Turnitin Originality Report

Panda by Asjad Iftikhar

From General1234 (General1234)

- Processed on 30-May-2022 14:41 PKT
- ID: 1847096533
- Word Count: 10656

Similarity Index
16%
Similarity by Source

Internet Sources:
3%
Publications:
1%
Student Papers:
16%

sources:

1 6% match (student papers from 03-Jun-2020)
Submitted to Higher Education Commission Pakistan on 2020-06-03

2 2% match (student papers from 03-Jun-2020)
Submitted to Higher Education Commission Pakistan on 2020-06-03

3 1% match (student papers from 14-Oct-2021)
Submitted to Chester College of Higher Education on 2021-10-14

4 1% match (student papers from 30-Mar-2022)
Submitted to King's College on 2022-03-30

5 < 1% match (student papers from 18-Dec-2014)
Submitted to Higher Education Commission Pakistan on 2014-12-18

6 < 1% match (student papers from 06-Jun-2014)
Submitted to Higher Education Commission Pakistan on 2014-06-06

7 < 1% match (student papers from 10-Nov-2012)
Submitted to Higher Education Commission Pakistan on 2012-11-10

8 < 1% match (student papers from 03-Jun-2020)
Submitted to Higher Education Commission Pakistan on 2020-06-03

< 1% match (student papers from 20-Mar-2018)

9 Submitted to Higher Education Commission Pakistan on 2018-03-20

10 < 1% match (student papers from 26-May-2016)
Submitted to Higher Education Commission Pakistan on 2016-05-26

11 < 1% match (student papers from 09-Aug-2012)
Submitted to Higher Education Commission Pakistan on 2012-08-09

12 < 1% match (student papers from 13-Nov-2014)
Submitted to Higher Education Commission Pakistan on 2014-11-13

13 < 1% match (student papers from 13-Jan-2021)
Submitted to De Montfort University on 2021-01-13

14 < 1% match (student papers from 10-Nov-2017)
Submitted to The University of the South Pacific on 2017-11-10

15 < 1% match (Internet from 04-Jun-2021)
https://assignmentfreelancers.com/software-design-specification/

16 < 1% match (student papers from 04-May-2022)
Submitted to University of Teesside on 2022-05-04

17 < 1% match (student papers from 25-Apr-2019)
Submitted to Universiti Tunku Abdul Rahman on 2019-04-25

18 < 1% match (student papers from 29-Mar-2022)
Submitted to Purdue University on 2022-03-29

19 < 1% match (student papers from 07-May-2021)
Submitted to University of Wolverhampton on 2021-05-07

20 < 1% match (student papers from 10-Jun-2019)
Submitted to Asia Pacific Instutute of Information Technology on 2019-06-10

21 < 1% match (student papers from 02-Jul-2020)
Submitted to Universiti Teknologi MARA on 2020-07-02

22 < 1% match (Internet from 23-Jan-2022)
https://webthesis.biblio.polito.it/21367/1/tesi.pdf

23 < 1% match (student papers from 24-May-2018)
Submitted to Sheffield Hallam University on 2018-05-24

24 < 1% match (student papers from 08-Sep-2014)
Submitted to University College London on 2014-09-08

25 < 1% match (Internet from 04-Jun-2019)
https://hci.ur.de/_media/news/raphael_wimmer_-_keeping_the_user_in_control_dresden_2018-01-10_slides_small.pdf

26 < 1% match (Internet from 13-Nov-2021)
https://www.coursehero.com/file/96695599/Letter-Taskpdf/

27 < 1% match (Internet from 19-Jan-2013)
http://www.viv-media.com/info-7714-Men-s-t-shirts-Men-s-polo-shirts-Men-s-trousers-Glory-International-Development-Ltd-.html

28 < 1% match (publications)
Guangming Xing. "Support vector machines for collaborative filtering", Proceedings of the 44th annual southeast regional conference on - ACM-SE 44 ACM-SE 44, 2006

29 < 1% match (Internet from 17-Feb-2022)
http://docplayer.net/52201055-Running-in-the-family-a-new-era-of-modular-application-ready-systems.html

30 < 1% match (Internet from 17-Oct-2020)
https://www.mdpi.com/2073-8994/8/12/138/htm

31 < 1% match (Internet from 25-Jan-2022)
https://123dok.com/article/use-case-diagram-case-narrative-perancangan-sistem-informasi.ye18vxez

32 < 1% match (publications)
Frank McCarey, Mel Cinneide, Nicholas Kushmerick. "A Recommender Agent for Software Libraries: An Evaluation of Memory-Based and Model-Based Collaborative Filtering", 2006 IEEE/WIC/ACM International Conference on Intelligent Agent Technology, 2006

33 < 1% match (publications)
iOS eCommerce App Development with Parse, 2015.

34 < 1% match (Internet from 03-Mar-2020)
https://link.springer.com/article/10.1007/s10009-015-0365-2

paper text:

Introduction During the COVID-19 pandemic online apparel shopping stores, in Pakistan, have grown exponentially in number. One major challenge for customers is that at any given time there are hundreds of stores to buy from, which makes shopping a tedious task. With advancement in machine learning and data science, an obvious solution is a recommendation system which assist the buyers in decision making, similar to how Netflix [5] provides a recommendation system for its subscribers to choose the best movies. The goal of this project is to develop an automated system that can scrape and collect data on clothing products, then optimize and analyze it using Artificial Intelligence and Data Science to make recommendations based on user preferences and purchase history. Our primary focus will be on gathering data by registration or automatic scraping, processing the data, and displaying the recommendations in a user-friendly online interface. We will also keep track of each user's previous purchases in order to improve over time. Data about brands and products will be obtained from: i. Websites of brands directly. ii. Registration at Panda Mall The results obtained from the model will be presented in a simplified web interface along with all the necessary information for customers to make a buying decision with ease. This document primarily covers how the system works and the concepts that were employed in its development. The work done in this scope is explained in the following chapter. Furthermore, in Chapter 3, all of our system's requirements, both functional and non-functional are described, as well as the system architecture, strategies and use cases. Chapter 4 covers the implementation and operation of our system. Finally, chapter 5 explains the overview, the obstacles that were encountered, and the breakdown of future work. Goals and Objectives The primary goals and objectives of this project are: - To design a user-friendly website that helps in recommending the best clothing products for the user. - To provide a platform where items of multiple stores will be listed for recommendation. - To use Machine Learning (ML) models such as collaborative filtering to produce efficient and effective recommendation engine. Scope of the Project The project will be implemented in three components. In first component, brands and product data will be collected from both automated web scrappers and a manual store registration process at Panda Mall website. After data analysis, it will be exported to the second component. Second component will use this data to train a machine learning model using a custom-built variation of collaborative filtering algorithm. Introduction Third component will consist of creating a website module for the users. User can register on the website. With the account user can search a product of their interest. The items can be filtered with the various options size, color and preferred brand etc. Items displayed on the result query will filter through the options before it is fed as input to the model trained in component two, the output will contain a filtered list of the best recommended products for the particular user. After the purchase of product, users will be reminded to fill out the feedback against the purchase. This feedback will be permanently recorded for future recommendations. The Scope of project doesn't include a payment gateway or a shopping a cart. However, as a point of future scope adding a shopping cart with payment and logistics could be a potential work to extend the scope of this project. Literature Survey / Related Work Functional Tensor Factorization Hu et al. [2] researched on personalized clothing recommendation systems. He presented a functional tensor factorization approach to describe user-item and item-item interaction. Collaborative Filtering Nogueira et al. [3] proposed a new collaborative filtering algorithm for better accuracy in clothing recommendation systems. After a lot of research, we have inclined towards collaborative filtering method for our recommendation system as we have found it more effective and accurate in these types of recommendation systems. Landia in [4] explains challenges faced during construction of fashion recommendation system. He has organized challenges into two categories namely retailer related and customer related. The prior consists of short lifetime of items and high volume of items, whereas seasonality and rapidly changing customer preferences make up some of the customer related challenges. Majority of the recommendation systems deal with products from a single clothing store unlike our system which will deal with different type of clothes from different clothing stores. Requirements and Design Functional Requirements 3.1.1

3**Functional requirements for** Users ? **System will allow** user **to** login by authenticating **user** login credentials. ? **System will allow** user **to**

create a new account by providing required information. ?

3**System will allow user to reset password via email. ? System will allow user to**

logout. ?

3**System will allow** user **to edit profile. ? System will**

display clothing products based on user’s profile. ?

13**System will allow user to filter** out **products** based on Type. ? **System will allow user to** filter out **products** based **on**

Sizes. ?

13**System will allow user to filter** out **products** based on Colors. ? **System will allow user to** filter out **products** based **on**

Waists. ? System

23**will allow user to filter** out **products based on**

Price Range. ? System

23**will allow user to filter** out **products based on**

Discount factor. ?

13**System will allow user to filter** out **products** based on Brands. ? **System will allow user to**

view selected product details. ? System will allow user to buy product by redirecting to respected page. ? System will allow user to add products to favorites. ? System will allow user to give feedback on the product bought. 3.1.2 ? ? ? Functional requirements for Administrators Administrators can approve/reject products. Administrators can update products. Administrators can view products. 3.1.3 Functional requirements for Stores ? Stores can add products. ? Stores can update products. ? Stores can view products. ? Stores can remove products. Non-Functional Requirements 3.2.1 Availability System will be available for the users for at-least 160 hours a week. 3.2.2 Usability System will have an intuitive design. ? Our system interface will be easy to learn and user interactive. A rookie will require at- most 5 seconds to learn the interface. 3.2.3 Reliability System will show the result of the desired filter within 3 seconds. 3.2.4 Scalability System will ensure there will be no significant performance degrade for at-most 100 users. 3.2.5 Data Integrity System will ensure user's personal data is not tempered by unauthorized source. 3.2.6 Performance All data views are paginated to allow limited and required information to prevent long load times.

29**Hardware and Software Requirements 3.3.1 ? Hardware Requirement**

A IOT device that supports a browser. ? Internet connection should have at least 1Mbps. ? A web server to host our website. 3.3.2 Software Requirement Web browser in the system.

30**System Architecture Figure 1: High Level System Architecture** This is **System**

Architecture Diagram for our system. As show in the diagram above we will be using 3-layer architecture for our system. Our users (Customer, Vendor and Admin) will interact with the presentation layer that will forward user requests to the server which will be acting as the business layer. It will process all the user request and send them back to the presentation layer. If needed it will interact with our cloud- based database that corresponds to the database layer to retrieve any kind of information needed. 3.4.1 Front-end Our front-end has the following components: 3.4.1.1 Customer Customer is the user that logs in to our system using his account and interacts with the application to buy products. Each customer request goes from the presentation layer to the server where the request is processed and the required data is fetched from the database layer and then the result in forwarded back to the presentation layer where the results are shown to the customer. Customer can perform following requests/functions: ? Login ? Sign up ? Reset password ? Logout ? Edit Profile ? Buy product ? Add products to favorites ? Give feedback on products ? Filter products based on several filters 3.4.1.2 Vendor Vendor is the user which creates a store and then adds his/her products to our website. Each vendor request goes from the presentation layer to the server where the request is processed and the given data is stored to the database layer. Vendor can perform following requests/functions ? Add products ? Update products ? View products ? Remove products 3.4.1.3 Admin Admin is the user of the system which approves the newly added products. If he approved a product then the product is added to the database otherwise the product is discarded. ? Approve/Reject products ? View products 3.4.2 Back-end Our Back-end has the following components: 3.4.2.1 Express Server Our server connects our database layer to the presentation layer. It processes all the requests coming from the users (presentation layer) in HTML format. After processing the request, it fetches the required information from the database layer in the form of JSON objects. After that the server responds back to the presentation layer by sending the requested information. 3.4.2.2 MongoDB This is our database where all the information about system users, products is stored in JSON format. Our database is hosted on the cloud so as a result it is scalable and efficient. It assists our server by providing required information needed to respond to user requests. 3.4.3 Subsystem Architecture There is no such component in our system architecture section that merits a detailed discussion Architectural Strategies 3.5.1 ReactJS, Node JS, Python, Express Server, Mongo DB When it comes to machine learning there is no better option than Python. As it provides different types of libraries to manipulate and perform different functions on large amount of data. We will be using python with Jupiter Notebook. Our system's front end will be designed using ReactJS. It provides us with JavaScript libraries that helps us to perform our work. Major benefit of using ReactJS is its reusable components that allows us to use same components for different pages wherever needed. Moreover, 22**ReactJS enables significant data changes that result in automatic alteration in the selected parts of user interfaces**

. We will be using NodeJS with python as our backend languages. Every other aspect expects for manipulating datasets will be handled by NodeJS. The reason for using NodeJS is its compatibility with ReactJS which we will be using as our frontend. For our server we have decided to use Express Server. As we are using JavaScript for our frontend and backend so why not using it to implement the server. Major benefit of using this server is its ability to handle several requests efficiently. Moreover, it also has a highly supportive open-source community. For our database we will be using MongoDB because of its scalability benefits. It has a very simple design with all the data stored as a single JSON object hence making key functions like saving and loading data relatively easier. 3.5.2 Future plans for enhancing software For this project we are limiting our scope to a web application but in the future, we are planning to scale our project by: ? We are planning to implement a cart system that we will integrate with the payment procedure to help our users to buy product directly from our website. ? We plan to develop a mobile application with all the features of our web application in order to assist the user. 3.5.3 User interface paradigms While making our UI we will implement the eight golden rules of user interaction in our Software Engineering course. Following are those

25**rules: 1. Strive for consistency. 2. Seek universal usability. 3. Offer informative feedback. 4. Design dialogs**

.

18**5. Prevent errors. 6. Permit easy reversal of actions. 7. Keep users in control. 8. Reduce short term memory load. This**

will help our UI to be able to yield better performance and efficiency. It will also make our UI user friendly. 3.5.4 Error detection and recovery In our situation, the most typical error occurs during the authentication process, when a user enters incorrect credentials, which our system authenticates and, if they are invalid, displays an error message to the user. Furthermore, if a user inputs incorrect information during registration or profile modification, our system will display an error message requiring the user to reenter the incorrect information. If a user attempts to purchase a product that the vendor has rendered unavailable, our system display popup menu to let them know of the product's unavailability. Furthermore, we will update the product list once a week to ensure that users have access to the most up-to-date products. 3.5.5 Concurrency and Synchronization Hopefully a lot of users will be accessing our website and as result making lot of simultaneous requests. So, we need to make our system synchronized so that we can treat each request independently. In order to do that we have to implement threading into our business logic. Moreover, with multithreading in place we will have to ensure synchronization of the entire system for users to avoid unexpected behaviors in the system. 3.5.6 Memory management policies As already stated above we will be using MongoDB which in an online database running in real-time on cloud. So, in order to access that DB our system users should be connected to the internet. Use Cases 3.6.1

3**Login Account Name Login Account Actors** Client, **Admin**, Store Owner **Summary** User **will be able to login**

into account. Pre- Conditions User should be registered in the system. User should

1**not already be logged in. Post- Conditions User**

shall be

3**logged in** successfully. **Special Requirements None Basic Flow Actor Action System Response 1 User** opens **the login page. 2** System displays **Login page asking for** username **and password. 3 User** enters valid login **credentials and** press login button. **4 System** verifies **the** user **credentials, establishes session** and redirects **the user** to **the home page. Alternative Flow**

14**Actor Action System Response** 3 **User** enters invalid **login** credentials 4-**A System** prompts the **error message**: and presses **the login**

button. Incorrect username or password entered. 3.6.2 Create Account Name Create Account Actors Client Summary User will be able to create a new account by providing the required information. Pre- Conditions User clicked on Sign-up button from the login page. Post- Conditions User's account shall be created successfully.

1**Special Requirements None Basic Flow Actor Action System Response 1 User clicks on the** Sign-up **button** from **the**

3**Login page. 2 System** displays Sign-up **page asking for** required information. **3 User** enters **valid** information **and**

presses submit button.

11**4 System verifies the information, creates a new account**

and redirects

3**the user to the login page. Alternative Flow Actor Action System Response 3 User** enters **invalid**

information and presses submit button. 4-A System prompts the error message: Incorrect/Missing required information 3.6.3 Reset

1**Password Name** Reset **Password Actors** Client **Summary User** will **be able to** reset **his/her** account **password. Pre- Conditions User**

should have an existing account. User clicked on “Forget Password?” button from Login page. Post- Conditions User shall receive reset password request on his/her email.

1**Special Requirements None Basic Flow Actor Action System Response 1 User clicks on the** “Forget **Password?” button** from **the**

Login page. 2 System displays a prompt asking for user’s username. 3 User enters username and presses submit button. 4 System verifies the username, sends an email to the respective user account and redirects the

3**user to the login page. Alternative Flow Actor Action System Response 3 User** enters **invalid** username and **4-A** System prompts **the**

error message: presses submit button. Username not found 3.6.4 Logout Name Logout Actors Client Summary User will

1**be able to** logout **his/her** account. **Pre- Conditions User** should have **logged in. User clicked** on “Logout” **button**

from Home page. Post- Conditions User shall be logged out.

1**Special Requirements None Basic Flow Actor Action System Response 1 User clicks on the** “Logout?” **button 2** System will logout **the user** and redirect from **the**

home page. the user to Login page. No Alternative Flow 3.6.5 Edit Profile Name Edit Profile Actors Client Summary User will

1**be able to** edit **his/her** profile. **Pre- Conditions User** should have **logged in. User clicked** on “**Edit** Profile” **button** from Home page. **Post- Conditions User’s** profile shall be **updated**

.

1**Special Requirements None Basic Flow Actor Action System Response 1 User clicks on the “Edit** Profile” **button** from **the**

Home page. 2 System displays a new page containing user’s current information. 3 User edits his/her profile by providing valid information and presses update button. 4 System validates the information, updates his/her profile and redirects the

3**user to the** Home **page. Alternative Flow Actor Action System Response 3 User** enters **invalid** profile **4-A** System prompts **the**

error message: information and presses update Information provided is incorrect. button. 3.6.6

20**View Product Details Name View Product Details** Actors Client Summary **User** will **be able to view the details of the selected product**

. Pre- Conditions User should have logged in. User should have clicked on a “Details” button. Post- Conditions Users shall be to see the product details.

1**Special Requirements None Basic Flow Actor Action System Response 1 User clicks on the** “Details” **button 2**

System displays a pop-up containing from the Home page. product details. No Alternative Flow 3.6.7 Buy Product Name Buy a Product Actors Client Summary User will be able to buy a product from the respected brand’s website.

1**Pre- Conditions User** must **be logged in. User must have clicked** on “Detail” **button** for **the**

selected product. Post- Conditions System will redirect user to the respective brand website.

1**Special Requirements None Basic Flow Actor Action System Response 1 User clicks on the** “Buy” **button** located in **the** product details. 2 System redirects the user to the respected link of the product. No Alternative Flow 3.7.1 Add to Favorites Name Add to favorites Actors Client Summary User will

33**be able to add** a **product to** his/her **favorites list**

. Pre- Conditions User must be logged in. Selected product should not be in the favorites list Post- Conditions System will update the favorites list by adding a product.

1**Special Requirements None Basic Flow Actor Action System Response 1 User clicks** on **the** “Favorites” icon **from** the selected product. **2** System adds **the** product **to**

his/her favorites list. No Alternative Flow 3.7.2 Remove from favorites Name Remove from favorites Actors Client Summary User will be able to remove a product from his/her favorites list. Pre- Conditions User must be logged in. Selected product should be in the favorites list Post- Conditions System will update the favorites list by removing a product.

1**Special Requirements None Basic Flow Actor Action System Response 1 User clicks** on **the** “Favorites” icon **from** the selected product. **2** System will remove **the**

product from his/her favorites list. No Alternative Flow 3.7.3 View favorites list Name View favorites list Actors Client Summary User will be able to view favorites list.

1**Pre- Conditions User must be logged in. Post- Conditions User shall be** able **to**

view his/her favorites list.

1**Special Requirements None Basic Flow Actor Action System Response 1 User clicks** on **the** “View favorites” button **from** the home page. **2** System redirects **the user to**

the new page containing a list of user’s favorite items. If there are no products in the favorites list then an empty list will be displayed. No Alternative Flow 3.7.4 Filter Product by Type Name Filter product by type Actors Client Summary User will be able to filter out products based on product type.

1**Pre- Conditions User must be logged in. Post- Conditions User will be** able to view filtered product list. Special Requirements Reliability (filtering takes less than 3 seconds).

1**Basic Flow Actor Action System Response 1 User clicks on the** “Filter” **button** from **the**

home page. 2 System displays a dropdown menu containing different types of filters. 3 User selects the “Type” filter from the dropdown menu. 4 System displays a pop-up of types for clothing products e.g. (

27**Men’s T- Shirts, Men’s Polo Shirts, Men’s Trousers**

). 5 User selects a type from the pop-up window. 6 System reloads the home page and displays the products based on the user’s filtering criteria. No Alternative Flow 3.7.5 Filter Product by Size Name Filter product by size Actors Client Summary User will be able to filter out products based on product sizes.

1**Pre- Conditions User must be logged in. Post- Conditions User will be**

able to view filtered product list. Special Requirements Reliability (filtering takes less than 3 seconds).

1**Basic Flow Actor Action System Response 1 User clicks on the** “Filter” **button** from **the**

home page. 2 System displays a dropdown menu containing different types of filters. 3 User selects the “Size” filter from the dropdown menu. 4 System displays a pop-up of sizes for clothing products e.g. (Small, Large). 5 User selects a size from the pop-up window. 6 System reloads the home page and displays the products based on the user’s filtering criteria. No Alternative Flow 3.7.6 Filter Product by Color Name Filter product by color Actors Client Summary User will be able to filter out products based on product color.

1**Pre- Conditions User must be logged in. Post- Conditions User will be**

able to view filtered product list. Special Requirements Reliability (filtering takes less than 3 seconds).

1**Basic Flow Actor Action System Response 1 User clicks on the** “Filter” **button** from **the**

home page. 2 System displays a dropdown menu containing different types of filters. 3 User selects the “Color” filter from the dropdown menu. 4 System displays a pop-up of colors for clothing products e.g. (Red, Green). 5 User selects a color from the pop-up window. 6 System reloads the home page and displays the products based on the user’s filtering criteria. No Alternative Flow 3.7.7 Filter Product by Waist Name Filter product by waist Actors Client Summary User will be able to filter out products based on product waist.

1**Pre- Conditions User must be logged in. Post- Conditions User will be**

able to view filtered product list. Special Requirements Reliability (filtering takes less than 3 seconds).

1**Basic Flow Actor Action System Response 1 User clicks on the** “Filter” **button**

from 2 System displays a dropdown menu the home page. containing different types of filters. 3 User selects the “Waist” filter from the dropdown menu. 4 System displays a pop-up of waists for clothing products e.g. (30-inch, 32- inch). 5 User selects a waist from the pop-up window. 6 System reloads the home page and displays the products based on the user’s filtering criteria. No Alternative Flow 3.7.8 Filter Product by Price Range Name Filter product by price range. Actors Client Summary User will be able to filter out products based on product price range.

1**Pre- Conditions User must be logged in. Post- Conditions User will be**

able to view filtered product list. Special Requirement s Reliability (filtering takes less than 3 seconds).

1**Basic Flow Actor Action System Response 1 User clicks on the** “Filter” **button** from **the**

home page. 2 System displays a dropdown menu containing different types of filters. 3 User selects the “Price” filter from the dropdown menu. 4 System displays a pop-up of price range for clothing products e.g. (Rs. 1200- 1800, Rs. 750-1000). 5 User selects a price range from the pop-up window. 6 System reloads the home page and displays the products based on the user’s filtering criteria. No Alternative Flow 3.7.9 Filter Product by Discount Factor Name Filter product by discount factor Actors Client Summary User will be able to filter out products based on product discount factor.

1**Pre- Conditions User must be logged in. Post- Conditions User will be**

able to view filtered product list. Special Requirement s Reliability (filtering takes less than 3 seconds).

1**Basic Flow Actor Action System Response 1 User clicks on the** “Filter” **button** from **the**

home page. 2 System displays a dropdown menu containing different types of filters. 3 User selects the “Discounted” filter from the dropdown menu. 4 System displays a pop-up of Discount range for clothing products e.g. (20% - 30% off). 5 User selects a Discount range from the pop-up window. 6 System reloads the home page and displays the products based on the user’s filtering criteria. No Alternative Flow 3.7.10 Approve Products Name Approve product Actors Admin Summary Admin will be able to see the details of a product and approve it to be published on the website Pre- Conditions Admin should be logged in the system and also on products list view page in the manage products section Post- Conditions Product is approved or rejected. Special Requirement s There should be an approval request pending. Basic Flow Actor Action System Response 1 Admin click on a store in manage products section. 2 Redirect the admin to a page showing all products of the store. Pending approval ordered on top. 3 Click on a particular product 4 Display all the attributes of the product. 5 Admin click on approve button at the bottom 6 System will approve the product and publish on the website and notify the store. 7 Admin click exit button. 8 Display list of products. Alternative Flow Actor Action System Response 5 Admin click on reject button 6-A System will not publish the product on alongside the approve button. the website and notify the store. 3.7.11 Suspend Product Name Suspend product Actors Admin Summary Admin will be able to view and terminate a published product. Pre- Conditions Admin should be logged in the system and also on products list view page in the manage products section. Post- Conditions System will immediately suspend the product from the website. Special Product must already be published Requirement s Basic Flow Actor Action System Response 1 Admin click on a store in manage products section. 2 Redirect the admin to a page showing all products of the store. Pending approval ordered on top. 3 Click on a particular product 4 Display all the attributes of the product. 5 Click on “terminate” button at the bottom. 6 System suspends the product and exit product view page. No Alternative Flow 3.7.12 View Product Name View products Actors Admin Summary Admin will be able to see the attribute of a product Pre- Conditions Admin should be logged in the system and also on manage products section. Post- Conditions System will display the attributes and values of the product.

1**Special Requirements None Basic Flow Actor Action System Response 1** Admin **click on** store entry in **the**

table ordered based on approval requests. 2 System will display the list of products sold by the store. 3 Admin click on a particular product from the table. 4 System will display the attributes and values of the product. No Alternative Flow 3.7.13 View home page Name View home page Actors Stores Summary User will be able to see all the products of their store.

1**Pre- Conditions User** should **be logged in. Post- Conditions User**

successfully view all of their products

1**Special Requirements None Basic Flow Actor Action System Response 1** Store **user**

logins successfully

17**2 System** loads **the** home **page with a** list **view** of **all**

products Alternative Flow

171 User clicks on the home from the navigation bar. 2-A System loads the home page with a list view of all

products 3.7.14 Add product Name Add product Actors Stores Summary User

3will be able to add products to their stores. Pre- Conditions User should be on logged in. Post- Conditions

User added product to store successfully.

1Special Requirements None Basic Flow Actor Action System Response 1 User clicks on Add product. 2

Create new product page is displayed, showing product detail, product description, product category, gallery and attachments needed to be filled. 3 User enters the details and press add button. 4 System checks the necessary fields needed to be filled and create that product and set “pending approval” for the product. System prompts the success message: Product sent for approval.

14Redirects the user back to home page. Alternative Flow Actor Action System Response 3 User

miss important fields and press 4-A System highlighted the required fields in save. red and prompts the error message: Please fill required fields! 3.7.15 Delete a product Name Delete a product Actors Store Summary User will be able to delete a product from store.

1Pre-Conditions User should be on logged in. Post- Conditions User

deleted a product successfully.

1Special Requirements None Basic Flow Actor Action System Response 1 User select a product(s) from the home page. 2 System records the selections. 3 User

clicks on the delete button. 4 System pops a small window with a warning. 5 User selects, “I’m sure” button. 6 System deletes the selected products and prompts the successful message: Products removed Successfully. Alternative Flow Actor Action System Response 5 User clicks “No” button. 6-A System returns to the home page. 3.7.16 Update product Name Update product Actors Store Summary User will be able to update the attribute values of a product from store like price or availability. Pre-Conditions User should be on logged in. Post- User updated a product attribute value successfully. Conditions Special None Requirements

3**Basic Flow Actor Action System Response 1 User select** a product **from** the home page. **2 System**

records the selection. 3 User clicks on the update button. 4 System displays an editable product form with values pre-filled. 5 User enters the details and press update button. 6 System checks the necessary fields needed to be filled and create that product and set “pending approval” for the product. System prompts the success message: Product sent for approval.

14**Redirects the user** back **to home page. Alternative Flow Actor Action System Response 5 User**

miss important fields and press 6-A System highlighted the required fields in save. red and prompts the error message: Please fill required fields! 5 User clicks ‘cancel’ 6-A Redirects the user back to home page. 3.7.17 Give Feedback Name Give feedback Actors Client Summary User will be able to give feedback on products bought. Pre- Conditions User should be logged in. User should have bought the product. User should be on the product details page. Post- Conditions User’s feedback will be stored. Special Requirement s None

31**Basic Flow Actor Action System Response 1 User**

will give feedback for the bought product by using star ratings. 2 System will store user’s feedback in the database. No Alternative Flow GUI 3.8.1 Admin panel This GUI shows login page for an admin. Figure 2: Login Screen Admin panel login screen. This GUI shows the “Manage Products” section of the admin panel. Figure 3: Manage Products Manage Products Section This GUI shows the edit page where an admin can approve, reject or suspend a product. Figure 4: Update product page Update product page at admin panel 3.8.2 Store GUI This GUI shows the home page of a store. Figure 5: Store home page Store home page This GUI shows the delete pop up window before a product is deleted. Figure 6: Pop up window Pop up window when delete button clicked This GUI shows the details of a product and allows to edit the product. This GUI is same for add products and update as the latter has prefilled values. Figure 7: Edit product Edit product page for store type user 3.8.3 Login GUI This GUI shows the login page of our website. Figure 8: Login page Website’s login page 3.8.4 Sign Up GUI This GUI shows the Sign-Up page of our website. Figure 9: Sign Up page Website’s sign-up page 3.8.5 User/Client Pages GUI This GUI shows the favourites page of our website. All of user’s favourites products are placed on this page. Figure 10: Favorite’s page Website’s favourites page This GUI shows the product details pop up window of our website. From this page we can buy a product along with viewing its details. Moreover, we can give feedback of a product that we have bought. Figure 11: Product Details page Products Detail page This GUI shows the Home page for the user where user can see products to buy from. User can add product to favorite’s list, can see the details of the product. Also, user can use filters to filter out specific product. User can also search a specific product and also can see his profile. Figure 12:3 Home Page Home page containing products This GUI shows the User’s profile where he can see his information. User can access feedback page and favorite’s page from here as well. Also user can sign out from here as well. Figure 13: Profile page Profile containing user information Database Design 3.9.1 ER Diagram Figure 14: ER Diagram This shows the ER Diagram of our system database 3.9.2 Data Dictionary 3.9.2.1 User Table 1: User Data Dictionary This is the data dictionary of user. Fields Data Types Example ID Long 1992 Name String Ben White Email String benwhite@gmail.com Password String ArsenalFC Phone Number String 0321-xxxxxxx Address String

26**852-B Milaad St, Block B Faisal Town, Lahore, Punjab 54770**

Profile Image String myWebsite.com/proiflepic.jpg 3.9.2.2 Favourite Table 2: Favourites Data Dictionary This is the data dictionary of favourites. Fields Data Types Examples ProductID (FK from ID of Product table) Long 221 UserID (FK from ID of Product table) Long 1919 TimeStamp datetime 2016-12-21 00:00:00.000 3.9.2.3 Product Table 3: Product Data Dictionary This is the data dictionary of Product. Fields Data Types Examples ID Long 23 SKU String K91919 Title String Shorts Description String Any description Price Float 99.99 Color String Red Size String Medium URL String My-website.com/product=10 Category String Men Approved Boolean True Suspended Boolean False TimeStamp DateTime 2016-12-21 00:00:00.000 Quantity Int 10 StoreID (FK) Long 101 3.9.2.4 Purchase History Table 4: Purchase History Data Dictionary This is the data dictionary of Purchase History. Field Data Types Examples ProductID (PK, FK) Long 23 UserID (PK, FK) Long 12 Clicks Int 20 TimeStamp DateTime 2016-12-21 00:00:00.000 Quantity Int 15 Feedback String Great Product 3.9.2.5 Store Table 5: Store Data Dictionary This is the data dictionary of Store. Field Data Types Examples ID Long 7 Name String Tayyab Email String tayyab@gmail.com Password String Qwerty1234 NTN String 231242134-9 Bank Account Number String HQM21HJ819 Address String 134 A Muslim Town Profile Image String myWebsite.com/proiflepic.jpg Description String Detailed Info 3.9.2.6 Admin Table 6: Admin Data Dictionary This is the data dictionary of Admin. Fields Data Types Examples ID Long 23 Name String Tayyab Email String tayyab@gmail.com Password String Tayyabqwerty Phone Number String 03401913211 Profile Image String myWebsite.com/proiflepic.jpg 3.9.2.7 Discount Factor Table 7: Discount Factor Data Dictionary This is the data dictionary of Discount. Field Data Types Examples ID Long 3 ProductID (FK) Long 29 Percentage Float 70.1 System Requirements 3.10.1 Hardware Requirements ? A IOT device that supports a browser. ? Internet connection should have at least 1Mbps. ? A web server to host our website. 3.10.2 Software Requirements ? Web browser in the system. Design Considerations Before attempting to build a complete design solution, this section explains many of the challenges that must be addressed or overcome. 3.11.

15**1 Assumptions and Dependencies** Following are the **assumptions or dependencies regarding the software and its use. These may concern such issues as**

: ? User has a desktop with 2GB RAM and 64-bit Operating system. ? Users have JS enabled browsers installed. ? Users must have internet connection. ? User is familiar with understanding of basic e-commerce. 3.11.2 General Constraints Following are the

15**global limitations or constraints that have a significant impact on the design of the system's software**

: 3.11.2.1 Hardware or software environment ● We are designing a web application, so targeted systems should have JS supported browsers such as Google Chrome or Mozilla Firefox. ● In the case of hardware, web application requires 2 GB RAM, 64-bit OS. 3.11.2.2 End-user environment ● Users can utilize the online application with high-speed internet and a browser that supports the idea of web caching and cookies to save data for future use, as specified in the software requirements. 3.11.2.3 Availability or volatility of resources ● High speed and uninterrupted internet is required for better performance. 3.11.2.4 Interoperability requirements ● Client and server have to share and store data using common standards.

3.11.2.5 Interface/protocol requirements ● Http protocol will be used for communication between client and server. ● Interface will be implemented using react framework and running on browsers which are mentioned in software requirement. 3.11.2.6 Data repository and distribution requirements ● For web application data storage and retrieval, online cloud-based databases will be employed. It can only be changed by the administrator. 3.11.2.7 Security requirements (or other such regulations) ● System will be performing actions using secure protocols, secure the personal information of users by using the CSRF tokens for forms and data transfer. ● System will perform authentication and authorization. 3.11.2.8 Memory and other capacity limitations ● System requires CPU, Memory, I/O capacity, Bandwidth and cache space for better performance. 3.11.2.9 Verification and validation requirements (testing) ● The system will allow those users who have correct login credentials to conduct actions. 3.11.3 Language Constraints ? This system is only useful to those who are familiar with English language. Development Methods We chose the agile model, specifically the scrum method [2]. When it comes to development, the main justification for using the scrum model is its productivity and quality. In addition, this paradigm aids in the rapid development of our software. In addition, we held daily meetings with each other to keep track of our development and performance. Another advantage that drew us to this approach over others was its flexibility to adapt to changing requirements as we went along. We used the Scrum paradigm to break tasks into sprints, which simplified our job. Following were the steps in each sprint: 1. Planning 2. Implementation 3. Review 4. Retrospect We initially examined two methods: the agile model, which we ultimately chose, and the prototype model. The following are the main reasons for not implementing the prototype model: 1. Poor documentation because of changing system requirements. 2. Incomplete problem analysis. 3. Increases the complexity of the system. Class diagram Figure 15: Class Diagram Diagram for high level system architecture. Sequence diagram Figure 16: Login Account This is the sequence diagram of Login Account for Users . Figure 17: Create Account/Signup This is the sequence diagram of Signup for Users. Figure 18: Reset/Forget Password This is the sequence diagram of Reset/Forget Password for Users. Figure 19: Logout This is the sequence diagram of Logout Procedure for Users. Figure 20: Edit Profile This is the sequence diagram of Edit Profile for Users . Figure 21: View Product Details This is the sequence diagram of View Product Details. Figure 22: Buy Product This is the sequence diagram of Buy a Product Procedure. Figure 23: Add/Remove favorites This is the sequence diagram of Removing and Adding Products from Favourites list . Figure 24: View Favorites List This is the sequence diagram of Viewing Favourites List. Figure 25: Filter by Type This is the sequence diagram of filtering products by their “Type”. Figure 26: Filter by Size This is the sequence diagram of filtering products by their “Size”. Figure 27: Filter by Color This is the sequence diagram of filtering products by their “Colour”. Figure 28: Filter by Waist This is the sequence diagram of filtering products by their “Waist”. Figure 29: Filter by Price Range This is the sequence diagram of filtering products by their “Price Range”. Figure 30: Filter by Discount Factor This is the sequence diagram of filtering products by their “Discount Factor”. Figure 31: Product Approval This is the sequence diagram of product approval, rejection or suspension. Figure 32: View Product This is the sequence diagram of admin view product details. Figure 33: Home Page This is the sequence diagram of store home page with product list. Figure 34: Add Product This is the sequence diagram of store adding a product. Figure 35: Update product This is the sequence diagram of store updating a product. Figure 36: Delete Product This is the sequence diagram of store deleting a product. Policies and Tactics 3.15.1 Product to use We will be using the latest version of python along with the Collaborative Filtering library. Moreover, we will use real-time cloud based databased named MongoDB. For running our Python code, we will use PyCharm. 3.15.2 Coding guidelines and conventions To make our code more structural and legible, all standard coding rules will be followed, including properly commented code and the use of OOP ideas while coding. Furthermore, as previously indicated, we will use Ben Schneiderman's 8 golden standards of UI design when creating our UI. 3.15.3 Testing the software Acceptance testing, unit testing, 34**functional testing, performance testing, stress testing**, and **usability testing**

are some of the testing methodologies we studied in software engineering. Furthermore, we will collect testing data from our university students and run it through our algorithm, after which we will compare the output of our algorithm to the testing data to ensure that our system is accurate enough. 3.15.4 Maintaining the software After our fyp is finished we will release improved versions of our software with added feature to assist the users. Moreover, we will fix any possible future faults in our system. 3.15.5 Protocol Http protocol will be used for communication between client and server. Interface will be implemented using react framework and running on browsers which are mentioned in software requirement. As we are using MongoDB, we will be storing data in JSON format hence data will flow over the network in JSON format. 3.15.6 Accessing the application The web application would be hosted on cloud accessible with a URL link. Users having an account can access the application by going through the authentication procedure. In case a user doesn't have an account he can sign up to make a new account. For vendors they must register their store on our website to add their products. 3.15.7 Choice of algorithm We will use brute force approach along with different variants of collaborative filtering. We will choose the algorithm that gives us the best results. 3.15.8 Web Scrapping We will scrap popular clothing brands website to retrieve their products by using a web scraping tool known as beautiful soup. Implementation and Test Cases This chapter provides the implementation details of our prototype. Implementation Initially we have scraped data from websites of few popular clothing brands of Pakistan using Beautiful Soup which is a library of Python. After that we have cleaned the data and after that we worked on extracting the features from our data set that will help us to optimize the accuracy of our prototype. We ran our data on different types of Collaborative filtering algorithms alongside brute force approach in order to test the accuracy of these algorithm. Moreover, we collected data from volunteers using google forms to train our model. We have almost completed the frontend of our website and we are turning our focus onto the backend. 4.1.1 Dataset Collection This project's data collection was created just for it. In this study, there were two sorts of datasets. We first required product information from a few well-known businesses. We scraped data from those brands using a website scraper. After that, we used a few of pre-processing procedures on it. To make it applicable to our model as a whole. The other dataset was gathered from university students and others in our usual surroundings. The dataset includes a sample of items from a few different companies. Each product had a rating scale ranging from 1 to 5, with a total of 50 goods to score. We acquired data from 50+ users through trusted sources. he data in the csv format sheet was included in the dataset. For use in the training model, the extracted data had to be preprocessed. The main problem with the dataset is that we were unable to gather it from a larger audience, and the number of goods was just too large for one user to score them all. 4.1.2 Dataset Preprocessing Preprocessing on data is a must if you are looking to train a model using this data. So the data that we scrapped as well as the data that we collected had some issues that needed preprocessing in order to make the data feasible for the model. We did EDA on the data in order to find some kind of relationships and patterns in the data. We handled null values from our data by either removing those value or by doing imputation on it. We also replaced some string value with integers in order to make the data type similar across all the dataset. 4.1.3 Product Recommendation Model Implementation We looked at four different options for implementing the concept. The first was dealing with our model in a basic manner. To create the forecast, our method simply sorted the results depending on the selling price of each product. A decision tree algorithm was used in the second technique.

16**We try to form a condition on the features at each step or node of a** decision **tree used for classification to separate all the labels or classes contained in the dataset to the** fullest purity. Collaborative filtering utilizing a model-based approach was the third option. Websites like Amazon, YouTube, and Netflix utilize a process called collaborative filtering. It eliminates products that a user could enjoy based on the reactions of other users.

32**Memory-based and model-based collaborative filtering** algorithms **are**

the two types. Building machine learning algorithms to anticipate user ratings is part of the model-based approach. They use dimensionality reduction techniques to replace a large matrix with a lot of missing values with a much smaller matrix in a lower-dimensional space. The purpose of this exercise is to apply SVD algorithms, experiment with different parameter combinations, and analyse the outcomes. Implementation and Test Cases Grid Search Cross Validation is a cross-validation process to calculate accuracy metrics for an algorithm based on numerous parameter combinations. It's handy for determining the ideal parameter setup. As a consequence, the default value is the best for the vast majority of parameters. The benefit gained through Grid Search is negligible. The findings generated by the SVD model that delivered the best RMSE score were investigated in depth. The fourth and last strategy was to use a collaborative filtering recommender system based on memory. Memory-based algorithms use statistical approaches to compute predictions over the full dataset. They're split

19**into two categories: user-item filtering and item-item filtering**. The **User-Item** technique entails identifying **a** group of **users** who **are** the most similar **to** the **user** U (**based on** their prior ratings) **and**

calculating the rating for the item I using the ratings supplied by those users. The item-by-item technique involves determining a collection of the most comparable items to the item I (based on previous user ratings) and calculating the rating for the item I using the ratings of similar things rated by user U.

21**Only Cosine similarity or Pearson correlation coefficients, which are** exclusively **based on** mathematical **operations, are used to** compute **the** closest users **or**

things. The system searches for a group of users with comparable transaction histories to the current user using statistical approaches. In the closest neighbour approach, this method is also known as nearest-neighbour or user-

28**based collaborative filtering. Memory-based collaborative filtering, on the**

other hand, performs poorly with high-sparsity data and has limited scalability for big datasets. 4.1.4 Website Implementation We have almost completed our frontend implementation and now we are moving to implementing backend for our website. Following section describe the implementation details for our frontend and backend. 4.1.4.1 Frontend Implementation We have made our frontend using React JS. We have used tailwind-css, bootstrap, material-ui and HTML to make our frontend. Our frontend is divided in two sections based on different users, one for normal customers and the other one for admin and store owners. Initially both users will be shown a login page where they can login using their credentials or they can make a new account. If they try to login, they will be redirected to their respective home pages based on whether the user is a customer or a store owner. Login credential verification will be done by sending an API request to our backend. A customer will be shown the recommended products on the home page along with a filter menu. A customer can also view the favorites page where he/she will be shown the list of his/her favorite products. Moreover, he/she can also update his/her profile by going to the profile page. All the updates will be updated in the database using API requests to the backend. A store owner will be shown his stores products that he has added on our website. From there he/she can add, edit and delete products. Moreover, he can also view a summary of his store's financial situation which includes his total revenue generated etc. 4.1.4.2 Backend Implementation We have decided to use Django (Python) in order to build our backend API. Our backend will be responsible to communicate with the database in order to efficiently respond to the API called from our frontend. It will respond to the API calls by sending the required data to our frontend. Test Case Design and Description 4.2.1 Login Account Login Module

2**Test Case ID: 1 QA Test Engineer**: Muhammad Zain CR7 **Test Case Version: 1 Reviewed by**: Tayyab Waseem **Test Date**: 14-05-2022 **Use Case Reference(s**): Login Account **Revision History**: None **Objective: User** should **be able to login successfully**

. Product/Ver/Module: Login module Environment: Software: Any Modern Browser Hardware: Laptop Assumptions: The computer of the user should be connected to Internet.

1**Pre-Requisite: User** should **be** registered **in** system. **Step No. Execution description Procedure result 1 User** opens **the** login page. **System displays** login page asking **for**

username and password. 2 Entered valid login credentials and pressed login button. System verifies the credentials,

1**establishes a session for the user and redirects the user to the home page**. Comments **The system**

works as expected.

1**Passed Failed Not Executed 4.2.2** Create Account **Login** Module **Test Case ID**

: 2

2**QA Test Engineer**: Muhammad Zain **Test Case Version: 1 Reviewed by**: Tayyab Waseem **Test Date**: 14-05-2022 **Use Case Reference(s**): Create Account **Revision**

History: None **Objective: User** should **be able to** create **his** account successfully. **Product/Ver/Module**: Signup **module Environment**

: Software: Any Modern Browser Hardware: Laptop Assumptions: The computer of the user should be connected to Internet. Pre-Requisite: User should enter valid credentials. Step No. Execution description Procedure result 1 Clicked in Sign-up button from the System displays Sign-up page asking for login page. required information. 2 Entered valid information and System verifies the information, creates a new pressed submit button. account

1**and redirects user to the home page**. Comments **The system**

works as expected.

2**Passed Failed Not Executed 4.2.3** Reset **Password** Reset **Password Test Case ID**: 3 **QA Test Engineer**: Muhammad Zain **Test Case Version: 1 Reviewed by**: Tayyab Waseem **Test Date**: 14-05-2022 **Use Case Reference(s)**: Reset Password **Revision History**: None **Objective: User** should **be able to** reset **his** account's **password** successfully. **Product/Ver/Module**: Login **module Environment**

: Software: Any Modern Browser Hardware: Laptop Assumptions: The computer

10**of the user should be** connected to Internet. **Pre**-Requisite: **User should be registered** in the system. **Step** No. **Execution description Procedure result 1**

Clicked on “Forget Password” System displays prompt asking for user's button from the login page. username. 2 Entered valid username and System verifies the username, sends an email to pressed submit button. the user's email address. Comments The system works as expected.

1**Passed Failed Not Executed 4.2**.4 **Email** Verification **Email**

Verification

1**Test Case ID: 4 QA Test Engineer**: Muhammad Zain **Test Case Version: 1 Reviewed by**: Tayyab Waseem **Test Date**: 14-05-2022 **Use Case Reference(s)**: Email Verification **Revision History**: None **Objective: User**

should have received the email containing the password. Product/Ver/Module: Login module Environment: Software: Any Modern Browser Hardware: Laptop Assumptions: The computer of the user should be connected to Internet and user should have opened his/her Gmail. Pre-Requisite: User should be have gone through the reset password procedure.

5**Step No. Execution description Procedure result 1 Clicks** on **the** view password **button System** displays **a**

password asking for user’s in the email. username. 2 Entered valid username and System verifies the username and displays the pressed show password button. password to the user. Comments The system works as expected.

2**Passed Failed Not Executed 4.2**.5 Logout Logout **Test Case ID**: 5 **QA Test Engineer**: Muhammad Zain **Test Case Version: 1 Reviewed by**: Tayyab Waseem **Test Date**: 14-05-2022 **Use Case Reference(s**): Logout **Revision History**: None **Objective: User** should **be able to**

logout from his/her account successfully. Product/Ver/Module: Logout module Environment: Software: Any Modern Browser Hardware: Laptop Assumptions: The computer of the user should be connected to Internet. Pre-Requisite: User

4**should be logged in to** his/her account. **Step** No. **Execution description Procedure result 1 Clicks on** the “Logout” **button** from **System will** logout **the user** and redirect **the user**

the home page. to the login page. Comments The system works as expected. Passed Failed Not Executed 4.2.6 View Profile View Profile Test Case ID: 6

2**QA Test Engineer**: Muhammad Zain **Test Case Version: 1 Reviewed by**: Tayyab Waseem **Test Date**: 14-05-2022 **Use Case Reference(s**): View Profile **Revision History**: None **Objective: User** should **be able to** view **his/her** profile information. **Product/Ver/Module**: Profile **module Environment**

: Software: Any Modern Browser Hardware: Laptop Assumptions: The computer of the user should be connected to Internet. Pre-Requisite:

4**User should be logged in to** his/her account. **Step** No. **Execution description Procedure result 1 Clicks on** the “Edit Profile” **button** System displays **a**

new page containing user’s from the home page. current information. Comments The system works as expected.

1**Passed Failed Not Executed 4.2.7 Edit** Profile **Edit** Profile **Test Case ID**

: 7

2**QA Test Engineer**: Muhammad Zain **Test Case Version: 1 Reviewed by**: Tayyab Waseem **Test Date**: 14-05-2022 **Use Case Reference(s**): Edit Profile **Revision History**: None **Objective: User** should **be able to** edit **his/her** profile information. **Product/Ver/Module**: Profile **module Environment**

: Software: Any Modern Browser Hardware: Laptop Assumptions: The computer of the user should be connected to Internet. Pre-Requisite: User should be logged in to his/her account and must be on the view profile page.

1**Step No. Execution description Procedure result 1 Clicks on the** “Edit Profile” button **System** displays **the**

user’s information in from the view profile page. editable form. 2 Entered updated information and System validates the information and updates pressed update profile button. user’s profile. Comments The system works as expected. Passed Failed Not Executed 4.2.8 View Product View Product Test Case ID: 8

2**QA Test Engineer**: Muhammad Zain **Test Case Version: 1 Reviewed by**: Tayyab Waseem **Test Date**: 14-05-2022 **Use Case Reference(s**): View Product **Revision History**: None **Objective: User** should **be able to** see **the** details of **the** selected **product**. Product/**Ver/Module**

: Product View module Environment: Software: Any Modern Browser Hardware: Laptop Assumptions: The computer of the user should be connected to Internet. Pre-Requisite: User should be logged in to his/her account and must click on the “Details” button.

1**Step No. Execution description Procedure result 1 Clicks on the** “Details” button from **System** displays a pop-**up** containing product **the**

home page. details. Comments The system works as expected. Passed Failed Not Executed 4.2.9 Buy Product Buy Product Test Case ID: 9

2**QA Test Engineer**: Muhammad Zain **Test Case Version: 1 Reviewed by**: Tayyab Waseem **Test Date**: 14-05-2022 **Use Case Reference(s**): Buy Product **Revision History**: None **Objective: User** should **be able to** buy **a**

product from the brand’s website. Product/Ver/Module: Product Buy module Environment: Software: Any Modern Browser Hardware: Laptop Assumptions: The computer of the user should be connected to Internet.

Pre-Requisite: User should be logged in to his/her account and must click on the “Details” button of the selected product. Step No.

4**Execution description Procedure result 1 Clicks on** the “Buy” **button** from **the System** redirects **the user to the**

respected link of product details pop-up. the product. Comments The system works as expected.

1**Passed Failed Not Executed 4.2**.10 **Add** to Favorites **Add** Favorites **Test Case ID**: 10 **QA Test Engineer**: Muhammad Zain **Test Case Version: 1 Reviewed by**: Tayyab Waseem **Test Date**: 14-05-2022 **Use Case Reference(s**

): Add Favorites

2**Revision History**: None **Objective: User** should **be able to** add **a** product **to**

his/her favorites list. Product/Ver/Module: Favorites module Environment: Software: Any Modern Browser Hardware: Laptop Assumptions: The computer of the user should be connected to Internet. Pre-Requisite: User should be logged in to his/her account and selected product should not be in the favorites list

1**Step No. Execution description Procedure result 1 Clicks on the** “Favorites” icon from **System** adds **the**

product to his/her favorites list. the selected product. Comments The system works as expected. Passed Failed Not Executed 4.2.11 Remove from Favorites Remove Favorites

5**Test Case ID: 11 QA Test Engineer**: Muhammad Zain **Test Case Version: 1 Reviewed by**: Tayyab Waseem **Test Date**: 14-05-2022 **Use Case Reference(s)**: Remove from Favorites **Revision History**

:

12**None Objective: User should be able to** remove **a**

product from his/her favorites list. Product/Ver/Module: Favorites module Environment: Software: Any Modern Browser Hardware: Laptop Assumptions: The computer of the user should be connected to Internet. Pre-Requisite: User should be logged in to his/her account and selected product should be in the favorites list Step No. Execution description Procedure result 1 Clicks on the “Favorites” icon from System will remove the product from his/her the selected product. favorites list. Comments The system works as expected. Passed Failed Not Executed 4.2.12 View Favorites View Favorites Test Case ID: 12

2**QA Test Engineer**: Muhammad Zain **Test Case Version: 1 Reviewed by**: Tayyab Waseem **Test Date**: 14-05-2022 **Use Case Reference(s**): View Favorites **Revision History**: None **Objective: User** should **be able to** view **his/her** favorites list. **Product/Ver/Module**: Favorites **module Environment**

: Software: Any Modern Browser Hardware: Laptop Assumptions: The computer of the user should be connected to Internet. Pre-Requisite: User

4**should be logged in to** his/her account. **Step** No. **Execution description Procedure result 1 Clicks on the** "View Favorite" **System** redirects **the** user **to the**

new page button from the home page. containing a list of user's favorite's items. If there are no products in the favorites list then empty list will be displayed. Comments The system works as expected. Passed Failed Not Executed 4.2.13 Filter Products Filter Products

9**Test Case ID**: 13 **QA Test Engineer**: Muhammad Zain **Test Case Version: 1 Reviewed by**: Tayyab Waseem **Test Date**: 14-05-2022 **Use Case Reference(s**): Filter Products **Revision History**: None **Objective: User** should be able **to**

filter products based on selected type.. Product/Ver/Module: Favorites module Environment: Software: Any Modern Browser Hardware: Laptop Assumptions: The computer of the user should be connected to Internet. Pre-Requisite:

4**User should be logged in to** his/her account. **Step** No. **Execution description Procedure result 1 Clicks on** the "Filter" **button** from **the** home page. **System**

displays a dropdown menu containing different types of filters. 2 Selects the "Type" filter from the dropdown menu. System displays a pop-up of the types for clothing products. 3 Selects a type from the pop-up window. System reloads the home page and displays the products based on the user's filtering criteria. Comments The system works as expected. Passed Failed Not Executed 4.2.14 View Home Page Home Page

5**Test Case ID**: 14 **QA Test Engineer**: Muhammad Zain **Test Case Version: 1 Reviewed by**: Tayyab Waseem **Test Date**: 14-05-2022 **Use Case Reference(s**): View Home Page **Revision History**

: None Objective: User should be able to see all products of their store. Product/Ver/Module: Favorites module Environment: Software: Any Modern Browser Hardware: Laptop Assumptions: The computer of the user should be connected to Internet. Pre-Requisite:

4**User should be logged in to** his/her account. **Step** No. **Execution description Procedure result 1 Clicks on the** “Home” icon from **the**

System displays all recommended products. navbar. Comments The system works as expected. Passed Failed Not Executed 4.2.15 Approve Product Approve

7**Test Case ID**: 15 **QA Test Engineer**: Asjad **Test Case Version: 1 Reviewed by**: Tayyab Waseem **Test Date**: 14-05-2022 **Use Case Reference(s**): Approve Product **Revision History: None Objective: Product**

is approved on the store. Product/Ver/Module: Administration module Environment: Software: Any Modern Browser Hardware: Laptop Assumptions: The computer of the user should be connected to Internet. Pre-Requisite:

4**User should be logged in to** his/her account. **Step** No. **Execution description Procedure result 1** Clicks on **a**

particular product. Display attribute of selected product 2 Click on approve Button System publish and push an email notification Click on suspend button System suspends and push an email notification Comments The system works as expected. Passed Failed Not Executed 4.2.16 View Product View Product

7**Test Case ID**: 16 **QA Test Engineer**: Asjad **Test Case Version: 1 Reviewed by**: Zain **Test Date**: 14-05-2022 **Use Case Reference(s**): View Product **Revision History: None Objective**

: List View of Products on store. Product/Ver/Module: Store module Environment: Software: Any Modern Browser Hardware: Laptop Assumptions: The computer of the user should be connected to Internet. Pre-Requisite:

4**User should be logged in to** his/her account. **Step** No. **Execution description Procedure result 1**

. Display all products of store 2 Click on particular product System display detail view Comments The system works as expected.

1**Passed Failed Not Executed 4.2**.17 **Add** Product **Add** Product **Test Case ID**: 17 **QA Test Engineer**: Asjad **Test Case Version: 1 Reviewed by**: Tayyab **Test Date**: 14-05-2022 **Use Case Reference(s**

): Add Product Revision History: None Objective: Add a new Product in store. Product/Ver/Module: Store module Environment: Software: Any Modern Browser Hardware: Laptop Assumptions: The computer of the user should be connected to Internet. Pre-Requisite:

> 4**User should be logged in to** his/her account. **Step** No. **Execution description Procedure result 1** . Click **on**

Add new product Display a form of product details 2 Fill form and click “Add Product” System add product for approval in store Comments The system works as expected. Passed Failed Not Executed 4.2.18 Delete Product Delete Product

> 6**Test Case ID**: 18 **QA Test Engineer**: Asjad **Test Case Version: 1 Reviewed by**: Tayyab **Test Date**: 14-**05**-2022 **Use Case Reference(s**): Delete Product **Revision History**: None **Objective**: Delete a **Product** from store. Product/**Ver/Module**: Store module **Environment**

: Software: Any Modern Browser Hardware: Laptop Assumptions: The computer of the user should be connected to Internet. Pre-Requisite:

> 4**User should be logged in to** his/her account. **Step** No. **Execution description Procedure result 1** . Select on **a**

particular product and Display a warning modal click “delete” 2 Click on confirmation Removes product from store Comments The system works as expected. Passed Failed Not Executed 4.2.19 Update Product Update Product

> 6**Test Case ID**: 19 **QA Test Engineer**: Asjad **Test Case Version: 1 Reviewed by**: Zain **Test Date**: 14-**05**-2022 **Use Case Reference(s**): Update Product **Revision History**: None **Objective**: Update a **Product** from store. Product/**Ver/Module**: Store module **Environment**

: Software: Any Modern Browser Hardware: Laptop Assumptions: The computer of the user should be connected to Internet. Pre-Requisite:

> 4**User should be logged in to** his/her account. **Step** No. **Execution description Procedure result 1** . Select on **a**

particular product and Display a model form with prefilled fields click “update” 2 Enters updated field data and click Product information is updated “update” button Comments The system works as expected. Passed Failed Not Executed Test Metrics The test metrics are discussed below:

8**Metric Value Number of Test Cases**: 19 **Number of Test Cases Passed**: 19 **Number of Test Cases Failed: 0 Test Case Defect Density: 0**/10 **Test Case Effectiveness**: 10/10 **Traceability Matrix**

: Included in a separate file Experimental Results and Analysis The experiments performed on the following Algorithms: 1. Naive Approach (Control) 2. Decision Tree 3. Support Vector Machines: Collaborative Filtering (Model based Approach) 4. Cosine Similarity: Collaborative Filtering (Memory based Approach) Produced vastly different results on the accuracy and error. We found that the cosine similarity based collaborative filtering works best with our use case and dataset. Experimental Results Cosine Similarity Support Vector Machine Decision Tree Naïve Approach 0 10 20 30 40 50 60 Accuracy Figure 37: Experimental Results This is the final results chart of the comparison between the recommendation system algorithms . Conclusion Conclusion The first part of our project involved deciding on the experimentation and testing of various algorithms to select the most suitable one. After that, we immediately started working on the implementation of the use cases of the project discussed in earlier sections of this document. We started off with the front-end development in React JS creating responsive web pages for user interactions. Then we created the backend models, serializers and restful APIs in Django Rest Framework to support and facilitate the front-end pages. Some of the challenges faced involved figuring out the communication part between the front end and back end for fast, efficient communication as well as writing code which is maintainable and extensible. As a solution we found “axios”, a handy library for making communication between React JS and Restful APIs easier and easy to maintain. Panda Mall 1 2 Panda Mall 3 Requirements and Design 4 Panda Mall 5 Requirements and Design 6 Panda Mall 7 Requirements and Design 8 Panda Mall 9 Requirements and Design 10 Panda Mall 11 Requirements and Design 12 Panda Mall 13 Requirements and Design 14 Panda Mall 15 Requirements and Design 16 Panda Mall

24**17 Requirements and Design 18** Panda Mall 19 **Requirements and Design**

20 Panda Mall

24**21 Requirements and Design 22** Panda Mall 23 **Requirements and Design**

24 Panda Mall 25 Requirements and Design 26 Panda Mall 27 Requirements and Design 28 Panda Mall 29 Requirements and Design 30 Panda Mall 31 Requirements and Design 32 Panda Mall 33 Requirements and Design 34 Panda Mall 35 Requirements and Design 36 Panda Mall 37 Requirements and Design 38 Panda Mall 39 Requirements and Design 40 Panda Mall 41 Requirements and Design 42 Panda Mall 43 Requirements and Design 44 Panda Mall 45 Requirements and Design 46 Panda Mall 47 48 Panda Mall 49 Implementation and Test Cases 50 Panda Mall 51 Implementation and Test Cases 52 Panda Mall 53 Implementation and Test Cases 54 Panda Mall 55 Implementation and Test Cases 56 Panda Mall 57 Implementation and Test Cases 58 Panda Mall 59 60 Panda Mall 61