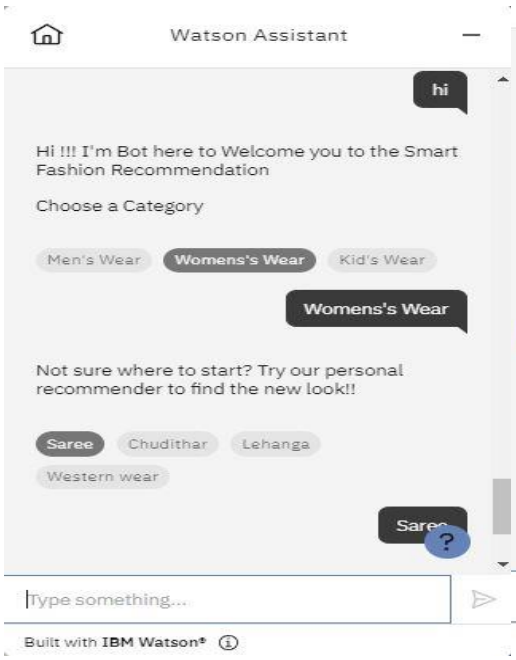

₹1400
₹500
₹800
₹400
₹1000
HOODIE
BLAZER
BLAZER
BLAZER
₹1400
₹1000
₹2500
₹600
Watson Assistant
hi
Hi !!! I'm Bot here to Welcome you to the Smart Fashion Recommendation
Choose a Category
Men's Wear
Womens's Wear
Kid's Wear
Men's Wear
I have got all the latest looks for you! I have a Feeling you are going to like what you see
Not sure where to start? Try our personal recommender to find the new look!!
Traditional Look
Western Outfit
Casuals
Formals
Type something...
Built with IBM Watson®

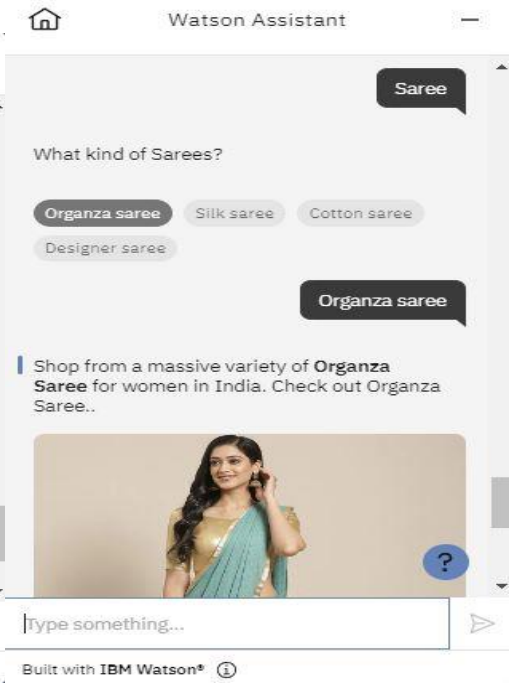

₹1400
₹500
₹800
₹400
₹1000
HOODIE
BLAZER
BLAZER
BLAZER
₹1400
₹1000
₹2500
₹600
Watson Assistant
Traditional Look
I have got just the thing...Some Traditional Sweet new looks are coming up...
Try this link...
Type something...
Built with IBM Watson®

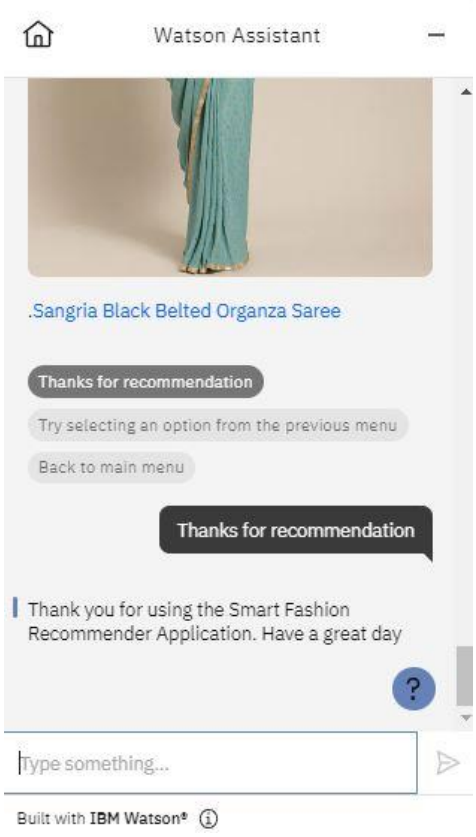

HOODIE
BLAZER
BLAZER
BLAZER
₹1400
₹1000
₹2500
₹600
Watson Assistant
Mens Readymade Adjustable Dhoti with Matching Shirt Full
Try Another
Thanks
Try Another
You may also try this link..
Type something...
Built with IBM Watson®

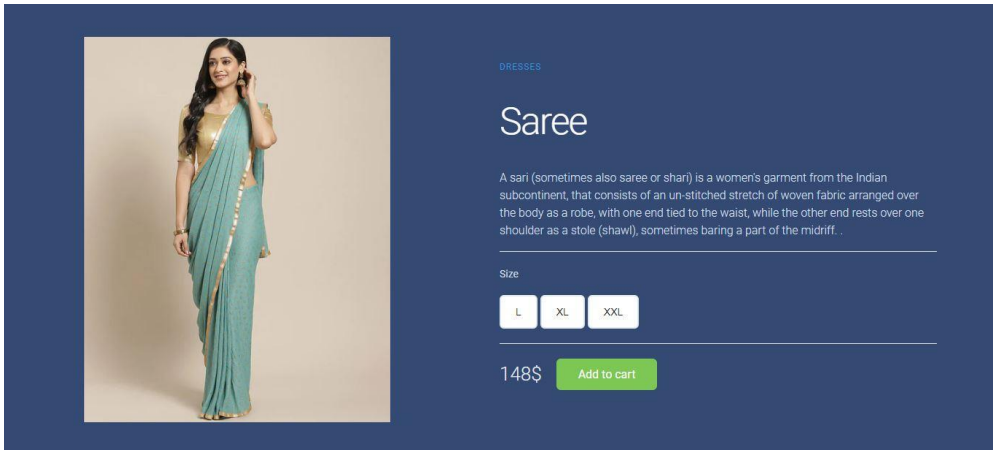

₹1400
₹500
₹800
₹400
₹1000
HOODIE
BLAZER
BLAZER
BLAZER
₹1400
₹1000
₹2500
₹600
Watson Assistant
Mens Matching Border Dhoti & Full Sleeves Shirt Set Trendy CC2
Type something...
Built with IBM Watson®


₹1400
₹500
₹800
₹400
₹1000
HOODIE
BLAZER
BLAZER
BLAZER
₹1400
₹1000
₹2500
₹600
Watson Assistant
Mens Matching Border Dhoti & Full Sleeves Shirt Set Trendy CC2
Thank You
Try selecting an option from the previous menu
Back to main menu
Thank You
Thank you for using the Smart Fashion Recommender Application. Have a great day
Type something...
Built with IBM Watson®


Watson Assistant
hi
Hi !!! I'm Bot here to Welcome you to the Smart Fashion Recommendation
Choose a Category
Men's Wear
Womens's Wear
Kid's Wear
Womens's Wear
Not sure where to start? Try our personal recommender to find the new look!!
Saree
Chudithar
Lehanga
Western wear
Saree
Type something...
Built with IBM Watson®


Watson Assistant
Saree
What kind of Sarees?
Organza saree
Silk saree
Cotton saree
Designer saree
Organza saree
Shop from a massive variety of Organza Saree for women in India. Check out Organza Saree..
Type something...
Built with IBM Watson®





Checkout.html

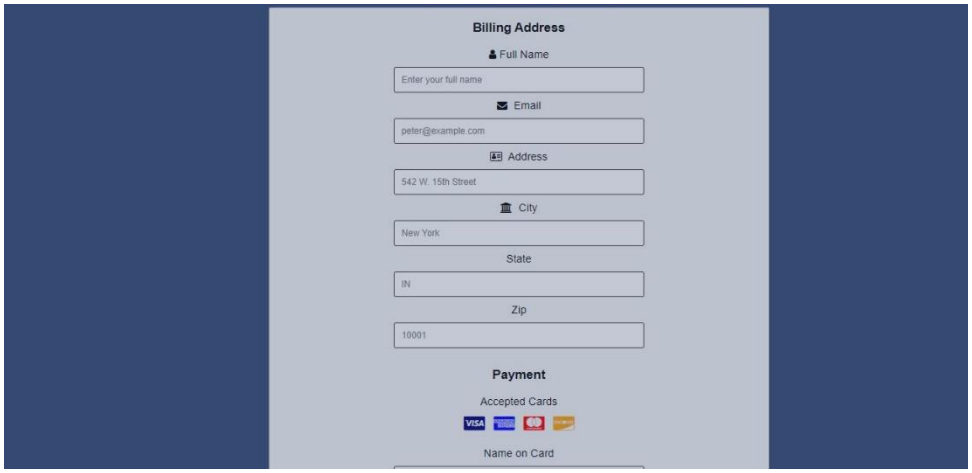
```
{%block head%}
<!DOCTYPE html>
<html lang="en">
<head>
```

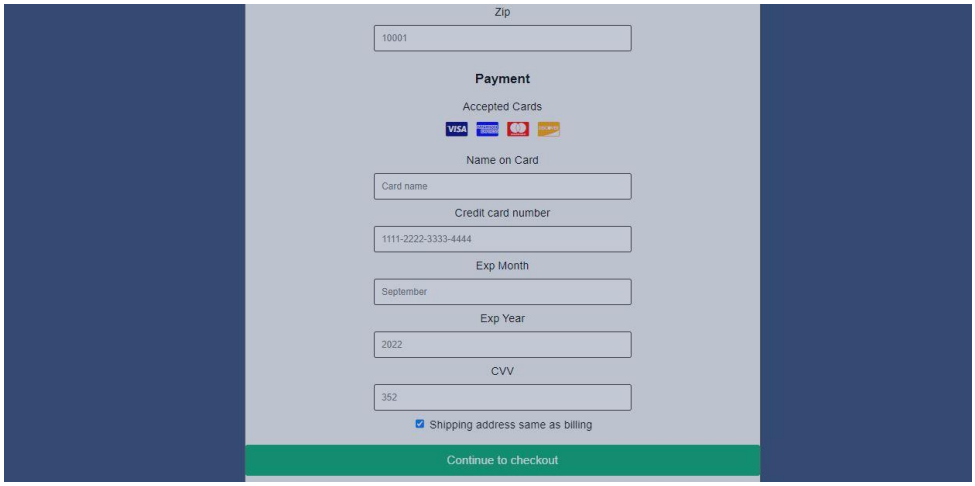
```
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
    <link href="{{ url_for('static', filename='checkoutstyle.css') }}" rel="stylesheet">

</head>
<body>



 <div class="chkrow">
  <div class="chkcol-75">
    <div class="chkcontainer">
      <form action="/action_page.php">

        <div class="chkrow">
          <div class="chkcol-50">
            <h3 style="text-align:center;">Billing Address</h3>
            <label for="fname"><i class="fa fa-user"></i> Full Name</label>
            <input type="text" id="fname" name="firstname" placeholder="Enter your full name">
            <label for="email"><i class="fa fa-envelope"></i>  Email</label>
```

```
            <input type="text" id="email" name="email"
placeholder="peter@example.com">
            <label for="adr"><i class="fa fa-address-card-o"></i> 
Address</label>
            <input type="text" id="adr" name="address" placeholder="542 W. 15th
Street">
            <label for="city"><i class="fa fa-institution"></i> 
City</label>
            <input type="text" id="city" name="city" placeholder="New York">
            <label for="state">State</label>
            <input type="text" id="state" name="state" placeholder="IN">
            <label for="zip">Zip</label>
            <input type="text" id="zip" name="zip" placeholder="10001">
          </div>
        </div>
        <div class="chkrow">
          <div class="chkcol-50">

            <h3>Payment</h3>
            <label for="fname">Accepted Cards</label>
            <div class="chkicon-container">
              <i class="fa fa-cc-visa" style="color:navy;"></i> 
              <i class="fa fa-cc-amex" style="color:blue;"></i> 
              <i class="fa fa-cc-mastercard" style="color:red;"></i> 
              <i class="fa fa-cc-discover" style="color:orange;"></i>
            </div>
```

```
            <label for="cname">Name on Card</label>
            <input type="text" id="cname" name="cardname" placeholder="Card
name">
            <label for="ccnum">Credit card number</label>
            <input type="text" id="ccnum" name="cardnumber" placeholder="1111-
2222-3333-4444">
            <label for="expmonth">Exp Month</label>
            <input type="text" id="expmonth" name="expmonth"
placeholder="September">
            <label for="expyear">Exp Year</label>
            <input type="text" id="expyear" name="expyear" placeholder="2022">
            <label for="cvv">CVV</label>
            <input type="text" id="cvv" name="cvv" placeholder="352">

          </div>



        </div>

        <label> <input type="checkbox" checked="checked" name="sameadr"> Shipping
address same as billing</label>

        <input type="submit" value="Continue to checkout" class="btn">
      </form>
    </div>
  </div>
```

```
 <!--</div>
</body>
</html>
{%endblock%}
```





App.py

```
from flask import Flask, render_template, session, url_for, redirect, flash,
request
import ibm_db;
con=ibm_db.connect("DATABASE=bludb;HOSTNAME=2d46b6b4-cbf6-40eb-bbce-
6251e6ba0300.bs2io90l08kqb1od8lcg.databases.appdomain.cloud;PORT=32328;SECURITY=S
SL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=bkj82229;PWD=KCwhio0Cb0XQmB5
H",'','')

app = Flask(__name__);

app.secret_key="Secret";

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

@app.route("/dashboard/", methods=['GET', 'POST'])
def dashboard():
    if session:
        products = []
        sql = "SELECT * FROM PRODUCTS"
        stmt = ibm_db.exec_immediate(con, sql)
        dictionary = ibm_db.fetch_both(stmt)
        while dictionary != False:
            # print ("The Name is : ",  dictionary)
            products.append(dictionary)
```

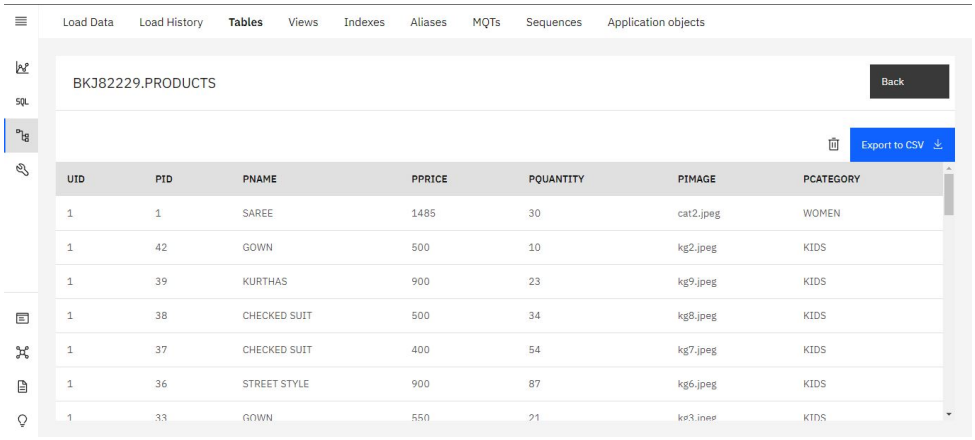
```
        dictionary = ibm_db.fetch_both(stmt)


    # test
    MEN = []
    sql = "SELECT * FROM products WHERE pcategory = 'MEN' "
    stmt = ibm_db.exec_immediate(con, sql)
    dictionary = ibm_db.fetch_both(stmt)
    while dictionary != False:
        # print ("The Name is : ",  dictionary)
        MEN.append(dictionary)
        dictionary = ibm_db.fetch_both(stmt)


    WOMEN = []
    sql = "SELECT * FROM products WHERE pcategory = 'WOMEN' "
    stmt = ibm_db.exec_immediate(con, sql)
    dictionary = ibm_db.fetch_both(stmt)
    while dictionary != False:
        # print ("The Name is : ",  dictionary)
        WOMEN.append(dictionary)
        dictionary = ibm_db.fetch_both(stmt)


    KIDS = []
    sql = "SELECT * FROM products WHERE pcategory = 'KIDS' "
    stmt = ibm_db.exec_immediate(con, sql)
    dictionary = ibm_db.fetch_both(stmt)
    while dictionary != False:
```

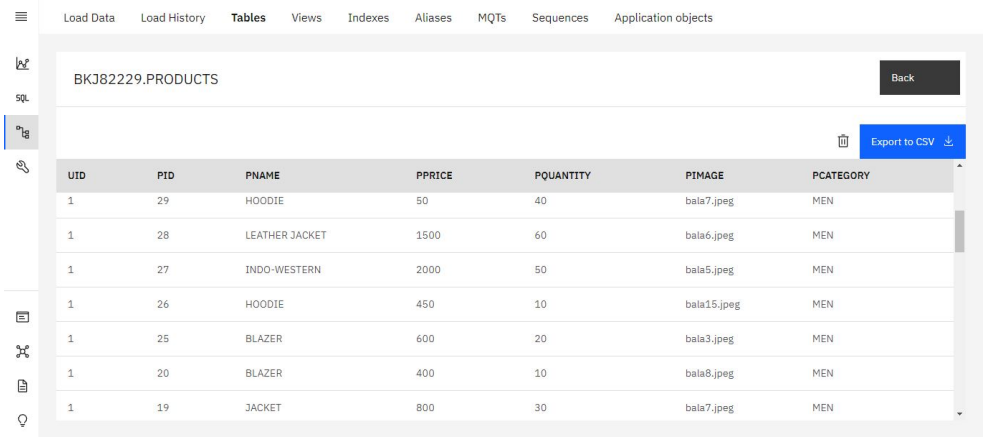
```
            # print ("The Name is : ",  dictionary)
            KIDS.append(dictionary)
            dictionary = ibm_db.fetch_both(stmt)

        if products:
            return render_template("dashboard.html", products = products,
MEN=MEN, WOMEN=WOMEN, KIDS=KIDS)
        return render_template("dashboard.html")

    flash("You're not logged in. Please login to enter into dashboard", "danger")
    return redirect(url_for('login'))

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

@app.route("/logout")
def logout():
    session.clear()
    return redirect(url_for('login'))

@app.route("/login/", methods=['GET','POST'])
def login():
    if request.method=="GET":
        return render_template("login.html")
```

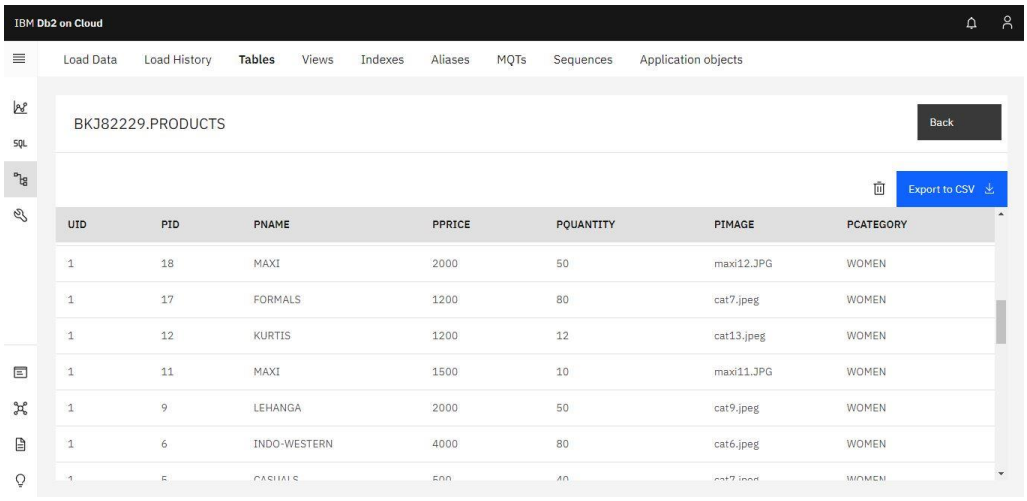
```
    elif request.method=="POST":
        User_mailid=request.form['mailid']
        User_pswd=request.form['pswd']

        sql="SELECT * FROM user_login WHERE User_mailid=? and User_pswd=?"
        stmt=ibm_db.prepare(con,sql)
        ibm_db.bind_param(stmt,1,User_mailid)
        ibm_db.bind_param(stmt,2,User_pswd)
        ibm_db.execute(stmt)
        data=ibm_db.fetch_assoc(stmt)

        if data:
            session["mailid"]= User_mailid
            return redirect(url_for("dashboard"))

        else:
            flash("E-mail & Password Mismatch","danger")

        return redirect(url_for("login"))


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

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

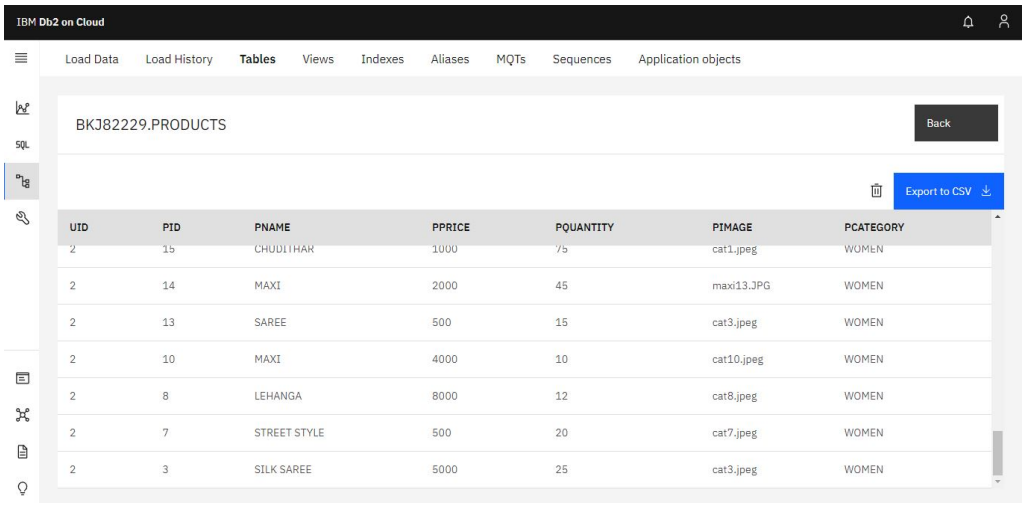
```
    if request.method == 'POST':
        # try:

            User_fname=request.form['User_fname'];
            User_lname=request.form['User_lname'];
            User_mailid=request.form['User_mailid'];
            User_pswd=request.form['User_pswd'];
            User_repswd=request.form['User_repswd'];
            User_phoneno=request.form['User_phoneno'];

            if(User_pswd==User_repswd):

                sql="SELECT * FROM user_login WHERE User_mailid=?"
                stmt=ibm_db.prepare(con,sql)
                ibm_db.bind_param(stmt,1,User_mailid)
                ibm_db.execute(stmt)
                account=ibm_db.fetch_assoc(stmt)

                if account:
                    flash("User already exists with the same email-id, Try
another one", "danger")
                    return redirect(url_for("registration"))


                stmt2="INSERT INTO user_login VALUES(?,?,?,?,?)"
```

```
            prep_stmt=ibm_db.prepare(con,stmt2)
            # ibm_db.bind_param(prep_stmt,1,'')
            ibm_db.bind_param(prep_stmt,1,User_fname)
            ibm_db.bind_param(prep_stmt,2,User_lname)
            ibm_db.bind_param(prep_stmt,3,User_mailid)
            ibm_db.bind_param(prep_stmt,4,User_pswd)
            ibm_db.bind_param(prep_stmt,5,User_phoneno)



            ibm_db.execute(prep_stmt)

            flash("Record added successfully", "success")
        else:
            flash("Password & Retype password mismatches","danger")
            return redirect(url_for("registration"))

    # except:
    #     flash("Error in Registration", "danger")


return redirect(url_for("login"))
```


Load Data
Load History
Tables
Views
Indexes
Aliases
MQTs
Sequences
Application objects
BKJ82229.PRODUCTS
Back
Export to CSV
UID PID PNAME PPRICE PQUANTITY PIMAGE PCATEGORY
1 1 SAREE 1485 30 cat2.jpeg WOMEN
1 42 GOWN 500 10 kg2.jpeg KIDS
1 39 KURTHAS 900 23 kg9.jpeg KIDS
1 38 CHECKED SUIT 500 34 kg8.jpeg KIDS
1 37 CHECKED SUIT 400 54 kg7.jpeg KIDS
1 36 STREET STYLE 900 87 kg6.jpeg KIDS


Load Data
Load History
Tables
Views
Indexes
Aliases
MQTs
Sequences
Application objects
BKJ82229.PRODUCTS
Back
Export to CSV
UID PID PNAME PPRICE PQUANTITY PIMAGE PCATEGORY
1 29 HOODIE 50 40 bala7.jpeg MEN
1 28 LEATHER JACKET 1500 60 bala6.jpeg MEN
1 27 INDO-WESTERN 2000 50 bala5.jpeg MEN
1 26 HOODIE 450 10 bala15.jpeg MEN
1 25 BLAZER 600 20 bala3.jpeg MEN
1 20 BLAZER 400 10 bala8.jpeg MEN
1 19 JACKET 800 30 bala7.jpeg MEN


IBM Db2 on Cloud
Load Data
Load History
Tables
Views
Indexes
Aliases
MQTs
Sequences
Application objects
BKJ82229.PRODUCTS
Back
Export to CSV
UID PID PNAME PPRICE PQUANTITY PIMAGE PCATEGORY
1 18 MAXI 2000 50 maxi12.JPG WOMEN
1 17 FORMALS 1200 80 cat7.jpeg WOMEN
1 12 KURTIS 1200 12 cat13.jpeg WOMEN
1 11 MAXI 1500 10 maxi11.JPG WOMEN
1 9 LEHANGA 2000 50 cat9.jpeg WOMEN
1 6 INDO-WESTERN 4000 80 cat6.jpeg WOMEN



GitHub & Project Demo Link

Github Link: https://github.com/IBM-EPBL/IBM-Project-38225-1660375306

Demo Link: https://youtu.be/O56yX6aId_w