Shoppick

Design specification

Student Number	Name
2013310500	Heesung Kim
2016315382	Seunghoon Lee
2017312668	Hyunjee Shin
2019319366	Misael Zapata

Contents

1. Preface

1.1 Objectives

In this section, we define the expected readership of the document, and briefly introduces the content of each chapter.

1.2 Readership

This document is written for project manager, project team and development team. Also, this document is for the end users and other various stakeholders who is involved in the support, maintenance of the system.

1.3 Document Structure

A. Preface

Preface talks about expected readers and introduces the structure of the document. Also, it describes the purpose and outline of each content when introducing the architecture.

B. Introduction

Introduction talks about the UML(Unified Modeling Language) which is used for system architecture and the diagrams and tools used. Also, it describes the scope of out system “Shoppick”.

C. System Architecture

System Architecture talks about general explanation of the “Shoppick” system we are going to develop. We explain the overall architecture of the system.

D. Protocol Design

Protocol design talks about the communication of the subsystems. The basic format of protocol is the JSON(JavaScript Object Notation) for server communication and the explanation is based on the parameters of each protocol.

E. Database Design

We wrote the database design based on the database requirement specification in the requirement specification. In database design, we rewrite the requirements modified from the original requirements. According to that requirements, we make the ER diagram. Based on that, we write the relational schema and the SQL DDL.

F. Testing Plan

Testing plan talks about the test policy and the test cases. In this section, we plan ahead the processes that verify whether the system runs on intended purpose. Testing policy describes a step-by-step plan of testing. Test cases determine the actors based on the example cases and describe the expected input and output actions of system.

G. Development Environment

Development environment talks about the development environments for developing front end and back end such as programming language and IDE.

H. Development Plan

Development plan talks about the plan of our development. Describe the environment of front end and back end. Also we explain our schedule for the development details.

I. Index

The index of the objects used in this document including pictures, tables and diagrams.

J. Reference

Describe a list of documents referenced in writing this document.

1.4 Version history

This table is for version history. It shows each update of this document.

Version	Date	Detail
0.0	2019.11.02	표지, Contents 작성
1.0	2019.11.03	Preface, Introduction 작성
2.0	2019.11.04	System Architecture, Protocol Design 작성
2.1	2019.11.06	System Architecture 수정
3.0	2019.11.07	Database Design 작성
4.0	2019.11.09	Testing Plan, Development Plan 작성
5.0	2019.11.10	Index, Reference 작성

2. Introduction

2.1 Objectives

This chapter explains UML used for system architecture and the development tools used for writing various diagrams of database models.

2.2 Applied Diagram

A. UML

Figure 1. UML logo

UML is a general-purpose modelling language consisting of an integrated set of diagrams developed to help system and software developers for specifying, visualizing, constructing, and documenting the artifacts of software system. The goal of UML is to provide a standard notation that can be used by all object-oriented methods and to select and integrate the best elements of precursor notations.

There are 13 diagrams in UML. We are going to use 6 diagrams including package diagram, deployment diagram, class diagram, state diagram, and sequence diagram. Also we will use ER diagram to express the database.

A general overview of the UML diagrams is provided below.

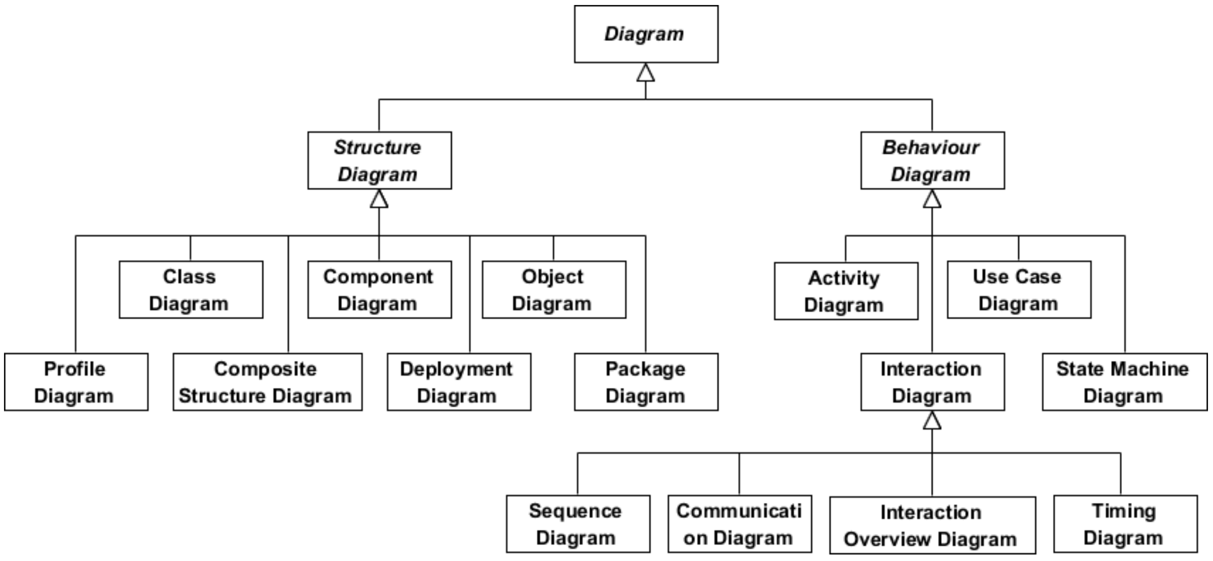

Diagram 1. UML diagram overview

B. Package Diagram

Package Diagram shows structures of the designed system at the level of packages. Package, packageable element, dependency, element import, package import and package merge are drawn in a package diagram.

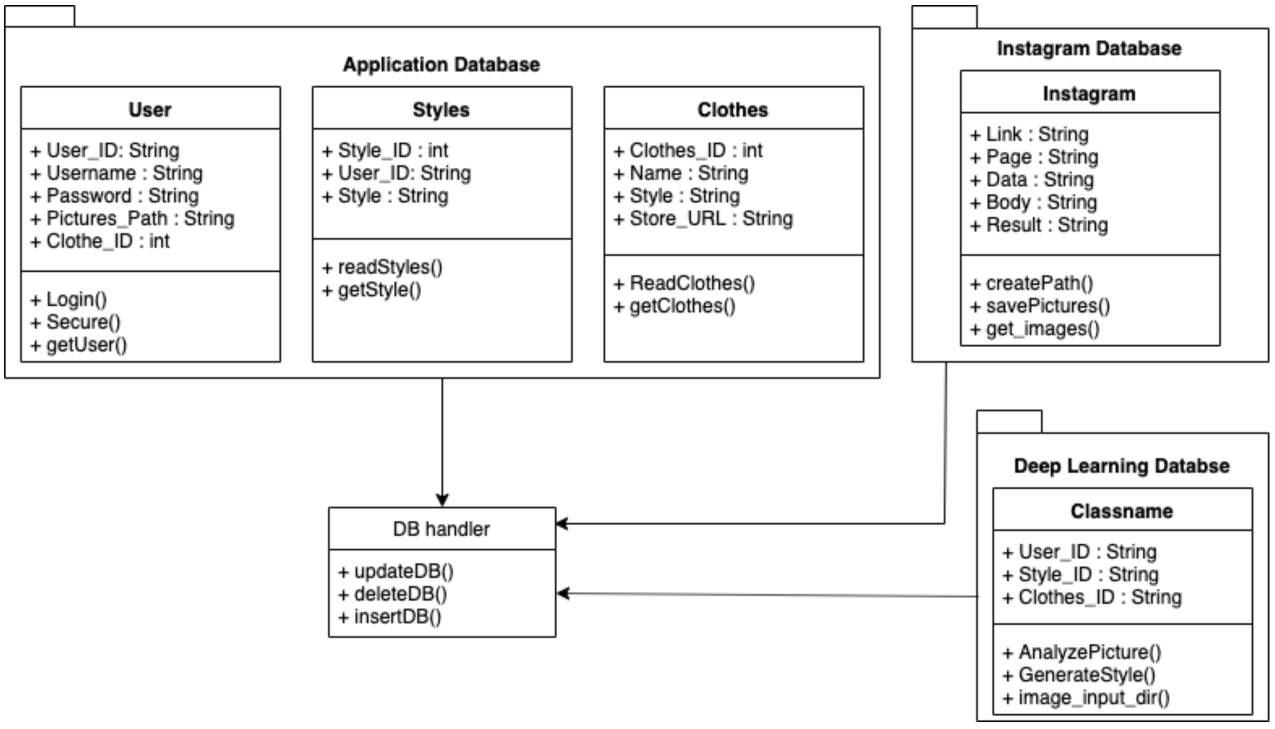

Diagram 2. Package Diagram

C. Deployment Diagram

Deployment Diagram expresses the communication between the hardware components of a system and the placement of software files on that hardware. Artifacts represent concrete elements in the physical world that are the result of a development process. It models the run-time configuration in a static view and visualizes the distribution of artifacts in an application. In most cases, it involves modeling the hardware configurations together with the software components that lived on.

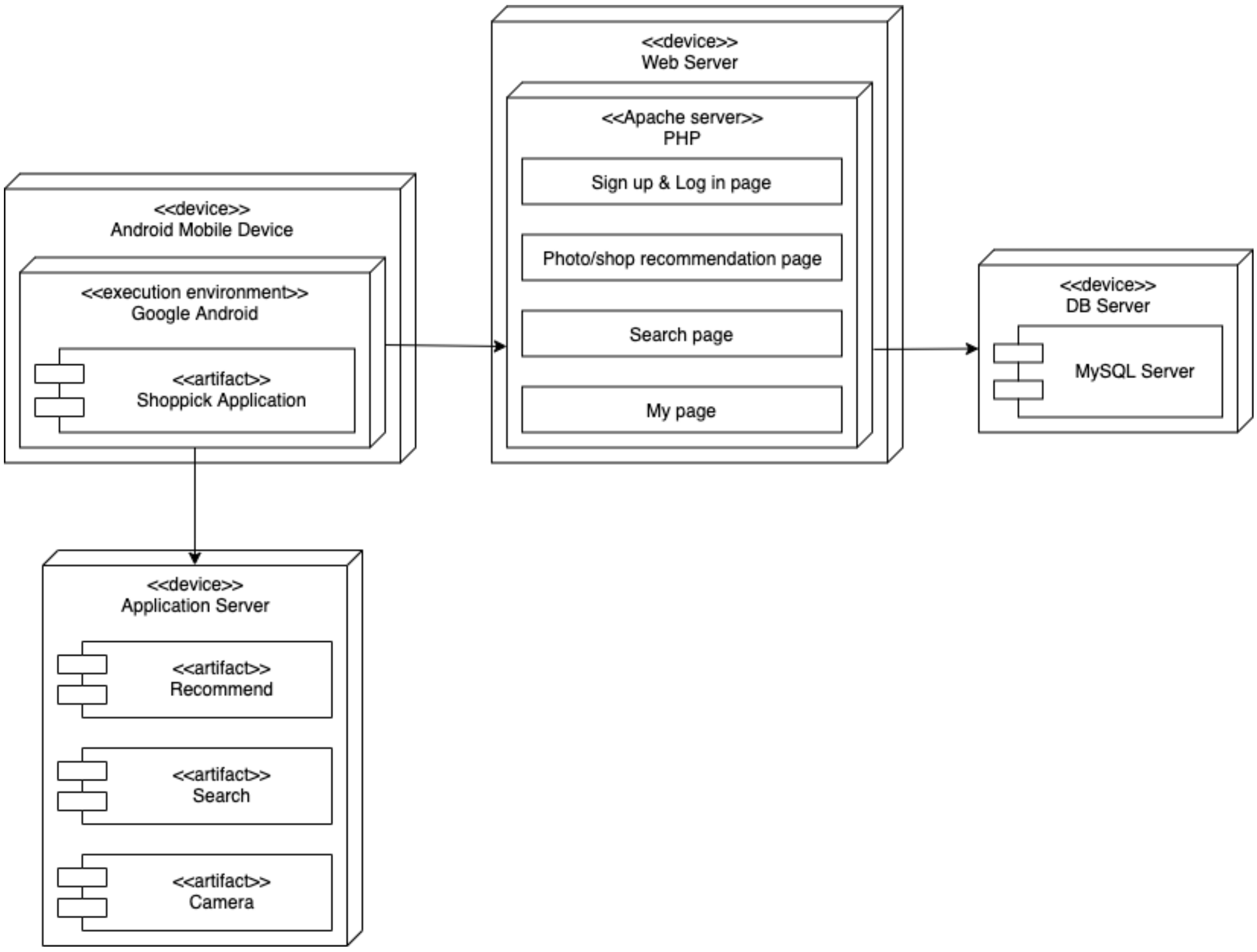

Diagram 3. Deployment Diagram

D. Class Diagram

Class diagram is a central modeling technique that runs through nearly all object-oriented methods. This diagram describes the types of objects in system and various kinds of static relationship between them.

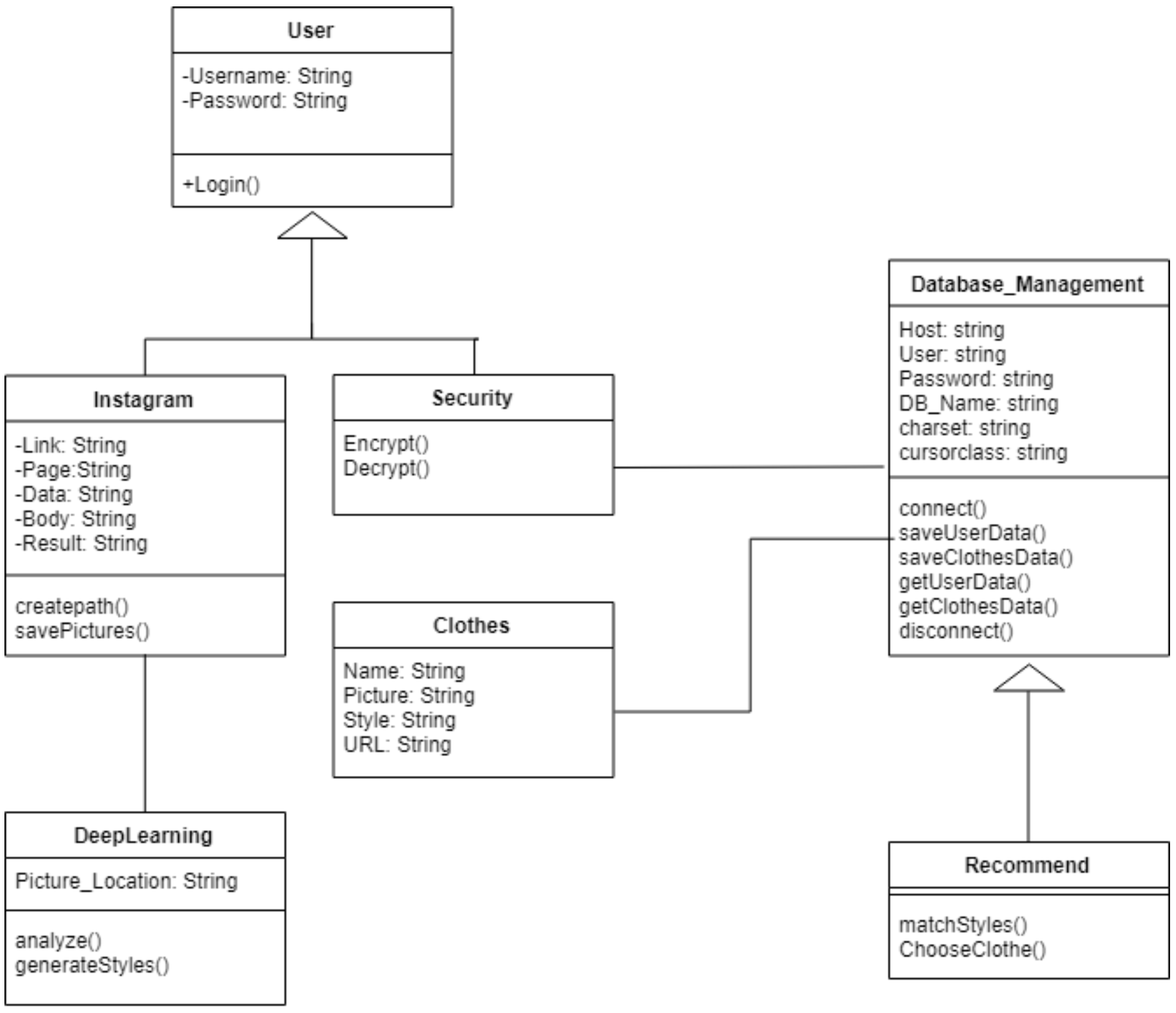

Diagram 4. Class Diagram

E. State Diagram

State diagram is a type of diagram used in UML to describe the behavior of systems. State diagram requires that the system described is composed of a finite number of states. It helps to visualize the entire lifecycle of objects and thus help to provide a better understanding of state-based systems.

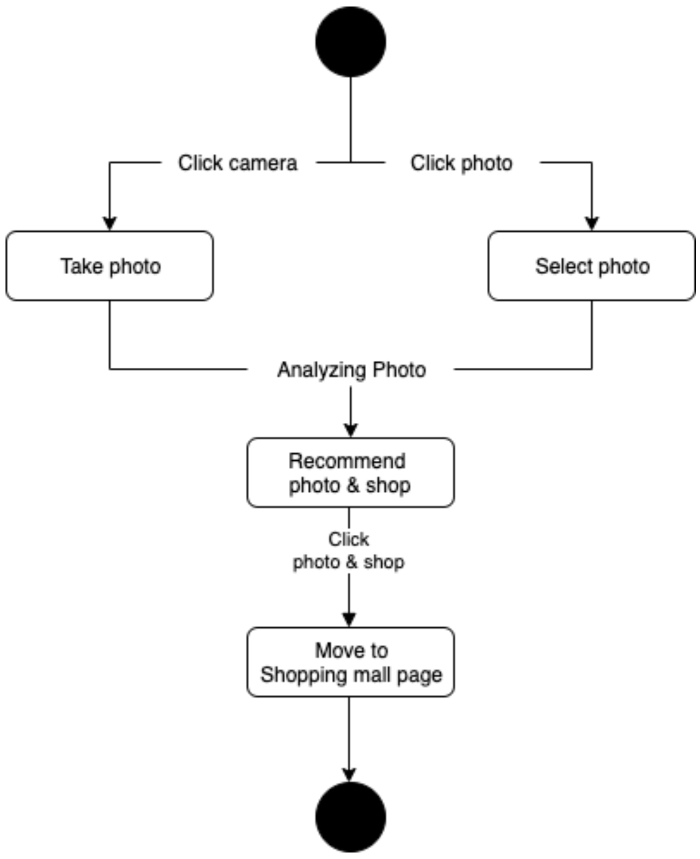

Diagram 5. State Diagram

F. Sequence Diagram

Sequence diagram models the interaction between objects based on a time sequence. It shows how the objects interact with others in a particular scenario of a use case. It is also called event diagram or event scenario. It describes how and what order the objects in a system function.

An actor in a UML diagram represents a type of role where it interacts with the system and its objects. It is important to note here that an actor is always outside the scope of the system we aim to model using the UML diagram.

A lifeline is a named element which depicts an individual participant in a sequence diagram. So basically each instance in a sequence diagram is represented by a lifeline. Lifeline elements are located at the top in a sequence diagram.

Communication between objects is depicted using messages. The messages appear in a sequential order on the lifeline. Lifelines and messages form the core of a sequence diagram.

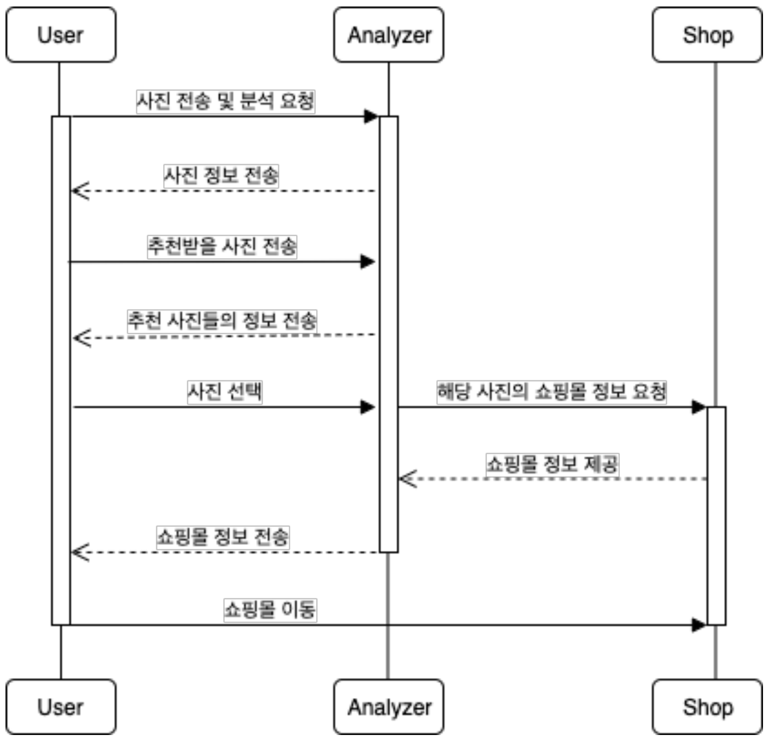

Diagram 6. Sequence Diagram

G. ER Diagram

ER diagram is a high-level conceptual data model diagram. Entity-relationship model is based on the notion of real-world entities and the relationship between them. ER modeling allows analyze data requirements systematically to produce well-designed database.

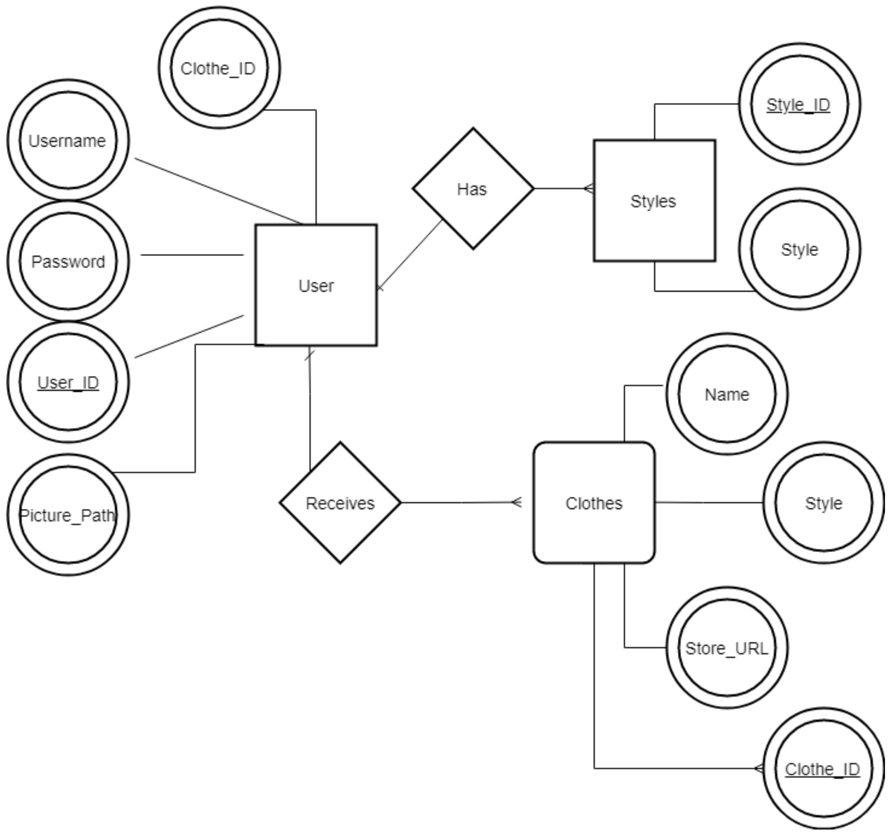

Diagram 7. ER diagram

2.3 Applied Tool

A. Android Studio

We use android studio to develop the front-end. Android Studio is the official integrated development environment for Google's Android operating system and designed specifically for Android development. It is available for download on Windows, macOS and Linux based operating systems.

Figure 2. Android Studio logo 1

Figure 3. Android Studio logo 2

B. Adobe photoshop

We use Adobe photoshop for making "Shoppick" logo. Adobe Photoshop is software that is extensively used for raster image editing, graphic design and digital art.

Figure 4. Adobe photoshop logo

C. PowerPoint

We use the power point for making UI design and presentation. Microsoft PowerPoint is a presentation program. It offers users many ways to display information from simple presentations to complex multimedia presentations.

Figure 5. PowerPoint logo

D. draw.io

We use draw.io for making various diagrams including package diagram, deployment diagram, state diagram and sequence diagram. draw.io is an open source technology stack for building diagramming applications, and the world's most widely used browser-based end-user diagramming application. It provides basic tools to draw diagrams including UML diagrams.

Figure 6. draw.io logo

2.4 Project Scope

“Shoppick” application is for people who are interested in fashion to get recommend clothes similar to their preference and find the shopping malls selling those clothes. “Shoppick” system is composed of three subsystems ; User Management system, Instagram Crawling system, Recommendation/Store matching system, Tag Searching system.

3. System Architecture – Overall

3.1 Objectives

This chapter shows the overall architecture of the "Shoppick" system we are going to develop. The overall structure of each system is represented as a block diagram. The relationship between subsystems and how the system actually used are described as a Block diagram and deployment diagram.

3.2 System Organization

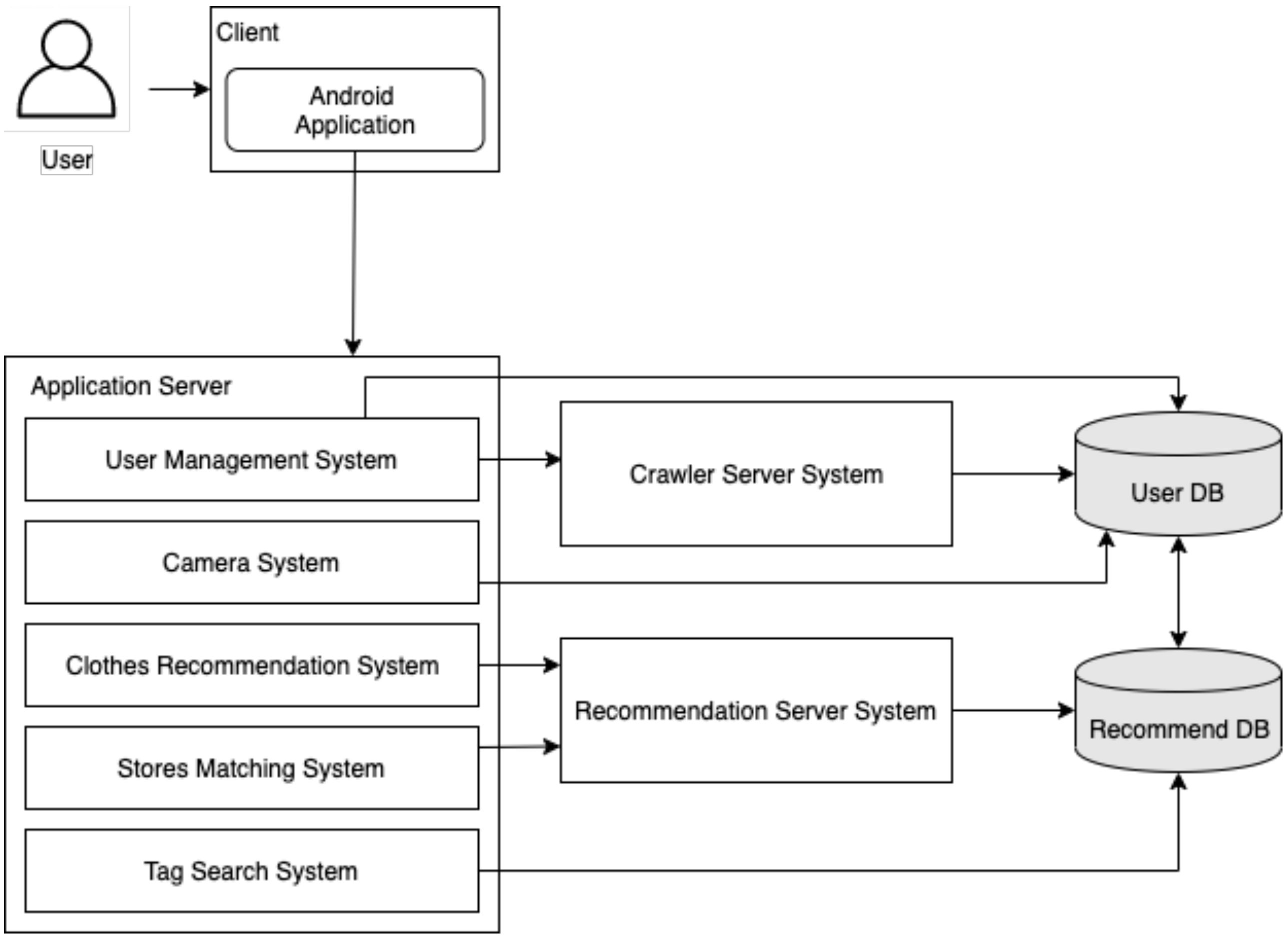

Diagram 8. Overall System Architecture

The overall architecture of "Shoppick" system Android application. The system is divided into user management system, camera system, clothes recommendation system, and stores matching system.

A. User Management System

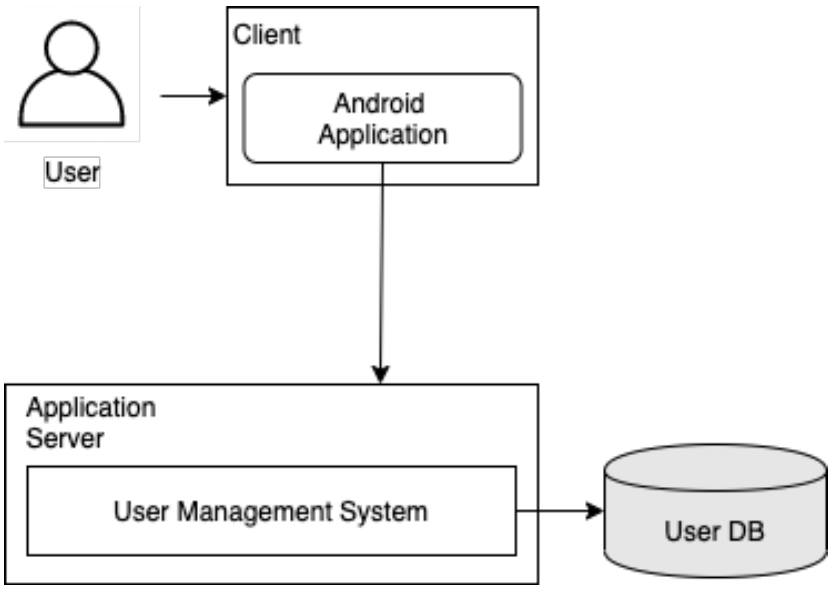

Diagram 9. User management Architecture

User management system carries out the services about the users such as signup, login and logout. The information about the user is stored in the User database.

B. Instagram Crawling System

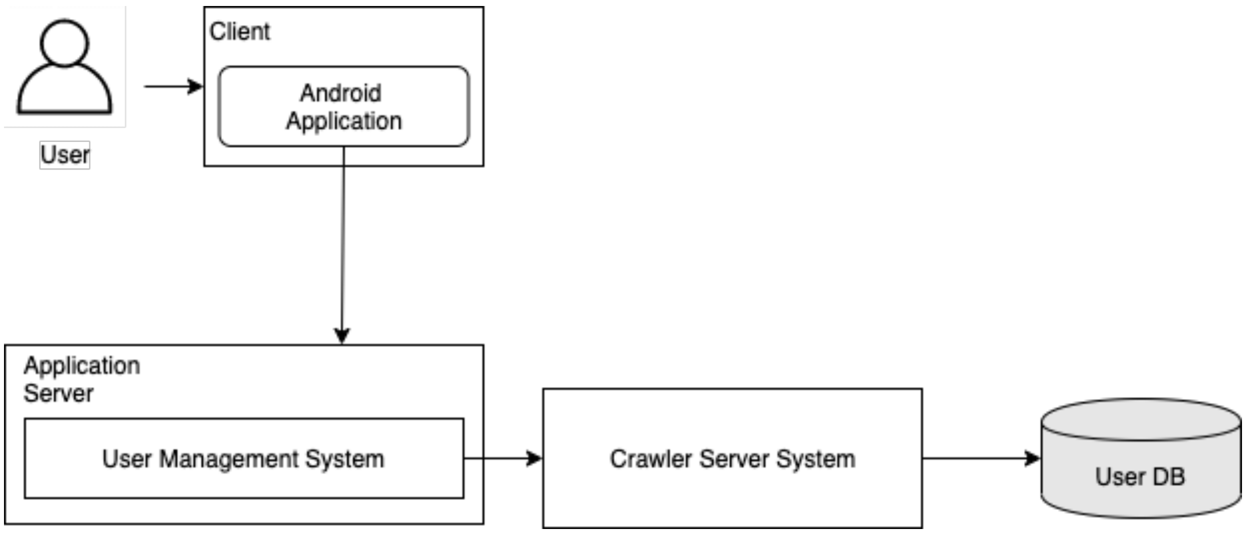

Diagram 10. Instagram Crawling System Architecture

Instagram crawling system is for the crawling the photos from the user's Instagram feed. The photos are used for the recommendation system. The information about the photos is stored in the User database.

C. Recommendation System

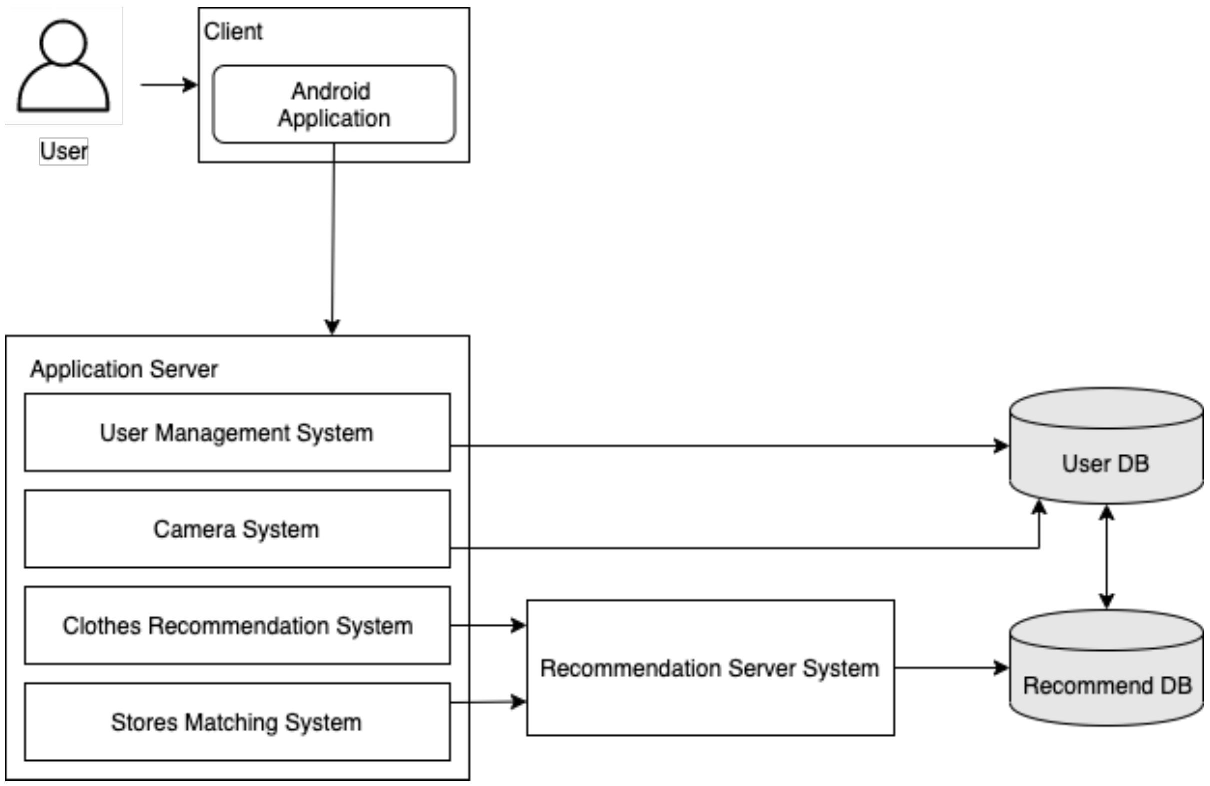

Diagram 11. Recommendation System Architecture

The 'Shoppick' recommends the clothes and stores by analyzing the user's selected photo. User can select photo from their Instagram feed or taking picture from camera. As the information of the photo is stored in User database, the system get the information from the User database. The process of analyzing photo is executed and tags of the photo are generated. The results of the process(tags) are used for the recommendation from the recommend database.

D. Tag Search System

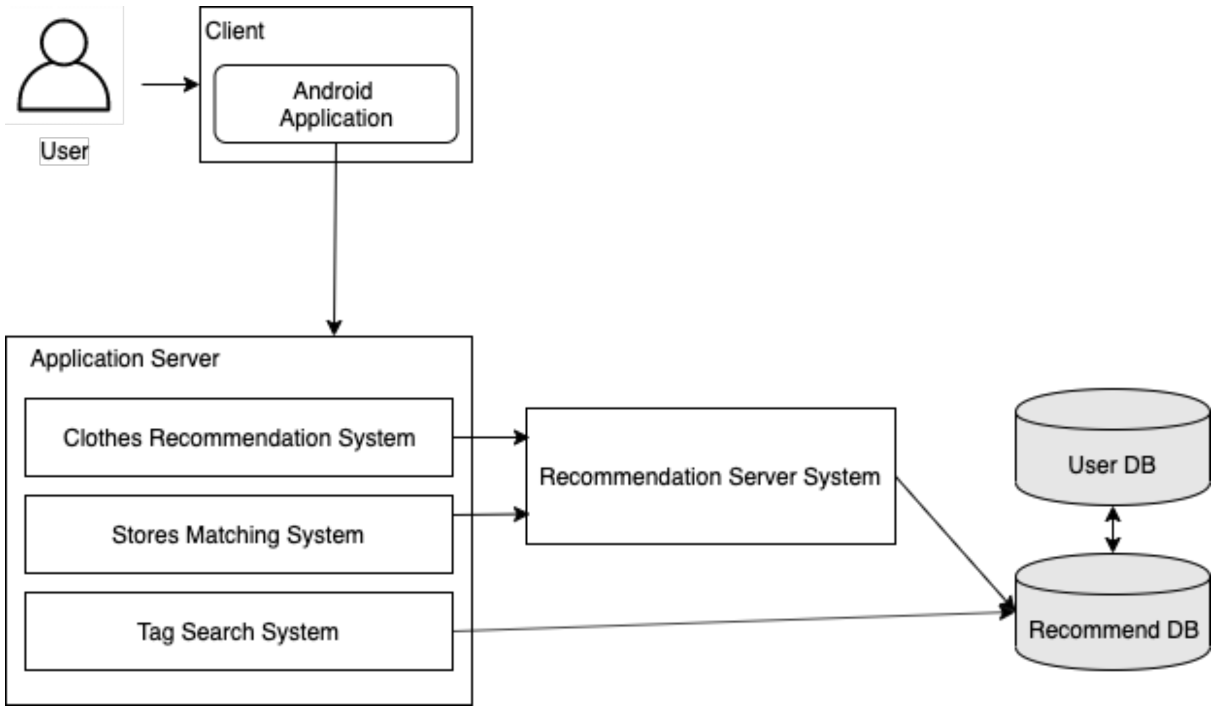

Diagram 12. Tag Search System Architecture

User can search the tag to be recommended particular clothes.

3.3 Deployment Diagram

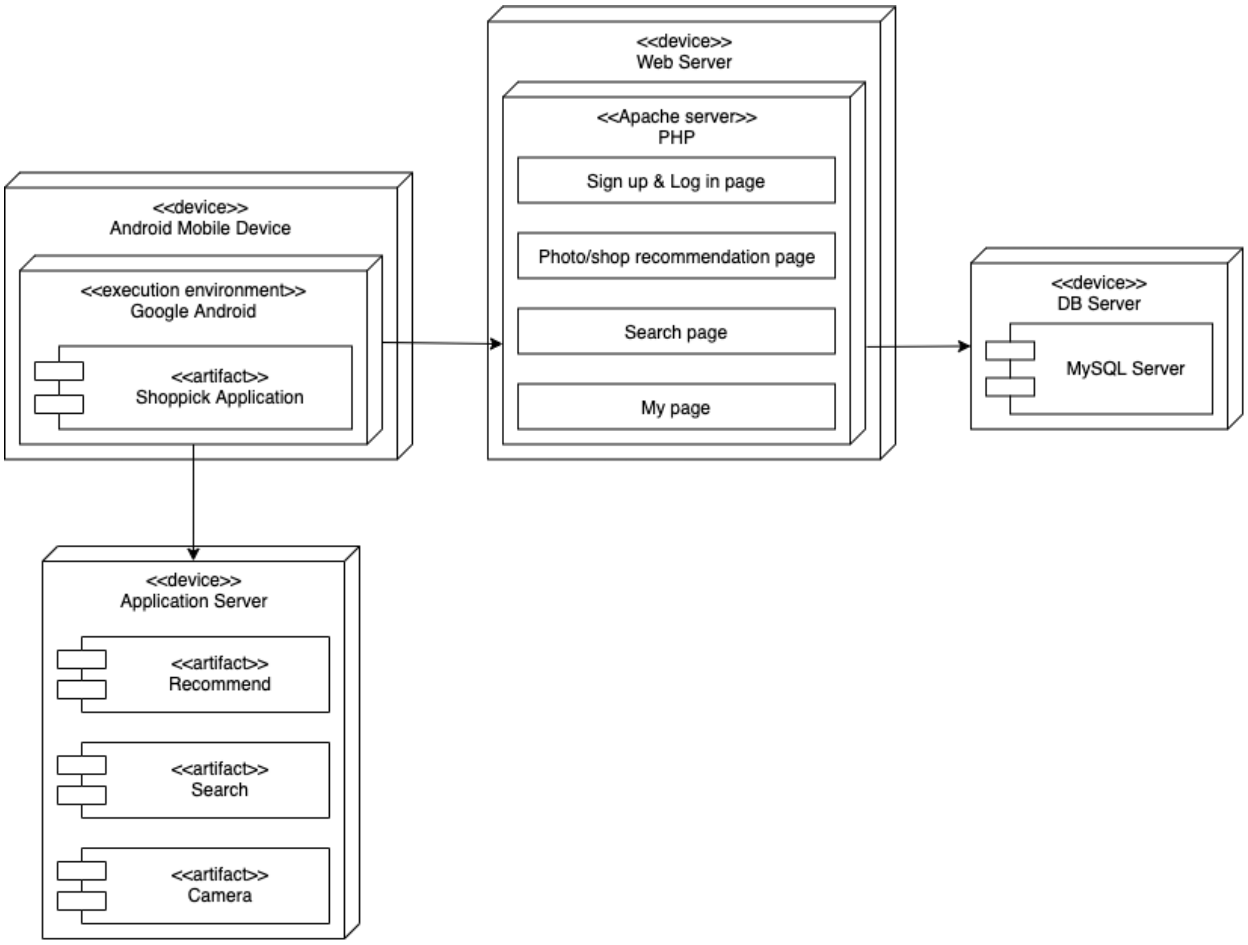

Diagram 13. Deployment Diagram

4. System Architecture – Front end

4.1 Objectives

Frontend 의 System Architecture 에서는 Frontend System 에 대하여 전반적으로 설명한다. SNS 와의 연동과 Backend 의 DB 연동에 대한 구조를 Diagram 을 이용하여 설명한다.

4.2 Subcomponents

A. Login

SNS 와 연동하여 회원이 인증될 경우 Main Activity 로 넘어가는 시스템이다. Main Activity 에서 Logout 을 하면 다시 Login Activity 로 넘어가게 된다.

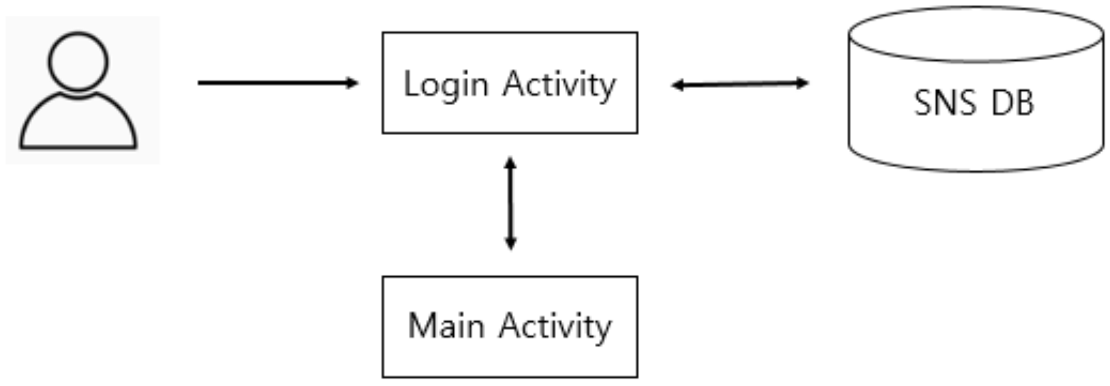

Diagram 14. Login

B. Camera

Camera intent 를 사용하여 사진을 Capture 한 후 Deep Learning Algorithm 을 하기 위해 Backend 로 Data 를 보내는 시스템이다.

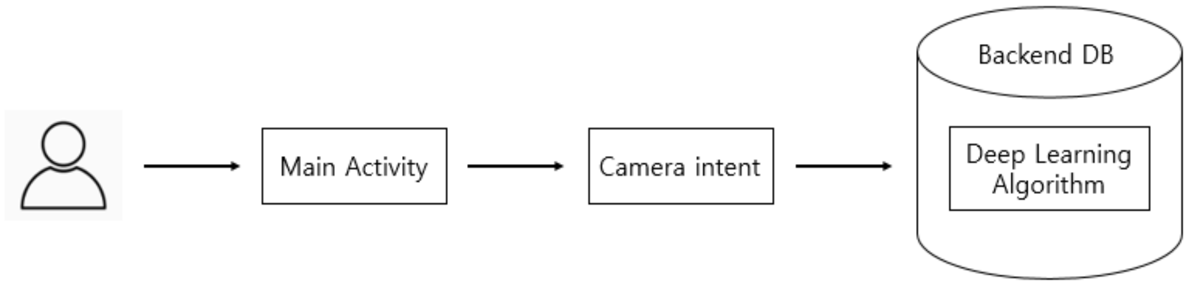

Diagram 15. Camera

C. Recommendation

Camera intent 와 함께 SNS 의 사진정보들을 Backend DB 로 전송한 후 그에 해당하는 결과물을 Recommend Activity 로 출력한다. 또한 다른 사진으로 시도하려면 다시 Main Activity 로 넘어간다.

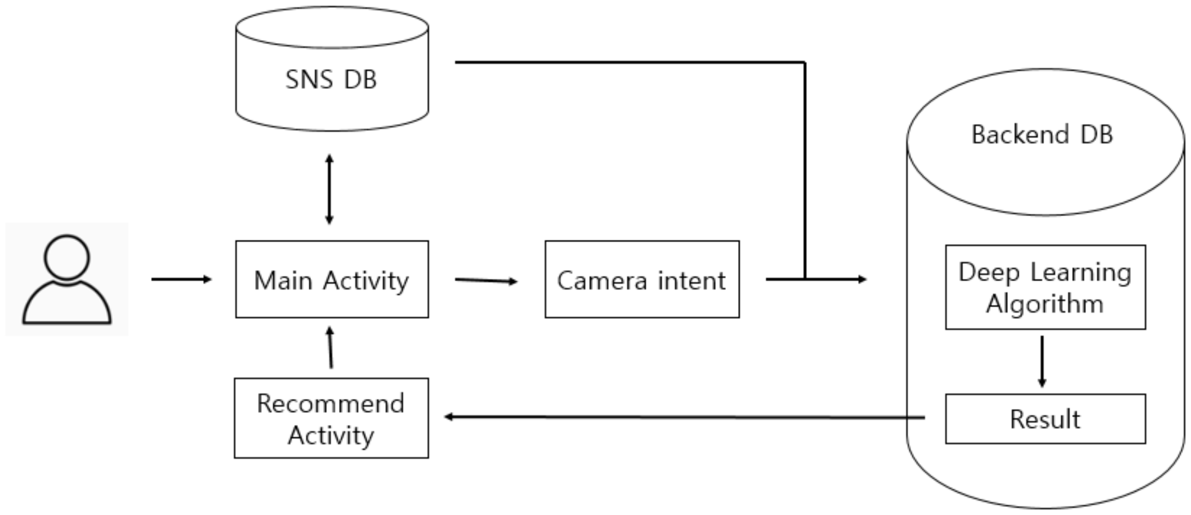

Diagram 16. Recommendation

D. Fashion Site

Backend 에서 받은 정보들을 토대로 해당 사이트의 URL 로 연결시켜준다.

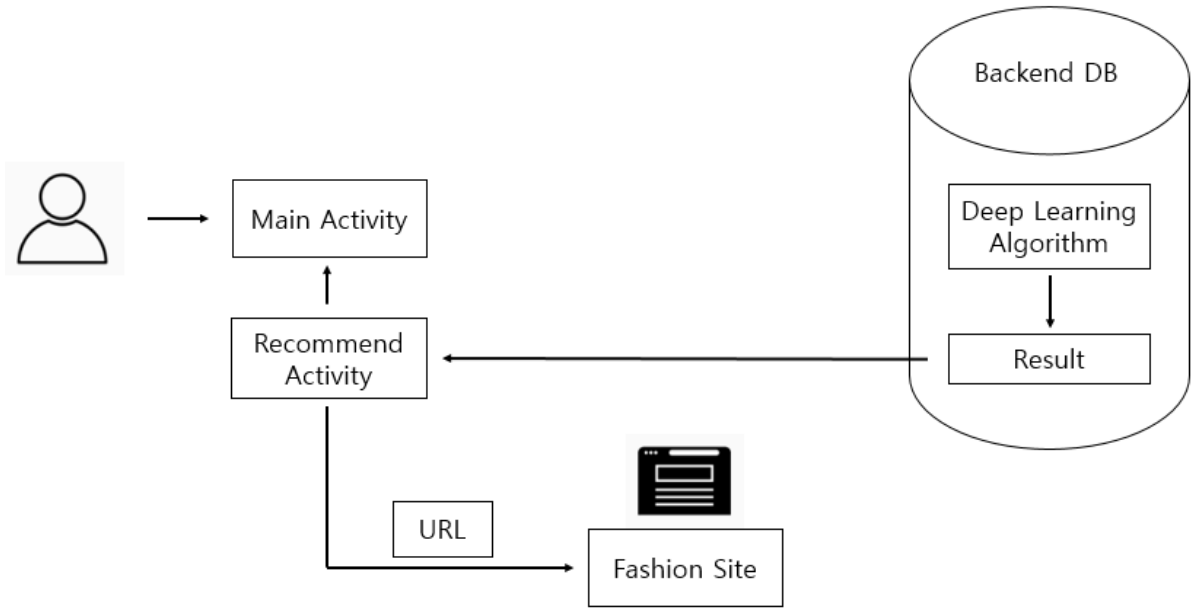

Diagram 17. Fashion Site

5. System Architecture – Back end

5.1 Objectives

In this section the structure of the back-end system is described in detail.

5.2 Overall Architecture

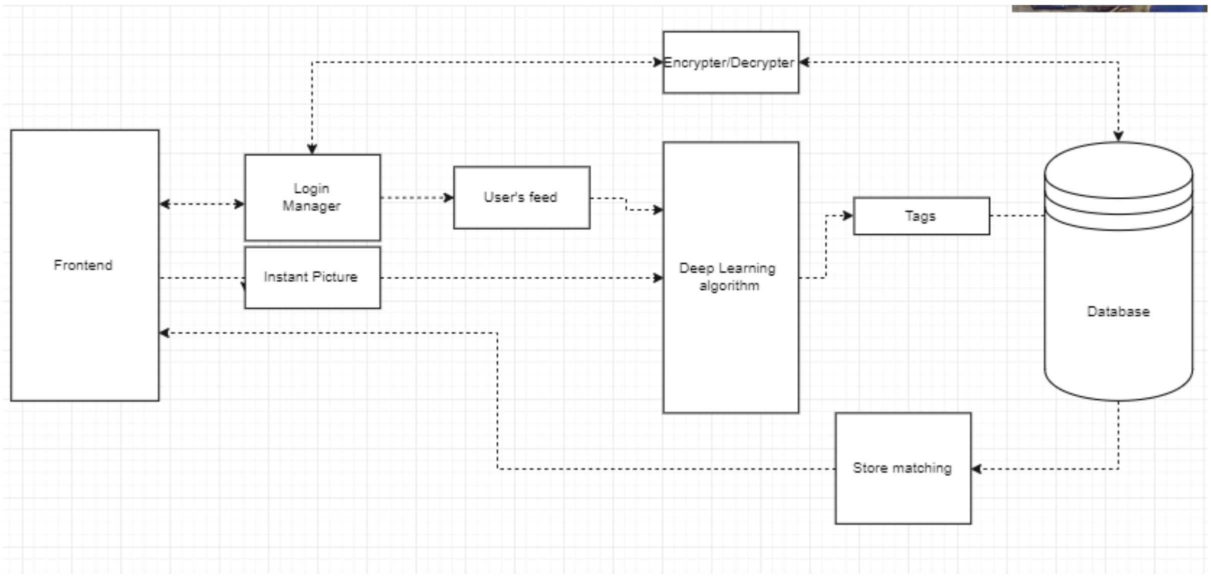

Diagram 18. Overall Architecture for backend

Backend Architecture is roughly consisted of three components, the web crawler, deep learning algorithm and database.

5.3 Subcomponents

A. Application Server

An application server is a server program in a device in a distributed network which provides the business logic for an application program. This server is frequently seen as a part of a three-level application

1. Graphic User Interface (GUI) being basically the front-end part of the program to use.
2. The application server
3. Backend-part of the program.

This server also includes middleware (connectivity software) to communicate with many services.

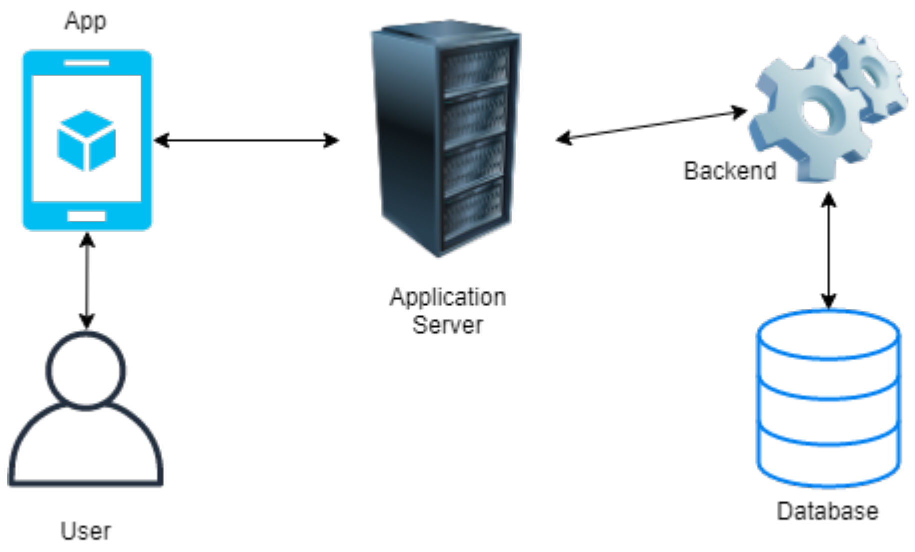

Figure 7. Server

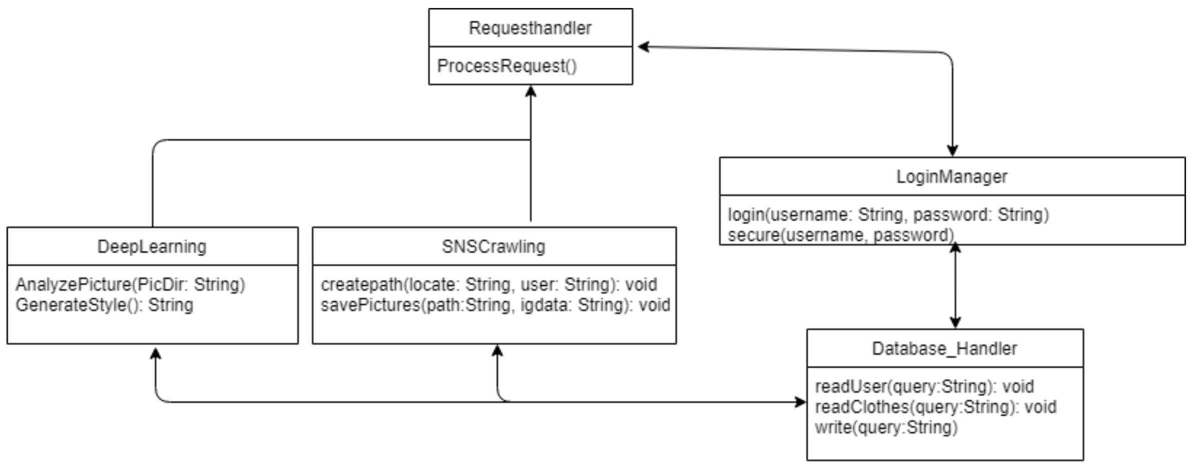
Requesthandler
ProcessRequest()
LoginManager
login(username: String, password: String)
secure(username, password)
DeepLearning
AnalyzePicture(PicDir: String)
GenerateStyle(): String
SNSCrawling
createpath(locate: String, user: String): void
savePictures(path:String, igdata: String): void
Database_Handler
readUser(query:String): void
readClothes(query:String): void
write(query:String)

Diagram 19. Diagram for server

The request handler will do the required action depending on the request received.

LoginManager will attempt to login to the user's SNS account, if successful will proceed to secure the username and password and save them to the database

The DeepLearning part will analyze the user's pictures and generate the styles according to what was found.G

SNSCrawling will retrieve the pictures from the user's feed and save the minto a new path in the host device.

Database_Handler will take care of accessing to the database, reading the stored information and adding new data sent by DeepLearning, SNSCrawling and LoginManager

How each of these works will be explained in more detail in their corresponding sections.

B. Instagram Crawling System (Instagram images Collecting System)

a. Class Diagram

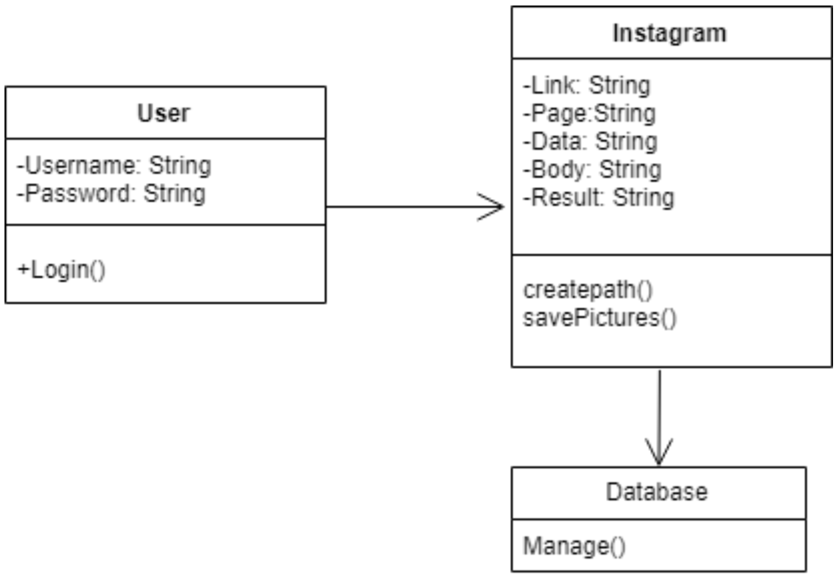

Diagram 20. Class Diagram for Instagram Crawling System

The Instagram class will get the login page from the user and start retrieving the pictures uploaded.

The fields Link, Page, Data, Body and Result are the information that the library

BeautifulSoup will need from the page in order to analyze it and get back the pictures.

1. Link refers to the user's profile URL
2. Page is the HTML format of the Link
3. Data is where we'll store all the information that appears in the page
4. Once we classify the data form the page (which parts are what), it'll be stored in Body
5. Result is where we store only the images from the page to save them to a new directory later.

The method createpath will create a new directory in a given location, assign the username as the directory name and savePictures will store all the images in the previously created directory.

b. Sequence diagram

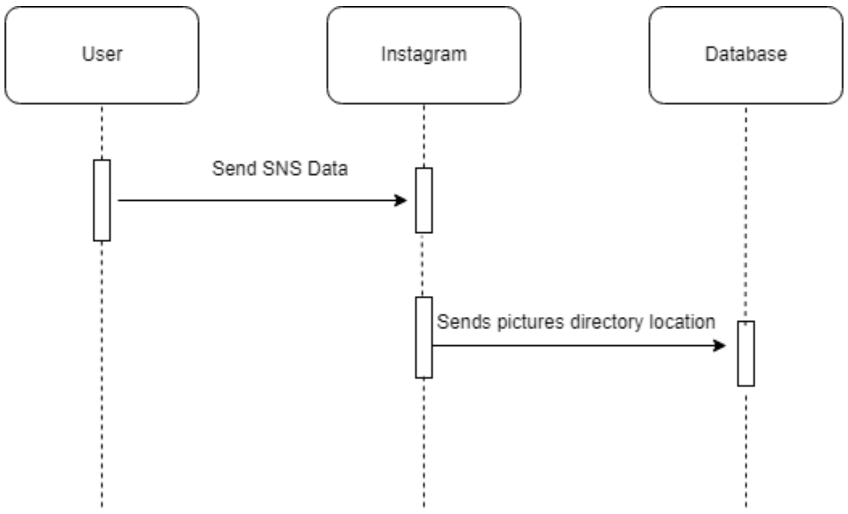

Diagram 21. Sequence Diagram for Instagram Crawling System

C. Clothes Recommendation System

a. Class Diagram

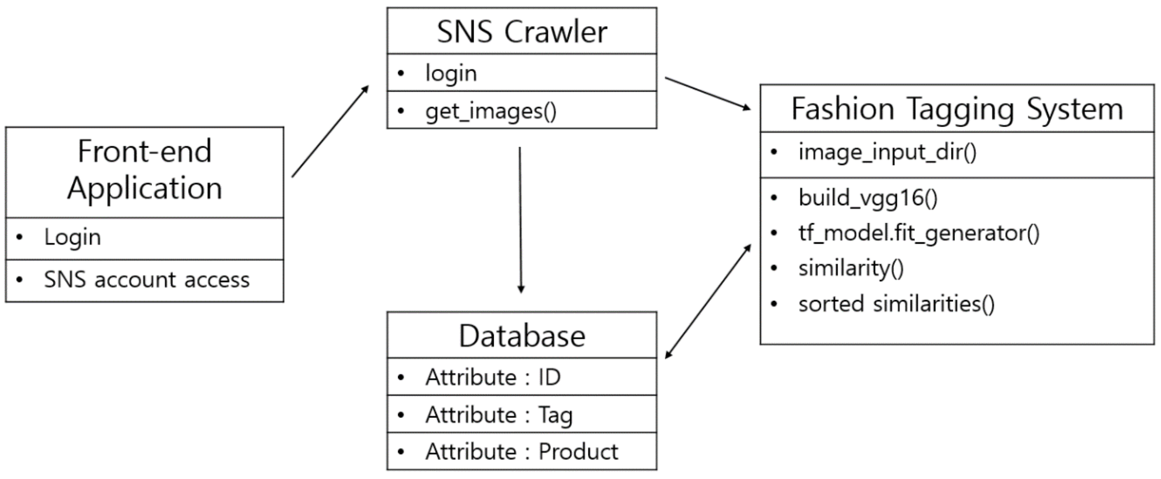

Diagram 22.Class Diagram for Clothes Recommendation System

1) SNS Crawler Class: This class is built in Python. It gets the social network service account ID and address and sends image data gathered from the data given.

   a) login field: logs in to the user's SNS account through a web browser from the server.

   b) get_images field: Using 'BeautifulSoap' library, scrapes all the image data posted on the user's SNS account and holds it.

2) Fashion Tagging System Class: This class uses python 3.x and tensorflow. It with the fashion tagging deep learning system which has been learned beforehand, tags each pictures given with specific styles.

   a) image_input_dir field: Gets the images processed from the SNS crawler.

   b) build_vgg16 field: This field is a structured deep learning algorithm based on VGG16. All pictures including pictures from retail stores to pictures from personal SNS accounts are processed through this algorithm and a style tag for each picture is formed.

c) similarity field: Gives out a score of 1 to 10 of how each picture suits each style tag.

d) sorted_similarities field: The highest score of tag given through the similarity field is tagged to the picture.

b. Sequence Diagram

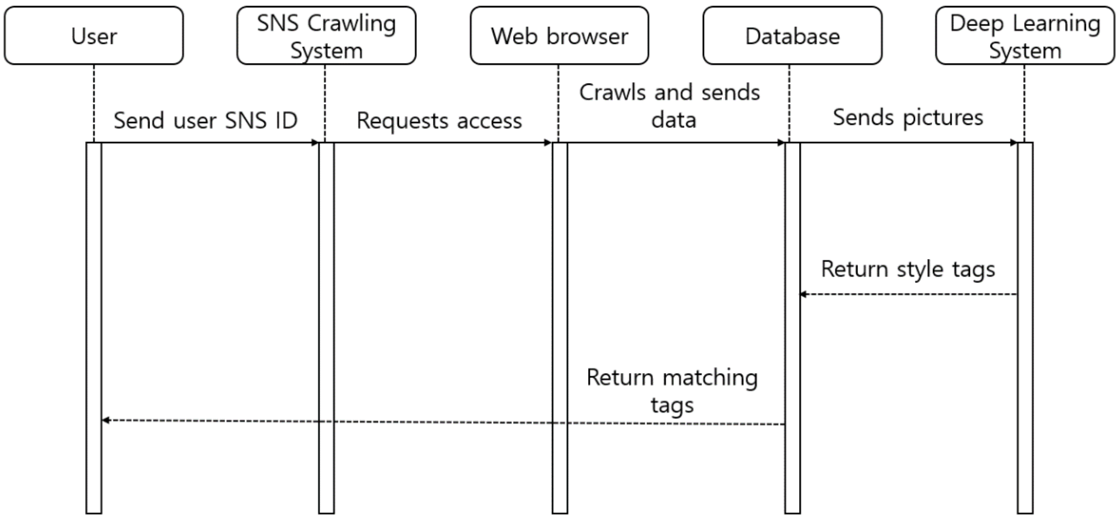

Diagram 23. Sequence Diagram for Clothes Recommendation System

c. Build_vgg16 architecture and code

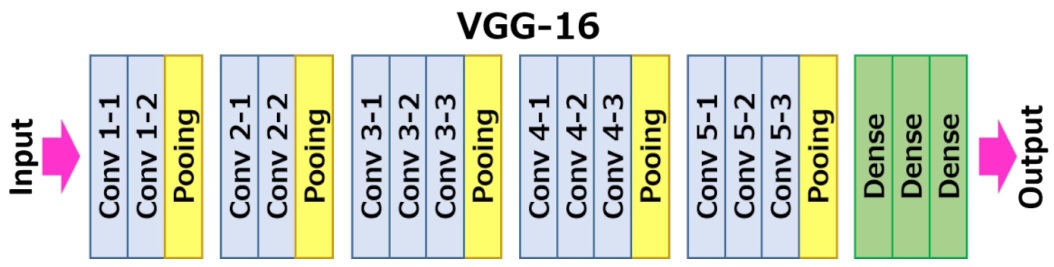

Figure 8. Build_vgg16 architecture

```
model.add(ZeroPadding2D((1, 1),
input_shape=(img_width, img_height, 3)))

   model.add(Convolution2D(64, 3, 3,
activation='relu', name='conv1_1'))
   model.add(ZeroPadding2D((1, 1)))
   model.add(Convolution2D(64, 3, 3,
activation='relu', name='conv1_2'))
   model.add(MaxPooling2D((2, 2), strides=(2, 2)))

   model.add(ZeroPadding2D((1, 1)))
   model.add(Convolution2D(128, 3, 3,
activation='relu', name='conv2_1'))
   model.add(ZeroPadding2D((1, 1)))
   model.add(Convolution2D(128, 3, 3,
activation='relu', name='conv2_2'))
   model.add(MaxPooling2D((2, 2), strides=(2, 2)))

   model.add(ZeroPadding2D((1, 1)))
   model.add(Convolution2D(256, 3, 3,
activation='relu', name='conv3_1'))
   model.add(ZeroPadding2D((1, 1)))
   model.add(Convolution2D(256, 3, 3,
activation='relu', name='conv3_2'))
   model.add(ZeroPadding2D((1, 1)))
   model.add(Convolution2D(256, 3, 3,
activation='relu', name='conv3_3'))
   model.add(MaxPooling2D((2, 2), strides=(2, 2)))

   model.add(ZeroPadding2D((1, 1)))
   model.add(Convolution2D(512, 3, 3,
activation='relu', name='conv4_1'))
   model.add(ZeroPadding2D((1, 1)))
   model.add(Convolution2D(512, 3, 3,
activation='relu', name='conv4_2'))
   model.add(ZeroPadding2D((1, 1)))
   model.add(Convolution2D(512, 3, 3,
activation='relu', name='conv4_3'))
   model.add(MaxPooling2D((2, 2), strides=(2, 2)))

   model.add(ZeroPadding2D((1, 1)))
   model.add(Convolution2D(512, 3, 3,
activation='relu', name='conv5_1'))
   model.add(ZeroPadding2D((1, 1)))
   model.add(Convolution2D(512, 3, 3,
activation='relu', name='conv5_2'))
   model.add(ZeroPadding2D((1, 1)))
   model.add(Convolution2D(512, 3, 3,
activation='relu', name='conv5_3'))
   model.add(MaxPooling2D((2, 2), strides=(2, 2)))
```

D. Store Matching System

a. Class Diagram

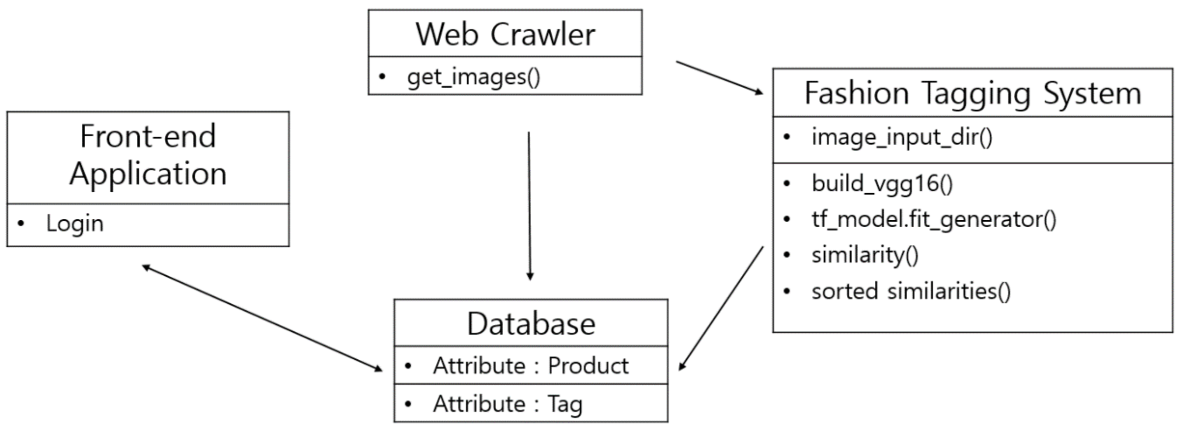

Diagram 24. Class Diagram for Store Matching System

1) Web Crawler Class: This class is built in Python. It gathers the images of products and their URL on retail store webpages. The URL information is sent to the Database

   a) get_images field: Using 'BeautifulSoap' library, scrapes each product image from the retail stores along with the URL.

2) Fashion Tagging System Class: This class uses python 3.x and tensorflow. It with the fashion tagging deep learning system which has been learned beforehand, tags each pictures given with specific styles.

   a) image_input_dir field: Gets the images processed from the Web Crawler.

   b) build_vgg16 field: This field is a structured deep learning algorithm based on VGG16. All pictures including pictures from retail stores to pictures from personal SNS accounts are processed through this algorithm and a style tag for each picture is formed.

   c) similarity field: Gives out a score of 1 to 10 of how each picture suits each style tag.

   d) sorted_similarities field: The highest score of tag given through the similarity field is tagged to the picture.

b. Sequence Diagram

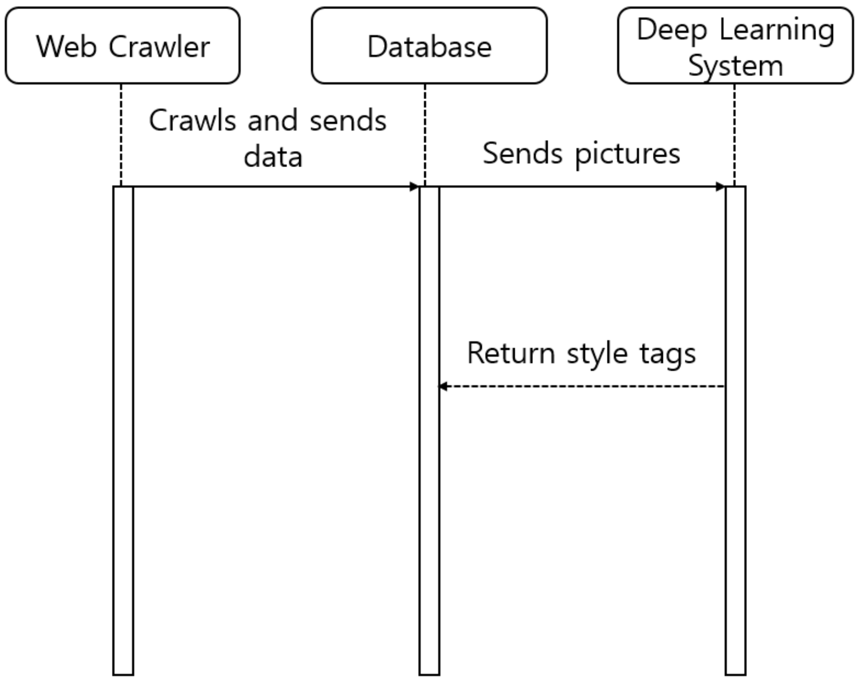

Diagram 25. Sequence Diagram for Store Matching System

6. Protocol Design

6.1 Objectives

Protocol Design에서는 Sub-System 간의 통신을 설명한다. Server 통신을 위해 텍스트를 사용하는 개방형 표준 format인 JSON(JavaScript Object Notation)을 사용하며 각 Protocol의 parameter를 기준으로 설명한다.

6.2 Details

A. Authentication

a. Sign up

Request	
Attribute	**Value**
User_ID	사용자의 ID (SNS ID 형식)
User_PW	사용자의 PW
Use_Name	사용자의 NAME

Table 1. Authentication sign up protocol Request

Response	
Attribute	**Value**
Regist_Success	True/False

Table 2. Authentication sign up protocol Response

b. Log in

Request	
Attribute	**Value**
User_ID	사용자의 ID (SNS ID 형식)
User_PW	사용자의 PW

Table 3. Authentication log in protocol request

Response	
Attribute	**Value**
Login_Success	True/False

Table 4. Authentication log in protocol response

B. User item

a. Send

Request	
Attribute	**Value**
Sns_Picture	SNS 에서 가져온 사용자의 사진 데이터
Capture_Picture	사용자가 직접 찍은 사진 데이터

Table 5. User item send protocol request

Response	
Attribute	**Value**
Picture_effective	이미지 정보에 대한 유효성 확인
Server_success	True/False(서버로의 전송 유무)

Table 6. User item send protocol response

C. Recommendation

a. Get

Request	
Attribute	**Value**
Sns_Picture	SNS 에서 가져온 사용자의 사진 데이터
Capture_Picture	사용자가 직접 찍은 사진 데이터
Picture_effective	이미지 정보에 대한 유효성 확인

Table 7. Recommendation get protocol request

Response	
Attribute	**Value**

Recommend_Picture	Backend 의 결과물에 대한 이미지
Recommend_Price	Backend 의 결과물에 대한 가격
Recommend_Site	Backend 의 결과물에 대한 브랜드 사이트

Table 8. Recommendation get protocol response

b. Search

Request	
Attribute	**Value**
Site_URL	해당 브랜드의 사이트 URL

Table 9. Recommendation search protocol request

Response	
Attribute	**Value**
Link_Success	True/False (URL 로 연결 여부)

Table 10. Recommendation search protocol response

7. Database design

7.1 Objectives

In this part we will describe how the database design was made and how all of the entities interact with each other.

7.2 ER Diagram

In here, we show how the entities interact with each other, the entities are in squares, attributes are in the circles and the relationships among them are in rhombus.

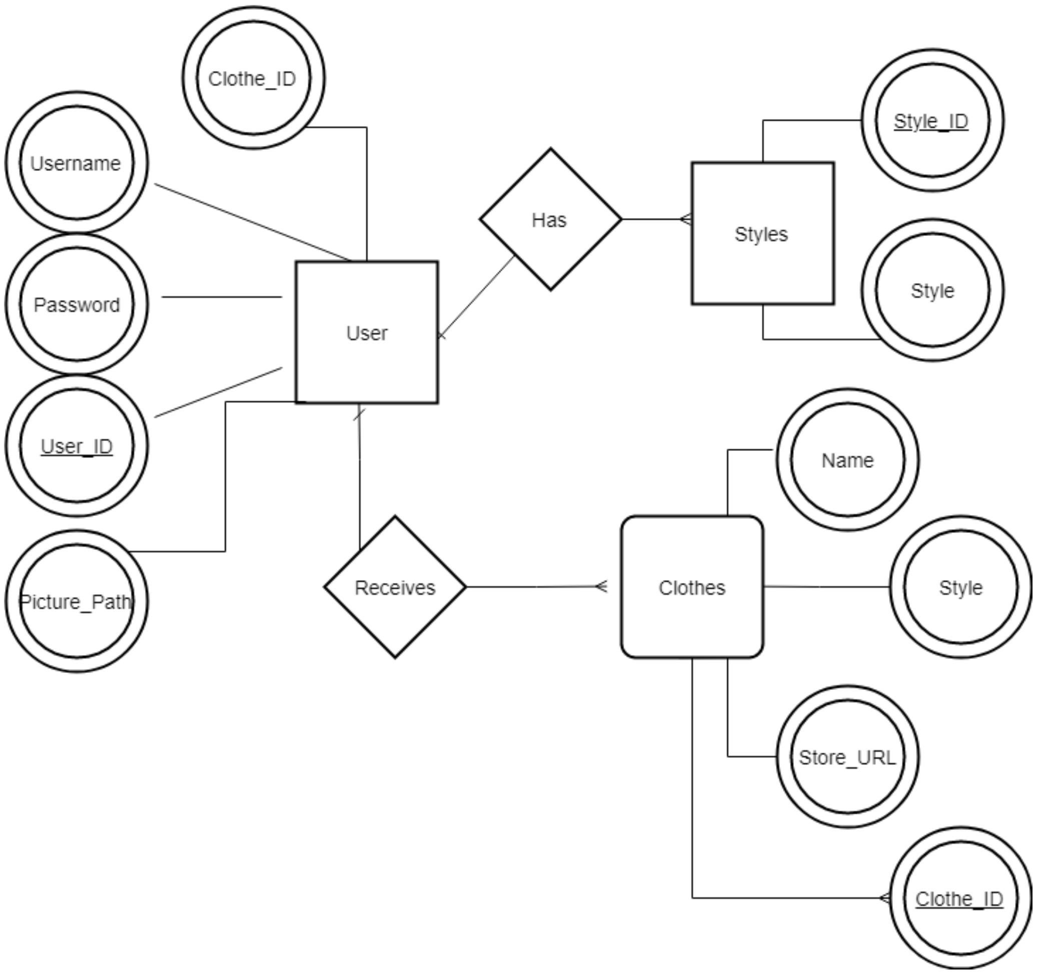

Diagram 26. ER diagram

A. Entities

a. User

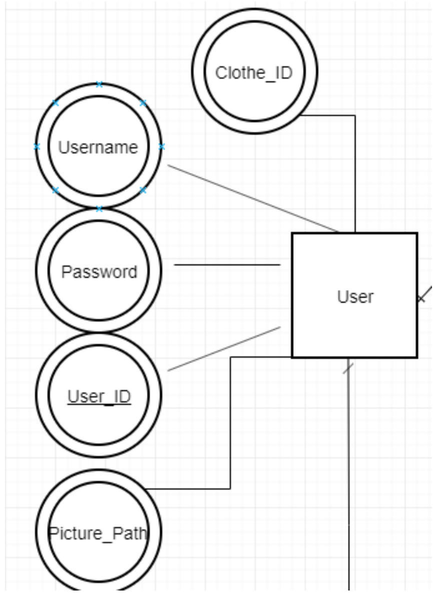

Diagram 27. User Entity

The user has 5 attributes: User_ID which will let us keep track of its position in a DB, username and password which are the login information that we are storing, picture_path which refers to the location in the host where the user's pictures are saved and finally clothe_ID which will store the ID of the recommended clothe.

b. Styles

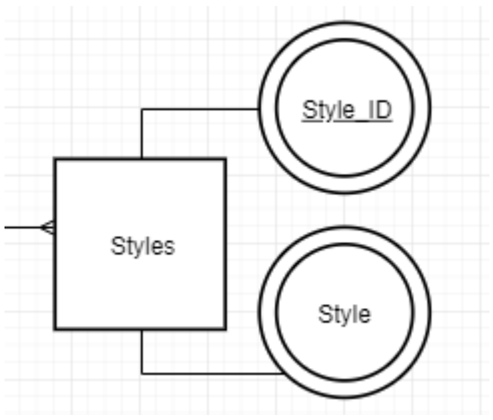

Diagram 28. Styles Entity

Each user will have a style, and each stored style will have an ID and the style obtained from the database

c. Clothes

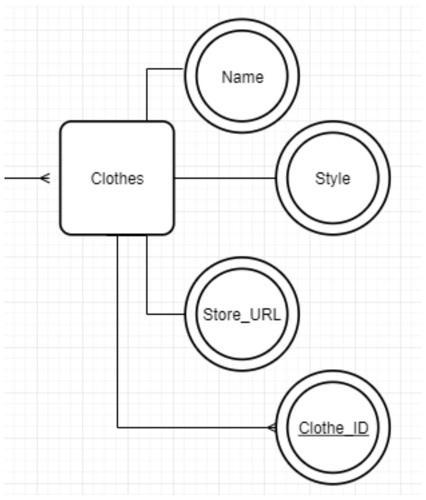

Diagram 29. Clothes Entity

Also each user will receive the clothing recommendation, each of these clothes will have a name, a style, an URL to the store where it is sold and an ID.

7.3 Relational Schema

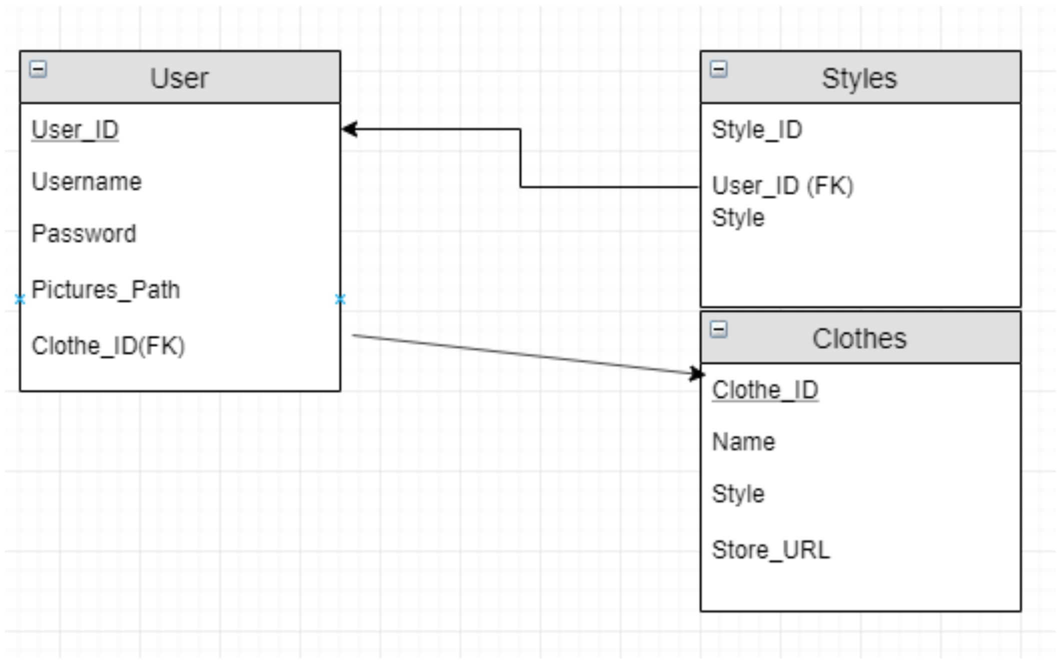

Figure 9. Relational Schema

7.4 SQL DDL

A. User

```
CREATE TABLE User
(
User_ID INT NOT NULL,
Username VARCHAR (30) NOT NULL,
Password VARCHAR (30) NOT NULL,

Pictures_path VARCHAR (30) NOT NULL,

PRIMARY KEY (User_ID),

Foreign KEY (Clothe_ID) REFERENCES Clothing (Clothe_ID)
);
```

B. Style

```
CREATE TABLE Styles
(
Style_ID INT NOT NULL,
Style VARCHAR (30) NOT NULL,

PRIMARY KEY (Style_ID),

Foreign KEY (User_ID) REFERENCES User (User_ID)
);
```

C. Clothing

```
CREATE TABLE Clothing
(
Clothe_ID INT NOT NULL,
Name VARCHAR (30) NOT NULL,
Style VARCHAR (30) NOT NULL ,

Store_URL VARCHAR (30) NOT NULL,

PRIMARY KEY (Clothe_ID),

);
```

8. Testing plan

8.1 Objectives

This section explains the testing policy and test cases. The directive is to see if the system runs as it has been planned. In testing policy, we illustrate testing methods step by step.

8.2 Testing Policy

Shoppick runs testing in the system development processes by parting them into three sections.

A. Development Testing

Development testing is aimed for synchronizing application of broad spectrum of defect prevention and detection strategies in order to reduce software development risks. This phase focuses on testing reliability, security.

a. Reliability

Deep learning process has to have a reliable match rate between the style tags and the pictures gathered from the SNS crawler. If the match rate is at least near 80 percent, the algorithm can be said is reliable since three tags are given to each users.

b. Security

The SNS pictures taken by the SNS crawler can be hijacked by hackers or get used in the wrong way by administrators. The developers must insure that the picture data doesn't get saved in the database.

The codes have to be reviewed to see if there are no loop holes for hackers to get in the system. The ID and password passed on from the users should be encrypted. The developers should revise the form the data is saved in the database also maybe mock-hacking routines can be helpful for making a sure security.

B. Release Testing

Release testing is a process to test the build of a software to make sure that it is flawless and doesn't have any issues and works as intended. It is done prior to the release and is one of the most critical part for success of a software aside from design and coding.

The release testing has to go thorough as the diagram above. Getting approval of building and implementing through-out the testing process.

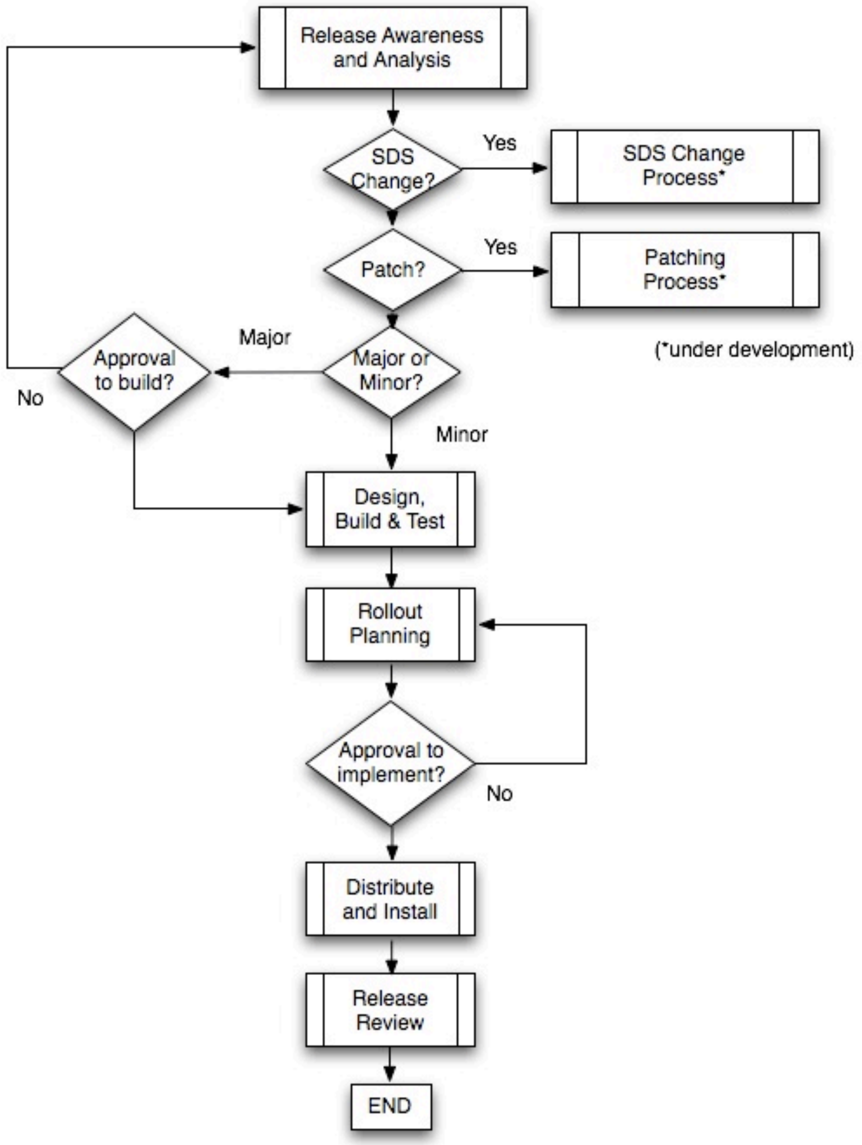

Diagram 30. Release testing diagram

C. User Testing

User testing can be done by the concept of alpha testing. Developers themselves can use the software themselves or make their close friends or families try it out. Around 30 people using this system can be an ideal amount for a pre-user test. The user’s opinions can be gathered by a survey or personal man-to-man interview. After this is done the developers can make changes with the good suggestions or criticisms.

8.3 Case testing

A. User information management System

1) The user sign in

User: clicks ‘sign in’
System: gives the user ‘ID’ and ‘password’ forms that can be filled in
User: types in the input info that they want to use
System: Checks if it is available

1-1) If is it available

System: saves the user info in the database encrypted.
User: gets logged in

1-2) If is it unavailable

System: outputs ‘ID is already taken’

2) The user logs in

User: input ‘ID’ and ‘password’

2-1) If the data is wrong

System: outputs wrong ID or password

2-2) If the data is right

System: user gets logged in

B. Recommendation System

1) When a user wants to make a style tag

User: Input SNS 'ID' and 'password'
System: runs the SNS crawler and gets pictures in the user's account. The pictures are then sent to the deep learning algorithm and saves the user's style tag in the database.

2) User wants to get shop recommendation

User: clicks search
System: gets the user style tag information saved in the database, searches for retail shops with matching style tags.

2-1) If the user doesn't have a style tag saved in the database

System: outputs 'style tag doesn't exist make a tag'

C. Search System

1) User wants to search for clothes

User: clicks the style tags that he or she wants to take a look at
System: gets the products matching the tag that the user has pressed

1-1) If there are no matching tags

System: send 'Sorry no matching tags'

1-2) If there are matching tags

System: sends the product list of the tags matching to the one user requested

9. Development plan

9.1 Objectives

In the section we illustrate the development environments and plans. The programming languages and IDE used in the system is specifically dealt with the information about them found in their web page explanation or wikipedia.

9.2 Frontend Environment

A. Android Studio

Figure 10. Android studio logo 3

Android Studio is the official Integrated Development Environment (IDE) for Android app development, based on IntelliJ IDEA. On top of IntelliJ's powerful code editor and developer tools, Android Studio offers even more features that enhance your productivity when building Android apps

9.3 Back-end Environment

A. Tensorflow

Figure 11. Tensorflow logo

Tensorflow is the core open source library to help you develop and train machine learning models. Get started quickly by running Colab notebooks directly in your browser. It is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural networks. It is used for both research and production at Google.

B. Jupyter Notebook

Figure 12. Jupyter Notebook logo

The *Jupyter Notebook* is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text. Uses include: data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more.

C. MySQL

Figure 13. MySQL logo

MySQL is a relational database management system based on SQL – Structured Query Language. The application is used for a wide range of purposes, including data warehousing, e-commerce, and logging applications. The most common use for mySQL however, is for the purpose of a web database.

9.4 Schedule

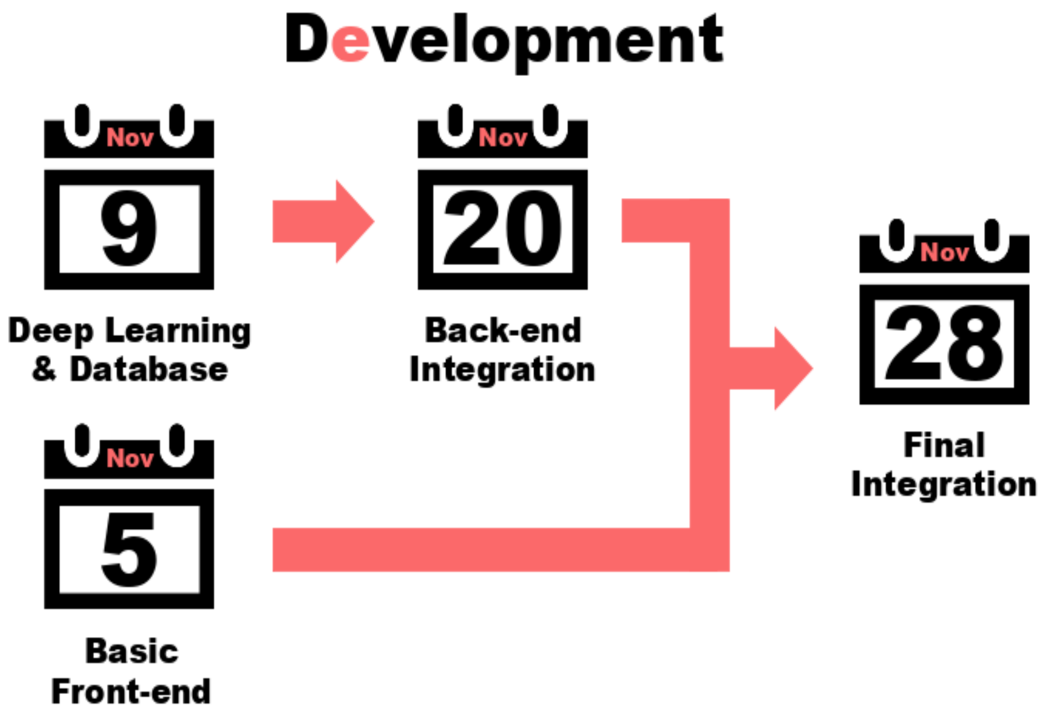

Figure 14. Schedule

The schedule can be changed or delayed due to the documentation but the final integration date is absolute.

10. Index

10.1 Objectives

This section shows the index of this document. It includes figure index, diagram index and table index.

10.2 Figure index

10.3 Diagram index

10.4 Table index

11. References

11.1 Objectives

Describe a list of documents referenced in writing this document.

11.2 Reference

- UML diagram
https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-uml/#class-diagram
- 위키피디아 Android studio
https://en.wikipedia.org/wiki/Android_Studio
- 위키피디아 MySQL
https://en.wikipedia.org/wiki/MySQL
- 위키피디아 Tensorflow
https://en.wikipedia.org/wiki/TensorFlow
- 위키피디아