CLOUD APPLICATION DEVELOPMENT

SMART FASHION RECOMMENDER APPLICATTION SYSTEM

PROJECT REPORT

IBM PROJECT -TEAM ID: PNT2022TMID39615

TEAM LEAD

SUSMITHA S G -510619104073

TEAM MEMBERS

MOHANA PRIYA V – 510619104050

HINITA PRIYANKA S – 510619104026

SIVAPRIYA T – 510619104065

PRIYADHARSHINI S – 510619104054

OF

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

C. ABDUL HAKEEM COLLEGE OF ENGINEERING AND TECHNOLOGY

ANNA UNIVERSITY::CHENNAI 600 025.

SMART FASHION RECOMMENDER APPLICATION SYSTEM

SMART FASHION RECOMMENDER APPLICATION SYSTEM

1. INTRODUCTION

1. 1 PROJECT OVERVIEW

Humans are inevitably drawn towards something that is visually more attractive. This tendency of humans has led to development of fashion industry over the course of time. With introduction of recommender systems in multiple domains, retail industries are coming forward with investments in latest technology to improve their business. Fashion has been in existence since centuries and will be prevalent in the coming days as well. Women are more correlated with fashion and style, and they have a larger product base to deal with making it difficult to take decisions. It has become an important aspect of life for modern families since a person is more often than not judged based on his attire. Moreover, apparel providers need their customers to explore their entire product line so they can choose what they like the most which is not possible by simply going into a cloth store.

With the increasing amount of information that people browse daily, how to quickly obtain Information Items that meet people's needs has become an urgent issue these days. The effort in information retrieval have brought great convenience to people who tend to retrieve information by entering a query or keywords. If some information-intensive websites can proactively suggest products or information items that users may be interested in, it will greatly improve the efficiency and satisfaction of users in obtaining information items. The research in the field of recommender systems precisely originated on this subject. Over the past years, tremendous progress has been made into this area, from non-personalised to personalised and to more recent deep learning recommender systems.

Although recommender systems have been widely applied, there are still many issues and challenges in designing high quality recommender systems. To measure the quality of a recommender system, a scientific and rigorous evaluation process is required.

This report reviews some existing well-established recommender systems and investigate some existing metrics for evaluating them. Besides, this report gives details of the project's implementation - a web application for the offline evaluation of three major collaborative filtering recommendation algorithms, item-based, user-based, and matrix-factorisation.

The application supports a wide range of readily configurable evaluation metrics for users to visualise the performance between different recommendation algorithms. The project aims to provide a comprehensive platform for designers to evaluate recommender systems and guide them to design better recommender systems

1.2 PURPOSE

The purpose is another important dimension to describe recommender systems. The purpose can be defined as the goals for which a recommender system is being used. The purpose can be the further study in recommendation algorithms for researchers, or online testing for evaluators, or simply the retrieval of relevant information for internet users. Understanding for the purpose is often important in real marketing because the purpose of users and information providers can sometimes be different. For instance, the users may be more interested in receiving high-quality recommendations, while the providers are more willing to suggest particular items or products to increase the revenue. That says the purpose guides the recommender employers to be on the right track. In it is emphasized that, in order to properly evaluate a recommender system, it is important to understand the user purpose. The paper points out that a comprehensive consideration for user goals makes it more sensible to decide what evaluation metrics are chosen to measure the quality of recommender systems.

2. LITERATURE SURVEY

Survey 1:
LIU, C., & WU, X. (2016):

A SURVEY ON E-COMMERCE RECOMMENDATION SYSTEM:

Several papers on RSs surveys had been published in the last decade in order to analyse major problems of traditional and non-traditional RSs. Li & Karahanna (2015) provide comprehensive research on e-commerce RSs addressing three major areas: understanding consumers, how recommendations work and the impacts of RSs. More akin to the methodology used in this paper, Lu et al, (2015) review the latest application developments of RSs in several application domains such as e-government, e-business and e-commerce through recommendation methods, software, and application platforms. While, Adomavicius and Tuzhilin (2005) reviewed the RSs, such as content-based, collaborative filtering-based and hybrid approaches; and discussed their limitations and possible solutions to enhance recommendation performance.

In this research, two types of articles that were published in the last three years were reviewed and classified:

1) Articles on RS techniques, specifically related methods and approaches;
2) Articles on RSs designed particularly for e-commerce domain.

The research and selection of these articles were performed as follows:

A. The following journal databases were used in order to select research papers on RSs for our study: ACM Digital Library, IEEE Xplore, Science Direct and SpringerLink.

B. Therefore, the search was implemented based on related keywords of RSs in e-commerce (such as, recommender system, e-commerce recommender system), while holding the following two criteria:

1) The articles must propose a novel recommender approach which addresses current limitations;
2) The articles must propose a novel recommender technique designed for e-commerce.
C. Based on the keywords related to e-commerce domain, these papers were divided by RS methods such as collaborative filtering-based, content-based, hybrid approaches and social-network based techniques.
Large e-vendors such as Amazon.com, eBay.com and Taobao.com are the best examples of massive implementers of recommender systems. 3
The products are usually recommended based on popularity, customer demographics and analysis of the customer's past purchase behaviours. (Schafer et al., 1999) Purchased product rating is a common function in e-shops. For example, in Amazon.com, feedbacks for purchased items are provided by giving a rating between 1 and 5. These ratings data can bee used to make recommendations. (Lu et al.,2015).

Survey 2:
ZHANG, Y.; CAVERLEE, J. (2019):

RECURRENT FASHION RECOMMENDATION WITH IMPLICIT VISUAL INFLUENCE:

Fashion-focused key opinion bloggers on Instagram, Facebook, and other social media platforms are fast becoming critical influencers. They can inspire consumer clothing purchases by linking high fashion visual evolution with daily street style. In this paper, they build the first visual influence-aware fashion recommender (FIRN) with leveraging fashion bloggers and their dynamic visual posts. Specifically, they extract the dynamic fashion features highlighted by these bloggers via a BiLSTM that integrates a large corpus of visual posts and community influence. Then they learnt the implicit visual influence funnel from bloggers to individual users via a personalized attention layer. Finally, they incorporate user personal style and her preferred fashion features across time in a recurrent recommendation network for dynamic fashion-updated clothing recommendation. Experiments show that FIRN outperforms state-of-the-art fashion recommenders, especially for users who are most impacted by fashion influencers, and utilizing fashion bloggers can bring greater improvements in recommendation compared with using other potential sources of visual information. They also release a large time-aware high-quality visual dataset of fashion influencers that can be exploited for future research.

Survey 3:
JH (JANGHYUN), BAEK; JOHN, TSAI; JUSTIN, SHAMOUN; MURIEL, MARABLE; YING CUI, YING; (2020):

AMAZON RECOMMENDED SYSTEM:

Amazon has been collecting, storing, processing, and analysing personal information from customers as a means of determining how they spend their money. Amazon currently

uses item-item collaborative filtering, which scales to massive datasets and produces high quality 4 recommendation systems in real time. This system is a kind of an information filtering system which seeks to predict the "rating" or preferences which user is interested in. Product recommendations tailored to a user are more likely to lead to higher conversion. Recommended products account for 35% of Amazon revenue (MacKenzie). Furthermore, users want recommendations of similar item.

Survey 4.

QINGQING TU, LE DONG (2010):

AN INTELLIGENT PERSONALIZED FASHION RECOMMENDATION SYSTEM:

In this paper, they proposed a novel system-Intelligent Personalized Fashion Recommendation System, which created a new space in web multimedia mining and recommendation. The proposed system of this project significantly helped customers to find their most suitable fashion choices in mass fashion information in the virtual space based on multimedia mining.

There are three stand-alone models developed in this paper to optimize the analysis of fashion features in mass fashion trend:

(i). Interaction and recommender model, which associated clients' personalized demand with the current fashion trend, and helps clients find the most favourable fashion factors in trend.

(ii). Evolutionary hierarchical fashion multimedia mining model, which created a hierarchical structure to filer the key components of fashion multimedia information in the virtual space, and it proved to be more efficient for web mass multimedia mining in an evolutionary way.

(iii). Colour tone analysis model, a relevant and straightforward approach for analysis of main colour tone as to the skin and clothing is used. In this model, a refined contour extraction of the fashion model method was also developed to solve the dilemma that the accuracy and efficiency of contour extraction in the dynamic and complex video scene. As evidenced by the experiment, the proposed system outperformed in effectiveness on mass fashion information in the virtual space compared with human, and thus developed a personalized and diversified way for fashion recommendation They developed a cloth recommendation system with using only single photo of user with scalable embedded system. This study led to important results and gave new opportunities for clothing companies and advertisements. In this study, they showed that how their system recommended a cloth options without user's previous shopping act data with embedded system and machine learning. In order to recommend a cloth, they developed two inception based convolutional neural networks as prediction part and one feed forward neural network as recommender. In this study, they reached to 98% accuracy on colour prediction, 86% accuracy on gender and cloth's pattern predictions and 75% accuracy on clothing recommendation

Survey 5:

BATUHAN AŞIROĞLU (2019):

A SMART CLOTHING RECOMMENDATION SYSTEM WITH DEEP LEARNING:

They developed a cloth recommendation system with using only single photo of user with scalable embedded system. This study led to important results and gave new opportunities for clothing companies and advertisements. In this study, they showed that how their system recommended a cloth options without user's previous shopping act data with embedded system and machine learning. In order to recommend a cloth, they developed two inception based convolutional neural networks as prediction part and one feed forward neural network as recommender. In this study, they reached to 98% accuracy on colour prediction, 86% accuracy on gender and cloth's pattern predictions and 75% accuracy on clothing recommendation.

2.2 REFERENCES

1. Liu, C., & Wu, X. (2016). Large-scale recommender system with compact latent factor model, 64, 467 475.doi:10.1016/j.eswa.2016.08.009.

2. Adomavicius, G., & Tuzhilin, A. (2005). Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. IEEE Transactions on Knowledge and Data Engineering, 17(6), 734–749. doi:10.1109/TKDE.2005.99.

3. Zhang, Y.; Caverlee, J. Instagrammers, Fashionistas, and Me: Recurrent Fashion Recommendation with Implicit Visual Influence. In Proceedings of the 28th ACM International Conference on Information and Knowledge Management, Beijing, China, 3–7 November 2019; pp. 1583–1592. [Google Scholar] [CrossRef][Green Version].

4. JH (Janghyun), Baek; John, Tsai; Justin, Shamoun; Muriel, Marable; Ying Cui,Ying;(2020) Amazon Recommender System.

5. Qingqing Tu,Le Dong -An Intelligent Personalized Fashion Recommendation System - 2010.

6. batuhan aşiroğlu- smart clothing recommendation system with deep learning 2019 3rd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT)11-13 Oct. 2019

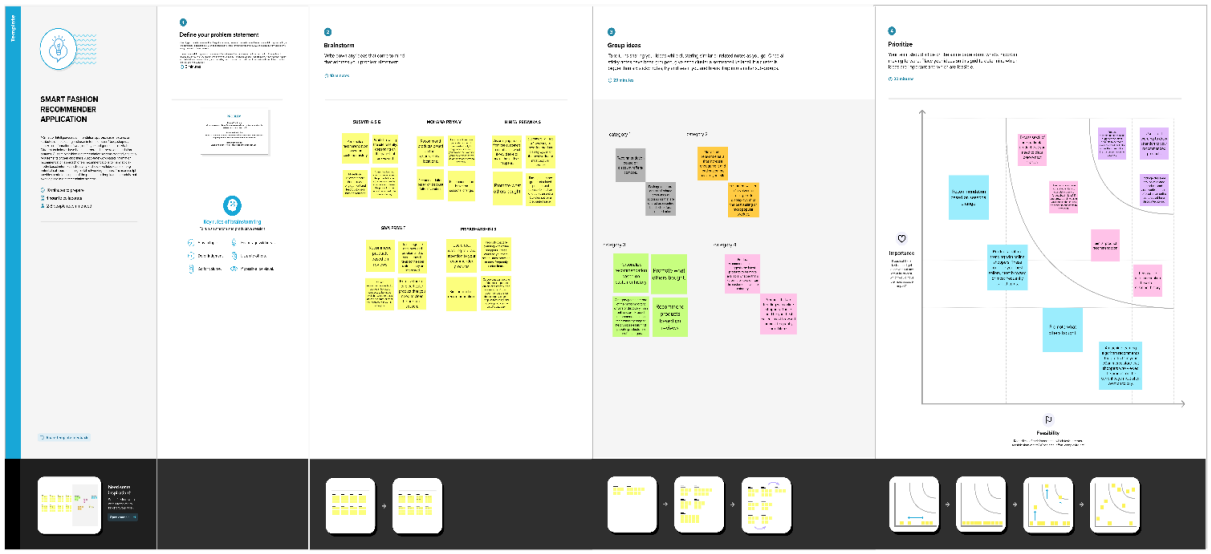
2.3 PROBLEM STATEMENT DEFINITIONS:

I AM	I'M TRYING TO	BUT	BECAUSE	WHICH MAKES ME
A Business man of an organization	Purchase blazers	I couldn't find What I search for	Presence of similar products with usual colours	Unsatisfied with the suggestions
A College Student	Shop for maxi dresses	I Can't get the filtered Product	There are less number of recommendation	Pleased not to acquire the product
A Customer	Buy number of specific products for my startup	There is no similar net price for all the products	It varies time to time	I couldn't complete my purchase

3. IDEATION AND PROPOSED SOLUTION:

3.1 EMPATHY MAP CANVAS

3.2 IDEATION AND BRAINSTROMING:



3.3 PROPOSED SOLUTION:

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	One biggest issue is the scalability of algorithms having real-world datasets under the recommendation system, a huge changing data is generated by user-item interactions in the form of ratings and reviews and consequently, scalability is a big concern for these datasets. The recommendation system is because of information overload, and we can call it an information filter system. It greatly influences what we interact with the world: shopping (Amazon, Best Buy), music(Spotify), Video (Youtube, Netfix), etc. To build a recommendation system providing recommendations to millions of users with millions of items, the first thing is, define the problem.
2.	Idea / Solution description	The goal of this survey is to provide a review of recommender systems that operate in the specific vertical domain of garment and fashion products. We have identified the most pressing challenges in fashion research and created a taxonomy that categorizes the literature according to the objective they are trying to accomplish (e.g., item or outfit recommendation, size recommendation, explainability, among others) and type of side information (users, items, context). We have also identified the most important evaluation goals and perspectives (outfit generation, outfit recommendation, pairing recommendation, and fill-in-the-blank outfit compatibility prediction) and the most commonly used datasets and evaluation metrics.
3.	Novelty / Uniqueness	Recommender systems help users navigate large collections of products to find items relevant to their interests leveraging large amounts of product information and user signals like product views, followed or ignored items, purchases or web-page visits to determine how, when and what to recommend to their customers. Recommender systems have grown to be an essential part of all large Internet retailers.

4.	Feasibility of Idea	Due to market dynamics and customer preferences, there is a large vocabulary of distinct fashion products, as well as high turnover. This leads to sparse purchase data, which challenges the usage of traditional recommender systems . Furthermore, precise and detailed product information is often not available, making it difficult to establish similarity between products. To deal with the aforementioned problems, and given the visual and aesthetic nature of fashion products, there is a growing body of computer vision research addressing tasks like localizing fashion items determining their category and attributes or establishing the degree of similarity to other products, to name only a few.
5.	Business model(Revenue model)	Traditional recommender systems such as Collaborative Filtering or Content-Based Filtering have difficulties in the fashion domain due to the sparsity of purchase data, or the insufficient detail about the visual appearance of the product in category names . Instead, more recent literature has leveraged models that capture a rich representation of fashion items through product images, text descriptions or customer reviews or videos which are often learned through surrogate tasks like classification or product retrieval.
6.	Social impact/ Customer Satisfaction	The textile and apparel industries have grown tremendously over the last years. Customers no longer have to visit many stores, stand in long queues, or try on garments in dressing rooms as millions of products are now available in online catalogs. However, given the plethora of options available, an effective recommendation system is necessary to properly sort, order, and communicate relevant product material or information to users. Effective fashion RS can have a noticeable impact on billions of customers' shopping experiences and increase sales and revenues on the provider-side.
7.	Scalability of the solution	By implementing this system , the people can efficiently and effectively predict the quality of the products. This system can also be integrated with the future Technologies.

3.4 PROBLEM SOLUTION FIT:

Define CS, fit into C

1. CUSTOMER SEGMENT(S)

- Demographic segmentation
- Technographic segmentation
- Geographic segmentation
- Behavioral segmentation
- Needs-based segmentation

6. CUSTOMER

With the rapid rising of living standard, people gradually developed higher shopping enthusiasm and increasing demand for garment. Nowadays, an increasing number of people pursue fashion.

- Customer should aware of updated application for recommendation.
- Smart devices with active Internet Connection.
- Customer should have installed application

5. AVAILABLE SOLUTIONS

- The system does a great job in inculcating a fashion sense among the users and can provide the best recommendations based on the user's wardrobe.
- Since the system is implemented as a website, it is very easy for the end users to access as well as use.
- The scope of this system can be expanded by including the ability to detect the various design and patterns on clothing, and to increase the number of occasions.

Explore AS, differentiate

Focus on J&P, tap into BE, understand

2. JOBS-TO-BE-DONE / PROBLEMS J&P

- Lack of Data. Maybe the biggest problem facing recommender systems is that they need a lot of data to effectively make suggestions.
- Changing Data.
- Changing User Preferences.
- Unforeseeable Items.
- This Stuff is Complex.

9. PROBLEM ROOT CAUSE

- Recommending items to users in case there is very little data available related to the user or item.
- If you do not have high-quality data, or cannot crunch and analyze it properly, you will not be able to make the most of the recommender application.

7. BEHAVIOUR

- Managing the User-Based Collaborative Filtering Model and making remarks.
- Many shopping websites have no website policies at all or have unclear and confusing user, return and refund policy.
- One of the biggest challenges faced is security breaches.
- Missing or Unclear Product Informations.

Focus on J&P, tap into BE, understand

3. TRIGGERS

Recommendation systems have been proposed in many domains, but have received limited attention in the area of End-User Development (EUD). We propose a novel approach for formulating recommendations in this area, based on deconstructing trigger-action rules into sequences of elements and the links between them.

4. EMOTIONS: BEFORE / AFTER

Before: Customers have individual knowledge and emotional feature.

After: Customer feel smart through the knowledge representation of the recommender application

10. YOUR SOLUTION

Using a mobile phone App, people can easily take of photo of the appealing clothes they saw on magazine, web page or even street, then get the recommended clothing with similar fashion and style in seconds. People can even directly link to the online shopping website to purchase if they like it. When people find a clothes they like but don't know where to buy it or how to find more similar clothing, the Clothing Fashion Style Recommendation System provides a convenient way to help find that. What's more, designed under the concept of Model-View-Presenter, the Clothing Fashion Style Recommendation System provides a highly flexible and extensible framework.
Also, if we want to significant improve the system in the future, we can let people who good at programming and aesthetic designing work on improve the view. Thus, this is a well-designed framework for long term maintaining and upgrading.

8.CHANNELS OF BEHAVIOUR

ONLINE

Through Advertising in social medias, news platforms makes customer to know and recognize the effectiveness of recommending system and their instant and secure features.

OFFLINE

Words of mouth among customers.

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 through Form Registration through Gmail
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	Product Master	It includes the information of the products, item no, size, categories etc.
FR-4	Mobile Friendliness	An analytic tool to study the audience and inquire about their devices. Know the position of essential buttons and options on the webpages, change them accordingly for a better experience.
FR-5	Price Master	Only for the price of the products and applicable discount of the products.
FR-6	Transaction	It is a payment method in which the transfer of money of buying products. This process is secure and password protected.
FR-7	Reporting	After ordering for the product, the system will send one copy of the bill to the customer's Email-address and another one for the system data base.
FR-8	Delivery Report	List of the products that can be delivered to the customer.
FR-9	Changes to cart	Changes to cart means the customer after login or registration can make order or cancel order of the product from the shopping cart.
FR-10	Payment	In this system we are dealing the mode of payment by cash. We will extend this to credit card, debit card etc. in the future.
FR-11	Interface aspects	Simulates and processes human conversation (either spoken or written), allowing humans to interact with digital devices as if they were communicating with a real person.
FR-12	Logout	After ordering or surfing for the product customer has to logout.

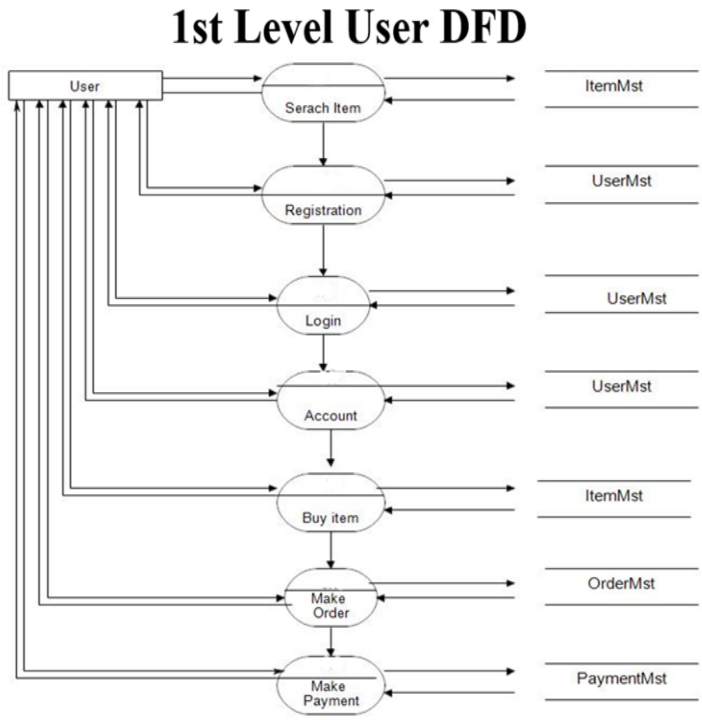
4.2 Non-Functional Requirements:

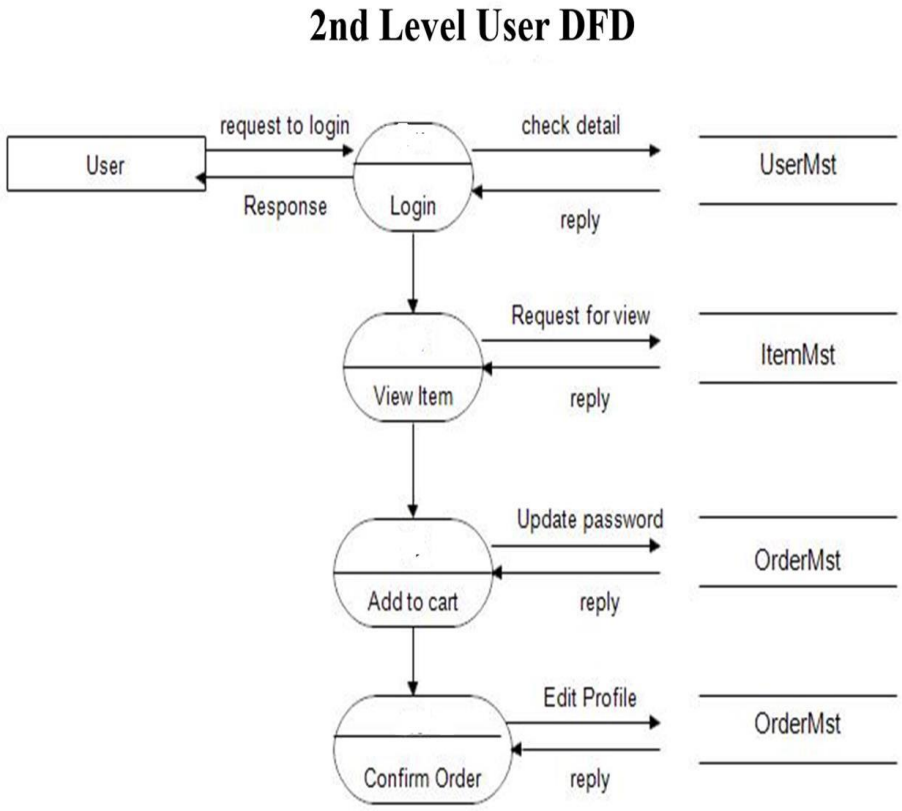
Following are the non-functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	Usability encapsulates the user experience. Essentially, it means the ease with which a visitor to the site can interact with it. If a site has strong usability, it provides an experience that is more comfortable and straightforward for its users to navigate.
NFR-2	**Security**	Security comes with utmost importance if a site is dealing with monetary transactions, users' financial and sensitive data. Privacy – The control over one's personal data. Security – The attempted access to data by unauthorized others.
NFR-3	**Maintainability**	Thriving the website maintenance from the initial development means cutting the time & cost to determine and resolve the faults of the system in the future
NFR-4	**Performance**	The focus is on loading the e-commerce store as fast as possible regardless of the number of integrations and traffic on the website.
NFR-5	**Availability**	It's available for 24x7 hours and in any browsers.
NFR-6	**Scalability**	It will define how the website can grow and increase its features and functionality without impacting the performance of the website.

5. PROJECT DESIGN:

5.1 DATA FLOW DIAGRAMS:





5.2. TECHNICAL ARCHITECTURE

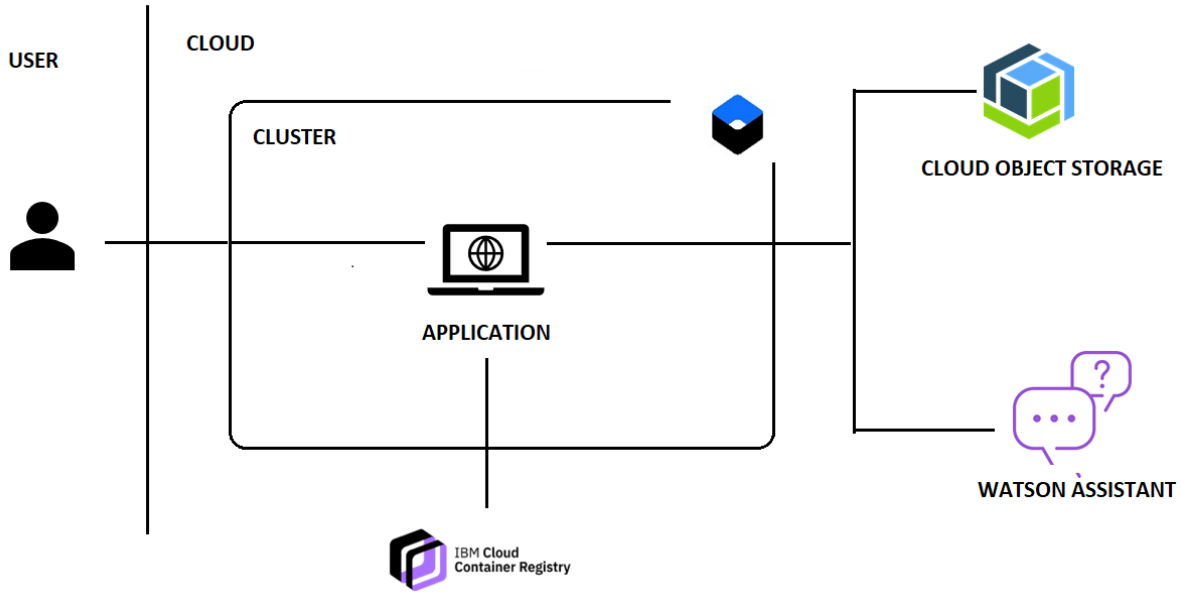


Table 1: Components and Technologies:

S.No	Component	Description	Technology
1.	User Interface	How user interacts with application e.g. Web UI, Mobile App, Chatbot etc.	HTML, CSS, JavaScript / Angular Js / React Js etc.
2.	Application Logic-1	Logic for a process in the application	Java / Python
3.	Application Logic-2	Logic for a process in the application	IBM Watson STT service
4.	Application Logic-3	Logic for a process in the application	IBM Watson Assistant
5.	Database	Data Type, Configurations etc.	MySQL, No SQL, etc.
6.	Cloud Database	Database Service on Cloud	IBM DB2, IBM Cloudant etc.
7.	File Storage	File storage requirements	IBM Block Storage or Other Storage Service or Local File system
8.	Infrastructure (Server / Cloud)	Application Deployment on Local System / Cloud Local Server Configuration: Cloud Server Configuration :	Local, Cloud Foundry, Kubernetes, etc.

Table 2: Application Characteristics

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	List the open-source frameworks used	Python flask
2.	Security Implementations	List all the security / access controls implemented, use of firewalls etc.	e.g. Encryptions, antivirus etc.
3.	Scalable Architecture	Justify the scalability of architecture (3 – tier, Micro-services)	Ability to increase or decrease IT resource as needed to meet changing demand
4.	Availability	Justify the availability of application (e.g. use of load balancers, distributed servers etc.)	For cloud infrastructure solutions, availability refers to time that the data center is accessible.
5.	Performance	Design consideration for the performance of the application (number of requests per sec, use of Cache, use of CDN's) etc.	A field of practice that users various tools, processors, and ideas in a scientific, systemetic manner to improve the desired outcomes of individual and organisations.

5.3 USER STORIES:

User Story Number	User Story / Task	Acceptance criteria	Priority	Release
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 can log into the application by entering registered email & password	I can access my account/ dashboard	High	Sprint-1

USN-3	As a user, I can view all the information of my orders	I can get information needed in my dashboard.	Low	Sprint-2
USN-4	As a user, I can place order by providing details.	I can come to know detailed information.	Medium	Sprint 2
USN-5	As a user, I can ask a query regarding anything with bot.	I can get clear among doubts.	Medium	Sprint-3
USN-6	As a user, I can easily pay for my order through online.	I can pay through credit card or UPI.	High	Sprint-4
USN-7	As a user, I can add my wish to cart.	I can view them later to place orders	High	Sprint-4
USN-8	As a user, I can give my experience of this application.	I can give feedback or compliant.	Medium	Sprint-3
USN-1	As a admin, I can log into the application by entering registered email & password	I can access my account / dashboard.	High	Sprint-1
USN-2	As a admin, I can view all orders placed in entire system and monitor the whole process of application.	I can access and monitor all process.	High	Sprint-2

6. PROJECT PLANNING AND SCHEDULING:

6.1 PROJECT PLANNING AND ESTIMATION

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	14	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	18	11 Oct 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	10	14 Oct 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	18	19 Oct 2022

Velocity:

Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day).

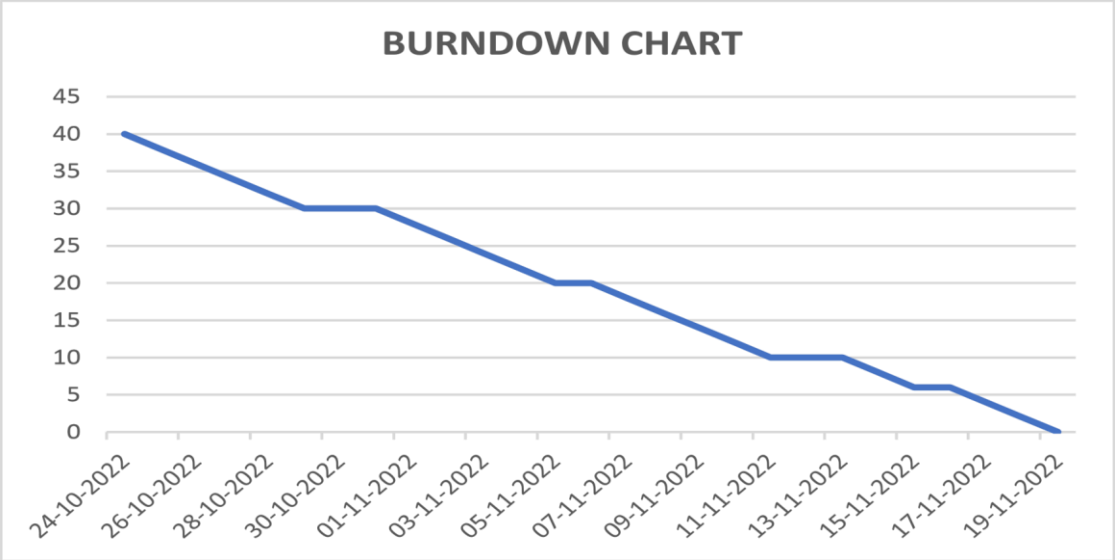
$$AV = \frac{sprint\ duration}{velocity} = \frac{20}{10} = 2$$

6.2 SPRINT DELIVERY SCHEDULE:

Sprint	Functional Requirement (Epic)	User Story Number	User Story/ Task	Story Points	Priority	Team Members
Sprint-1	Setting up an Environment	USN-1	Creating an IBM Cloud account	5	High	Susmitha S.G HinitaPriyanka S Mohana Priya V Siva Priya T Priyadharshini S
		USN-2	Installation of IBM Cloud CLI, Docker CLI And IBM Cloud Plugins	4	Medium	
		USN-3	Installation flask framework and start working on it.	3	High	
		USN-4	Creating an SendGrid Account.	2	Medium	

Sprint-2	Creating web application	USN-5	Creating UI to interact with application	7	High	Susmitha S.G Mohana Priya V Priyadharshini S
		USN-6	Creating IBM DB2 to connect with python.	6	Medium	HinitaPriyanka S Siva Priya T
		USN-7	Integrating SendGrid with python code.	5	Low	Mohana Priya V HinitaPriyanka S
Sprint-3	Chatbot Development	USN-8	Developing a chatbot for web application and integrating to HTML page.	10	Medium	Susmitha S.G Siva Priya T
Sprint-4	Final delivery	USN-9	Containerizing a developed app and uploading an image to IBM Container Registry.	9	High	Priyadharshini S Hinita Priyanka S Susmitha S.G
		USN-10	Deployment of application in Kubernetes.	10	High	Susmitha S.G HinitaPriyanka S Mohana Priya V Siva Priya T Priyadharshini S
		USN-11	Final documentation.	10	High	Mohana Priya Siva Priya T

BURNDOWN



7. CODING AND SOLUTIONS:

app.py:

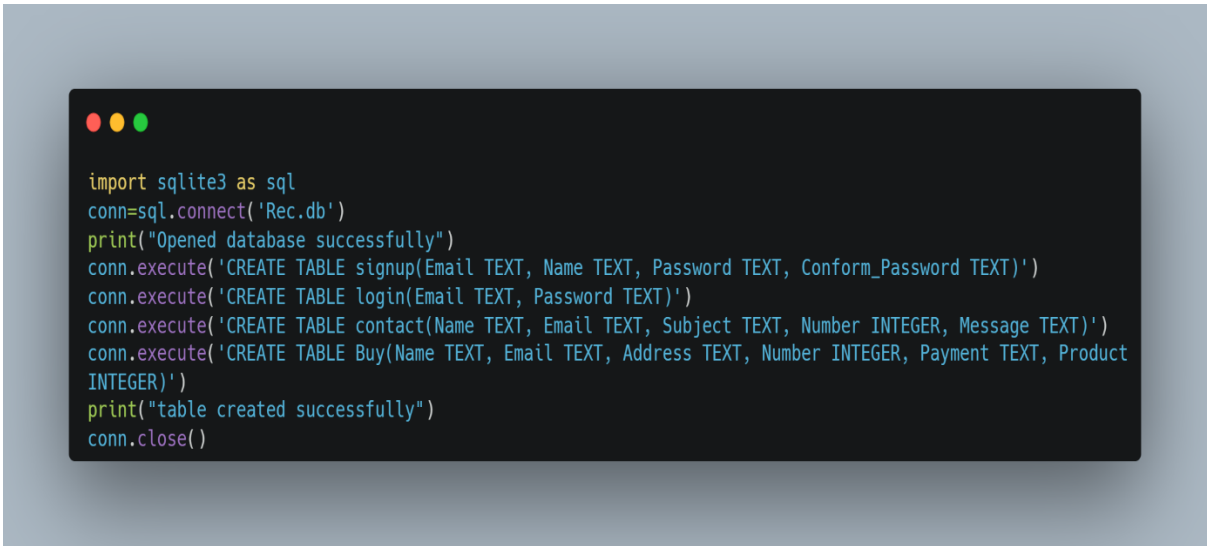
```
from flask import Flask, render_template, redirect, url_for, request

import sqlite3 as sql
app = Flask(__name__)
@app.route("/")
def hello():
    return redirect(url_for('home'))
@app.route("/home")
def home():
    return render_template('index.html')
@app.route("/signup")
def signup():
     return render_template('signup.html')
@app.route("/register",methods=['GET','POST'])
def register():
    if request.method=='POST':
        try:
            Email=request.form['email']
            Name=request.form['usrname']
            Password=request.form['psw']
            Conform_Password=request.form['psw-repeat']
            with sql.connect("Rec.db") as con:
                cur=con.cursor()
```

```
                cur.execute("INSERT INTO signup (Email, Name, Password,
Conform_Password) VALUES (?,?,?,?)",(Email, Name, Password, Conform_Password))
                con.commit()
        finally:
            return render_template('index.html')
    con.close()

@app.route("/logging",methods=['GET','POST'])
def logging():
    if request.method=='POST':
        try:
            Email=request.form['Email']
            Password=request.form['Password']
            with sql.connect("Rec.db") as con:
                cur=con.cursor()
                cur.execute("INSERT INTO login (Email, Password) VALUES
(?,?)",(Email, Password))
                con.commit()

        finally:
            return render_template('index.html')
    con.close()
@app.route("/contact1",methods=['GET','POST'])
def contact1():
    if request.method=='POST':
        try:
            Email=request.form['Email']
            Name=request.form['Name']
            Number=request.form['Number']
            Subject=request.form['Text']
            Message=request.form['Content']
            with sql.connect("Rec.db") as con:
                cur=con.cursor()
                cur.execute("INSERT INTO contact (Email, Name, Number,Subject,
Message) VALUES (?,?,?,?,?)",(Email, Name, Number,Subject, Message))
                con.commit()

        finally:
            return render_template("index.html")
    con.close()
@app.route("/order",methods=['GET','POST'])
def order():
    if request.method=='POST':
        try:
            Email=request.form['Email']
            Name=request.form['Name']
            Number=request.form['Number']
            Address=request.form['Address']
            Payment=request.form['Payment']
```

```
            Product=request.form['Product']
            with sql.connect("Rec.db") as con:
                cur=con.cursor()

                cur.execute("INSERT INTO Buy(Email, Name, Number, Address,
Payment, Product) VALUES (?,?,?,?,?,?)",(Email, Name, Number, Address,
Payment, Product))

                con.commit()

        finally:
            return render_template("index.html")

    con.close()
@app.route('/admin1')
def admin():
    con=sql.connect("Rec.db")
    con.row_factory=sql.Row
    cur=con.cursor()
    cur.execute("select * from signup")
    signup=cur.fetchall();
    cur.execute("select * from login")
    login=cur.fetchall();
    cur.execute("select * from contact")
    contact=cur.fetchall();
    cur.execute("select * from Buy")
    Buy=cur.fetchall();
    return
render_template("admin1.html",signup=signup,login=login,contact=contact,Buy=Bu
y)



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

@app.route("/about")
def about():
    return render_template('#about')

@app.route("/contact")
def contact():
    return render_template('contact.html')
@app.route("/cart")
def cart():
    return render_template('cart.html')
@app.route("/logout")
def logout():
```

```
    return render_template('logout.html')
@app.route("/loggedout")
def loggedout():
    return render_template('loggedout.html')
@app.route("/Women")
def Women():
    return render_template('Women.html')
@app.route("/Mens")
def Men():
    return render_template('Mens.html')
@app.route("/Kids")
def kids():
    return render_template('kids.html')
@app.route("/product")
def product():
    return render_template('product.html')
@app.route("/buy")
def buy():
    return render_template('Buy.html')


if __name__=='__main__':
    app.run()
```

app 1.py:



```
import sqlite3 as sql
conn=sql.connect('Rec.db')
print("Opened database successfully")
conn.execute('CREATE TABLE signup(Email TEXT, Name TEXT, Password TEXT, Conform_Password TEXT)')
conn.execute('CREATE TABLE login(Email TEXT, Password TEXT)')
conn.execute('CREATE TABLE contact(Name TEXT, Email TEXT, Subject TEXT, Number INTEGER, Message TEXT)')
conn.execute('CREATE TABLE Buy(Name TEXT, Email TEXT, Address TEXT, Number INTEGER, Payment TEXT, Product
INTEGER)')
print("table created successfully")
conn.close()
```

```python
from flask import Flask, render_template, redirect, url_for, request
import sqlite3 as sql
app = Flask(__name__)
@app.route("/")
def hello():
    return redirect(url_for('home'))
@app.route("/home")
def home():
    return render_template('index.html')
@app.route("/signup")
def signup():
     return render_template('signup.html')
@app.route("/register",methods=['GET','POST'])
def register():
    if request.method=='POST':
        try:
            Email=request.form['email']
            Name=request.form['usrname']
            Password=request.form['psw']
            Conform_Password=request.form['psw-repeat']
            with sql.connect("Rec.db") as con:
                cur=con.cursor()
                cur.execute("INSERT INTO signup (Email, Name, Password, Conform_Password) VALUES (?,?,?,?)",
(Email, Name, Password, Conform_Password))
                con.commit()
        finally:
            return render_template('index.html')
    con.close()

@app.route("/logging",methods=['GET','POST'])
def logging():
    if request.method=='POST':
        try:
            Email=request.form['Email']
            Password=request.form['Password']
            with sql.connect("Rec.db") as con:
                cur=con.cursor()
                cur.execute("INSERT INTO login (Email, Password) VALUES (?,?)",(Email, Password))
                con.commit()

        finally:
            return render_template('index.html')
    con.close()
@app.route("/contact1",methods=['GET','POST'])
def contact1():
    if request.method=='POST':
        try:
            Email=request.form['Email']
            Name=request.form['Name']
            Number=request.form['Number']
            Subject=request.form['Text']
            Message=request.form['Content']
            with sql.connect("Rec.db") as con:
                cur=con.cursor()
                cur.execute("INSERT INTO contact (Email, Name, Number,Subject, Message) VALUES (?,?,?,?,?)",
(Email, Name, Number,Subject, Message))
                con.commit()

        finally:
            return render_template("index.html")
    con.close()
@app.route("/order",methods=['GET','POST'])
def order():
    if request.method=='POST':
        try:
            Email=request.form['Email']
            Name=request.form['Name']
            Number=request.form['Number']
            Address=request.form['Address']
            Payment=request.form['Payment']
            Product=request.form['Product']
            with sql.connect("Rec.db") as con:
                cur=con.cursor()

                cur.execute("INSERT INTO Buy(Email, Name, Number, Address, Payment, Product) VALUES
(?,?,?,?,?,?)",(Email, Name, Number, Address, Payment, Product))

                con.commit()

        finally:
            return render_template("index.html")

    con.close()
@app.route('/admin1')
def admin():
    con=sql.connect("Rec.db")
    con.row_factory=sql.Row
    cur=con.cursor()
    cur.execute("select * from signup")
    signup=cur.fetchall();
    cur.execute("select * from login")
    login=cur.fetchall();
    cur.execute("select * from contact")
    contact=cur.fetchall();
    cur.execute("select * from Buy")
    Buy=cur.fetchall();
    return render_template("admin1.html",signup=signup,login=login,contact=contact,Buy=Buy)


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

@app.route("/about")
def about():
    return render_template('#about')

@app.route("/contact")
def contact():
    return render_template('contact.html')
@app.route("/cart")
def cart():
    return render_template('cart.html')
@app.route("/logout")
def logout():
    return render_template('logout.html')
@app.route("/loggedout")
def loggedout():
    return render_template('loggedout.html')
@app.route("/Women")
def Women():
    return render_template('Women.html')
@app.route("/Mens")
def Men():
    return render_template('Mens.html')
@app.route("/Kids")
def kids():
    return render_template('kids.html')
@app.route("/product")
def product():
    return render_template('product.html')
@app.route("/buy")
def buy():
    return render_template('Buy.html')


if __name__=='__main__':
    app.run()
```

8. TESTING:

8.1 TEST CASES

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

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

8.2 USER ACCEPTANCE TESTING

Purpose of Document

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

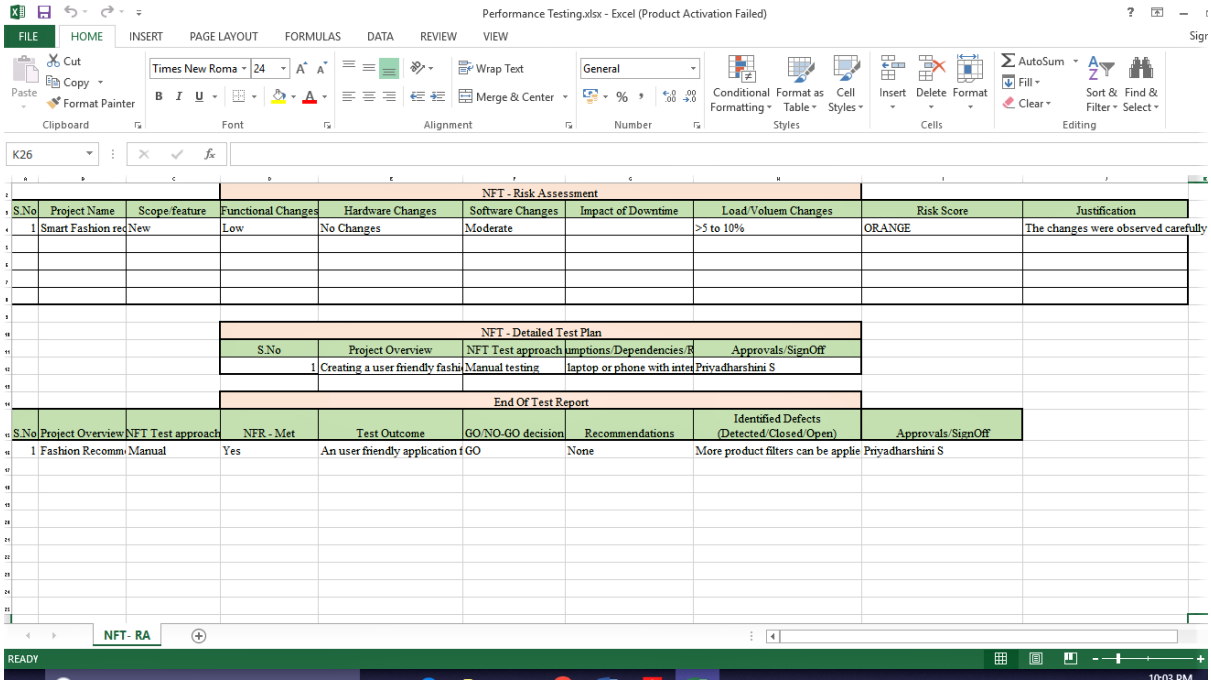
Defect Analysis

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

Resolution	**Severity 1**	**Severity2**	**Severity3**	**Severity4**	**Subtotal**
By Design	5	5	2	3	21
Duplicate	1	0	3	0	4
External	2	3	0	1	6
Fixed	11	2	4	20	37
Not Reproduced	0	0	0	0	1
Skipped	0	0	0	1	2
Won’t fix	0	5	2	1	8
Total	24	14	13	26	77

9. RESULTS:

9.1 PERFORMANCE METRICS:



Performance Testing.xlsx - Excel (Product Activation Failed)

NFT - Risk Assessment

S.No	Project Name	Scope/feature	Functional Changes	Hardware Changes	Software Changes	Impact of Downtime	Load/Voluem Changes	Risk Score	Justification
1	Smart Fashion rec	New	Low	No Changes	Moderate		>5 to 10%	ORANGE	The changes were observed carefully

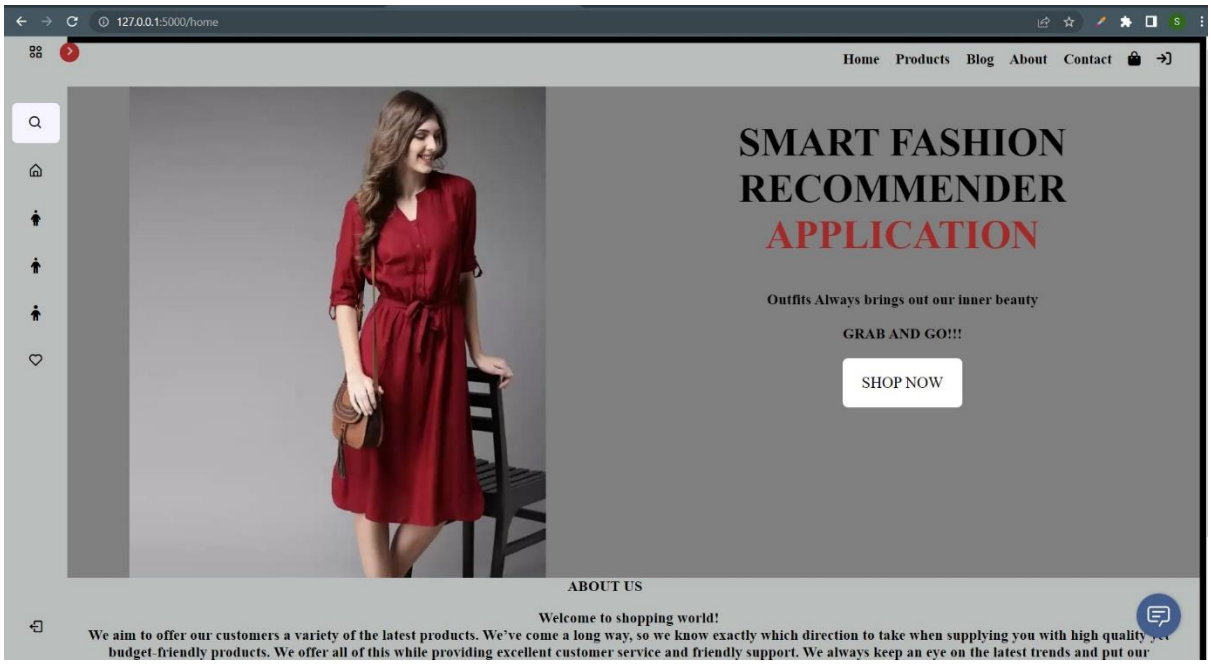
NFT - Detailed Test Plan

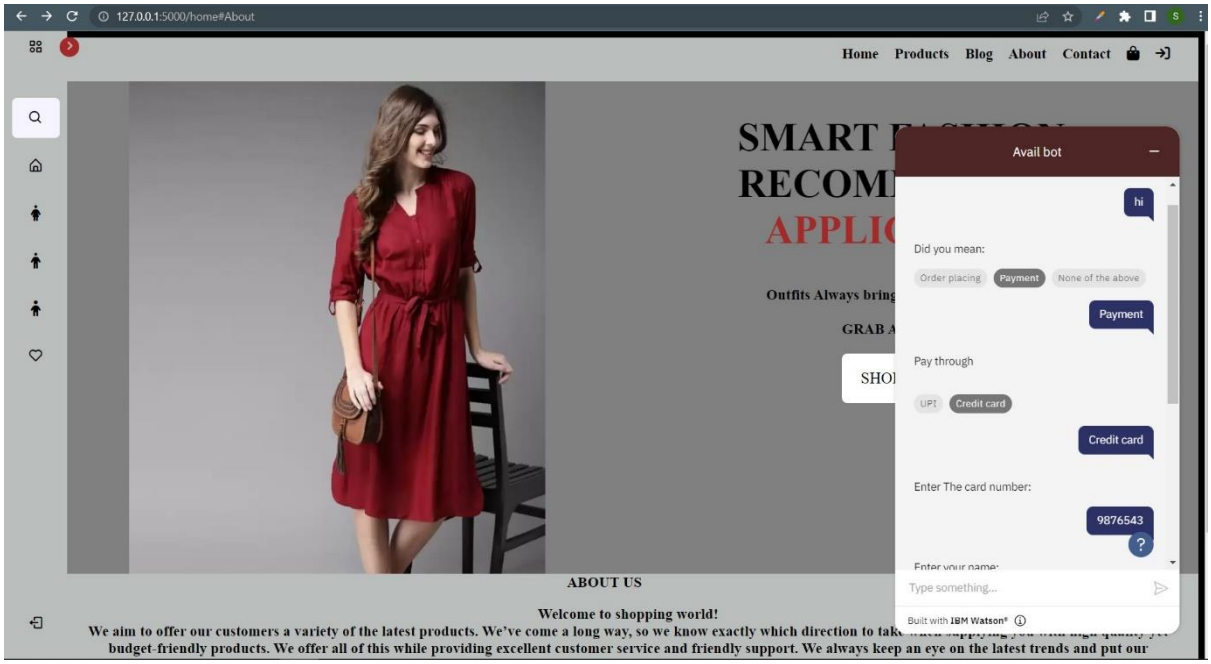
S.No	Project Overview	NFT Test approach	umptions/Dependencies/R	Approvals/SignOff
1	Creating a user friendly fashi	Manual testing	laptop or phone with inter	Priyadharshini S

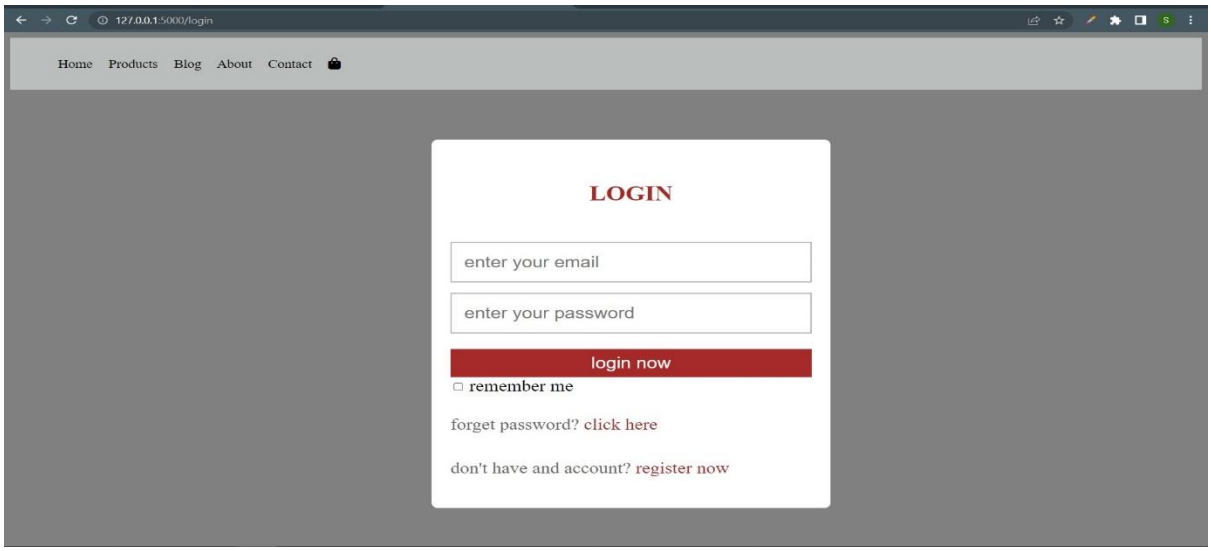
End Of Test Report

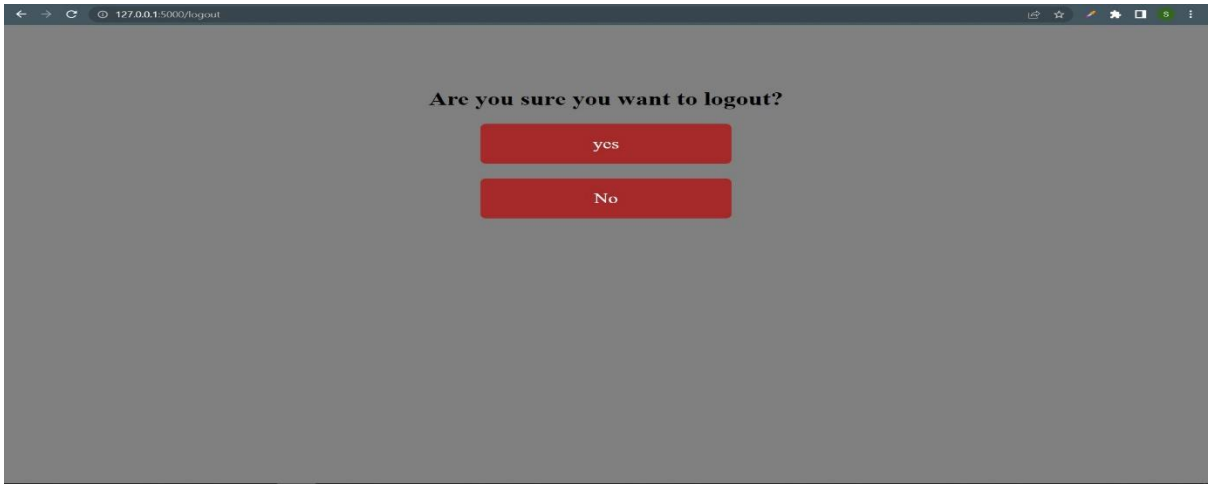
S.No	Project Overview	NFT Test approach	NFR - Met	Test Outcome	GO/NO-GO decision	Recommendations	Identified Defects (Detected/Closed/Open)	Approvals/SignOff
1	Fashion Recomm	Manual	Yes	An user friendly application f	GO	None	More product filters can be applie	Priyadharshini S

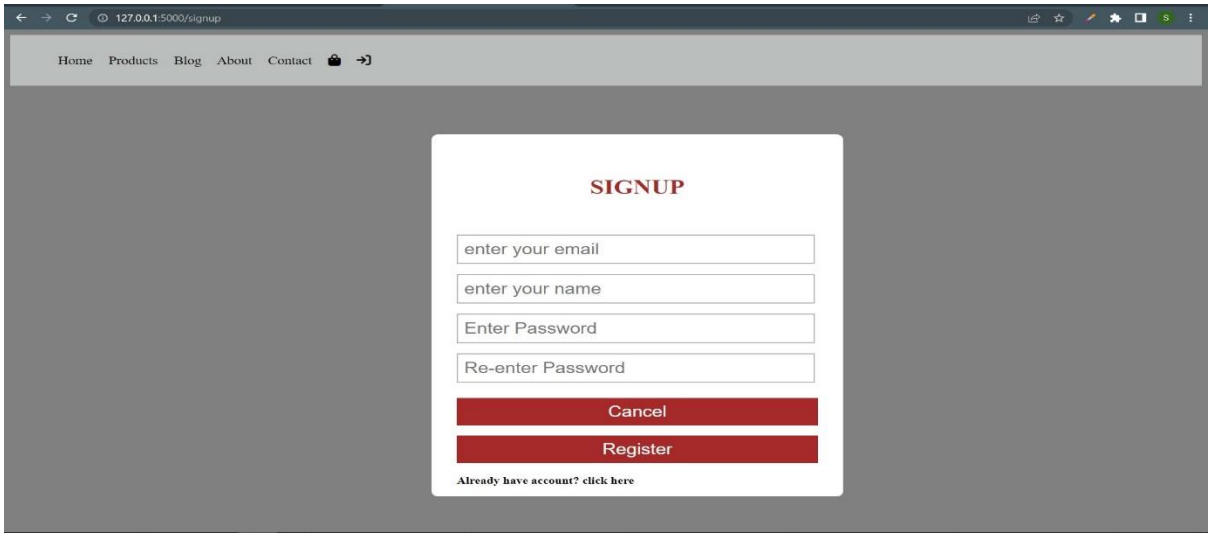
9.2 OUTPUT:

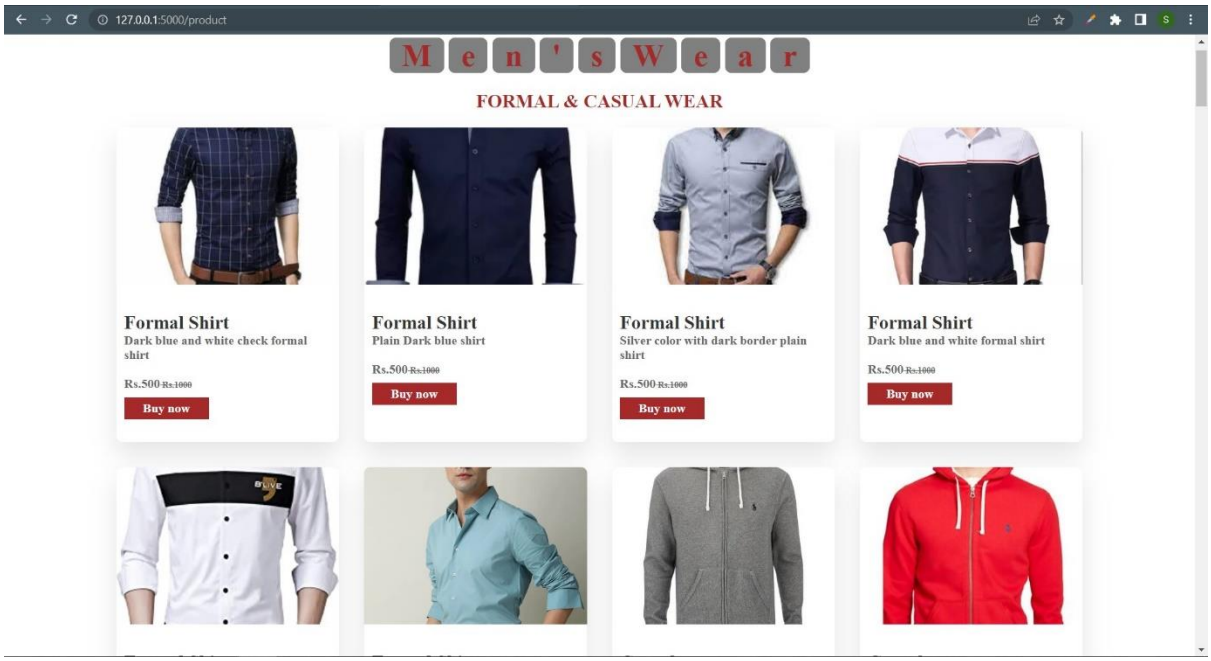


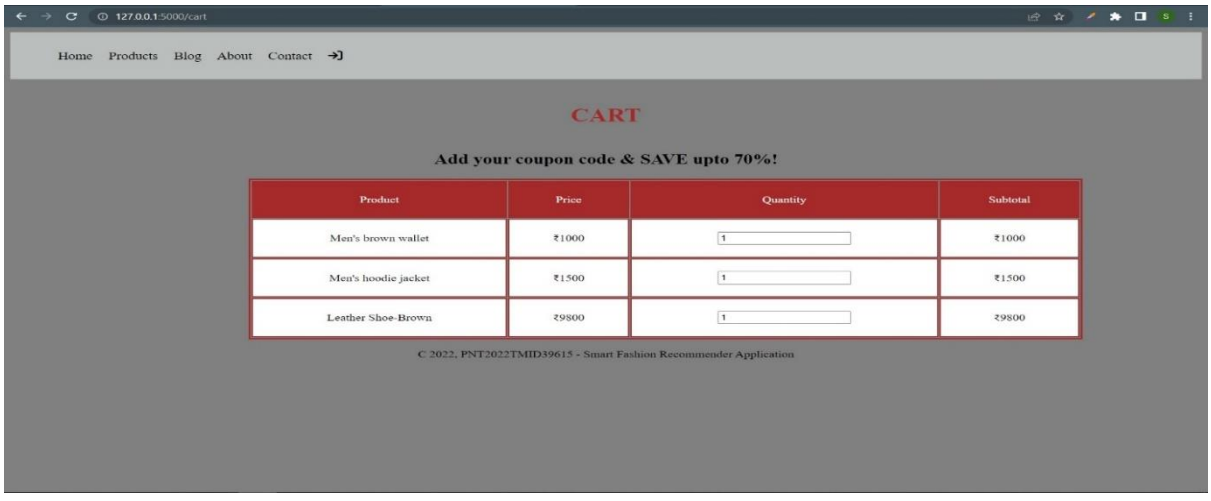

127.0.0.1:5000/home#About
Home Products Blog About Contact
SMART
RECOM
APPLI
Outfits Always bring
GRAB
SHO
Avail bot
hi
Did you mean:
Order placing
Payment
None of the above
Payment
Pay through
UPI
Credit card
Credit card
Enter The card number:
9876543
Enter your name:
Type something...
Built with IBM Watson®
ABOUT US
Welcome to shopping world!
We aim to offer our customers a variety of the latest products. We've come a long way, so we know exactly which direction to tak
budget-friendly products. We offer all of this while providing excellent customer service and friendly support. We always keep an eye on the latest trends and put our

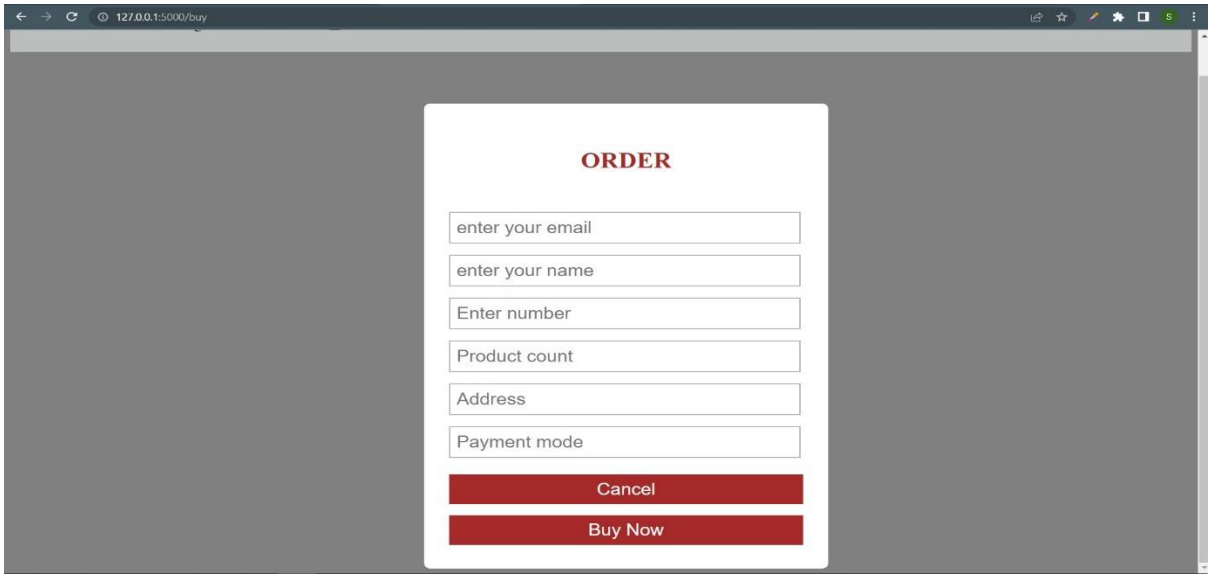

127.0.0.1:5000/login
Home Products Blog About Contact
LOGIN
enter your email
enter your password
login now
remember me
forget password? click here
don't have and account? register now

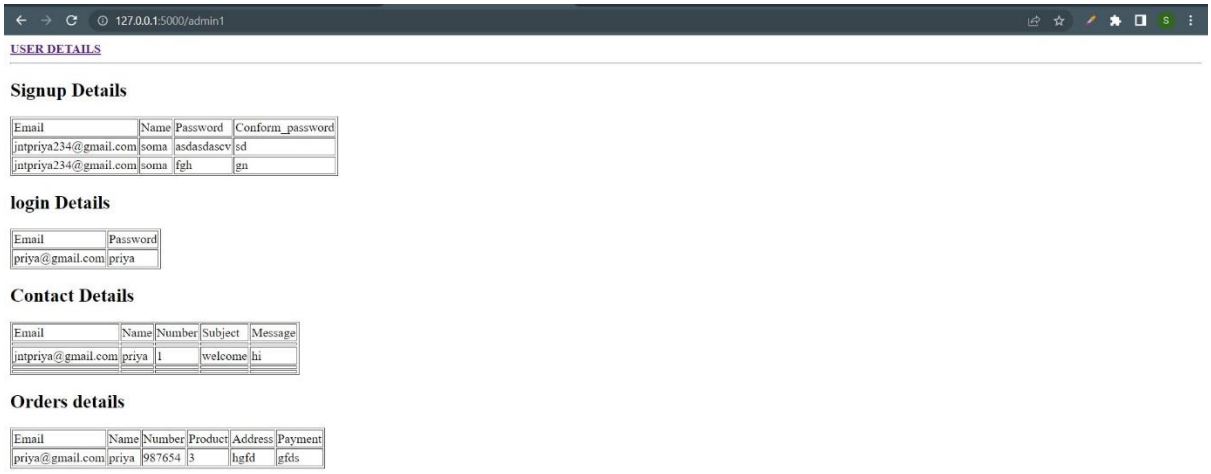

127.0.0.1:5000/logout
Are you sure you want to logout?
yes
No


127.0.0.1:5000/signup
Home
Products
Blog
About
Contact
SIGNUP
enter your email
enter your name
Enter Password
Re-enter Password
Cancel
Register
Already have account? click here


127.0.0.1:5000/product
M e n ' s W e a r
FORMAL & CASUAL WEAR
Formal Shirt
Dark blue and white check formal shirt
Rs.500 Rs:1000
Buy now
Formal Shirt
Plain Dark blue shirt
Rs.500 Rs:1000
Buy now
Formal Shirt
Silver color with dark border plain shirt
Rs.500 Rs:1000
Buy now
Formal Shirt
Dark blue and white formal shirt
Rs.500 Rs:1000
Buy now


127.0.0.1:5000/cart
Home
Products
Blog
About
Contact
CART
Add your coupon code & SAVE upto 70%!
Product
Price
Quantity
Subtotal
Men's brown wallet
₹1000
1
₹1000
Men's hoodie jacket
₹1500
1
₹1500
Leather Shoe-Brown
₹9800
1
₹9800
© 2022, PNT2022TMID39615 - Smart Fashion Recommender Application





---

10. ADVANTAGES AND DISADVANTAGES:

ADVANTAGES:

1) Easy recommendations make less search and sometimes end up un good deals

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

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

4) The model can capture the specific interests of a user, and can recommend niche items that very few other users are interested in.

DISADVANTAGES:

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

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

3) Chances are that some websites may suggest products wrongly based on analysis of little information gathered.

11. CONCLUSION:

Recommendation systems have the potential to explore new opportunities for retailers by enabling them to provide customized recommendations to consumers based on information retrieved from the Internet. They help consumers to instantly find the products and services that closely match with their choices. Moreover, different stat-of-the-art algorithms have been developed to recommend products based on users' interactions with their social groups. Therefore, research on embedding social media images within fashion recommendation systems has gained huge popularity in recent times. This paper presented a review of the fashion recommendation systems, algorithmic models and filtering techniques based on the academic articles related to this topic. The technical aspects, strengths and weaknesses of the filtering techniques have been discussed elaborately, which will help future researchers gain an in-depth understanding of fashion recommender systems. However, the proposed prototypes should be tested in commercial applications to understand their feasibility and accuracy in the retail market, because inaccurate recommendations can produce a negative impact on a customer. Moreover, future research should concentrate on including time series analysis and accurate categorization of product images based on the variation in color, trend and clothing style in order to develop an effective recommendation system. The proposed model will follow brand specific personalization campaigns and hence it will ensure highly curated and tailored offerings for users. Hence, this research will be highly beneficial for researchers interested in using augmented and virtual reality features to develop recommendation system.

12. FUTURE SCOPE:

There has been significant progress recently in fashion recommendation system research, which will benefit both consumers and retailers soon. The use of product and user images, textual content, demographic history, and cultural information is crucial in developing recommendation frameworks. Product attributes and clothing style matching are common features of collaborative and content-based filtering techniques. Researchers can develop more sophisticated hyper personalized filtering techniques considering the correlation between consumers' clothing styles and personalities. The methods based on employing a scoring system for quantifying each product attribute will be helpful in increasing the precision of the model. The use of virtual sales advisers in an online shopping portal would provide consumers with a real time offline shopping experience. Retailers can collect the data on users' purchase history and product reviews from the recommendation system and subsequently use them in style prediction for the upcoming seasons. The integration of different domain information strengthens the deep learning paradigm by enabling the detection of design component variation, which improves the performance of the recommendation system in the long run. Deep learning approaches should be more frequently used to quickly explore fashion items from different online databases to provide prompt recommendations to users or consumers.

13. APPENDIX:

SOURCE CODE:

index.html:

```
<html>

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Smart fashion recommender application</title>
<link href='https://unpkg.com/boxicons@2.0.7/css/boxicons.min.css'
rel='stylesheet'>
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/6.1.1/css/all.min.css"/>
<link rel="stylesheet" type="text/css" href="https://fashionapp.s3.jp-
tok.cloud-object-storage.appdomain.cloud/Style/style.css"/>
<script>
    window.watsonAssistantChatOptions = {
      integrationID: "18d8562f-cf49-4d58-859b-3fd02cf9bdc7", // The ID of this
integration.
      region: "jp-tok", // The region your integration is hosted in.
      serviceInstanceID: "a96cd1a0-8940-4a51-9668-99d54ef694b8", // 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 bgcolor="black">

<nav class="sidebar close">
        <header>

         <div class="image-text">
                <span class="image">
                    <i class='bx bx-category-alt icon'></i>
                </span>
                <div class="text logo-text">
                    <span class="name">Categories</span>

                </div>

             </div>
            <i class='bx bx-chevron-right toggle'></i>
        </header>

        <div class="menu-bar">
            <div class="menu">

                <li class="search-box">
                    <i class='bx bx-search icon'></i>
                    <input type="text" placeholder="Search...">
                </li>

                <ul class="menu-links">
                    <li class="nav-link">
                        <a href="/home">
                            <i class='bx bx-home-alt icon' ></i>
                            <span class="text nav-text">Home</span>
                        </a>
                    </li>

                    <li class="nav-link">
```

```
                    <a href="/Women">
                       <i class="fa-solid fa-person-dress icon"></i>
                        <span class="text nav-text">Women</span>
                    </a>
                </li>

                <li class="nav-link">
                    <a href="/Mens">
                        <i class="fa-solid fa-person icon"></i>
                        <span class="text nav-text">Men</span>
                    </a>
                </li>

                <li class="nav-link">
                    <a href="/Kids">
                        <i class="fa-solid fa-child icon"></i>
                        <span class="text nav-text">Kids</span>
                    </a>
                </li>

                <li class="nav-link">
                    <a href="#">
                        <i class='bx bx-heart icon' ></i>
                        <span class="text nav-text">Likes</span>
                    </a>
                </li>



            </ul>
        </div>

        <div class="bottom-content">
            <li class="">
                <a href="/logout">
                    <i class='bx bx-log-out icon' ></i>
                    <span class="text nav-text">Logout</span>
                </a>
            </li>



        </div>
    </div>

</nav>
<section>
<div style="background-color:#BABEBD" id="home">
```

```
    <b><a href="/home">Home</a>
        <a href="/product">Products</a>
        <a href="#blog">Blog</a>
        <a href="#About">About</a>
        <a href="/contact">Contact</a></b>
    <a href="/cart"><i class="fa fa-shopping-bag"></i></a>
    <a href="/login"><i class="fa-solid fa-arrow-right-to-bracket" id="login-
btn"></i></a>
</div>

<div id="home_1">
<img src="https://fashionapp.s3.jp-tok.cloud-object-
storage.appdomain.cloud/Images/Home.jpg"/>
</div>

<div id="home_2">
<h1>SMART FASHION RECOMMENDER <span
style="color:brown">APPLICATION</span></h1>
<h3>Outfits Always brings out our inner beauty<br><br>
GRAB AND GO!!!</h3>
<p id="shop">
<a href="/signup">SHOP NOW</a>
</p>
</div></section>
<section>
<div id="About" style="background-color:#BABEBD">
<center><br><h3 style="padding-top:40px">ABOUT US</h3>
<h3 style="padding-left:80px;padding-bottom:20px;padding-right:20px">Welcome
to shopping world!<br>

We aim to offer our customers a variety of the latest products. We've come a
long way, so we know exactly which direction to take when supplying you with
high quality yet budget-friendly products. We offer all of this while
providing excellent customer service and friendly support.

We always keep an eye on the latest trends and put our customers' wishes
first. That is why we have satisfied customers all over the world, and are
thrilled to be a part of the fashion industry.

The interests of our customers are always top priority for us, so we hope you
will enjoy our products as much as we enjoy making them available to
you.</h3></center>
</div>

<center>
                <div class="icon" style="color:white">
                    <b>Follow Us</b><br><br>
```

```
                    <i class="fa-brands fa-facebook"></i>
                    <i class="fa-brands fa-twitter"></i>
                    <i class="fa-brands fa-instagram"></i>
                    <i class="fa-brands fa-youtube"></i>
                </div> </center>


<div class="copyright"style="color:white">
<center><p>© 2022, PNT2022TMID39615 - Smart Fashion Recommender Application
</p></center>
</div>
</section>
<script>
      const body = document.querySelector('body'),
      sidebar = body.querySelector('nav'),
      toggle = body.querySelector(".toggle"),
      searchBtn = body.querySelector(".search-box");


toggle.addEventListener("click" , () =>{
    sidebar.classList.toggle("close");
})

searchBtn.addEventListener("click" , () =>{
    sidebar.classList.remove("close");
})

    </script>


</body>
</html>
```

app.py:

```
from flask import Flask, render_template, redirect, url_for, request
import sqlite3 as sql
app = Flask(__name__)
@app.route("/")
def hello():
    return redirect(url_for('home'))
@app.route("/home")
def home():
    return render_template('index.html')
@app.route("/signup")
def signup():
     return render_template('signup.html')
@app.route("/register",methods=['GET','POST'])
```

```
def register():
    if request.method=='POST':
        try:
            Email=request.form['email']
            Name=request.form['usrname']
            Password=request.form['psw']
            Conform_Password=request.form['psw-repeat']
            with sql.connect("Rec.db") as con:
                cur=con.cursor()
                cur.execute("INSERT INTO signup (Email, Name, Password,
Conform_Password) VALUES (?,?,?,?)",(Email, Name, Password, Conform_Password))
                con.commit()
        finally:
            return render_template('index.html')
    con.close()

@app.route("/logging",methods=['GET','POST'])
def logging():
    if request.method=='POST':
        try:
            Email=request.form['Email']
            Password=request.form['Password']
            with sql.connect("Rec.db") as con:
                cur=con.cursor()
                cur.execute("INSERT INTO login (Email, Password) VALUES
(?,?)",(Email, Password))
                con.commit()

        finally:
            return render_template('index.html')
    con.close()
@app.route("/contact1",methods=['GET','POST'])
def contact1():
    if request.method=='POST':
        try:
            Email=request.form['Email']
            Name=request.form['Name']
            Number=request.form['Number']
            Subject=request.form['Text']
            Message=request.form['Content']
            with sql.connect("Rec.db") as con:
                cur=con.cursor()
                cur.execute("INSERT INTO contact (Email, Name, Number,Subject,
Message) VALUES (?,?,?,?,?)",(Email, Name, Number,Subject, Message))
                con.commit()

        finally:
            return render_template("index.html")
```

```
    con.close()
@app.route("/order",methods=['GET','POST'])
def order():
    if request.method=='POST':
        try:
            Email=request.form['Email']
            Name=request.form['Name']
            Number=request.form['Number']
            Address=request.form['Address']
            Payment=request.form['Payment']
            Product=request.form['Product']
            with sql.connect("Rec.db") as con:
                cur=con.cursor()

                cur.execute("INSERT INTO Buy(Email, Name, Number, Address,
Payment, Product) VALUES (?,?,?,?,?,?)",(Email, Name, Number, Address,
Payment, Product))

                con.commit()

        finally:
            return render_template("index.html")

    con.close()
@app.route('/admin1')
def admin():
    con=sql.connect("Rec.db")
    con.row_factory=sql.Row
    cur=con.cursor()
    cur.execute("select * from signup")
    signup=cur.fetchall();
    cur.execute("select * from login")
    login=cur.fetchall();
    cur.execute("select * from contact")
    contact=cur.fetchall();
    cur.execute("select * from Buy")
    Buy=cur.fetchall();
    return
render_template("admin1.html",signup=signup,login=login,contact=contact,Buy=Bu
y)



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

@app.route("/about")
def about():
```

```
    return render_template('#about')

@app.route("/contact")
def contact():
    return render_template('contact.html')
@app.route("/cart")
def cart():
    return render_template('cart.html')
@app.route("/logout")
def logout():
    return render_template('logout.html')
@app.route("/loggedout")
def loggedout():
    return render_template('loggedout.html')
@app.route("/Women")
def Women():
    return render_template('Women.html')
@app.route("/Mens")
def Men():
    return render_template('Mens.html')
@app.route("/Kids")
def kids():
    return render_template('kids.html')
@app.route("/product")
def product():
    return render_template('product.html')
@app.route("/buy")
def buy():
    return render_template('Buy.html')


if __name__=='__main__':
    app.run(host="0.0.0.0")
```

admin1.html:

```
<!doctype html>
<html>
   <body>


   <a href="/admin1"><b>USER DETAILS</b></a>
    <br><hr>


    <h2>Signup Details</h2>
    <table border = 1>
```

```
    <thead>
       <td>Email</td>
       <td>Name</td>
       <td>Password</td>
       <td>Conform_password</td>
    </thead>

    {% for row in signup %}
       <tr>
          <td>{{row["Email"]}}</td>
          <td>{{row["Name"]}}</td>
          <td> {{ row["Password"]}}</td>
          <td>{{row["Conform_Password"]}}</td>
       </tr>
    {% endfor %}
 </table>
 <h2>login Details</h2>
  <table border = 1>
    <thead>
       <td>Email</td>
       <td>Password</td>
    </thead>

    {% for row in login %}
       <tr>
          <td>{{row["Email"]}}</td>

          <td> {{ row["Password"]}}</td>

       </tr>
    {% endfor %}
 </table>

 <h2>Contact Details</h2>
 <table border = 1>
   <thead>
      <td>Email</td>
      <td>Name</td>
      <td>Number</td>
      <td>Subject</td>
      <td>Message</td>
   </thead>

   {% for row in contact %}
      <tr>
         <td>{{row["Email"]}}</td>
         <td>{{row["Name"]}}</td>
         <td> {{ row["Number"]}}</td>
```

```
            <td>{{row["Subject"]}}</td>
            <td>{{row["Message"]}}</td>
        </tr>
    {% endfor %}
  </table>
  <h2>Orders details</h2>
   <table border = 1>
     <thead>
        <td>Email</td>
        <td>Name</td>
        <td>Number</td>
        <td>Product</td>
        <td>Address</td>
        <td>Payment</td>
     </thead>

     {% for row in Buy %}
        <tr>
           <td>{{row["Email"]}}</td>
           <td>{{row["Name"]}}</td>
           <td> {{ row["Number"]}}</td>
           <td>{{row["Product"]}}</td>
           <td>{{row["Address"]}}</td>
           <td>{{row["Payment"]}}</td>
        </tr>
     {% endfor %}
  </table>



         </body>
</html>
```

app1.py:

```
import sqlite3 as sql
conn=sql.connect('Rec.db')
print("Opened database successfully")
conn.execute('CREATE TABLE signup(Email TEXT, Name TEXT, Password TEXT, Conform_Password TEXT)')
conn.execute('CREATE TABLE login(Email TEXT, Password TEXT)')
conn.execute('CREATE TABLE contact(Name TEXT, Email TEXT, Subject TEXT, Number INTEGER, Message TEXT)')
conn.execute('CREATE TABLE Buy(Name TEXT, Email TEXT, Address TEXT, Number INTEGER, Payment TEXT, Product
INTEGER)')
print("table created successfully")
conn.close()
```

GITHUB AND PROJECT DEMO LINK:

GITHUB LINK: https://github.com/IBM-EPBL/IBM-Project-17977-1659677649

PROJECT DEMO LINK: **https://youtu.be/DnTZ687GjyQ**