Software Requirements Specification (SRS) Document

Project Name: Intelligent Wardrobe Management System

Team Number: 42

Team Members: Shaunak Biswas, Vansh Motwani,

Harpreet Singh, Jayesh Rajesh Sutar, Raveesh Vyas

Brief problem statement

The project aims to enable efficient wardrobe organization through implementing an Intelligent Wardrobe Management System. Utilizing deep learning, artificial intelligence, and machine learning, the system aims to transform how users engage with their wardrobes. The main objective is to develop a website and a mobile app that efficiently categorizes clothing items and offers personalized outfit suggestions from the clothes in their wardrobe. The application also presents the user with contemporary fashion trends to keep their style updated. The system strives to enhance the wardrobe management experience.

System requirements

Software: The web application should be developed using the MERN (MongoDB, Express.js, React.js, Node.js) stack, while the mobile app should be developed using React Native.

Technologies: Integration of machine learning models will require the use of FastAPI for API development and integration with ML models.

Programming Languages: JavaScript for both frontend and backend development.

Operational Environment: Microsoft Azure for hosting Databases and ML Models.

Users profile

The system is designed to be used by individuals who are fashion-conscious and seek assistance in managing their wardrobe efficiently. Users are expected to be familiar with basic computer operations and comfortable navigating through mobile and web applications. They may vary in their proficiency with technology, but the system aims to provide a user-friendly interface that accommodates users of all skill levels.

Feature requirements (described using use cases)

Website Interface

No.	Use Case Name	Description	Release
		A) **User Profile**	
A1	**Signing up**	The user will be prompted to enter the following details if they choose to create a new account They will be asked to fill up a form consisting of the following entries 1. Email id* 2. Password* 3. Name 4. Gender* 5. Phone number 6. Skin tone* (mandatory fields marked with asterisk*) After signing up, the newly created profile will already be logged in.	R1
A2	**Signing in**	The user will be prompted to enter their email id and password. If the password is incorrect, or the email id has not been registered, the user will be prompted to fill in their details again, until the correct password is entered.	R1
A3	**User Profile Customization**	Users should be able to change their existing password by providing old and new password Any other change in the profile (i.e. Name, Gender, Phone Number etc.) can be made without entering a password.	R1
A4	**Log out**	On clicking the log out button, the logged in profile will be logged out and be redirected to the home page.	R1
		B) **Organize Wardrobe**	
B1	**Viewing Clothing Collection**	Users should be able to apply filters and/or view their existing collection of clothes stored in the user database. This is only accessible if a profile is logged in. All clothing items have categories assigned by the ML model.	R2

		If a profile is not logged in, the user will be prompted to either log in or sign up.	
B2	**Adding New Clothing Items**	Users should be able to add new clothing to their wardrobe if they have logged in. To add clothing items, a button will be presented, which will open a pop-up window providing 'Drag and drop' and 'Browse from device' options. If the wardrobe is empty, the application should prompt the user to add their first item. Upon adding a new item, the application should update the user's wardrobe in the database with the new clothing details.	R2
B3	**Deleting Clothing Items**	Users should be able to remove unwanted clothing items from their wardrobe. Again, this option is only available if the user is logged in with a profile. The website should provide a delete option for each clothing item displayed in the collection. Upon deletion, the application should update the user's wardrobe in the database by removing the selected item.	R2
B4	**Add To Favourites**	The user will be presented with a button to add clothes to Favourites.	R2
C) **Home Page**			
C1	**Navigation Options**	The Home Page shall provide clear navigation options to direct users to other pages within the application: Current Trends, their wardrobe, Outfit Recommendations	R1
C2	**Slideshow of Fashion Trends**	The Home Page shall include a slideshow component showcasing current fashion trends. The slideshow shall display a series of images representing popular fashion trends, with each image transitioning smoothly to the next. Users shall have the option to manually navigate through the slideshow or allow it to auto-advance at a predefined interval.	R2
D) **Trend Analysis**			

D1	**Trends page**	The user is displayed clothing items currently trending (not from the user's wardrobe) under a specified category (such as Gen Z, Ethnic, etc.).	R2
		E) **Outfit Recommendation**	
E1	**User Input for Clothing Preferences**	Users will provide input regarding their preferred types of clothing through options such as Formals, Partywear, Casual, Business Attire, Sportswear, Menswear, Womenswear, etc. This option is only available if the user has signed in. Moreover, if the user's wardrobe is empty, 'Your wardrobe is empty' will be displayed instead.	R1
E2	**Display of Recommended Clothing**	Upon processing user input by passing through ML models the system will display recommended clothing items that match the specified criteria. Present recommended clothing items to users in a visually appealing and user-friendly manner, facilitating easy exploration and selection.	R1

Mobile App Interface (M)

No.	Use Case Name	Description	Release
		A) User Authentication	
A1	**Signing up**	The user will be prompted to enter the following details if they choose to create a new account They will be asked to fill up a form consisting of the following entries 1. Email id* 2. Password* 3. Name 4. Gender* 5. Phone number 6. Skin tone* (mandatory fields marked with asterisk*) After signing up, the newly created profile will already be logged in.	R1
A2	**Signing in**	If the user instead chooses to sign in, they will be prompted to enter their email id and password. If the	R1

		password is incorrect, or the email id has not been registered, the user will be prompted to fill in their details again, until the correct password is entered.	
A3	**User Profile Customization**	Users should be able to change their existing password by providing old and new password Any other change in the profile (i.e. Name, Gender, Phone Number etc.) can be made without entering a password	R1
B) Organize Wardrobe			
B1	**Viewing Clothing Collection**	Users should be able to view their existing collection of clothes stored in the user database. This is only accessible if a profile is logged in. All clothing items have categories assigned by the ML model. If a profile is not logged in, the user will be prompted to either log in or sign up.	R1
B2	**Adding New Clothing Items**	Users should be able to add new clothing to their wardrobe if they have logged in. To add clothing items, a button will be presented, which will open a pop-up window providing 'Drag and drop' and 'Browse from device' options. If the wardrobe is empty, the application should prompt the user to add their first item. Upon adding a new item, the application should update the user's wardrobe in the database with the new clothing details.	R1
B3	**Deleting Clothing Items**	Users should be able to remove unwanted clothing items from their wardrobe. Again, this option is only available if the user is logged in with a profile. The website should provide a delete option for each clothing item displayed in the collection. Upon deletion, the application should update the user's wardrobe in the database by removing the selected item.	R1
B4	**Add to favourites**	The user will be presented with a button to add clothes to Favourites.	R1
C) Home Page			

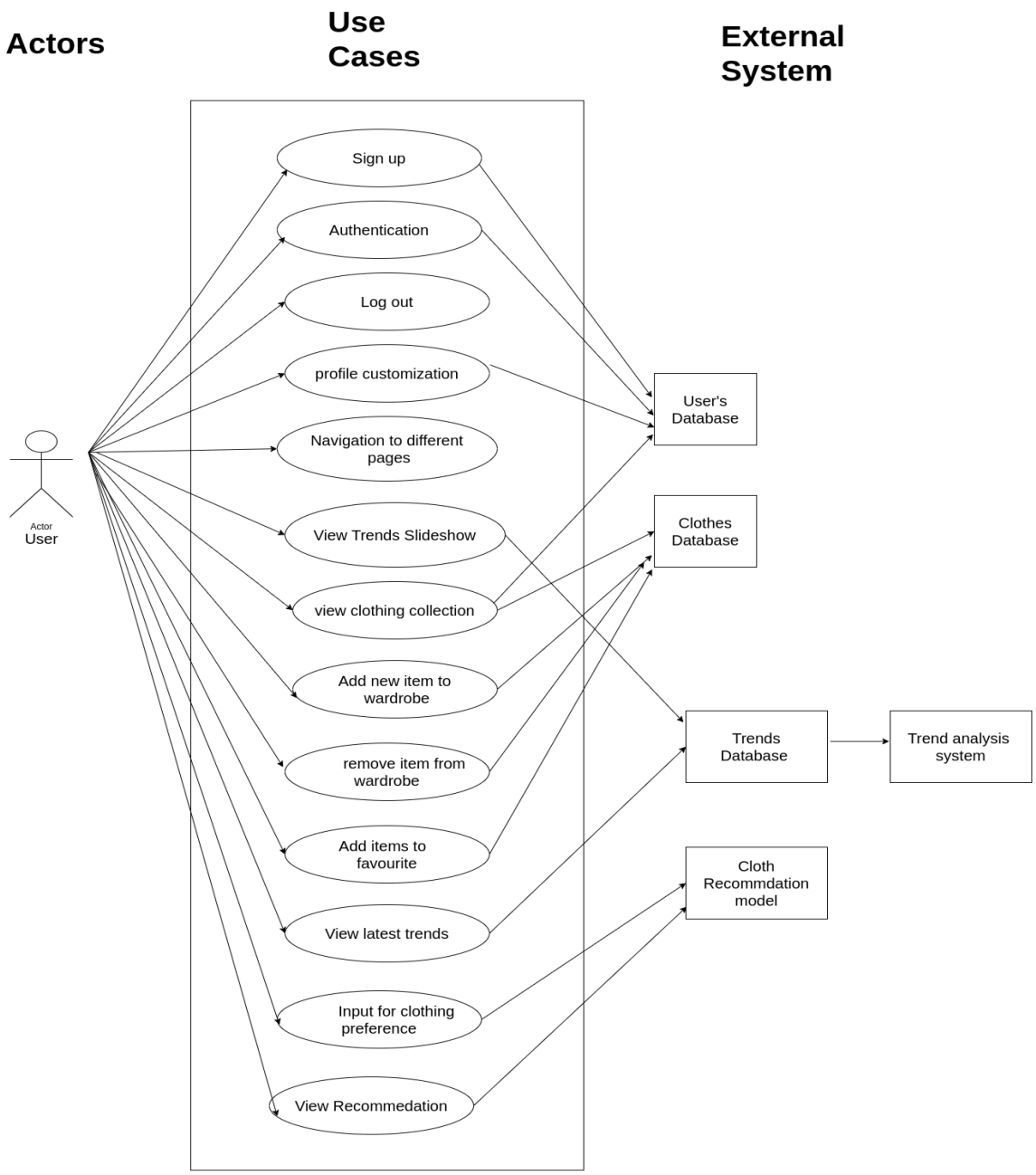
C1	**Navigation Menu**	The Home Page shall feature a compact navigation menu accessible via a hamburger icon or swipe gesture, allowing users to access various sections of the app, including Clothing Collection, Outfit Recommendations, Settings, etc.	R1
C2	**Featured Fashion Trends**	The Home Page shall include a dynamic section showcasing featured fashion trends, displayed as visually appealing cards or tiles.	R2
		D) Trend Analysis	
D1	**Trends page**	On the home page, there is also an option to visit the 'trends' page. On the trends page, the user can see a list of all items that are currently in trend. These items are not from the user's wardrobe, but rather a collection of items that are trendy.	R2
		E) Outfit Recommendation	
E1	**User Input for Clothing Preferences**	Users will provide input regarding their preferred types of clothing, including options such as Formals, Partywear, Casual, Business Attire, Sportswear, Menswear, Womenswear, etc. This option is only available if the user has signed in.	R2
E2	**Display of Recommended Clothing**	Upon processing user input by passing through ML models the system will display recommended clothing items that match the specified criteria. Present recommended clothing items to users in a visually appealing and user-friendly manner, facilitating easy exploration and selection.	R2

Database and ML Model Integration (D)

No.	Use Case Name	Description	Release
		A) Confirmation Page	
DA1	**Feedback Data Structure**	Include fields for a unique identifier, occasion prompting the recommendation, user review, and thumbs up/thumbs down response to be stored in the database.	R2

DA2	**Feedback Storage Endpoint**	Implement an endpoint in the application backend to receive and process user feedback submissions.	R2
		B) User Information and Wardrobe	
DB1	**Sensitive Information Encryption and Storage**	User details such as age, gender, username, and password entered during signup must undergo encryption before storage in the database. Storage of the encrypted user information in the database.	R1
DB2	**Image Storage**	Images uploaded by users for their wardrobe must be securely stored in the database.	R1
		C) ML Model Integration	
DC1	**FastAPI Integration**	Implement FastAPI endpoints to serve as interfaces for communicating with the ML models. Define routes for handling requests related to clothes classification, user recommendation, and fashion trend analysis.	R2
DC2	**Clothes Classification Model Integration**	Develop a FastAPI endpoint to receive image inputs for clothing items and invoke the ML model for classification. Ensure the endpoint returns the predicted category for each clothing item, facilitating organization within the user's wardrobe.	R2
DC3	**User Recommendation Model Integration**	Design a FastAPI endpoint to accept user profile information and selected occasions as input parameters. Integrate the ML model to generate personalized outfit recommendations based on the provided data. Return the recommended outfits to the application backend for display to the user.	R2
DC4	**Fashion Trend Analysis Model Integration**	Create a FastAPI endpoint to trigger the extraction and analysis of new fashion trends from external sources. Implement logic to update the database with the latest trend data obtained from the ML model's predictions.	R2

Use case diagram



Use case description

Use Case Number:	UC-A1
Use Case Name:	Signing Up
Overview:	New users can create an account by providing necessary details such as email, password, name, gender, phone number, and skin tone.
Actors:	User
Pre-condition:	The user is not logged in to the system.
Flow:	Main Flow: 1. User selects the option to create a new account. 2. User fills out the sign-up form with the required information (email, password, name, gender, phone number, skin tone). 3. User submits the form. 4. System validates the entered information. 5. If validation is successful, the system creates a new user account. 6. The newly created profile is automatically logged in to the system.
	Alternate Flow: 5. If the user fails to provide any mandatory information (marked with asterisk), the system displays an error message prompting the user to fill in all required fields.
	Alternate Flow: 5. If the email entered is already associated with an existing account, the system displays an error message indicating that the email is already in use.
Post Condition:	The user's new account is successfully created, and they are logged in to the system.

Use Case Number:	UC-A2
Use Case Name:	Signing In
Overview:	Signing In allows registered users to access their accounts by entering their email and password.
Actors:	User
Pre-condition:	Enter the condition that must be true before the main flow is executed.
Flow:	Main Flow: 1. User selects the option to sign in. 2. User enters their email ID and password.

	3. System verifies the entered credentials. 4. If the credentials are correct, the user is successfully logged in to the system.
	Alternate Flow: 4.1 If the entered email ID is not registered in the system, or the password is incorrect, the system displays an error message indicating invalid credentials. 4.2 The user is prompted to re-enter their email ID and password until the correct credentials are provided.
Post Condition:	The user is successfully logged in to the system.

Use Case Number:	UC-A3
Use Case Name:	User Profile Customization
Overview:	User Profile Customization enables users to modify their account settings and personal information, including updating passwords and making changes to profile details such as name, gender, and phone number.
Actors:	User
Pre-condition:	The user is logged in to the system.
Flow:	Main (success) Flow: 1. User navigates to the profile customization section. 2. User selects the option to change the password. 3. User enters their old password and the new password. 4. System verifies the entered old password and updates the password to the new one if validation is successful. 5. User selects the option to edit other profile information (Name, Gender, Phone Number, etc.). 6. User makes the desired changes to the profile information. 7. User saves the changes.
	Alternate Flow: 2. The user selects the option to change other details. 2.2. User makes the desired changes to the profile information. 2.3 User saves the changes.
	Alternate Flow: 4. If the old password entered is incorrect, the system displays an error message indicating invalid credentials for password change.
Post Condition:	The user's profile settings are successfully updated according to the changes made.

Use Case Number:	UC-A4
Use Case Name:	Log Out
Overview:	The Log Out feature allows users to securely sign out of their accounts, ensuring the protection of their personal information and maintaining account privacy.
Actors:	User
Pre-condition:	The user is logged in to the system.
Flow:	Main (success) Flow: 1. User clicks on the log out button. 2. The system logs out the currently logged-in profile.
	Alternate Flows: None
Post Condition:	The user is successfully logged out from the system, and the session ends.

Use Case Number:	UC-B1
Use Case Name:	Viewing Clothing Collection
Overview:	Enables users to filter through, access and browse their existing collection of clothing items stored in the user database.
Actors:	User
Pre-condition:	The user is logged in to the system.
Flow:	Main (success) Flow: 1. User navigates to the "Clothing Collection" section of the application. 2. The user may select filters for specific clothing categories. 3. The system retrieves the user's clothing collection from the database. 4. The system displays the clothing collection to the user along with categories assigned to clothing items by the ML model.
	Alternate Flow: If a profile is not logged in: 3.1 The user is prompted to either log in or sign up. 3.2 The flow redirects to the login/signup process.
Post Condition:	The user successfully views their clothing collection stored in the user database.

Use Case Number:	UC-B2

Use Case Name:	Adding New Clothing Items
Overview:	Allows users to add new clothing items to their wardrobe. Users can upload photos or the new clothing item, which will be stored in the user database for future reference.
Actors:	User
Pre-condition:	The user is logged in to the system.
Flow:	Main (success) Flow: 1. User navigates to the "Add New Clothing" section of the application. 2. The system presents a button for adding new clothing items. 3. User clicks on the button, opening a pop-up window with options to 'Drag and drop' or 'Browse from device'. 4. User selects the desired option and uploads the image of the new clothing item. 5. If the wardrobe is empty, the application prompts the user to add their first item. 6. Upon adding a new item, the application updates the user's wardrobe in the database with the new clothing details.
	Alternate Flow: None
Post Condition:	The user successfully adds a new clothing item to their wardrobe, and the wardrobe database is updated accordingly.

Use Case Number:	UC-B3
Use Case Name:	Deleting Clothing Items
Overview:	The Delete Clothing Items feature allows users to remove unwanted items from their wardrobe.
Actors:	User
Pre-condition:	The user is logged in to the system and has access to their wardrobe.
Flow:	Main (success) Flow: 1. User navigates to the "View Clothing Collection" section of the application. 2. The system displays the user's collection of clothing items with a delete option for each item. 3. User selects the clothing item they want to remove and clicks on the delete option. 4. The application prompts the user to confirm the deletion. 5. User confirms the deletion, and the selected clothing item is removed from their wardrobe.
	Alternate Flows: None
Post Condition:	The user successfully deletes the unwanted clothing item from their wardrobe, and the wardrobe database is updated accordingly.

Use Case Number:	UC-B4
Use Case Name:	Add To Favourites
Overview:	Enables users to mark specific clothing items in their wardrobe as favorites for quick access and reference.
Actors:	User
Pre-condition:	The user is logged in to the system and has access to their wardrobe.
Flow:	Flow: Main (success) Flow: 1. User navigates to the "View Clothing Collection" section of the application. 2. The system displays the user's collection of clothing items with an "Add to Favorites" button for each item. 3. User selects the clothing item they want to add to favorites and clicks on the "Add to Favorites" button. 4. The application adds the selected clothing item to the user's favorites list.
	Alternate Flow: None
Post Condition:	The user successfully adds the selected clothing item to their favorites list, which is stored in the database.

Use Case Number:	UC-C1
Use Case Name:	Navigation Buttons
Overview:	The Home Page shall have 3 icons to access the major sections of the app, namely User Wardrobe, Trending Clothes Page and Personal Recommendations Page.
Actors:	User
Pre-condition:	Home page must be opened.
Flow:	Main (success) Flow: 1. User clicks on Wardrobe / Trending / Recommended Labelled Button. 2. User is redirected to Wardrobe Page / Trending Clothes Page / Personal Recommendations Page respectively.
	Alternate Flow: 2. If the user is not logged in and presses Wardrobe / Recommendations Button, then user is redirected to the login/sign up page.
Post Condition:	User must be on Wardrobe / Personal Recommendations / Trending Clothes Page.

Use Case Number:	UC-C2
Use Case Name:	Featured Fashion Trends
Overview:	The Home Page shall have pictures of some of the currently trending clothes displayed in the form of tiles. These are a subset of the pictures shown on the Fashion Trends page.
Actors:	User
Pre-condition:	Home page must be opened.
Flow:	Main (success) Flow: 1. User scrolls down on the Home Page. 2. User is able to view the current trending clothes as tiles, with more coming in view after scrolling. 3. After a set number of pictures, the display ends and a “see more” button is displayed. 4. User clicks the “see more” button. 5. User is redirected to the Current Fashion Trends Page.
	Alternate Flow: 2.1 If the user is not logged in, a message prompting the user to log in and a “Log In” button is displayed. 2.2 On clicking “Log In”, user is redirected to Login Page.
Post Condition:	None

Use Case Number:	UC-D1
Use Case Name:	Fashion Trends Page
Overview:	On the trends page, the user can see a list of all currently trending clothes. These items are not from the user’s wardrobe.
Actors:	User
Pre-condition:	Trends Page must be opened.
Flow:	Main (success) Flow: *1.* The user can scroll down on the Trends Page, viewing different clothes which are currently trending under various categories. *2.* User continues scrolling until the pictures run out.

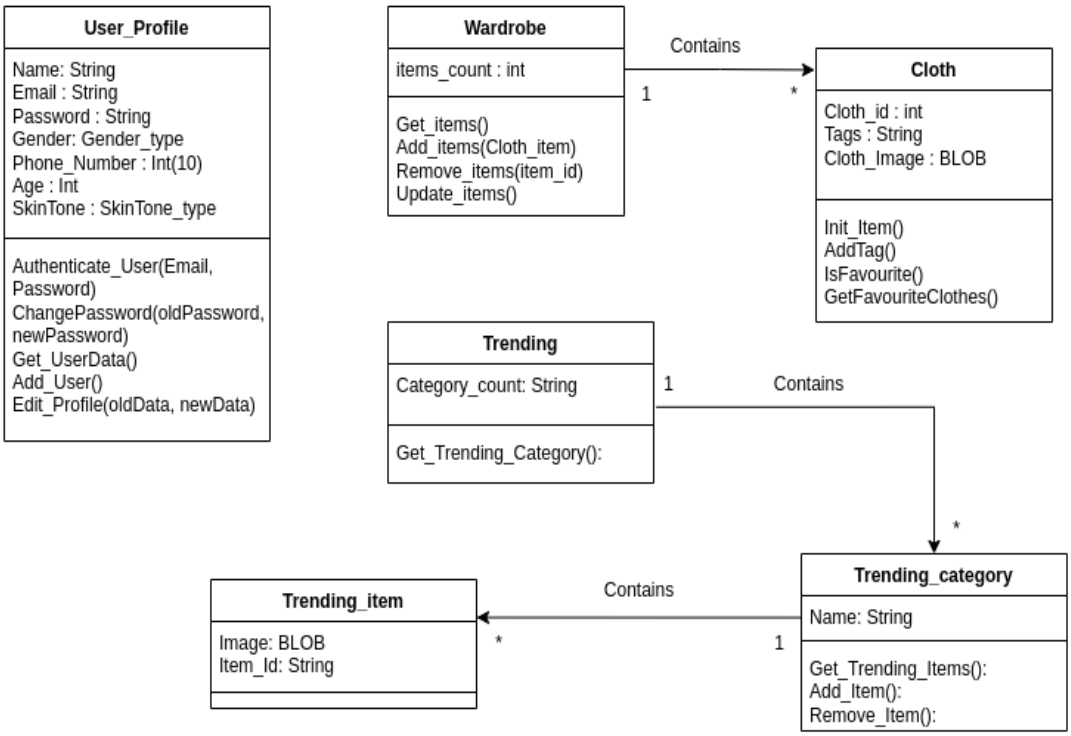
	3. At the end, the user is prompted to find more trending clothes on websites like Marks and Spencer, Myntra, etc.
	Alternate Flows: None
Post Condition:	All items added by user to their wardrobe from current trends page are stored in the DB as a part of their wardrobe.

Use Case Number:	UC-E1
Use Case Name:	Clothing recommendations
Overview:	Taking input from user about the type of clothing he wants to see and displaying them the specified clothes.
Actors:	User
Pre-condition:	The user must have signed in and must have at least 1 item added to his wardrobe.
Flow:	Main (success) Flow: 1. The user chooses the type of clothes they want to see from a drop-down menu consisting of the options 'party-wear', 'casuals', 'formals'. 2. The system shows all the clothes in the user's wardrobe that match the given tag.
	Alternate Flows: If no clothes in the user's wardrobe match the given tag, a message 'No such clothes found' will be shown.
Post Condition:	In either case (main flow or alternative flow) the user will be on the 'get recommendations' page and can change the tag to see other clothes.

UML Class diagram

Software Classes**:**

- User_Profile
- Wardrobe
- Cloth
- Trending
- Trending_category
- Trending_item


User_Profile
Name: String
Email : String
Password : String
Gender: Gender_type
Phone_Number : Int(10)
Age : Int
SkinTone : SkinTone_type
Authenticate_User(Email, Password)
ChangePassword(oldPassword, newPassword)
Get_UserData()
Add_User()
Edit_Profile(oldData, newData)
Wardrobe
items_count : int
Get_items()
Add_items(Cloth_item)
Remove_items(item_id)
Update_items()
Contains
1
*
Cloth
Cloth_id : int
Tags : String
Cloth_Image : BLOB
Init_Item()
AddTag()
IsFavourite()
GetFavouriteClothes()
Trending
Category_count: String
Get_Trending_Category():
1
Contains
*
Trending_category
Name: String
Get_Trending_Items():
Add_Item():
Remove_Item():
Contains
*
1
Trending_item
Image: BLOB
Item_Id: String