**Prototype Requirements**

**<P14>:<Shop Savvy>**

|  |  |
| --- | --- |
| **Student ID** | **Name** |
| **25100002** | **Ahmad Kashif Jabbar** |
| **25100105** | **Husnain Ali** |
| **25100179** | **Musa Aftab Ahmed** |
| **25100229** | **Syed Messam Ali** |
| **25100208** | **Zainab Fatima** |

|  |  |  |
| --- | --- | --- |
| **Content** | **Totals** | **Obtained** |
| Prototype Requirements Completed | 10 | 10 |
| Properly tested working system | 30 | 25 |
| Deployed on an online hosting platform. | 20 | 10 |
| Code with readable comments | 10 | 10 |
| 3-4 minutes video | 10 | 8 |
| Updated Readme file | 10 | 10 |
| Uploaded to GitHub’s "Prototype" folder. | 10 | 10 |
| GitHub folder structure penalty | -15 | - |
| Late submission penalty | -20 | - |
| **Grand Total** | **100** | **83** |
| **General Comments/Individual Grading:**  Explaining Code in video was not required  No Products available on online app | | |

**Table of Contents**

[1. Introduction 3](#_Toc180427639)

[2. Instructions 4](#_Toc180427640)

[3. List of Requirements for Prototype 5](#_Toc180427641)

[4. Review checklist 6](#_Toc180427642)

# Introduction

The AI-driven Personalized Clothing Recommendation Platform aims to enhance the online shopping experience for Pakistani consumers by bringing together local clothing brands on a single platform. This allows users to explore clothing from various brands with ease and convenience. The platform will focus on showcasing at least 10 local Pakistani clothing brands, providing users with a curated selection of their favorite products. This approach caters specifically to the local fashion market, giving shoppers access to multiple brands in one place.

The platform’s primary benefit is convenience. Users can find their preferred clothing items from different brands all in one location, eliminating the need to visit multiple websites. The experience will be personalized, offering tailored suggestions that align with their style and preferences.

The recommendation system will be powered by an AI model that learns from users’ previous purchases, browsing history, and personal taste. This data-driven approach allows for highly relevant, personalized product suggestions, making the shopping experience more enjoyable and efficient.

The platform also has the potential to generate revenue through affiliate marketing. By partnering with local brands, the platform can earn commissions for purchases made through the site. This creates a mutually beneficial relationship between the platform and the brands, helping to drive visibility and sales for both parties.

Ultimately, the goal is to provide a seamless, personalized, and engaging shopping experience for Pakistani consumers. By offering a convenient, AI-enhanced platform, the project will simplify the process of discovering and purchasing clothing from a variety of local brands.

# List of Requirements for Prototype

### **1. Website Setup**

#### **1.1 Initial Website Layout**

* Define website structure: homepage, product pages, login/signup pages, and wishlist.
* Create wireframes for each page.
* Set up routing for navigation across different pages.

#### **1.2 Backend Server Setup**

* Configure Node.js server for handling requests.
* Integrate MongoDB for user and product data storage.
* Set up RESTful API endpoints to serve data to the frontend.

### **2. Data Scraping for Product Information**

#### **2.1 Scraping Module Setup**

* Define target websites and establish scraping parameters (brands, product categories).
* Write scripts for scraping essential product information (name, price, images, availability).

#### **2.2 Data Storage and Updating**

* Design MongoDB schema to store product details and links.
* Schedule automated tasks to refresh and update product data periodically.

#### **2.3 Error Handling & Data Validation**

* Implement error-handling mechanisms for potential scraping issues (e.g., broken links, timeouts).
* Validate scraped data to ensure consistency (e.g., price formats, valid URLs).

### **3. Product Viewing Functionality**

#### **3.1 Product Listing Page**

* Create a frontend component to display products in a grid or list view.
* Configure API calls to fetch product data dynamically.

#### **3.2 Product Details Page**

* Set up a dedicated page for each product showing images, price, brand, description, and link.
* Enable real-time fetching of product details via API on the product details page.

#### **3.3 Pagination or Infinite Scroll**

* Implement pagination or an infinite scroll for efficient browsing on the product listing page.

### **4. User Authentication (Login & Signup)**

#### **4.1 User Registration**

* Create a signup form that captures username, email, and password.
* Implement backend logic to securely store user information in the MongoDB database.

#### **4.2 User Login**

* Create a login form allowing existing users to access their accounts.
* Configure session or token-based authentication for secure login sessions.

#### **4.3 Authentication Error Handling**

* Display error messages for incorrect credentials or validation issues.
* Implement a secure password reset option.

### **5. Wishlist Functionality**

#### **5.1 Add to Wishlist Feature**

* Add a button on the product details page that allows users to save items to their wishlist.
* Create backend logic to store wishlist items associated with the user’s profile.

#### **5.2 View Wishlist**

* Create a dedicated wishlist page displaying all saved items.
* Allow users to view product details or remove items from the wishlist.

#### **5.3 Persistent Wishlist Data**

* Ensure the wishlist data remains consistent across user sessions and devices by storing it in MongoDB.

### **6. Product Filtering**

#### **6.1 Filter by Category**

* Implement a category filter (e.g., Men’s, Women’s, Sale) for product listings.
* Enable dynamic updating of products displayed based on selected categories.

#### **6.2 Filter by Price Range and Brand**

* Add a filter for setting a price range and brand preference.
* Configure frontend and backend to apply filters simultaneously.

#### **6.3 Sorting Products**

* Implement sorting options (e.g., price low to high, newest arrivals).
* Ensure sorted and filtered views update dynamically without page refreshes.

### **7. Product Redirection through Links**

#### **7.1 Redirection Setup**

* For each product, add a “Buy Now” or “View on Site” button linking to the product’s original page on the brand’s site.

#### **7.2 Link Verification**

* Ensure all links redirect accurately, checking for broken or outdated links.
* Set up logging for any errors encountered in redirections for future debugging.

### **8. User Interface and Quality Assurance**

#### **8.1 User Interface Design**

* Finalize the UI for each page to ensure consistency in style, fonts, and branding.
* Make the UI responsive for different device screen sizes.

#### **8.2 Testing and Debugging**

* Conduct functional testing on each feature (login, wishlist, filters).
* Collect feedback from team members for UI improvements and bug fixes.

#### **8.3 Optimization**

* Optimize image sizes and load times for a smoother user experience.
* Evaluate and minimize API call frequency to reduce server load.

# Review checklist

Before submission of this deliverable, the team must perform an internal review. Each team member will review one or more sections of the deliverable.

|  |  |
| --- | --- |
| **Section** **Title** | **Reviewer Name(s)** |
| Req 1 2 | Zainab Fatima |
| Req 2 3 | Messam Ali |
| Req 3 4 | Husnain Ali |
| Req 5 6 | Musa Aftab |
| Req 7 8 | Ahmad jabbar |