Project Report Format

1. **INTRODUCTION**
    a. Project Overview
    b. Purpose
2. **LITERATURE SURVEY**
    a. Existing problem
    b. References
    c. Problem Statement Definition
3. **IDEATION & PROPOSED SOLUTION**
    a. Empathy Map Canvas
    b. Ideation & Brainstorming
    c. Proposed Solution
    d. Problem Solution fit
4. **REQUIREMENT ANALYSIS**
    a. Functional requirement
    b. Non-Functional requirements
5. **PROJECT DESIGN**
    a. Data Flow Diagrams
    b. Solution & Technical Architecture
    c. User Stories
6. **PROJECT PLANNING & SCHEDULING**
    a. Sprint Planning & Estimation
    b. Sprint Delivery Schedule
    c. Reports from JIRA
7. **CODING & SOLUTIONING (Explain the features added in the project along with code)**

1. Introduction

1.1 Project Overview:

User is buy the product online by chatbot instead of keyboard search.Keeyboard Search Is not all time recommends correct product.Chatbot is normally recoomends the product by user interest.The keyboard may not recoomend the product user interest.The chat also manage the order details in the project.It is very easy the user is to order without any worry about.The user is only focus on the product not all other things in the website.The user is login the webpage.After the dashboard page is shows the dress.In the side the chatbot is here.The chatbot is use the user order the product.The is user selected.The chatbot is sent the mail to user email.Chatbot is send the notification when the product is arrived in the user location.The admin is login the website then the admin dashboard is open.The admin dashboard is gives the user product.The admin can view the user details.The admin dashboard have the update stock.The admin can update the stock using to update the stock.The website use the external chatbot.the chatbot are IBM Watson Assistance.The Website store data at the cloud databse.the database are IBM DB2.It is sql based database.The Website is upload the project in the cloud.It the project is accessed using the IBM Object Storage.The Object storage is use bucket to store the project.The website use the container.The container is Docker.It is used to upload the project to the cloud.The user is click the website to manage the massive amount of user.

1.2 Purpose:

Users to buy product to chatbot.It is very easy the user is use the website.

User can manage the order by chatbot.User can display the product by the user interest.

User can find the product with less time.

2. LITERATURE SURVEY

2.1 Exsisting Problem:

Title	Year	Technology	Problem
Outfit Recommender System	2018	E-Commerce, Collaborative filtering,Cloud Computing EngCine,Python,html.	Grey-sheep problem refers to users with unique preferences and tastes that make it difficult to develop accurate profiles.
Clothing fashion Recommendation system	2020	Singular value Decomposition method,Azure ML Studio,Collaborative filtering.	Some offer up too many lowest common denominator recommendation artificially.
Image base fashion recommender system	2021	Cross domain recommendation system,Flask,DevOps,Html,Css	Some don’t support the long tail enough and just recommend obivious items,outliers can be a problem.
Modern Fashion recommender system	2022	AWS,Docker,Artificial Intelligence,python,google cloud computing engine.	Inaccurately estimate consumer’s true preference stand to pull down willingness to pay for some items and increase of the likehood of actual it.

2.2 References:

Mohamed Elleuch, Anis Mezghani, Mariem Khemakhem, Monji Kherallah “Clothing Classification using Deep CNN Architecture based on Transfer Learning” ,2021 DOI:10.1007/978-3-030-49336-3_24 [2] Saurabh Gupta, Siddartha Agarwal, Apoorve Dave. “Apparel Classifier and Recommender using Deep Learning.” (2015). [3] Bossard, Lukas, Matthias Dantone, Christian Leistner, Christian Wengert, Till Quack and Luc Van Gool. “Apparel Classification with Style.” ACCV (2012). [4] Krizhevsky, Alex, Ilya Sutskever and Geoffrey E. Hinton. “ImageNet classification with deep convolutional neural networks.” Communications of the ACM 60 (2012): 84 - 90. [5] Congying Guan, Shengfeng Qin, Yang Long, (2019) \"Apparel-based deep learning system design for apparel style recommendation\", International Journal of Clothing Science and Technology. [6] Stephen Marsland, ?Machine Learning – An Algorithmic Perspective?, Second Edition, Chapman and Hall/CRC Machine Learning and Pattern Recognition Series, 20

2.3 Problem Definition Statement

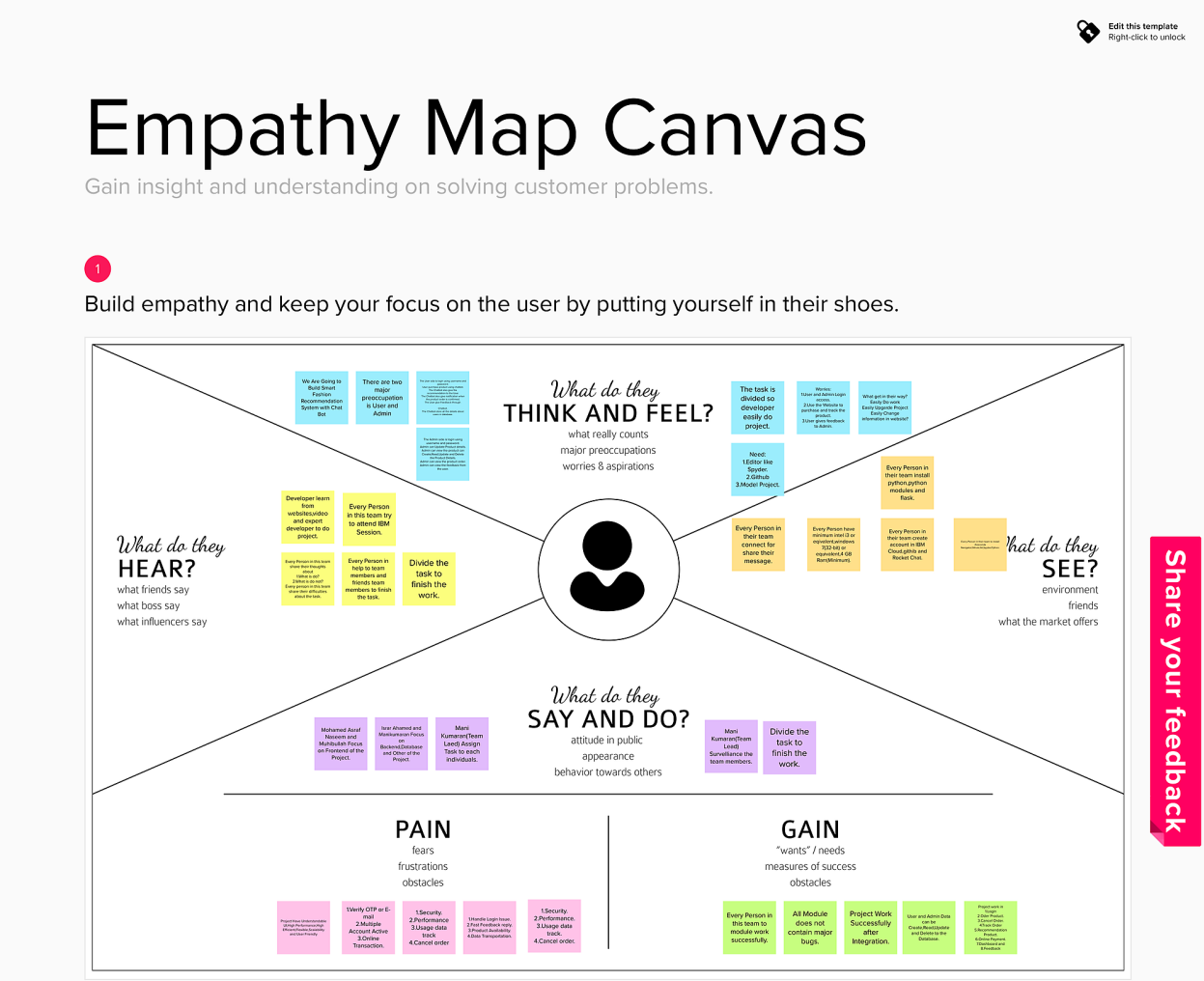
User is enter the wrong keyword to search keyboard it is recommend wrong product.

Users is give the option to the chatbot to recommend the correct product.

3. Ideation and Proposed Solution

3.1 Empathy map & Canvas

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.



Reference:

https://app.mural.co/invitation/mural/ibmproject0250/1663489514513?sender=u11a15f7b9d6bacf44a890331&key=9537ddbf-520c-44a0-8c57-37939aba8c63

3.2 Brainstorm & Ideation

Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem solving. Prioritizing volume over value, out-of-the-box ideas are welcome and built upon, and all participants are encouraged to

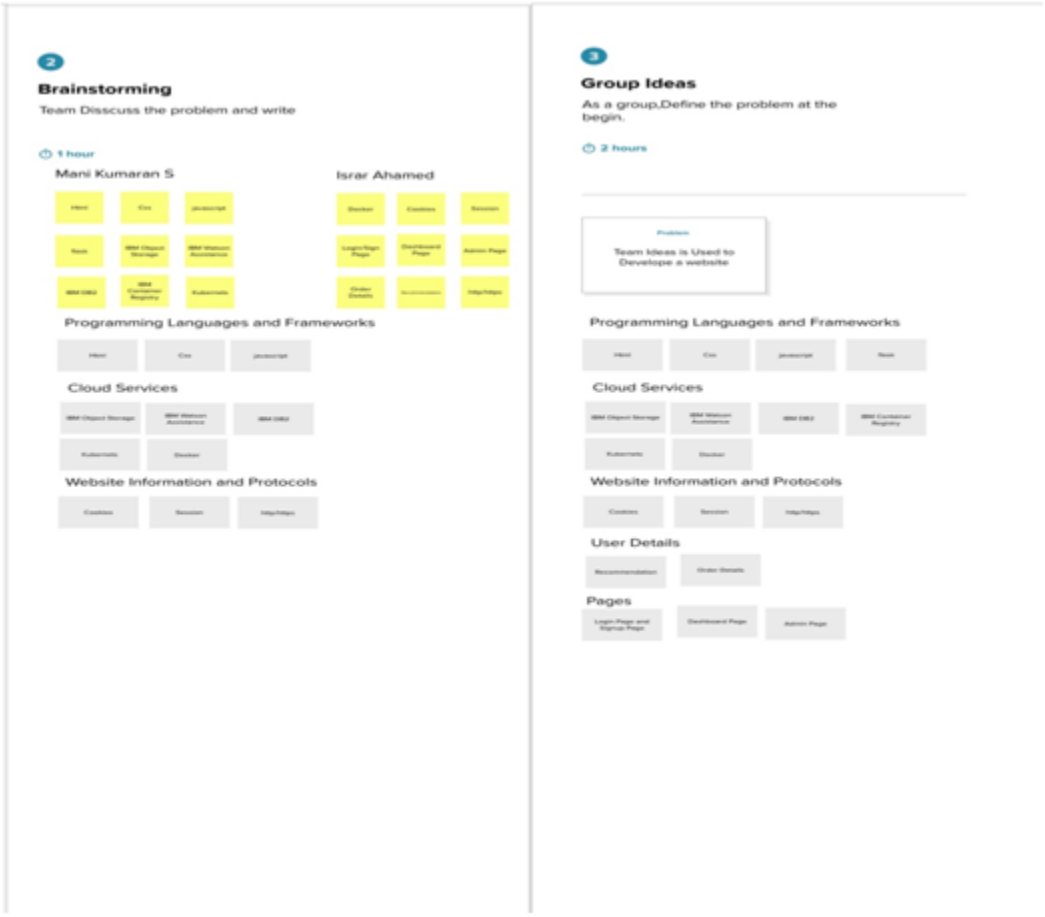
collaborate, helping each other develop a rich amount of creative solutions. Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room

Reference:

https://app.mural.co/t/ibmproject0250/m/ibmproject0250/1668146454106/db7d236756f32bba505a2712c7ba94299cc51e2e?sender=ud60e8640702a4e97caed3020

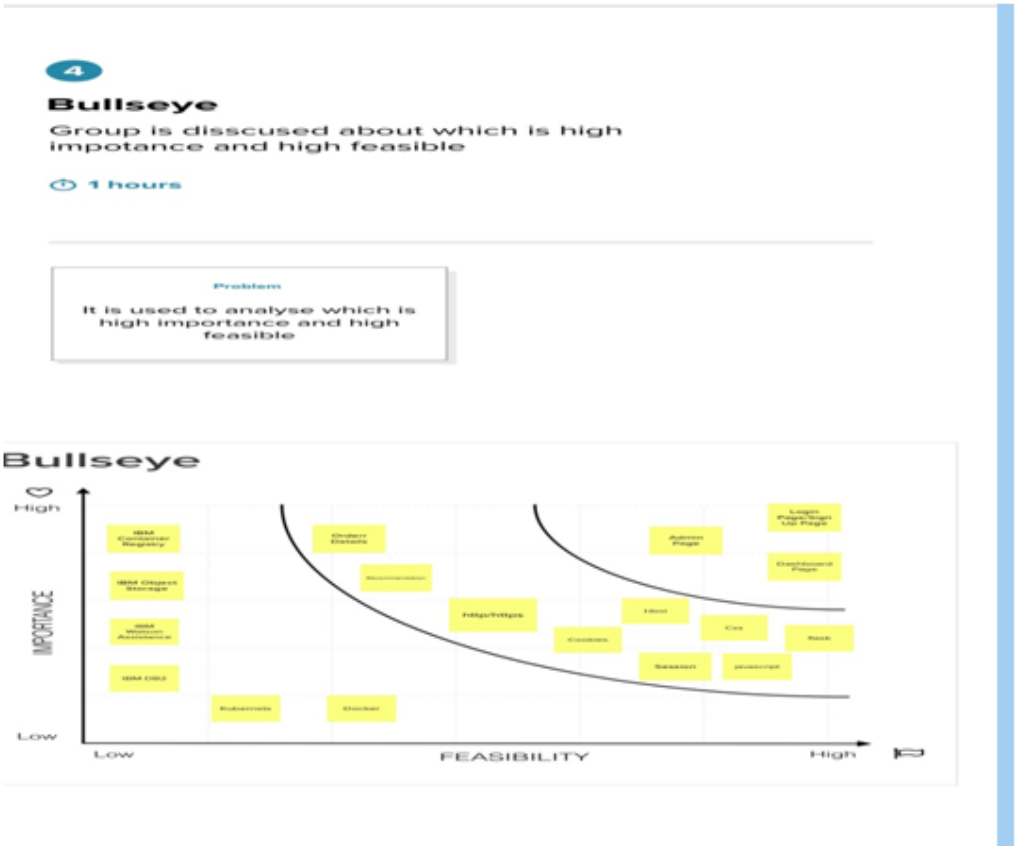
Step-1: Team Gathering, Collaboration and Select the Problem Statement

Step-2: Brainstorm, Idea Listing and Grouping



Step-3: Idea Prioritization

Step-3: Idea Prioritization



3.3 Proposed Solution

Project team shallfill the following information in proposedsolution template

S.No.	Parameter	Description
1.	Problem Statement (Problem to besolved)	Many of the website is use a keyboard search for searching the correct product.The customeris type the wrong word it would recommend wrong product.It is major problem most of the online purchasing website.

2.	Idea / Solution description	We have a chatbotit is choose theoption to display the product by the recommendation thecorrect product.
3.	Novelty / Uniqueness	It Provides the correct product in the onlinepurchasing website. Customer can find the product using therecommendation.
4.	Social Impact / Customer Satisfaction	Customer can easily to find the product usingchatbot.
5.	Business Model (Revenue Model)	It provide more sales because that gives thegood result.The website display ads and purchase get the commission.
6.	Scalability of the Solution	At starting it is website and after we develop toapplication for all platform.

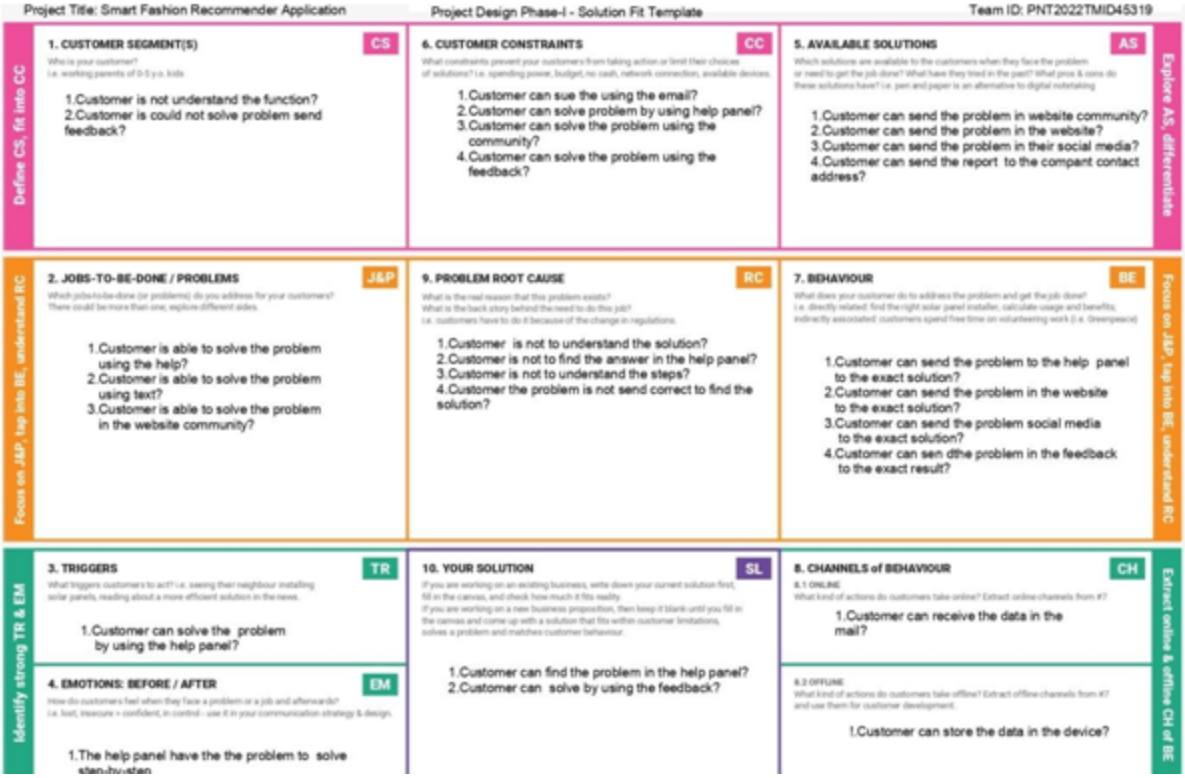
3.4 Problem Solution Fit

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



References:

1. https://gustdebacker.com/problem-solution-fit/#:~:text=What%20is%20a,the%20customer%E2%80%99s%20problem.
2. https://www.feedough.com/problem-solution-fit/#:~:text=Why%20Achieving%20A,guessing%20their%20needs.

4.Requirement Analysis

4.1 Functional Requirements

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration throughForm
FR-2	User Interaction	Interact through the Chat Bot
FR-3	Buying Products	Through the chatBot Recommendation
FR-4	Track Products	Ask the Chat Bot to Track my Orders
FR-5	Return Products	Through the chat Bot
FR_6	New Collections	Recommended from chatBot

4.2 Non Functional Requirements

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

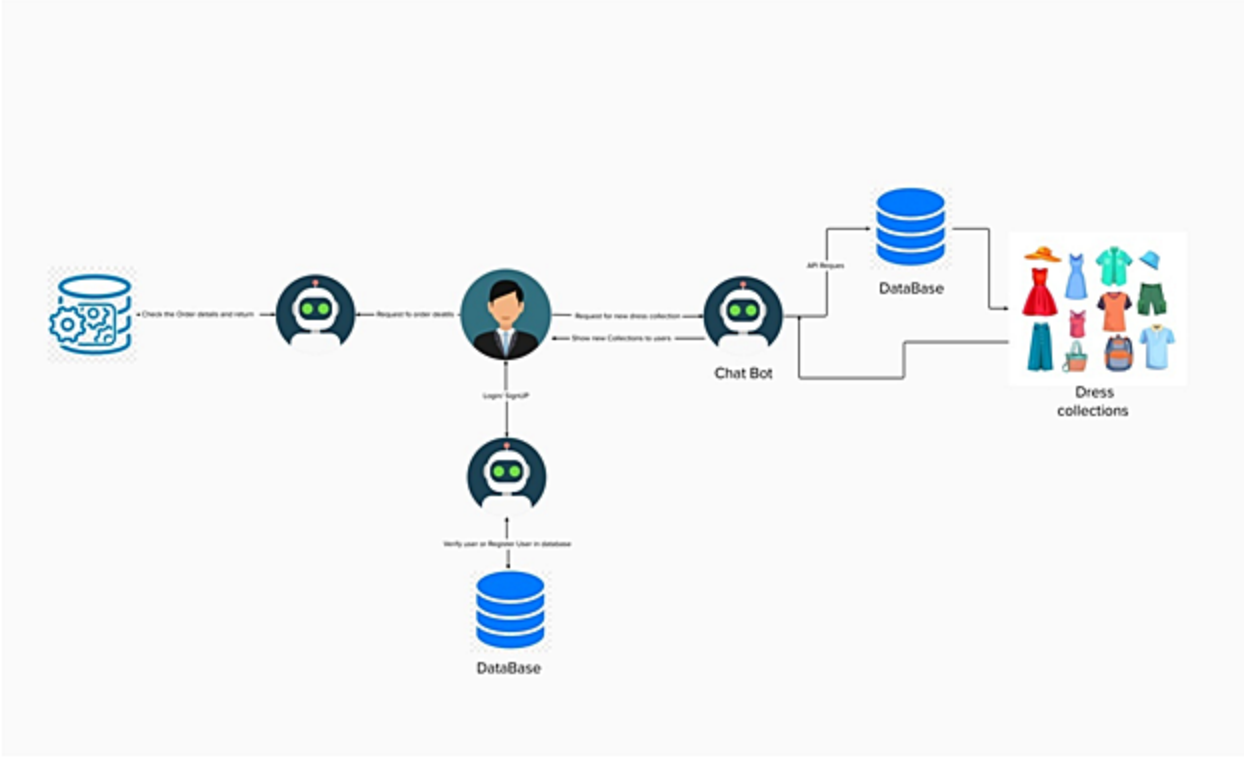
FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	Using Android or IOS or windows applications.
NFR-2	**Security**	The user datais stored securely in IBM cloud.
NFR-3	**Reliability**	The Quality of the services are trusted.
NFR-4	**Performance**	Its Provide smooth user experience.
NFR-5	**Availability**	The services are available for 24/7.
NFR-6	**Scalability**	Its easy to scalable size of usersand products.

5.Project Design

5.1 Data Flow Diagram

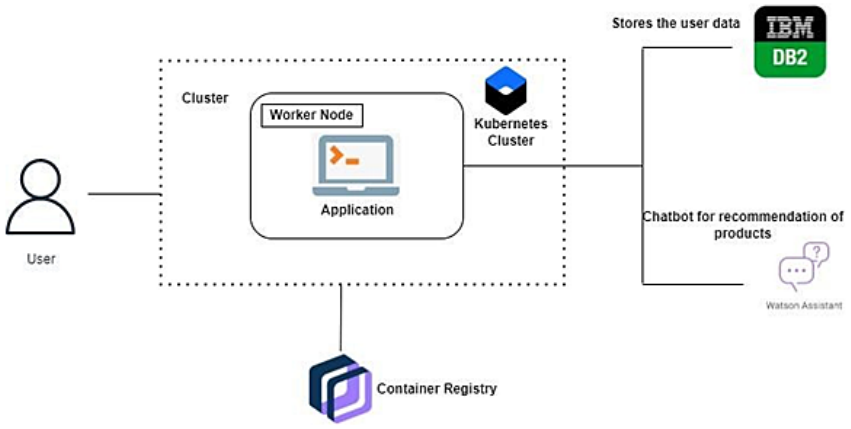
Data Flow Diagrams:

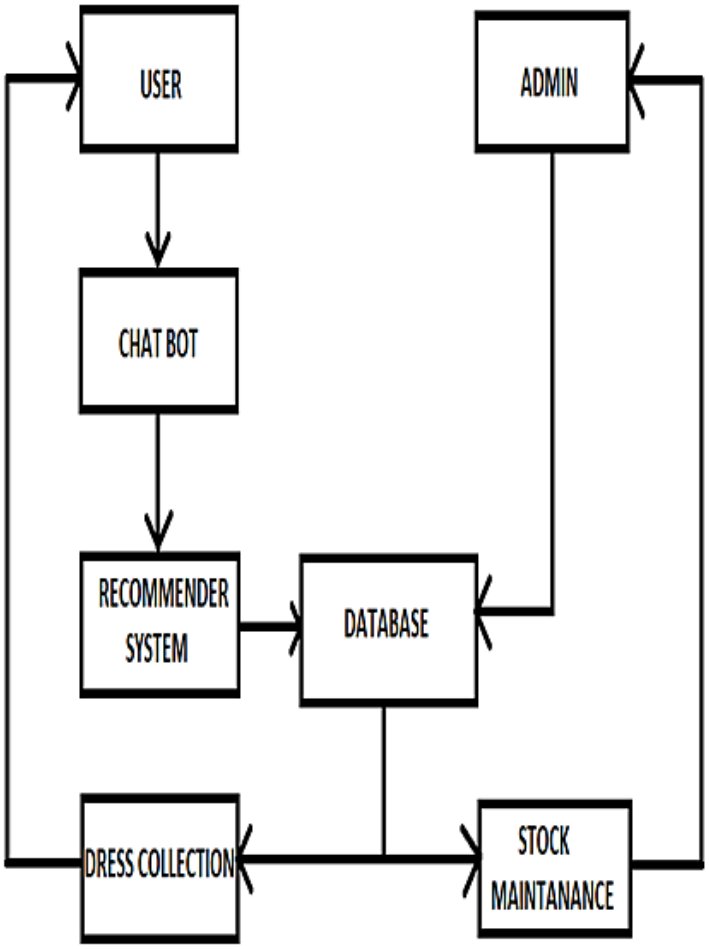
A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the rightamount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.



5.2 Solution & Technical Architecture

Solution Architecture





5.3 User Stories

Spirint	Functional Requirement(Epic)	User Story Number	User Story/Task	Story Points	Priority	Team Members
Spirint-1	User Panel	USN-1	The User will login into the website and go through the products available on the website	20	High	1.MANIKUMARAN S 2.ISRAR AHAMED M 3.MOHAMED ASRAF NASEEM S 4.MUHIBULLA M

Spirint-2	Admin Panel	USN-2	The role of the admin is to check out the database about the stock and have a track of all the things that the users are purchasing.	20	High	1.MANIKUMARAN S 2.ISRAR AHAMED M 3.MOHAMED ASRAF NASEEM S 4.MUHIBULLA M
Spirint-3	Chat Bot	USN-3	The User can directly talk to Chatbot regarding the products.Get the recommendations based on information provided by the user.	20	High	1.MANIKUMARAN S 2.ISRAR AHAMED M 3.MOAHMED ASRAF NASEEM S 4.MUHIBULLA M
Spirint-4	Final delivery	USN-4	Container of applications using docker kubernets and deployment the application.Create thedocumentation and final submit the application	20	High	1.MANIKUMARAN S 2.ISRAR AHAMED M 3.MOHAMED ASRAF NASEEM S 4.MUHIBULLA M

6.Project Planning & Scheduling

6.1 Spirint Planning & Estimation

Spirint	Functional Requirement(Epic)	User Story Number	User Story/Task	Story Points	Priority	Team Members
Spirint-1	User Panel	USN-1	The User will login into the website and go through the products available on the website	20	High	1.MANIKUMARAN S 2.ISRAR AHAMED M 3.MOHAMED ASRAF NASEEM S 4.MUHIBULLA M
Spirint-2	Admin Panel	USN-2	The role of the admin is to check out the database about the stock and have a track of all the things that the users are purchasing.	20	High	1.MANIKUMARAN S 2.ISRAR AHAMED M 3.MOHAMED ASRAF NASEEM S 4.MUHIBULLA M
Spirint-3	Chat Bot	USN-3	The User can directly talk to Chatbot regarding the products.Get the recommendations based on information provided by the user.	20	High	1.MANIKUMARAN S 2.ISRAR AHAMED M 3.MOAHMED ASRAF NASEEM S 4.MUHIBULLA M

Spirint-4	Final delivery	USN-4	Container of applications using docker kubernets and deployment the application.Create thedocumentation and final submit the application	20	High	1.MANIKUMARAN S 2.ISRAR AHAMED M 3.MOHAMED ASRAF NASEEM S 4.MUHIBULLA M

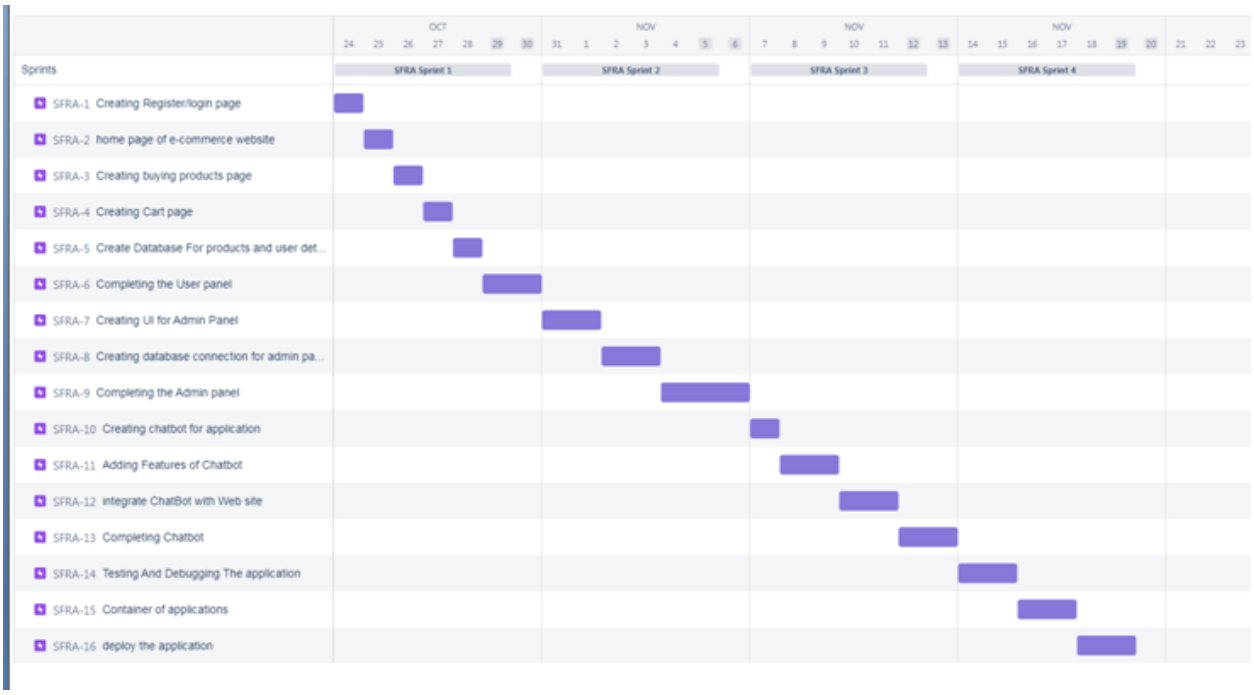
6.2 Sprint Delivery & Schedule

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

Sprint	Total StoryPoints	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		29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022		05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022		12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022		19 Nov 2022

6.3 Report Jira Files

Burndown Chart:



7.Coding & Solution

7.1 Login

Customer is login using this module

login.html

```
<!DOCTYPE html>

<html lang="en">

  <head>

    <meta content="text/html;charset=utf-8" http-equiv="Content-Type">

    <meta content="utf-8" http-equiv="encoding">

    <meta charset="utf-8">

    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-
fit=no">

    <meta name="theme-color" content="#000000">

    <link rel="shortcut icon" href="%PUBLIC_URL%/favicon.ico">

    <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
integrity="sha384-
```

```
Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"

            crossorigin="anonymous">

        <link href="{{ url_for('static',filename='css/custom.css') }}" rel="stylesheet"
type="text/css" />

        <script defer src="https://use.fontawesome.com/releases/v5.0.6/js/all.js"></script>

        <script src="https://code.jquery.com/jquery-1.11.0.min.js"></script>



        <title>Legendry Fashion - Log In</title>

    </head>

    <body>

    <header>

        <nav class="navbar fixed-top navbar-dark bg-dark navbar-expand-sm box-shadow">

          <a href="/" class="navbar-brand d-flex align-items-center">

            <strong><i class="fa  fa-cart-plus"></i>Online Clothing Store</strong>

          </a>

        </nav>
```

```
        </header><br />
        <main>
          <div class="container">
            <div class="row">
              <div class="col-sm">
                <h2>Log In to Buy</h2>
                <p>{{ msg }}</p>
                <div>
                  <form action="/logged/" class="form" method="post">
                    <div>
                      <input type="text" name="username" autofocus
placeholder="Username">
                      <input type="password" name="password" placeholder="Password">
                      <button type="submit" class="btn btn-primary">Login</button>
                    </div>
                  </form>
```

```
            </div>
          </div>
        </div>
      </div>
    </main>
    </body>
</html>
```

7.2 Signup

Users is signup in the signup Module

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta content="text/html;charset=utf-8" http-equiv="Content-Type">
        <meta content="utf-8" http-equiv="encoding">
        <meta charset="utf-8">
        <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-
```

```
fit=no">

        <meta name="theme-color" content="#000000">

        <link rel="shortcut icon" href="%PUBLIC_URL%/favicon.ico">

        <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
integrity="sha384-
Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"

            crossorigin="anonymous">

        <link href="{{ url_for('static',filename='css/custom.css') }}" rel="stylesheet"
type="text/css" />

        <script defer src="https://use.fontawesome.com/releases/v5.0.6/js/all.js"></script>

        <script src="https://code.jquery.com/jquery-1.11.0.min.js"></script>



        <title>Trendy Clothing Store - Register</title>

    </head>

    <body>

        <header>
```

```
        <nav class="navbar fixed-top navbar-dark bg-dark navbar-expand-sm box-shadow">
          <a href="/" class="navbar-brand d-flex align-items-center">
            <strong><i class="fa fa-shopping-bag"></i> Sample Clothing Store</strong>
          </a>
        </nav>
      </header><br />
      <main>
        <div class="container">
          <div class="row">
            <div class="col-sm">
              <h2>Register</h2>
              <p>{{msg}}</p>
              <form action="/register/" class="form" method="post">
                <input type="text" name="username" id="username"
placeholder="Username" autofocus required > <span id="user-msg" class="alert alert-
danger"></span><br /><br />
```

```
                    <input type="password" name="password" id="password"
placeholder="Password" required > <span id="password-msg" class="alert alert-
danger"></span><br /><br />

                    <input type="password" name="confirm" id="confirm"
placeholder="Confirm Password" required> <span id="confirm-msg" class="alert alert-
danger"></span><br /><br />

                    <input type="text" name="fname" id="fname" placeholder="First Name"
required> <span id="fname-msg" class="alert alert-danger"></span><br /><br />

                    <input type="text" name="lname" id="lname" placeholder="Last Name"
required> <span id="lname-msg" class="alert alert-danger"></span><br /><br />

                    <input type="email" name="email" id="email" placeholder="Email"
required> <span id="email-msg" class="alert alert-danger"></span><br /><br /><br />

                    <button type="reset" class="btn btn-secondary">Clear</button>

                    <button type="submit" id="submit" class="btn btn-
primary">Register</button>

                </form>

            </div>

        </div>
```

```
        </div>
      </main>
  <!-- Custom JS Scripts -->
      <script src="{{ url_for('static',filename='js/validate.js') }}"></script>
  </body>
</html>
```

7.3 Mainpage

Customer order the product on the mainpage

index.html

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



{% block title %}
Legendry Fashion- Home
{% endblock %}
```

```
{% block body %}

<!-- Main Store Body -->

  {% if session['user'] %}

    <div class="alert alert-warning alert-dismissible fade show" role="alert">

      <button type="button" class="close" data-dismiss="alert" aria-label="Close">

        <span aria-hidden="true">&times;</span>

      </button>

        <strong>Welcome, {{ session['user'] }}</strong> Hope you have a pleasant experience shopping with us.

    </div>



  {% endif %}

    <div class="row" id="shirtCard">

    {% for i in range(shirtsLen) %}

      <div class="col-sm">

        <div class="card text-center">
```

```
                    <div class="card-body">
                      <form action="/buy/" methods="POST">
                        <h5 class="card-title">{{shirts[i]["typeClothes"].capitalize()}}</h5>
                        <img src="/static/img/{{shirts[i]["image"]}}" class="shirt" alt="" />
                        <h5 class="card-text">{{shirts[i]["samplename"]}}</h5>
                        {% if shirts[i]["onSale"] %}
                          <img src="/static/img/sale-icon.png" width="26px" />
                          <h4 class="card-text price" style="color:red; display:inline">{{
'{:,.2f}'.format(shirts[i]["onSalePrice"]) }}</h4>
                        {% else %}
                          <h4 class="card-text price">{{ '{:,.2f}'.format(shirts[i]["price"]) }}</h4>
                        {% endif %}
                        <div class="stepper-input">
                          <span class="decrement target">-</span>
                          <input class="quantity" name="quantity" value='0' />
                          <span class="increment target">+</span>
```

```
                    </div>

                    <input type="hidden" name="id" value="{{shirts[i]["id"]}}" />

                    {% if not session %}

                    <input type="hidden" name="loggedin" value="0" />

                    {% else %}

                    <input type="hidden" name="loggedin" value="1" />



                    {% endif %}

                    <input type="submit" class="btn btn-primary addToCart" value="Add To Cart"
/><br /><br />

                    <div class="alert alert-danger flashMessage" style="text-align: center;
display:none; font-size:0.9em;"></div>

                  </form>

                </div>

              </div>

            </div>
```

```
        {% endfor %}

        </div>

      </div>

    </main>

  {% endblock %}
```

admin.html

```
<html>

<head>

    <meta content="text/html;charset=utf-8" http-equiv="Content-Type">

        <meta content="utf-8" http-equiv="encoding">

        <meta charset="utf-8">

        <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-
fit=no">

        <meta name="theme-color" content="#000000">

        <link rel="shortcut icon" href="%PUBLIC_URL%/favicon.ico">

        <link rel="stylesheet"
```

```
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
integrity="sha384-
Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"
        crossorigin="anonymous">
    <link href="{{ url_for('static',filename='css/custom.css') }}" rel="stylesheet"
type="text/css" />
    <script defer src="https://use.fontawesome.com/releases/v5.0.6/js/all.js"></script>
    <script src="https://code.jquery.com/jquery-1.11.0.min.js"></script>

  <title>Admin Dashboard</title>
  <style>
  .model{
  text-align:center;
  }
  .container-color{
        background-color:#102ABD;
```

```
        }
    </style>
    </head>
    <body>
      <header>
        <nav class="navbar fixed-top navbar-dark container-color navbar-expand-sm box-
shadow">
          <a href="/" class="navbar-brand d-flex align-items-center">
            <strong><i class="fa fa-shopping-bag"></i> Sample Clothing Store</strong>
          </a>
        </nav>
      </header><br />
    <p class="model">Admin Dashboard</p>
    <pre>Admin Page</pre>
    <script>
    function website()
```

```
{

i=1;

window.open("/admin",target="_self");

}

</script>

<button type="button" onClick="website()">Open</button>

{orders}

</body>

</html>
```

application.py

```
from cs50 import SQL

from flask_session import Session

from flask import Flask, render_template, redirect, request, session, jsonify

from datetime import datetime



# # Instantiate Flask object named app
```

```
app = Flask(__name__)



# # Configure sessions

app.config["SESSION_PERMANENT"] = False

app.config["SESSION_TYPE"] = "filesystem"

Session(app)



# Creates a connection to the database

db = SQL ( "sqlite:///data.db" )

uid=[]



@app.route("/")

def index():

    shirts = db.execute("SELECT * FROM shirts ORDER BY onSalePrice")

    shirtsLen = len(shirts)

    # Initialize variables
```

```
    shoppingCart = []

    shopLen = len(shoppingCart)

    totItems, total, display = 0, 0, 0

    if 'user' in session:

        shoppingCart = db.execute("SELECT samplename, image, SUM(qty), SUM(subTotal),
price, id FROM cart GROUP BY samplename")

        shopLen = len(shoppingCart)

        for i in range(shopLen):

            total += shoppingCart[i]["SUM(subTotal)"]

            totItems += shoppingCart[i]["SUM(qty)"]

        shirts = db.execute("SELECT * FROM shirts ORDER BY onSalePrice ASC")

        shirtsLen = len(shirts)

        return render_template ("index.html", shoppingCart=shoppingCart, shirts=shirts,
shopLen=shopLen, shirtsLen=shirtsLen, total=total, totItems=totItems, display=display,
session=session )

    return render_template ( "index.html", shirts=shirts, shoppingCart=shoppingCart,
shirtsLen=shirtsLen, shopLen=shopLen, total=total, totItems=totItems, display=display)
```

```
@app.route("/buy/")
def buy():
    # Initialize shopping cart variables
    shoppingCart = []
    shopLen = len(shoppingCart)
    totItems, total, display = 0, 0, 0
    qty = int(request.args.get('quantity'))
    if session:
        # Store id of the selected shirt
        id = int(request.args.get('id'))
        # Select info of selected shirt from database
        goods = db.execute("SELECT * FROM shirts WHERE id = :id", id=id)
        # Extract values from selected shirt record
        # Check if shirt is on sale to determine price
```

```
            if(goods[0]["onSale"] == 1):
                price = goods[0]["onSalePrice"]
            else:
                price = goods[0]["price"]
            samplename = goods[0]["samplename"]
            image = goods[0]["image"]
            subTotal = qty * price
            # Insert selected shirt into shopping cart
            db.execute("INSERT INTO cart (id, qty, samplename, image, price, subTotal) VALUES (:id,
:qty, :samplename, :image, :price, :subTotal)", id=id, qty=qty, samplename=samplename,
image=image, price=price, subTotal=subTotal)
            shoppingCart = db.execute("SELECT samplename, image, SUM(qty), SUM(subTotal),
price, id FROM cart GROUP BY samplename")
            shopLen = len(shoppingCart)
            # Rebuild shopping cart
            for i in range(shopLen):
```

```
            total += shoppingCart[i]["SUM(subTotal)"]

            totItems += shoppingCart[i]["SUM(qty)"]

        # Select all shirts for home page view

        shirts = db.execute("SELECT * FROM shirts ORDER BY samplename ASC")

        shirtsLen = len(shirts)

        # Go back to home page

        return render_template ("index.html", shoppingCart=shoppingCart, shirts=shirts,
shopLen=shopLen, shirtsLen=shirtsLen, total=total, totItems=totItems, display=display,
session=session )




@app.route("/update/")

def update():

    # Initialize shopping cart variables

    shoppingCart = []

    shopLen = len(shoppingCart)
```

```
totItems, total, display = 0, 0, 0

qty = int(request.args.get('quantity'))

if session:

    # Store id of the selected shirt

    id = int(request.args.get('id'))

    db.execute("DELETE FROM cart WHERE id = :id", id=id)

    # Select info of selected shirt from database

    goods = db.execute("SELECT * FROM shirts WHERE id = :id", id=id)

    # Extract values from selected shirt record

    # Check if shirt is on sale to determine price

    if(goods[0]["onSale"] == 1):

        price = goods[0]["onSalePrice"]

    else:

        price = goods[0]["price"]

    samplename = goods[0]["samplename"]

    image = goods[0]["image"]
```

```
        subTotal = qty * price

        # Insert selected shirt into shopping cart

        db.execute("INSERT INTO cart (id, qty, samplename, image, price, subTotal) VALUES (:id,
:qty, :samplename, :image, :price, :subTotal)", id=id, qty=qty, samplename=samplename,
image=image, price=price, subTotal=subTotal)

        shoppingCart = db.execute("SELECT samplename, image, SUM(qty), SUM(subTotal),
price, id FROM cart GROUP BY samplename")

        shopLen = len(shoppingCart)

        # Rebuild shopping cart

        for i in range(shopLen):

            total += shoppingCart[i]["SUM(subTotal)"]

            totItems += shoppingCart[i]["SUM(qty)"]

        # Go back to cart page

        return render_template ("cart.html", shoppingCart=shoppingCart, shopLen=shopLen,
total=total, totItems=totItems, display=display, session=session )
```

```
@app.route("/filter/")
def filter():
    if request.args.get('typeClothes'):
        query = request.args.get('typeClothes')
        shirts = db.execute("SELECT * FROM shirts WHERE typeClothes = :query ORDER BY samplename ASC", query=query )
    if request.args.get('sale'):
        query = request.args.get('sale')
        shirts = db.execute("SELECT * FROM shirts WHERE onSale = :query ORDER BY samplename ASC", query=query)
    if request.args.get('id'):
        query = int(request.args.get('id'))
        shirts = db.execute("SELECT * FROM shirts WHERE id = :query ORDER BY samplename ASC", query=query)
    if request.args.get('kind'):
        query = request.args.get('kind')
        shirts = db.execute("SELECT * FROM shirts WHERE kind = :query ORDER BY samplename
```

```
ASC", query=query)

        if request.args.get('price'):

            query = request.args.get('price')

            shirts = db.execute("SELECT * FROM shirts ORDER BY onSalePrice ASC")

        shirtsLen = len(shirts)

        # Initialize shopping cart variables

        shoppingCart = []

        shopLen = len(shoppingCart)

        totItems, total, display = 0, 0, 0

        if 'user' in session:

            # Rebuild shopping cart

            shoppingCart = db.execute("SELECT samplename, image, SUM(qty), SUM(subTotal),
price, id FROM cart GROUP BY samplename")

            shopLen = len(shoppingCart)

            for i in range(shopLen):

                total += shoppingCart[i]["SUM(subTotal)"]
```

```
            totItems += shoppingCart[i]["SUM(qty)"]

        # Render filtered view

        return render_template ("index.html", shoppingCart=shoppingCart, shirts=shirts,
shopLen=shopLen, shirtsLen=shirtsLen, total=total, totItems=totItems, display=display,
session=session )

    # Render filtered view

    return render_template ( "index.html", shirts=shirts, shoppingCart=shoppingCart,
shirtsLen=shirtsLen, shopLen=shopLen, total=total, totItems=totItems, display=display)




@app.route("/checkout/")

def checkout():

    order = db.execute("SELECT * from cart")

    # Update purchase history of current customer

    for item in order:

        db.execute("INSERT INTO purchases (uid, id, samplename, image, quantity)
VALUES(:uid, :id, :samplename, :image, :quantity)", uid=session["uid"], id=item["id"],
```

```
samplename=item["samplename"], image=item["image"], quantity=item["qty"] )

        # Clear shopping cart

        db.execute("DELETE from cart")

        shoppingCart = []

        shopLen = len(shoppingCart)

        totItems, total, display = 0, 0, 0

        # Redirect to home page

        return redirect('/')




    @app.route("/remove/", methods=["GET"])

    def remove():

        # Get the id of shirt selected to be removed

        out = int(request.args.get("id"))

        # Remove shirt from shopping cart

        db.execute("DELETE from cart WHERE id=:id", id=out)
```

```
        # Initialize shopping cart variables

        totItems, total, display = 0, 0, 0

        # Rebuild shopping cart

        shoppingCart = db.execute("SELECT samplename, image, SUM(qty), SUM(subTotal), price,
id FROM cart GROUP BY samplename")

        shopLen = len(shoppingCart)

        for i in range(shopLen):

            total += shoppingCart[i]["SUM(subTotal)"]

            totItems += shoppingCart[i]["SUM(qty)"]

        # Turn on "remove success" flag

        display = 1

        # Render shopping cart

        return render_template ("cart.html", shoppingCart=shoppingCart, shopLen=shopLen,
total=total, totItems=totItems, display=display, session=session )
```

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

def login():

    return render_template("login.html")




@app.route("/new/", methods=["GET"])

def new():

    # Render log in page

    return render_template("new.html")




@app.route("/logged/", methods=["POST"] )

def logged():

    # Get log in info from log in form

    user = request.form["username"].lower()

    pwd = request.form["password"]
```

```
#pwd = str(sha1(request.form["password"].encode('utf-8')).hexdigest())

# Make sure form input is not blank and re-render log in page if blank

if user == "" or pwd == "":

    return render_template ( "login.html" )

# Find out if info in form matches a record in user database

if user=="admin" and pwd=="management":

    return render_template("admin.html",order="table")

query = "SELECT * FROM users WHERE username = :user AND password = :pwd"

rows = db.execute ( query, user=user, pwd=pwd )



# If username and password match a record in database, set session variables

if len(rows) == 1:

    session['user'] = user

    session['time'] = datetime.now( )

    session['uid'] = rows[0]["id"]

# Redirect to Home Page
```

```
    if 'user' in session:
        return redirect ( "/" )
    # If username is not in the database return the log in page
    return render_template ( "login.html", msg="Wrong username or password." )


@app.route("/history/")
def history():
    # Initialize shopping cart variables
    shoppingCart = []
    shopLen = len(shoppingCart)
    totItems, total, display = 0, 0, 0
    # Retrieve all shirts ever bought by current user
    uid.append(session["uid"])
    myShirts = db.execute("SELECT * FROM purchases WHERE uid=:uid", uid=session["uid"])
    myShirtsLen = len(myShirts)
```

```
        # Render table with shopping history of current user

        return render_template("history.html", shoppingCart=shoppingCart, shopLen=shopLen,
total=total, totItems=totItems, display=display, session=session, myShirts=myShirts,
myShirtsLen=myShirtsLen)




    @app.route("/logout/")

    def logout():

        # clear shopping cart

        db.execute("DELETE from cart")

        # Forget any user_id

        session.clear()

        # Redirect user to login form

        return redirect("/")
```

```
@app.route("/register/", methods=["POST"] )
def registration():
    # Get info from form
    username = request.form["username"]
    password = request.form["password"]
    confirm = request.form["confirm"]
    fname = request.form["fname"]
    lname = request.form["lname"]
    email = request.form["email"]
    # See if username already in the database
    rows = db.execute( "SELECT * FROM users WHERE username = :username ", username = username )
    # If username already exists, alert user
    if len( rows ) > 0:
        return render_template ( "new.html", msg="Username already exists!" )
    # If new user, upload his/her info into the users database
```

```
        new = db.execute ( "INSERT INTO users (username, password, fname, lname, email)
VALUES (:username, :password, :fname, :lname, :email)",
                    username=username, password=password, fname=fname, lname=lname,
email=email )
        # Render login template
        return render_template ( "login.html" )



    @app.route("/cart/")
    def cart():
        if 'user' in session:
            # Clear shopping cart variables
            totItems, total, display = 0, 0, 0
            # Grab info currently in database
            shoppingCart = db.execute("SELECT samplename, image, SUM(qty), SUM(subTotal),
price, id FROM cart GROUP BY samplename")
```

```
            # Get variable values
            shopLen = len(shoppingCart)
            for i in range(shopLen):
                total += shoppingCart[i]["SUM(subTotal)"]
                totItems += shoppingCart[i]["SUM(qty)"]
        # Render shopping cart
        return render_template("cart.html", shoppingCart=shoppingCart, shopLen=shopLen,
total=total, totItems=totItems, display=display, session=session)


    @app.route("/admin")
    def page():
        # Initialize shopping cart variables
        shoppingCart = []
        shopLen = len(shoppingCart)
        totItems, total, display = 0, 0, 0
        # Retrieve all shirts ever bought by current user
```

```
    myShirts = db.execute("SELECT * FROM purchases WHERE uid=:uid",uid=uid[0])

    myShirtsLen = len(myShirts)

    # Render table with shopping history of current user

    return render_template("history.html", shoppingCart=shoppingCart, shopLen=shopLen,
total=total, totItems=totItems, display=display, session=session, myShirts=myShirts,
myShirtsLen=myShirtsLen)
```

custom.css

```
.card:hover {

    border-color: #999;

    box-shadow: 1px 2px #999;

}



.card {

    margin-bottom: 1em;

    background-color: palegoldenrod;

}
```

```
.price {
    color: seagreen;
    font-weight: bold;
}

.price:before {
    content: '$';
}

.shirt {
    margin-bottom: 10px;
    width: 200px;
    height: 200px;
}
```

```
.stepper-input{
    display: flex;
    display: -webkit-flex;
    color: #222;
    max-width: 120px;
    margin: 10px auto;
    text-align: center;
}

header {
    margin-bottom: 50px;
}

.shirtCart {
    width: 25px;
}
```

```
.add {
    text-transform: uppercase;
    font-size: 0.8em;
    font-weight: bold;
    color: white;
}

.checkout {
    text-transform: uppercase;
    font-size: 0.8em;
    font-weight: bold;
}

.add:hover {
    background-color: deepskyblue;
```

```
    border-color: deepskyblue;
}

tr {
    text-align: center;
}

.modal-header {
    border-bottom: 0px;
}

.counter {
    font-size: 0.6em;
    margin-left: 1em;
    font-weight: bold;
}
```

```
.increment,

.decrement{

        height: 24px;

        width: 24px;

        border: 1px solid #222;

        text-align: center;

        box-sizing: border-box;

        border-radius: 50%;

        text-decoration: none;

        color: #222;

        font-size: 24px;

        line-height: 22px;

        display: inline-block;

        cursor: pointer;
```

```
    }



.decrement:hover,

.increment:hover {

    color: green;

}



.decrement:active,

.increment:active {

    background-color: green;

    color: white;

}



.quantity{

        height: 24px;
```

```
        width: 48px;
        text-align: center;
        margin: 0 12px;
        border-radius: 2px;
        border: 1px solid #222;

    }

    body {
        margin: 0;
        font-family: -apple-system,BlinkMacSystemFont,"Segoe UI",Roboto,"Helvetica
Neue",Arial,sans-serif,"Apple Color Emoji","Segoe UI Emoji","Segoe UI Symbol";
        font-size: 1rem;
        font-weight: 400;
        line-height: 1.5;
        color: #212529;
```

```
    text-align: left;
    background-color: beige;
}

.bg-dark {
    background-color: saddlebrown!important;
}
```

myscript.js

```
$(".target").on("click", function() {
    let $button = $(this);
    let oldVal = parseInt($button.parent().find("input").val());
    let newVal = 0;

    if ($button.text() == '+') {
        newVal = oldVal + 1;
    }
```

```
    else {
        if (oldVal > 0) {
            newVal = oldVal - 1;
        }
        else {
            newVal = 0;
        }
    }

    $button.parent().find("input").val(newVal);
});
```

```
$('.addToCart').on("click", function(event) {
    console.log('hello');
    if($(this).prev().prev().prev().find("input").val() == '0') {
        event.preventDefault();
        $(this).next().next().next().html("You need to select at least one clothing.");
        $(this).next().next().next().css("display", "block");
        $(this).next().next().next().delay(3000).slideUp();
    }

    if ($(this).prev().val() == "0") {
            event.preventDefault();
            $(this).next().next().next().html("You need to log in to buy.");
            $(this).next().next().next().css("display", "block");
            $(this).next().next().next().delay(3000).slideUp();
        }
```

```
});




$(".flashMessage").delay(3000).slideUp();
```

validate.js

```
// The submit button

const SUBMIT = $( "#submit" );



// Each of the fields and error message divs

const USERNAME = $( "#username" );

const USERNAME_MSG = $( "#user-msg" );



const PASSWORD = $( "#password" );

const PASSWORD_MSG = $( "#password-msg" );



const CONFIRM = $( "#confirm" );
```

```
const CONFIRM_MSG = $( "#confirm-msg" );



const FNAME = $( "#fname" );

const FNAME_MSG = $( "#fname-msg" );



const LNAME = $( "#lname" );

const LNAME_MSG = $( "#lname-msg" );



const EMAIL = $( "#email" );

const EMAIL_MSG = $( "#email-msg" );



/**

 * Resets the error message fields and makes the submit

 * button visible.

 */

function reset_form ( )
```

```
{
    USERNAME_MSG.html( "" );
    USERNAME_MSG.hide();
    PASSWORD_MSG.html( "" );
    PASSWORD_MSG.hide();
    CONFIRM_MSG.html( "" );
    CONFIRM_MSG.hide();
    LNAME_MSG.html( "" );
    LNAME_MSG.hide();
    FNAME_MSG.html( "" );
    FNAME_MSG.hide();
    EMAIL_MSG.html( "" );
    EMAIL_MSG.hide();
    SUBMIT.show();
}
```

```
/**
 * Validates the information in the register form so that
 * the server is not required to check this information.
 */
function validate ( )
{
    let valid = true;
    reset_form ( );



    // This currently checks to see if the username is
    // present and if it is at least 5 characters in length.
    if ( !USERNAME.val() || USERNAME.val().length < 5  )
    {
        // Show an invalid input message
        USERNAME_MSG.html( "Username must be 5 characters or more" );
```

```
    USERNAME_MSG.show();

    // Indicate the type of bad input in the console.

    console.log( "Bad username" );

    // Indicate that the form is invalid.

    valid = false;

}

// TODO: Add your additional checks here.




if ( USERNAME.val() != USERNAME.val().toLowerCase())

{

    USERNAME_MSG.html("Username must be all lowercase");

    USERNAME_MSG.show();

    valid = false;

}
```

```
if ( !PASSWORD.val() || PASSWORD.val().length < 8 )

{

    PASSWORD_MSG.html("Password needs to be at least 8 characters long");

    PASSWORD_MSG.show();

    valid = false;

}



if ( !CONFIRM.val() || PASSWORD.val() != CONFIRM.val() )

{

    CONFIRM_MSG.html("Passwords don't match");

    CONFIRM_MSG.show();

    valid = false;

}



if ( !FNAME.val() )

{
```

```
    FNAME_MSG.html("First name must not be empty");
    FNAME_MSG.show();
    valid = false;
}

if ( !LNAME.val() )
{
    LNAME_MSG.html("Last name must not be empty");
    LNAME_MSG.show();
    valid = false;
}

var x = EMAIL.val().trim();
var atpos = x.indexOf("@");
var dotpos = x.lastIndexOf(".");
if ( atpos < 1 || dotpos < atpos + 2 || dotpos + 2 >= x.length ) {
```

```
        EMAIL_MSG.html("You need to enter a valid email address");
        EMAIL_MSG.show();
        valid = false;
    }

    // If the form is valid, reset error messages
    if ( valid )
    {
        reset_form ( );
    }
}

// Bind the validate function to the required events.
$(document).ready ( validate );
USERNAME.change ( validate );
PASSWORD.change ( validate );
```

CONFIRM.change (validate);

LNAME.change (validate);

FNAME.change (validate);

EMAIL.change (validate);

7.4 IBM DB2

Customer data is stored in the IBM Cloud

{

“network”:

{

“host”:host,

“port”:website port

}

“db”

{

“method”:”direct”,

“username”:username,

“password”:password

}

}

8.Testing

8.1 Test Cases

A test case has components that describe input, action, and an expected response, in order to determine if a feature of an application works correctly.

8.2 User Acceptance Testing

User Acceptance Testing (UAT), also known as beta or end-user testing, is defined as testing the software by the user or client to determine whether it can be accepted or not. This is the final testing performed once the functional, system and regression testing are completed.

9.Results

9.1 Performance & Metrics

Performance Metrics track and measure how well employees are performing in their jobs. HRs, Managers, and leaders use tools and their own methods to measure productivity and efficiency against set parameters. These parameters can vary from employee to employee and also from one department to another. Employee performance metrics benefit both the organization and the employee by aligning them towards a single direction and company goals.

10.Advantage & Disadvantage

Advantage:

East to user order

Personal interest is easily get a user

To view the order easily

To send notification to user

Disadvantage:

Less recommendation

User may not satisfied

User is select only using button

User should know how to use

11.Conclusion

Chat is used to user easily order than the keyword search

12.Future Scope

We will devlope app for all devices

13.Appendix

Source Code

```
from cs50 import SQL

from flask_session import Session

from flask import Flask, render_template, redirect, request, session, jsonify

from datetime import datetime
```

```
# # Instantiate Flask object named app

app = Flask(__name__)



# # Configure sessions

app.config["SESSION_PERMANENT"] = False

app.config["SESSION_TYPE"] = "filesystem"

Session(app)



# Creates a connection to the database

db = SQL ( "sqlite:///data.db" )




@app.route("/")

def index():

    shirts = db.execute("SELECT * FROM shirts ORDER BY onSalePrice")

    shirtsLen = len(shirts)
```

```
    # Initialize variables
    shoppingCart = []
    shopLen = len(shoppingCart)
    totItems, total, display = 0, 0, 0
    if 'user' in session:
        shoppingCart = db.execute("SELECT samplename, image, SUM(qty),
SUM(subTotal), price, id FROM cart GROUP BY samplename")
        shopLen = len(shoppingCart)
        for i in range(shopLen):
            total += shoppingCart[i]["SUM(subTotal)"]
            totItems += shoppingCart[i]["SUM(qty)"]
        shirts = db.execute("SELECT * FROM shirts ORDER BY onSalePrice ASC")
        shirtsLen = len(shirts)
        return render_template ("index.html", shoppingCart=shoppingCart,
shirts=shirts, shopLen=shopLen, shirtsLen=shirtsLen, total=total, totItems=totItems,
display=display, session=session )
    return render_template ( "index.html", shirts=shirts, shoppingCart=shoppingCart,
```

```
shirtsLen=shirtsLen, shopLen=shopLen, total=total, totItems=totItems,
display=display)



@app.route("/buy/")
def buy():
    # Initialize shopping cart variables
    shoppingCart = []
    shopLen = len(shoppingCart)
    totItems, total, display = 0, 0, 0
    qty = int(request.args.get('quantity'))
    if session:
        # Store id of the selected shirt
        id = int(request.args.get('id'))
        # Select info of selected shirt from database
        goods = db.execute("SELECT * FROM shirts WHERE id = :id", id=id)
```

```
        # Extract values from selected shirt record

        # Check if shirt is on sale to determine price

        if(goods[0]["onSale"] == 1):

            price = goods[0]["onSalePrice"]

        else:

            price = goods[0]["price"]

        samplename = goods[0]["samplename"]

        image = goods[0]["image"]

        subTotal = qty * price

        # Insert selected shirt into shopping cart

        db.execute("INSERT INTO cart (id, qty, samplename, image, price, subTotal)
VALUES (:id, :qty, :samplename, :image, :price, :subTotal)", id=id, qty=qty,
samplename=samplename, image=image, price=price, subTotal=subTotal)

        shoppingCart = db.execute("SELECT samplename, image, SUM(qty),
SUM(subTotal), price, id FROM cart GROUP BY samplename")

        shopLen = len(shoppingCart)
```

```
        # Rebuild shopping cart
        for i in range(shopLen):
            total += shoppingCart[i]["SUM(subTotal)"]
            totItems += shoppingCart[i]["SUM(qty)"]
        # Select all shirts for home page view
        shirts = db.execute("SELECT * FROM shirts ORDER BY samplename ASC")
        shirtsLen = len(shirts)
        # Go back to home page
        return render_template ("index.html", shoppingCart=shoppingCart,
shirts=shirts, shopLen=shopLen, shirtsLen=shirtsLen, total=total, totItems=totItems,
display=display, session=session )



@app.route("/update/")
def update():
    # Initialize shopping cart variables
```

```
shoppingCart = []

shopLen = len(shoppingCart)

totItems, total, display = 0, 0, 0

qty = int(request.args.get('quantity'))

if session:

    # Store id of the selected shirt

    id = int(request.args.get('id'))

    db.execute("DELETE FROM cart WHERE id = :id", id=id)

    # Select info of selected shirt from database

    goods = db.execute("SELECT * FROM shirts WHERE id = :id", id=id)

    # Extract values from selected shirt record

    # Check if shirt is on sale to determine price

    if(goods[0]["onSale"] == 1):

        price = goods[0]["onSalePrice"]

    else:

        price = goods[0]["price"]
```

```
        samplename = goods[0]["samplename"]

        image = goods[0]["image"]

        subTotal = qty * price

        # Insert selected shirt into shopping cart

        db.execute("INSERT INTO cart (id, qty, samplename, image, price, subTotal)
VALUES (:id, :qty, :samplename, :image, :price, :subTotal)", id=id, qty=qty,
samplename=samplename, image=image, price=price, subTotal=subTotal)

        shoppingCart = db.execute("SELECT samplename, image, SUM(qty),
SUM(subTotal), price, id FROM cart GROUP BY samplename")

        shopLen = len(shoppingCart)

        # Rebuild shopping cart

        for i in range(shopLen):

            total += shoppingCart[i]["SUM(subTotal)"]

            totItems += shoppingCart[i]["SUM(qty)"]

        # Go back to cart page

        return render_template ("cart.html", shoppingCart=shoppingCart,
shopLen=shopLen, total=total, totItems=totItems, display=display, session=session )
```

```
@app.route("/filter/")
def filter():
    if request.args.get('typeClothes'):
        query = request.args.get('typeClothes')
        shirts = db.execute("SELECT * FROM shirts WHERE typeClothes = :query
ORDER BY samplename ASC", query=query )
    if request.args.get('sale'):
        query = request.args.get('sale')
        shirts = db.execute("SELECT * FROM shirts WHERE onSale = :query ORDER
BY samplename ASC", query=query)
    if request.args.get('id'):
        query = int(request.args.get('id'))
        shirts = db.execute("SELECT * FROM shirts WHERE id = :query ORDER BY
samplename ASC", query=query)
```

```
    if request.args.get('kind'):
        query = request.args.get('kind')
        shirts = db.execute("SELECT * FROM shirts WHERE kind = :query ORDER BY samplename ASC", query=query)
    if request.args.get('price'):
        query = request.args.get('price')
        shirts = db.execute("SELECT * FROM shirts ORDER BY onSalePrice ASC")
    shirtsLen = len(shirts)
    # Initialize shopping cart variables
    shoppingCart = []
    shopLen = len(shoppingCart)
    totItems, total, display = 0, 0, 0
    if 'user' in session:
        # Rebuild shopping cart
        shoppingCart = db.execute("SELECT samplename, image, SUM(qty), SUM(subTotal), price, id FROM cart GROUP BY samplename")
```

```
        shopLen = len(shoppingCart)
        for i in range(shopLen):
            total += shoppingCart[i]["SUM(subTotal)"]
            totItems += shoppingCart[i]["SUM(qty)"]
        # Render filtered view
        return render_template ("index.html", shoppingCart=shoppingCart,
shirts=shirts, shopLen=shopLen, shirtsLen=shirtsLen, total=total, totItems=totItems,
display=display, session=session )
    # Render filtered view
    return render_template ( "index.html", shirts=shirts, shoppingCart=shoppingCart,
shirtsLen=shirtsLen, shopLen=shopLen, total=total, totItems=totItems,
display=display)



@app.route("/checkout/")
def checkout():
    order = db.execute("SELECT * from cart")
```

```
    # Update purchase history of current customer
    for item in order:
        db.execute("INSERT INTO purchases (uid, id, samplename, image, quantity)
VALUES(:uid, :id, :samplename, :image, :quantity)", uid=session["uid"], id=item["id"],
samplename=item["samplename"], image=item["image"], quantity=item["qty"] )
    # Clear shopping cart
    db.execute("DELETE from cart")
    shoppingCart = []
    shopLen = len(shoppingCart)
    totItems, total, display = 0, 0, 0
    # Redirect to home page
    return redirect('/')



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

```
    # Get the id of shirt selected to be removed

    out = int(request.args.get("id"))

    # Remove shirt from shopping cart

    db.execute("DELETE from cart WHERE id=:id", id=out)

    # Initialize shopping cart variables

    totItems, total, display = 0, 0, 0

    # Rebuild shopping cart

    shoppingCart = db.execute("SELECT samplename, image, SUM(qty),
SUM(subTotal), price, id FROM cart GROUP BY samplename")

    shopLen = len(shoppingCart)

    for i in range(shopLen):

        total += shoppingCart[i]["SUM(subTotal)"]

        totItems += shoppingCart[i]["SUM(qty)"]

    # Turn on "remove success" flag

    display = 1

    # Render shopping cart
```

```
    return render_template ("cart.html", shoppingCart=shoppingCart,
shopLen=shopLen, total=total, totItems=totItems, display=display, session=session )




@app.route("/login/", methods=["GET"])
def login():
    return render_template("login.html")




@app.route("/new/", methods=["GET"])
def new():
    # Render log in page
    return render_template("new.html")




@app.route("/logged/", methods=["POST"] )
```

```
def logged():
    # Get log in info from log in form
    user = request.form["username"].lower()
    pwd = request.form["password"]
    #pwd = str(sha1(request.form["password"].encode('utf-8')).hexdigest())
    # Make sure form input is not blank and re-render log in page if blank
    if user == "" or pwd == "":
        return render_template ( "login.html" )
    # Find out if info in form matches a record in user database
    query = "SELECT * FROM users WHERE username = :user AND password =
:pwd"
    rows = db.execute ( query, user=user, pwd=pwd )


    # If username and password match a record in database, set session variables
    if len(rows) == 1:
        session['user'] = user
```

```
        session['time'] = datetime.now( )
        session['uid'] = rows[0]["id"]
    # Redirect to Home Page
    if 'user' in session:
        return redirect ( "/" )
    # If username is not in the database return the log in page
    return render_template ( "login.html", msg="Wrong username or password." )



@app.route("/history/")
def history():
    # Initialize shopping cart variables
    shoppingCart = []
    shopLen = len(shoppingCart)
    totItems, total, display = 0, 0, 0
    # Retrieve all shirts ever bought by current user
```

```
    myShirts = db.execute("SELECT * FROM purchases WHERE uid=:uid",
uid=session["uid"])

    myShirtsLen = len(myShirts)

    # Render table with shopping history of current user

    return render_template("history.html", shoppingCart=shoppingCart,
shopLen=shopLen, total=total, totItems=totItems, display=display, session=session,
myShirts=myShirts, myShirtsLen=myShirtsLen)




@app.route("/logout/")

def logout():

    # clear shopping cart

    db.execute("DELETE from cart")

    # Forget any user_id

    session.clear()

    # Redirect user to login form
```

```
    return redirect("/")




@app.route("/register/", methods=["POST"] )
def registration():
    # Get info from form
    username = request.form["username"]
    password = request.form["password"]
    confirm = request.form["confirm"]
    fname = request.form["fname"]
    lname = request.form["lname"]
    email = request.form["email"]
    # See if username already in the database
    rows = db.execute( "SELECT * FROM users WHERE username = :username ",
username = username )
    # If username already exists, alert user
```

```
    if len( rows ) > 0:
        return render_template ( "new.html", msg="Username already exists!" )
    # If new user, upload his/her info into the users database
    new = db.execute ( "INSERT INTO users (username, password, fname, lname,
email) VALUES (:username, :password, :fname, :lname, :email)",
                    username=username, password=password, fname=fname,
lname=lname, email=email )
    # Render login template
    return render_template ( "login.html" )



@app.route("/cart/")
def cart():
    if 'user' in session:
        # Clear shopping cart variables
        totItems, total, display = 0, 0, 0
```

```
        # Grab info currently in database
        shoppingCart = db.execute("SELECT samplename, image, SUM(qty),
SUM(subTotal), price, id FROM cart GROUP BY samplename")
        # Get variable values
        shopLen = len(shoppingCart)
        for i in range(shopLen):
            total += shoppingCart[i]["SUM(subTotal)"]
            totItems += shoppingCart[i]["SUM(qty)"]
    # Render shopping cart
    return render_template("cart.html", shoppingCart=shoppingCart,
shopLen=shopLen, total=total, totItems=totItems, display=display, session=session)
```

templates/login.html

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta content="text/html;charset=utf-8" http-equiv="Content-Type">
```

```
    <meta content="utf-8" http-equiv="encoding">

    <meta charset="utf-8">

    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-
fit=no">

    <meta name="theme-color" content="#000000">

    <link rel="shortcut icon" href="%PUBLIC_URL%/favicon.ico">

    <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
integrity="sha384-
Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JX
m"

        crossorigin="anonymous">

    <link href="{{ url_for('static',filename='css/custom.css') }}" rel="stylesheet"
type="text/css" />

    <script defer
src="https://use.fontawesome.com/releases/v5.0.6/js/all.js"></script>

    <script src="https://code.jquery.com/jquery-1.11.0.min.js"></script>
```

```
        <title>Legendry Fashion - Log In</title>
    </head>
    <body>
    <header>
        <nav class="navbar fixed-top navbar-dark bg-dark navbar-expand-sm box-
shadow">
          <a href="/" class="navbar-brand d-flex align-items-center">
             <strong><i class="fa  fa-cart-plus"></i>Online Clothing Store</strong>
          </a>
        </nav>
    </header><br />
    <main>
        <div class="container">
           <div class="row">
              <div class="col-sm">
                 <h2>Log In to Buy</h2>
```

```
                <p>{{ msg }}</p>

                <div>

                    <form action="/logged/" class="form" method="post">

                        <div>

                            <input type="text" name="username" autofocus
placeholder="Username">

                            <input type="password" name="password"
placeholder="Password">

                            <button type="submit" class="btn btn-primary">Login</button>

                        </div>

                    </form>

                </div>

            </div>

        </div>

    </main>
```

```
    </body>
</html>
```

new.html

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta content="text/html;charset=utf-8" http-equiv="Content-Type">
        <meta content="utf-8" http-equiv="encoding">
        <meta charset="utf-8">
        <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
        <meta name="theme-color" content="#000000">
        <link rel="shortcut icon" href="%PUBLIC_URL%/favicon.ico">
        <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
integrity="sha384-
Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JX
```

```
m"
            crossorigin="anonymous">
        <link href="{{ url_for('static',filename='css/custom.css') }}" rel="stylesheet"
type="text/css" />
        <script defer
src="https://use.fontawesome.com/releases/v5.0.6/js/all.js"></script>
        <script src="https://code.jquery.com/jquery-1.11.0.min.js"></script>


        <title>Trendy Clothing Store - Register</title>
    </head>
    <body>
        <header>
            <nav class="navbar fixed-top navbar-dark bg-dark navbar-expand-sm box-
shadow">
              <a href="/" class="navbar-brand d-flex align-items-center">
                  <strong><i class="fa fa-shopping-bag"></i> Sample Clothing
Store</strong>
```

```
            </a>
          </nav>
        </header><br />
        <main>
          <div class="container">
            <div class="row">
              <div class="col-sm">
                <h2>Register</h2>
                <p>{{msg}}</p>
                <form action="/register/" class="form" method="post">
                  <input type="text" name="username" id="username"
placeholder="Username" autofocus required > <span id="user-msg" class="alert
alert-danger"></span><br /><br />
                  <input type="password" name="password" id="password"
placeholder="Password" required > <span id="password-msg" class="alert alert-
danger"></span><br /><br />
                  <input type="password" name="confirm" id="confirm"
```

```
placeholder="Confirm Password" required> <span id="confirm-msg" class="alert
alert-danger"></span><br /><br />

                    <input type="text" name="fname" id="fname" placeholder="First
Name" required> <span id="fname-msg" class="alert alert-danger"></span><br
/><br />

                    <input type="text" name="lname" id="lname" placeholder="Last
Name" required> <span id="lname-msg" class="alert alert-danger"></span><br
/><br />

                    <input type="email" name="email" id="email" placeholder="Email"
required> <span id="email-msg" class="alert alert-danger"></span><br /><br /><br
/>

                    <button type="reset" class="btn btn-secondary">Clear</button>

                    <button type="submit" id="submit" class="btn btn-
primary">Register</button>

                </form>

            </div>

        </div>

    </div>
```

```
        </main>
    <!-- Custom JS Scripts -->
        <script src="{{ url_for('static',filename='js/validate.js') }}"></script>
    </body>
</html>
```

index.html

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



{% block title %}
Legendry Fashion- Home
{% endblock %}



{% block body %}
<!-- Main Store Body -->
    {% if session['user'] %}
        <div class="alert alert-warning alert-dismissible fade show" role="alert">
```

```
        <button type="button" class="close" data-dismiss="alert" aria-label="Close">
          <span aria-hidden="true">&times;</span>
        </button>
          <strong>Welcome, {{ session['user'] }}</strong> Hope you have a pleasant experience shopping with us.
      </div>


  {% endif %}
    <div class="row" id="shirtCard">
    {% for i in range(shirtsLen) %}
      <div class="col-sm">
        <div class="card text-center">
          <div class="card-body">
            <form action="/buy/" methods="POST">
              <h5 class="card-title">{{shirts[i]["typeClothes"].capitalize()}}</h5>
            <img src="/static/img/{{shirts[i]["image"]}}" class="shirt" alt="" />
```

```
                    <h5 class="card-text">{{shirts[i]["samplename"]}}</h5>

                    {% if shirts[i]["onSale"] %}

                      <img src="/static/img/sale-icon.png" width="26px" />

                      <h4 class="card-text price" style="color:red; display:inline">{{
'{:,.2f}'.format(shirts[i]["onSalePrice"]) }}</h4>

                    {% else %}

                      <h4 class="card-text price">{{ '{:,.2f}'.format(shirts[i]["price"]) }}</h4>

                    {% endif %}

                    <div class="stepper-input">

                       <span class="decrement target">-</span>

                       <input class="quantity" name="quantity" value='0' />

                       <span class="increment target">+</span>

                    </div>

                    <input type="hidden" name="id" value="{{shirts[i]["id"]}}" />

                    {% if not session %}

                    <input type="hidden" name="loggedin" value="0" />
```

```
                {% else %}

                <input type="hidden" name="loggedin" value="1" />



                {% endif %}

                <input type="submit" class="btn btn-primary addToCart" value="Add To Cart" /><br /><br />

                <div class="alert alert-danger flashMessage" style="text-align: center; display:none; font-size:0.9em;"></div>

              </form>

            </div>

          </div>

        </div>

    {% endfor %}

    </div>

  </div>

</main>
```

{% endblock %}

history.html

{% extends "base.html" %}

{% block title %}

Trendy Clothing Store - Home

{% endblock %}

{% block body %}

<!-- Main Store Body -->

<h2>Your Shopping History</h2>

Items you've bought in the past.

<table class="table table-sm">

```
<thead>
  <tr>
    <th scope="col">#</th>
    <th scope="col">Item</th>
    <th scope="col">Name</th>
    <th scope="col">Quantity</th>
    <th scope="col">Date</th>
    <th scope="col"></th>
  </tr>
</thead>
<tbody>
<!-- For Each shirt -->
{% for i in range(myShirtsLen) %}
  <tr>
    <th scope="row">{{ i + 1 }}</th>
    <td><img src="/static/img/{{ myShirts[i]["image"] }}" width="30px" alt="{{
```

```
myShirts[i]["samplename"] }}" /></td>

                <td>{{ myShirts[i]["samplename"] }}</td>

                <td>{{ myShirts[i]["quantity"] }}</td>

                <td>{{ myShirts[i]["date"] }}</td>

                <td><a href="/filter/?id={{ myShirts[i]["id"] }}"><button type="button"
class="btn btn-warning">Buy Again</button></a></td>

              </tr>

            </tbody>

            {% endfor %}

            <tfoot>

            </tfoot>

          </table>

        </div>

      </div>

    </div>

  </main>
```

```
{% endblock %}
```

cart.html

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



{% block title %}
Trendy Clothing Store - Home
{% endblock %}



{% block body %}
<!-- Main Store Body -->
  <div aria-hidden="true">
    <div>
      <div>
        <div>
          <h5 class="modal-title" id="exampleModalLongTitle">Shopping Cart</h5>
```

```
  <button type="button" class="close" data-dismiss="modal" aria-label="Close">
  </button>
</div>
<div>
  <div id="shoppingCart">
    <div class="container">
      <div class="row">
        <div class="col-sm">
          <table class="table table-sm">
            <thead>
              <tr>
                <th scope="col">#</th>
                <th scope="col">Item</th>
                <th scope="col">samplename</th>
                <th scope="col">Quantity</th>
                <th scope="col">Unit Price</th>
```

```
                    <th scope="col">Sub-Total</th>

                    <th scope="col"></th>

                  </tr>

                </thead>

                <tbody>

                <!-- For Each shirt -->

                {% if shopLen != 0 %}

                {% for i in range(shopLen) %}

                  <tr>

                    <th scope="row">{{ i + 1 }}</th>

                    <td><img src="/static/img/{{ shoppingCart[i]["image"] }}" width="30px"
alt="{{ shoppingCart[i]["samplename"] }}" /></td>

                    <td>{{ shoppingCart[i]["samplename"] }}</td>

                    <td><form action="/update/">

                      <input type="hidden" name="id" value="{{shoppingCart[i]["id"]}}" />

                      <input type="number" name="quantity" min="1" max="10" size="5"
```

```
value="{{ shoppingCart[i]['SUM(qty)'] }}">

                        <button type="submit" class="btn btn-warning
checkout">Update</button>

                      </form></td>

                    <td>{{ '${:,.2f}'.format(shoppingCart[i]["price"]) }}</td>

                    <td>{{ '${:,.2f}'.format(shoppingCart[i]['SUM(subTotal)']) }}</td>

                    <td>

                      <form action="/remove/" methods="GET">

                        <input type="hidden" name="id" value="{{ shoppingCart[i]["id"] }}"
/>

                        <button type="submit" class="btn btn-secondary btn-sm"
id="removeFromCart">Remove</button>

                      </form>

                    </td>

                  </tr>

                </tbody>
```

```
                {% endfor %}

                <tfoot>

                  <tr>

                    <td colSpan="7">Total: {{ '${:,.2f}'.format(total) }}<br /><br />

                      <div class="modal-footer">

                        <a href="/"><button type="button" class="btn btn-primary
checkout">Continue Shopping</button></a>

                        <a href="/checkout/"><button type="button" class="btn btn-success
checkout">Proceed to Checkout</button></a>

                      </div>

                    </td>

                  </tr>

                </tfoot>

                {% else %}

                  <tr>

                    <td colSpan="7"><h3>Your cart is empty :\</h3></td>
```

```
                </tr>
              </tbody>
              <tfoot>
                <tr>
                  <td colSpan="7">Get some shirts now!<br />
                    <div>
                      <a href="/"><button type="button" class="btn btn-secondary" data-dismiss="modal">Continue Shopping</button></a>
                    </div>
                  </td>
                </tr>
              </tfoot>
              {% endif %}
            </table>
          </div>
        </div>
```

```
            </div>
          </div>
        </div>
      </div>
    </div>
  </div>
  </main>

{% endblock %}
```

base.html

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

    <head>
        <meta content="text/html;charset=utf-8" http-equiv="Content-Type">
```

```
    <meta content="utf-8" http-equiv="encoding">

    <meta charset="utf-8">

    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-
fit=no">

    <meta name="theme-color" content="#000000">

    <link rel="shortcut icon" href="%PUBLIC_URL%/favicon.ico">

    <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
integrity="sha384-
Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JX
m"

        crossorigin="anonymous">

    <link href="{{ url_for('static',filename='css/custom.css') }}" rel="stylesheet"
type="text/css" />

    <script defer
src="https://use.fontawesome.com/releases/v5.0.6/js/all.js"></script>

    <script src="https://code.jquery.com/jquery-1.11.0.min.js"></script>
```

```
        <title>{% block title %}{% endblock %}</title>
    </head>
    <body>
  <!-- Modal -->
  <div class="modal fade" id="modalCenter" tabindex="-1" role="dialog" aria-
labelledby="exampleModalCenterTitle" aria-hidden="true">
    <div class="modal-dialog modal-dialog-centered modal-lg" role="document">
      <div class="modal-content">
        <div class="modal-header">
          <h5 class="modal-title" id="exampleModalLongTitle">Dashboard Page</h5>
          <button type="button" class="close" data-dismiss="modal" aria-label="Close">
            <span aria-hidden="true">&times;</span>
          </button>
        </div>
        <div class="modal-body">
          <div id="shoppingCart">
```

```
<div class="container">
  <div class="row">
    <div class="col-sm">
      <table class="table table-sm">
        <thead>
          <tr>
            <th scope="col">#</th>
            <th scope="col">Item</th>
            <th scope="col">Name</th>
            <th scope="col">Quantity</th>
            <th scope="col">Unit Price</th>
            <th scope="col">Sub-Total</th>
            <th scope="col"></th>
          </tr>
        </thead>
        <tbody>
```

```
                <!-- For Each shirt -->

                {% if shopLen != 0 %}

                {% for i in range(shopLen) %}

                  <tr>

                    <th scope="row">{{ i + 1 }}</th>

                    <td><img src="/static/img/{{ shoppingCart[i]["image"] }}" width="30px"
alt="{{ shoppingCart[i]["samplename"] }}" /></td>

                    <td>{{ shoppingCart[i]["samplename"] }}</td>

                    <td>{{ shoppingCart[i]['SUM(qty)'] }}</td>

                    <td>{{ '${:,.2f}'.format(shoppingCart[i]["price"]) }}</td>

                    <td>{{ '${:,.2f}'.format(shoppingCart[i]['SUM(subTotal)']) }}</td><!--

                    <td>

                      <form action="/remove/" methods="GET">

                        <input type="hidden" name="id" value="{{ shoppingCart[i]["id"] }}"
/>

                        <button type="submit" class="btn btn-secondary btn-sm"
```

```
id="removeFromCart">Remove</button>

                    </form>

                  </td>-->

                </tr>

              </tbody>

              {% endfor %}

              <tfoot>

                <tr>

                  <td colSpan="7">Total: {{ '${:,.2f}'.format(total) }}<br /><br />

                    <div class="modal-footer">

                      <a href="/cart/"><button type="button" class="btn btn-primary
checkout">Make Changes</button></a>

                      <button type="button" class="btn btn-primary checkout" data-
dismiss="modal">Continue Shopping</button>

                      <a href="/checkout/"><button type="button" class="btn btn-success
checkout">Quick Checkout</button></a>
```

```
        </div>
      </td>
    </tr>
  </tfoot>
  {% else %}
    <tr>
      <td colSpan="7"><h3>Your cart is empty :\</h3></td>
    </tr>
  </tbody>
  <tfoot>
    <tr>
      <td colSpan="7">Get some shirts now!<br />
        <div class="modal-footer">
          <button type="button" class="btn btn-primary" data-
dismiss="modal">Continue Shopping</button>
        </div>
```

```
                    </td>
                  </tr>
                </tfoot>
                {% endif %}
              </table>
            </div>
          </div>
        </div>
      </div>
    </div>
  </div>
  <header>
    <nav class="navbar fixed-top navbar-dark bg-dark navbar-expand-sm box-
shadow">
```

```
<a href="/" class="navbar-brand d-flex align-items-center">
    <strong><i class="fa  fa-cart-plus">Trendy Clothing Store</i></strong>
</a>
{% if session %}
<ul class="navbar-nav mr-auto">
  <li class="nav-item"><a href="/logout/" class="nav-link">Logout</a></li>
  <li class="nav-item"><a href="/history/" class="nav-link">You Bought</a></li>
{% else %}
<ul class="navbar-nav mr-auto">
  <li class="nav-item"><a href="/new/" class="nav-link">Register</a></li>
  <li class="nav-item"><a href="/login/" class="nav-link">Login</a></li>

{% endif %}
  <li class="nav-item dropdown">
    <a class="nav-link dropdown-toggle" href="#" id="navbardrop" data-
toggle="dropdown">
```

```
      Filter By
    </a>
    <div class="dropdown-menu">
      <a class="dropdown-item" href="/">All</a>
      <a class="dropdown-item" href="/filter/?typeClothes=shirt">Shirts</a>
      <a class="dropdown-item" href="/filter/?typeClothes=pant">Trousers</a>
      <a class="dropdown-item" href="/filter/?typeClothes=shoe">Shoes</a>
      <a class="dropdown-item" href="/filter/?kind=casual">Casual Clothing</a>
      <a class="dropdown-item" href="/filter/?kind=formal">Formal Clothing</a>
      <a class="dropdown-item" href="/filter/?sale=1">On Sale</a>
      <a class="dropdown-item" href="/filter/?price=1">Price $0-$000</a>
    </div>
  </li>
</ul>
<div>
  <button class="navbar-toggler" style="display:inline" type="button" data-
```

```
toggle="modal" data-target="#modalCenter">
        <span class="glyphicon glyphicon-shopping-cart" data-toggle="modal" data-
target="">
          <i class="fas fa-shopping-cart"></i>
          <span class="counter">No. of Items: {{ totItems }}</span>
          <span class="counter">Total: ${{ '{:,.2f}'.format(total) }}</span>
        </span>
      </button>
    </div>
  </nav>
</header><br />
<main>
  <div class="container">
    {% if display == 1 %}
    <div class="alert alert-success flashMessage" style="text-align:center">
      <strong>Your item was successfully removed from shopping cart!</strong>
```

```
  </div>

  {% endif %}

{% block body %}{% endblock %}

<footer>

    <div class="container">

        <div class="row">

            <div class="col-md">

                <hr />

                <p>&#169; <a href="/">Trendy Clothing Store</a></p>

            </div>

        </div>

    </div>

</footer>



    <!-- jQuery first, then Popper.js, then Bootstrap JS -->

    <script src="https://code.jquery.com/jquery-1.11.0.min.js"></script>
```

```
        <!-- <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js"
integrity="sha384-
ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q"
            crossorigin="anonymous"></script>-->
        <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"
integrity="sha384-
JZR6Spejh4U02d8jOt6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYl"
            crossorigin="anonymous"></script>
        <!-- Custom JS Scripts -->
        <script src="{{ url_for('static',filename='js/myscripts.js') }}"></script>
        <script src="{{ url_for('static',filename='js/validate.js') }}"></script>
    </body>
</html>
```

GitHub & Project Demo Link

GitHub link :

https://github.com/IBM-EPBL/IBM-Project-32683-1660211401

Video Link:

https://vimeo.com/775213077