

TEAM #3 FLIXTYLE

REQUIREMENT SPECIFICATION

horizontal line

**Team 3** April 26th, 2019

Flixtyle

**Software Requirements Specification**

Introduction to Software Engineering-41

Table of Contents

[**1.** **Introduction** 5](#_Toc8132708)

[**1.1** **Purpose** 5](#_Toc8132709)

[**1.2** **Intended Audience and Reading Suggestions** 5](#_Toc8132710)

[Sub-chapter Descriptions: 5](#_Toc8132711)

[1. Introduction 5](#_Toc8132712)

[2. Overall Description 5](#_Toc8132713)

[3. User Requirements Definition 5](#_Toc8132714)

[4. System Requirements Definition 5](#_Toc8132715)

[5. System Architecture and Models 5](#_Toc8132716)

[6. Appendices 5](#_Toc8132717)

[**1.3** **Product Scope** 6](#_Toc8132718)

[**1.4** **References** 6](#_Toc8132719)

[**2.** **Overall Description** 6](#_Toc8132720)

[**2.1** **Product Perspective** 7](#_Toc8132721)

[**2.2** **Product Functions** 7](#_Toc8132722)

[**2.3** **User Classes and Characteristics** 8](#_Toc8132723)

[**2.4** **Operating Environment** 8](#_Toc8132724)

[**2.5** **Design and Implementation Constraints** 8](#_Toc8132725)

[**2.6** **User Documentation** 9](#_Toc8132726)

[**2.7** **Assumptions and Dependencies** 9](#_Toc8132727)

[**3.** **User Requirements Definition** 10](#_Toc8132728)

[**3.1 General User Stories** 10](#_Toc8132729)

[**3.2**  **Functional Requirements** 10](#_Toc8132730)

[3.2.1 Authentication Page 10](#_Toc8132731)

[3.2.1.1 Create Account Page 10](#_Toc8132732)

[3.2.1.2 SNS First-login Page 10](#_Toc8132733)

[3.2.2 Discovery Page 10](#_Toc8132734)

[3.2.3 Recommendations Page 11](#_Toc8132735)

[3.2.3.1 Recommendations Tab 11](#_Toc8132736)

[3.2.3.2 Search & Filter Popup 11](#_Toc8132737)

[3.2.4 Account Page 11](#_Toc8132738)

[**3.3** **Non-functional Requirements** 11](#_Toc8132739)

[3.3.1 Product requirement 11](#_Toc8132740)

[3.3.1.1 Performance 11](#_Toc8132741)

[3.3.1.2 Security 11](#_Toc8132742)

[3.3.1.3 Efficiency 12](#_Toc8132743)

[3.3.1.4 Usability 12](#_Toc8132744)

[3.3.2 Organizational requirement 12](#_Toc8132745)

[3.3.2.1 Environment 12](#_Toc8132746)

[3.3.2.2 Operation 12](#_Toc8132747)

[**4. System Architecture and Models** 12](#_Toc8132748)

[**4.1** **General Structure of Flixtyle** 12](#_Toc8132749)

[4.1.1 Account Creation 13](#_Toc8132750)

[4.1.2 First Login Setup 13](#_Toc8132751)

[4.1.3 Discovery Loop 13](#_Toc8132752)

[4.1.4 Recommendation Listing 14](#_Toc8132753)

[4.1.5 Adding to “Heart” List 14](#_Toc8132754)

[4.1.6 “Heart” Listing 14](#_Toc8132755)

[4.1.7 Filtering Function 15](#_Toc8132756)

[4.1.8 Show User Account Page 15](#_Toc8132757)

[4.1.9 User Account Information Editing 15](#_Toc8132758)

[4.1.10 Item Classification Processing 16](#_Toc8132759)

[**5.1** **Functional Requirements** 16](#_Toc8132760)

[5.1.1 Account Creation 16](#_Toc8132761)

[5.1.2 First Login Setup 16](#_Toc8132762)

[5.1.3 Discovery Loop 17](#_Toc8132763)

[5.1.4 Recommendation Listing 17](#_Toc8132764)

[5.1.5 Adding to “Heart” List 18](#_Toc8132765)

[5.1.6 “Heart” Listing 18](#_Toc8132766)

[5.1.7 Filtering Function 19](#_Toc8132767)

[5.1.8 Show User Account Page 19](#_Toc8132768)

[5.1.9 User Account Information Editing 19](#_Toc8132769)

[**5.2** **Non-functional Requirements** 20](#_Toc8132770)

[**5.3** **Scenario** 20](#_Toc8132771)

[5.3.1. User Account Scenarios 20](#_Toc8132772)

[5.3.1.1. Sign Up 20](#_Toc8132773)

[5.3.1.2 Login 20](#_Toc8132774)

[5.3.1.3 Check Account Information 20](#_Toc8132775)

[5.3.1.4 Editing Account Information 20](#_Toc8132776)

[5.3.1.5 Membership Withdrawal 21](#_Toc8132777)

[5.3.2. Recommended Scenario 21](#_Toc8132778)

[5.3.2.1. Recommendation 21](#_Toc8132779)

[5.3.2.2. Check the Heart list 21](#_Toc8132780)

[5.3.2.3. Connect to Online-Seller 21](#_Toc8132781)

[**6.** **Appendices** 21](#_Toc8132782)

[**6.1** **Appendix A: Glossary** 21](#_Toc8132783)

[Seller 21](#_Toc8132784)

[User 21](#_Toc8132785)

[Fashion item 22](#_Toc8132786)

[Product 22](#_Toc8132787)

[Recommendations 22](#_Toc8132788)

[CPI 22](#_Toc8132789)

[Discovery 22](#_Toc8132790)

[Save 22](#_Toc8132791)

[“Heart” List 22](#_Toc8132792)

[Swipe 22](#_Toc8132793)

[GUI 22](#_Toc8132794)

[GCP 22](#_Toc8132795)

[**6.2** **Appendix B: Analysis Models** 23](#_Toc8132796)

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
| Initial | 04/26/2019 | Initial version | 0.1 |
|  |  |  |  |

# **1.** **Introduction**

## **1.1** **Purpose**

The purpose of this document is to outline the specifications of the “Flixtyle” application. The document encompasses the goal, characteristics, and the interface of the system, in order to show the workings of the system and its components. It is oriented to both end-users and developers.

## **1.2** **Intended Audience and Reading Suggestions**

This document is written not only for the developers, but also for the end users and other various stakeholders like project managers and testers. It is recommended to know most of the terms in the glossary section before reading through the document.

### Sub-chapter Descriptions:

#### Introduction

An introduction to the purpose of the document is provided, as well as a description of the intended audience and product scope. A short reference list for the open-source software used is provided.

#### Overall Description

The context and origin of the product is explained, along with operating environment, product functions and characteristics.

#### User Requirements Definition

Sample user stories are provided and function and non-functional user requirements are derived and defined.

#### System Requirements Definition

Functional and non-functional system requirements are described from a technical perspective.

#### System Architecture and Models

A general model of the system architecture is provided.

#### Appendices

Various references is provided.

## **1.3** **Product Scope**

Flixtyle is a type of platform business. Platform business is a business model that sells specific services or products based on the social relationships, with a structure which producers and consumers are freely connected in. These platform business models are largely divided into Intermediate platforms and advertising platforms. Intermediate platforms have a profit structure that receives certain fees from producers and consumers. On the other hand, advertising platforms have a profit from advertising after collecting a lot of users.

Flixtyle has a business structure similar with intermediate platform that connects producers and consumers. Flixtyle recommends new products to its users based on the previous Likes from the customer. It utilizes the overlapping community detection to form user groups invisible from the user to get effective recommendations to the user. After recommendation, Flixtyle connects online shopping mall and customer. Through this process, user will be able to get recommendations relevant to them, which saves them time, and also helps find new styles.

However, its structure for profit has to be an advertising platform because it cannot expect fees from large shopping mall sites. Therefore, The system will act as an advertising platform which connects potential customers to partnered sellers, and collect advertisement fee from the sellers on a CPI basis. Flixtyle will obtain user’s favorite fashion items and advertise based on this user data. More specific, Flixtyle’s ads will be designed like existing content in the service which is tailored to users. By this way, Flixtyle can reduces users' reluctance to advertise and maximize the effectiveness of advertising.

In fact, Pinterest, which have similar platform structure with Flixtyle, generated more than 500 million dollars last year, and this year's advertising revenue is expected to exceed 1 billion dollars. According to the Internet Trend Report, 55 percent of Pinterest users make actual purchases through their websites. On the other hands, only 12 percent of Facebook and Instagram user and only 9 percent of Twitter user make actual purchases through their websites. This survey shows the fact that Pinterest’s structure is more successful than other platforms.

## **1.4** **References**

<https://nodejs.org/en/about/>

<https://firebase.google.com/>

# **2.** **Overall Description**

## **2.1** **Product Perspective**

As the global fashion market is increasing in size, the online shopping malls have grown bigger. Shopping malls around the world have tried to get more consumers by providing a variety of technologies and services. These technologies and services are starting to deploy internationally and online customers are not afraid to buy items from other countries. In fact, according to the National Statistical Office, the amount of oversea purchases is increasing. However, it is difficult for customers to navigate the enormous amount of products available. Online shopping is characterized by short customer attention-spans and efficiency is of the essence.

To battle this problem, Flixtyle gathers and organizes products from all over the world, displaying them on a single application. Moreover, Flixtyle provide an automated and intelligent recommendation system that can classify fashion items which the user likes and provide similar items as recommendations. This enhances efficiency, reduces the time needed for discovering likeable fashion items and is overall an entertaining activity.

## **2.2** **Product Functions**

The application will begin by checking whether the user session is authenticated. If it isn’t, the user will be provided a choice between logging in by inputting an ID and password, making a new account or logging in through a third-party SNS (namely Facebook, Twitter and Google).

After login, the application will consist of three main tabs: the Discovery tab, the Recommendations tab and the Account tab. The application will launch in the Discovery tab by default.

The Discovery tab is the main part of the application and the data necessary for making recommendations will originate from this tab. Various fashion items gathered from participating online shopping malls have been gathered and is shown to the user in sequence. This gathering session will execute periodically to ensure that the items are still available.

When a user goes through a session of swiping, he/she must choose to swipe left or right depending on their preference for the specific item. Swiping left will signify that the user doesn’t like the item and swiping right will signify that the user likes the item. Swiping up at an item will add the item to the “Heart” List, a list of the customer’s favourite items. The user can at any time access this list and press the items saved there to get redirected to an online shopping mall selling the item. Then, according to the user’s preferences, the application will generate tailored recommendations. This is done by saving the liked items to a database and applying a clustering algorithm to the data.

These recommendations can be found in the Recommendations tab. This list will be updated every time the user signify that he/she likes a particular fashion item and will also regularly check the validity of the items (i.e. that the item is still available). The user may press the items to get redirected to an online shopping mall selling the item.

In the Account tab, the user will have the options of changing his/her password, linking accounts (in case of third-party login) or changing the account information. Also, user will view “Heart” list which contains user’s favorite items that swiping up in the discovery tab.

Models of the data flow and system architecture can be found in chapter 4.

## **2.3** **User Classes and Characteristics**

The main user class will consist of users who wants to use the application to discover new fashion items and are interested in buying them. This class is characterized by actively using the discovery and recommendation functionality and the functionality of clicking the items to be redirected to the shopping site.

A second user class is similar, but will mainly use the application for entertainment and inspiration purposes. These users will use the discovery and recommendation functionality actively, but are less likely to actually wanting to buy the items (i.e. clicking the items to be redirected). For this user class, the requirement of frequently checking the validity of the items in the recommendation list is less important.

We may also divide the users as to how frequently they’ll use the application, but as the requirements will remain more or less equal, this is not necessary.

## **2.4** **Operating Environment**

**2.4.1 Back-end**

A VPC with Ubuntu 18.04 LTS installed will be used as the server. It will have node.js 10.15.3 LTS and Python 3.5.7 installed. Node.js will handle the networking, and Python and the classification algorithm based on Python are to be installed in the VPC hosted by GCP.

Firebase (a Google Cloud product) will be used as the database of the application.

**2.4.2 Front-end**

React version 16.8 will be used to build the user interface. The interface should run in any modern browser, such as Chrome, Safari and Firefox. The user should access said browsers through a touch-enabled device.

## **2.5** **Design and Implementation Constraints**

Using node.js requires javascript. This limits the front-end implementation tools, but as the plan always was to use React (which uses javascript), this is not a problem.

Firebase is a Google cloud platform and will naturally result in costs if the application is to grow in customers in the long run. Firebase is free of charge for up to 1GB of hosting storage, 100 simultaneous connections and the data transfer of 10GB/month. The computation and access speed is also somewhat reduced in the free version. However, the paid versions of Firebase are reasonably affordable, considering the long-term income of the application.

As the application will use clustering algorithms to make the recommendations, Big Data analytics tools are needed. Firebase offers built-in machine learning tools that may handle this functionality. This tool is also available in the free version of Firebase. Alternatively, Python will be used to handle the analytics, as there are lots of libraries and documentation available for Python online. This doesn’t notably restrain the system operation.

In addition, as the application will be a Progressive Web Application (PWA), hardware constraints on the client-side are restrained to having a touch-enabled device with Internet connection, running on any modern browser. On the host-side, no specific hardware constraints exists as we will use a cloud product for the back-end of the system.

## **2.6** **User Documentation**

There won’t be any user manuals or tutorials accompanying the product. The web application will be intuitive and will provide hints by overlays and icons to guide users through the use of the application. We are confident that this deployment strategy will be satisfiable.

## **2.7** **Assumptions and Dependencies**

The application will rely on the third-party product Firebase for the back-end, and will therefore be dependant on the Firebase-platform being online and available. As Firebase is a well-tested Google product, we don’t expect any major problems emerging from this dependency. Nevertheless, this will be the main external dependency that may greatly affect the applications operation.

In addition, the application will rely on the online shopping malls being online, displaying correct information and generally operating as expected. The shopping malls should update their catalogues regularly and make sure that the information they display is precise.

Lastly, the application relies on the user device being connected to the Internet. Some data will be cached and stored locally (namely the upcoming Discovery-items), but the application will in general require a stable Internet-connection.

# **3.** **User Requirements Definition**

## **3.1 General User Stories**

User story 1: A User logs in to the application through Facebook credentials and proceeds to swipe right to indicate the she likes items.

User story 2: Users get recommendations for the proper product based on their consumption history.

User story 3: As a clothing shopper, I can read reviews of selected clothes to help me decide to buy it or not.

User story 4: As a student with not a lot of money, I want to see clothes that are in my budget.

## **3.2** **Functional Requirements**

### 3.2.1 Authentication Page

The authentication page is the first page the user is presented with. The user can create an account in our system and sign in with the previously made account or social media account, such as Google, Facebook, and Twitter. The user should also be able to reset their password with their ID.

#### 3.2.1.1 Create Account Page

The user will fill out six fields: E-mail, password, password confirmation, gender, age, and country of residence. The email, password, and password confirmation fields will be mandatory. The three fields gender, age, and country of residence are for better recommendations and does not necessarily have to be filled out. Users will be able to change their minds about the three fields later on in their account page. The users will be able to connect to their SNS accounts later on.

#### 3.2.1.2 SNS First-login Page

When the user has created an account using SNS account, they will be asked to fill out the gender, age, and country of residence. It should be possible to skip this step if the user does not want to provide these information. However, if the user ever changes his or her mind, they can go to their account page and provide the information for left out parts.

### 3.2.2 Discovery Page

The user is presented with products the recommendation system finds appropriate to find more about the user’s likes and dislikes.The user then can swipe the item left, right, or up, depending on whether the user likes the product or not. Swiping left will indicate that the user is not interested and will show a frowny face in the back. Swiping right will indicate that the user is interested in the product and show a smiley face in the back. Swiping up will indicate that the user is very interested in buying the product, and save it to the “Heart” List. Basically, swiping left will discard the product, swiping right will keep the product in the recommendation page, and swiping up will keep the product in the recommendation page and also save it in the “Heart” List. The user will be notified every few likes that the system has brought new Recommendations to the user.

### 3.2.3 Recommendations Page

The Recommendations page will contains recommendations provided by the Community Detection algorithm.

#### 3.2.3.1 Recommendations Tab

The user is presented with multiple fashion items that the recommendation system approximates to match the user’s taste, as well as the items that the user has swiped right or up in the Discovery page. The user will be able to able to scroll through the items, and click on the items to be redirected to the product page of the seller. Moreover, the user can save favorite items in “Heart” list by pressing heart. It is also possible to open the “Search & Filter Popup” from this tab.

#### 3.2.3.2 Search & Filter Popup

The “popup” will include every possible filtering option, such as price range and product category. After applying these filters, a filtered list of products will be shown.

### 3.2.4 Account Page

In the account page, the user is able to view the email, linked SNS accounts, as well as age, gender, and country of residence if the user has provided the information. The actions the user will be able to take are update/delete age, gender, and country of residence, change password, and link new SNS account. Moreover, the user can scroll down to view the “Heart” list, which will contain all the items the user has saved while swiping up in the Discovery page, or pressed heart on the Recommendations page.

## **3.3** **Non-functional Requirements**

### 3.3.1 Product requirement

#### 3.3.1.1 Performance

After the user has exited the Discovery page after swiping some items, the user should be able to see the products that they have swiped right and up for in the recommendation page right away, along with new recommendations brought by the system. The app should not leave out any of the items and keep up with all the delete and “Heart” actions. There should be no mistakes in the product information and no matter how many products the app has to show, it should be able to apply the app UI well.

#### 3.3.1.2 Security

The app shall provide basic security and respect privacy concerns and relevant regulations such as GDPR and similar regulations. Passwords should be stored using a cryptographic hash function, and login information should not be leaked.

#### 3.3.1.3 Efficiency

The application should be lightweight and not take up more than 10MB of local storage. The classification algorithm runs every 12 hours, and should complete within 11 hours in order to allow for refresh. Also, the recommendations given out to each users should be calculated within 0.5 seconds (excluding network delays). First-time users should be able to load the page within 5 seconds, and returning users should be able to load within 2 seconds. The images for the Recommendations and Discovery should be pre-emptively cached to ensure low load time when using the functionality. Code refactoring should be performed frequently to ensure maintainability.

#### 3.3.1.4 Usability

The application should be easy to use for all people. The UI should be intuitive enough for everybody to understand so that the application can be used without extensive explanation on the UI.

### 3.3.2 Organizational requirement

#### 3.3.2.1 Environment

The application should be able to operate correctly on all devices with a touch input running a modern web browser which supports PWA.

#### 3.3.2.2 Operation

The application interacts with the user in real-time, so the server should be ready to respond to the requests made by the application on any time. Therefore, the classification process and the recommendation providing server should be able to run in parallel.

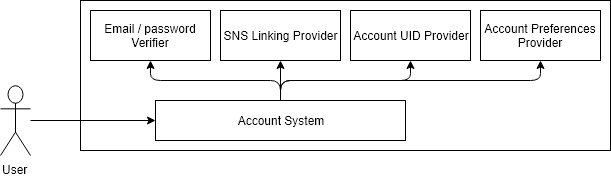
# **4. System Architecture and Models**

## **4.1** **General Structure of Flixtyle**

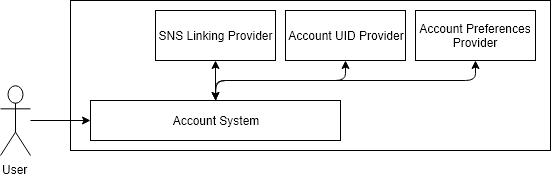
The system mainly consists of two parts: backend, and the frontend. The backend runs algorithms to classify users based on their likes and recommend liked items from other users of the same group. The application’s GUI will consist of four main pages; the login page, the account page, the discovery page and the recommendations page.

After login, the user will be presented to the three main functional pages. The account page, the discovery page and the recommendation page. The account page will contain functionality for changing the password, gender, age and country of residence, in addition to an option to link accounts (if the user has logged in previously through another authentication method).

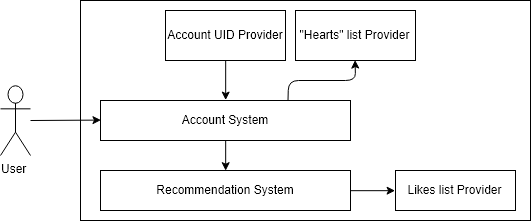
### 4.1.1 Account Creation



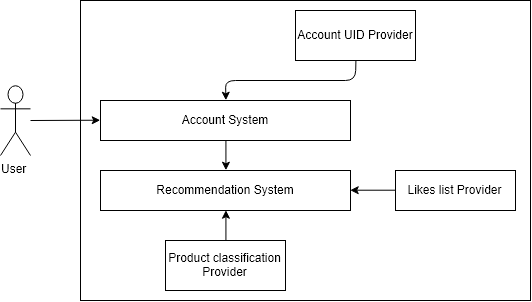
### 4.1.2 First Login Setup



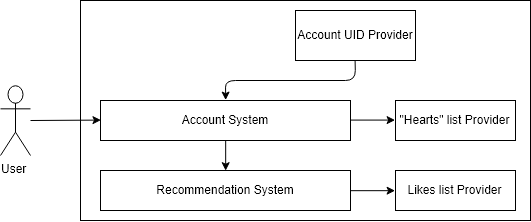
### 4.1.3 Discovery Loop



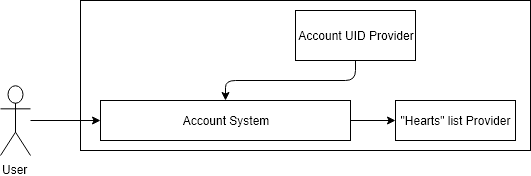
### 4.1.4 Recommendation Listing



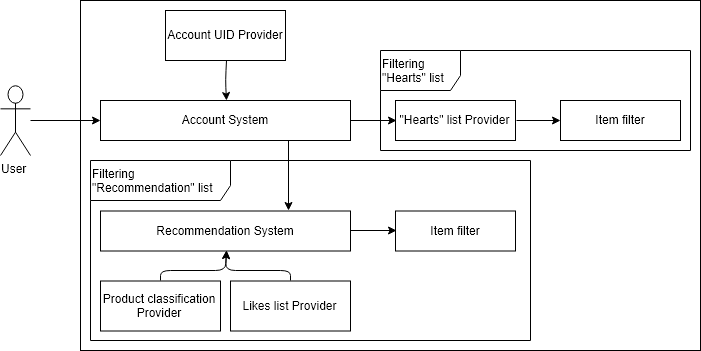
### 4.1.5 Adding to “Heart” List



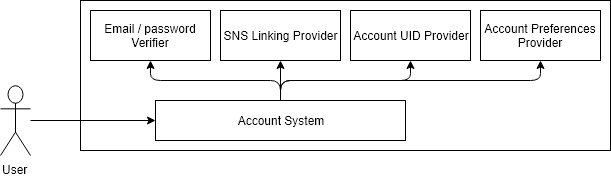
### 4.1.6 “Heart” Listing



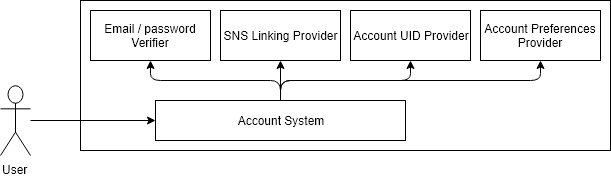
### 4.1.7 Filtering Function



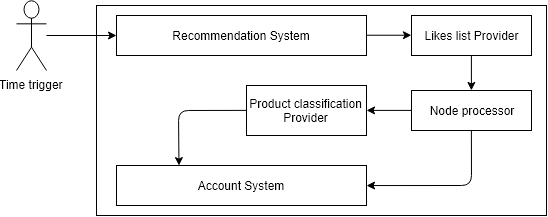
### 4.1.8 Show User Account Page



### 4.1.9 User Account Information Editing



### 4.1.10 Item Classification Processing



**5.** **System Requirements Definition**

## **5.1** **Functional Requirements**

### 5.1.1 Account Creation

|  |  |
| --- | --- |
| Function | Account creation |
| Description | Creating a user account |
| Input | User account information |
| Output | Authenticated token for client |
| Action | The user fills input fields for gender, age and country of residence, email, password and password confirmation and submits it, sending the request to the server where the request is validated and if the validation passes, the user account is created in the database and an authentication token is sent to the user. |

### 5.1.2 First Login Setup

|  |  |
| --- | --- |
| Function | First login setup |
| Description | Updates the account information for user |
| Input | User information or nothing |
| Output | Updated user account information |
| Action | The user fills input fields for gender, age and country of residence and submits it, sending the request to the server where the request is validated and if the validation passes, the user account is updated in the database. |

### 5.1.3 Discovery Loop

|  |  |
| --- | --- |
| Function | Discovery loop |
| Description | Loop where user is presented with a fashion item and will swipe either left, right, down or up depending if the item is to their tastes. |
| Input | Fashion item the recommendation system deems that it is beneficial to know if the user likes it or not. |
| Output | Updated user likes database |
| Action | Fashion items are requested from the server until the buffer of these items is filled. These images are provided by the recommendation system and they are images that the system finds are needed to know more about. The user is presented one of these fashion items and will swipe in a direction depending if they are interested in the item or not. This swipe will act as a request to the server, where the database for the user’s likes is updated. Then the loop will start from the beginning. |

### 5.1.4 Recommendation Listing

|  |  |
| --- | --- |
| Function | Recommendation listing |
| Description | Fashion items based on the recommendation system are listed to the user. |
| Input | Fashion items the recommendation system classifies as similar to the user’s likes. |
| Output | Listing of fashion items in the application window |
| Action | Some amount of fashion items is requested from the recommendation system and are listed in the application view. The user then can scroll down in the window to view more items and when the listing reaches the end of the scrolling space, the application will request for new recommendations. The user can click on a heart shaped button on the fashion item to save it to their “Heart” list |

### 5.1.5 Adding to “Heart” List

|  |  |
| --- | --- |
| Function | Adding to “Heart” list |
| Description | User can add an item to the “Heart” list. |
| Input | The fashion item the user has “Hearted”. |
| Output | Updated list of user’s “Heart” list. |
| Action | The user can click on a heart-shaped button in the Recommendation Listing or swipe up or down in the Discovery Loop which will send a request to the server which adds the fashion item to the user’s “Heart” list. |

### 5.1.6 “Heart” Listing

|  |  |
| --- | --- |
| Function | “Heart” listing |
| Description | Fashion items the user has added to their “Heart” list. |
| Input | Fashion items in user’s saved “Heart” list. |
| Output | Listing of fashion items in the application window. |
| Action | Fashion items the user has saved in their “heart” list are listed in the application view. The user then can scroll down in the window to view more items. The user can click on the fashion item which will open up a website location in a browser where the item was scraped from. |

### 5.1.7 Filtering Function

|  |  |
| --- | --- |
| Function | Filtering function |
| Description | Listed fashion items filtering option. |
| Input | User’s filtering options |
| Output | Listing of filtered fashion items in the application window. |
| Action | Fashion items listed in the window can be filtered by clicking the filter button in the upper part of the window and a filtering popup will appear, where the user can filter the fashion items by category or price. When the apply button is pressed, a request to the server is sent where a list of filtered fashion items from the previously mentioned list will be listed in the window. |

### 5.1.8 Show User Account Page

|  |  |
| --- | --- |
| Function | Show user account page |
| Description | User account page where user account’s information is shown. |
| Input | User account’s information from database. |
| Output | Listed user account information. |
| Action | The user account’s settings, email, linked SNS accounts, age, gender and country of residence are shown in the user window. |

### 5.1.9 User Account Information Editing

|  |  |
| --- | --- |
| Function | User Account Information Editing |
| Description | User account information updating function in user account page. |
| Input | User’s input, either password, email, linked SNS accounts, age, gender and country of residence. |
| Output | Updated user account information in database. |
| Action | The user account’s settings can be edited in the window, and if the changes are submitted, a request is sent to the server where the request is validated and when the validation passes, the user account’s information is updated. |

## **5.2** **Non-functional Requirements**

The classification algorithm runs every 12 hours, and should complete within 11 hours in order to allow for refresh. Also, the recommendations given out to each users should be calculated within 0.5 seconds (excluding network delays).

First-time users should be able to load the page within 5 seconds, and returning users should be able to load within 2 seconds.

The images for the Recommendations and Discovery should be pre-emptively cached to ensure low load time when using the functionality.

Code refactoring should be performed frequently to ensure maintainability.

## **5.3** **Scenario**

### 5.3.1. User Account Scenarios

#### 5.3.1.1. Sign Up

If customer doesn’t have a flixtyle account, customer can go to the sign up screen by clicking on the sign up button on the login screen. The user must enter information about ID, PW, PW confirm, gender, age and country of residence. If all the information has been entered and valid, the information is sent to the server and the Sign up is completed.

#### 5.3.1.2 Login

Log in to our app (Flixtyle) by typing ID and PW. Login is successful if it is valid against the login database stored in the server. If it is invalid, an error message is displayed. If login is successful, go to main page.

#### 5.3.1.3 Check Account Information

Click the Settings button on the main page and select Account Information to proceed to the Account Information page. You can see the account information entered at the time of registration on this page.

#### 5.3.1.4 Editing Account Information

If you click the edit button on the account information page, you can edit the account information except the ID. When modifying the PW, it must go through the PW confilm process. After editing the information, press the OK button and the changes are sent to the server and the editing is completed.

#### 5.3.1.5 Membership Withdrawal

If you click the withdraw button on the account information page, you will be asked to confirm your intention to withdraw. If you are sure you want to unsubscribe, delete the account.

### 5.3.2. Recommended Scenario

#### 5.3.2.1. Recommendation

The user selects the type of fashion item desired by the user. This type of fashion item is recommended by the recommendation system of the server. If user like or does not like the item, user can swipe it left or right. Products that users like are stored in the user’s 'Heart list'. If the user is satisfied or the server has provided enough recommendations, the recommendation is terminated.

#### 5.3.2.2. Check the Heart list

The user can click the Account tab to view his saved Heart list. The Heart list has a filter. Users can filter and sort the products in their Heart list using filters.

#### 5.3.2.3. Connect to Online-Seller

If user has a product on the Heart list that user would like to buy, then the user can go to the sales site if he or she wants to purchase the recommended product. In this process, a predetermined advertising fee is paid to us from the seller.

# **6.** **Appendices**

## **6.1** **Appendix A: Glossary**

###### Seller

The affiliates to the system who sells products on the Flixtyle service.

###### User

Customers who use Flixtyle services to purchase products.

###### Fashion item

Wearable items. Limited to clothing, hats, swimming suits

###### Product

A fashion item the seller has registered to the system

###### Recommendations

Products which are shown to relevant customers and acts as an advertisement to the user

###### CPI

Cost Per Interaction, one is charged with fee proportional to the number of interactions to the advertisement

###### Discovery

The process of verifying whether the user is actually interested in the products the system predicts.

###### Save

The act of adding a product to the “Heart” List, so that the user will be able to quickly find the product in the future.

###### “Heart” List

The list of the products that a user has saved.

###### Swipe

The act of swiping across the touchscreen.

###### GUI

Graphical User Interface

###### GCP

Abbreviation for “Google Cloud Platform”, a cloud service provided by Google

*Authentication*

The process of verifying the user’s identity.

*Request to the server*

A message sent with the HTTP protocol to the server with information from client.

*Validation*

The action of verifying that the request sent to the server is correctly formatted and doesn’t contain harmful information.

## 

## **6.2** **Appendix B: Analysis Models**

