Fooding

Design Specification

Team 6

하예송 양주원 유병서 전재혁 정석원 Algi

Contents

1. Preface

1.1. Readership

This document is written for and considering variable readers. Before starting to describe our application, we will first script expecting readers. Basically, there's no specific reader to this design specification in contrast to requirement specification. But maybe who needs information about design specification, such as system architecture and each system's operation and DB will read this document.

1.2. Document Structure

A. Preface

Describe about expected readers. And with document structure, show outline of document and each section.

B. Introduction

Introduction contains objective of introduction, needs and system overview. At Introduction, we'll explain why our system is needed with various grounds. And give purpose of the system with expected result.

C. System Architecture

This part describes architecture of system. In system architecture, how systems are organized is explained. In this part, overall application organization and frontend, backend organization are described.

D. Sign up/Login System

Sign up is a function that requests the personal information of users which are necessary to use our service and stores the information in DB. Sign up/Login System part show how these user management systems work, specifically how class and its attributes and methods are organized. And with sequence diagram, this part show how this system works.

E. Preference Survey System

Preference survey is conducted to identify user's preferences for food. Preference survey system show how this preference survey is conducted, describing with class diagram and sequence diagram.

F. Product Detail System

Product detail shows details, user reviews and average ratings of a specific product. Users can directly buy a product or put it in shopping cart from the product detail page. This part shows how this product detail system works.

G. Search System

This part describes how search system is performed. Search system is used when user goes to search tab of application. Search is a function that users can use to search specific products. Users can search products by entering search keyword. There are filters to user, such as allergens, and the food contains that filter is not shown to search result.

H. Recommendation System

Recommendation function shows a list of recommended products that users will like based on the preference survey. This part shows system divided by frontend and backend. From frontend, how recommendation implemented is described. At Backend, how recommend system connected to other data is shown.

I. Review & Rating System

Review & Rating function allows users to write reviews and rate products. This part describes review system with dividing whole system into review and review handling system.

J. Protocol Design

K. Database Design

Database design is the part describing details about database of our application. With whole ER Diagram, this part will show how database is organized. And showing each entities' database, we'll show what attributes are in each entity. With relational schema, we'll give information about which is primary key and foreign key. We'll also show JSON Tree and thus indicate implicatively how DB is organized.

L. Testing Plan

At testing plan, testing policy will be described. More specifically, how test will be performed while development and after application released will be described.

M. Development Plan

Development play is about workflow dates, and also introduces tools we'll use in developing this application. There's also version management tool we using is introduced.

N. Index

This part is for tables and figures used in this document. In index, there are brief information such as which page they are.

2. Introduction

2.1 Objectives

Objectives of introduction is giving readers explanation of diagrams and tools for design system and introduce the scope of our project.

2.2. Applied Diagram

A. UML

UML is a general purpose and developmental modeling language and technique that combines different aspects of a system to represent relations, processes or results of an overall model or system. It is essential to mention that we have used it thoroughly in this document to visualize the workflow of the system.

Since it provides different modeling techniques and a handful subset of diagrams, it can be efficiently used to provide means of communication between developers and users as it covers wide range of symbols and definitions, and it consists of the following diagrams: Package Diagram, Deployment Diagram, Class Diagram, State Diagram, Sequence Diagram and ER Diagram

B. Package Diagram

Package diagram is kind of structural diagrams which show the arrangement and organization of model elements. Package diagram can show both structure and dependencies between sub-systems or modules in a more abstract way than other types of UML diagrams. This abstraction leads to the use of package diagrams in simplifying complex class diagrams by grouping them in packages.

A. Deployment Diagram

Deployment diagram describes the physical deployment of information generated by the software program on hardware components. Deployment diagrams is made up of several UML shapes. The 3D boxes, known as nodes, represent the basic software or hardware elements, or nodes, in the system. Lines from node to node indicate relationships, and the smaller shapes contained within the boxes represent the software artifacts that are deployed.

B. Class Diagram

Class diagram is used to showcase the object classes of a system and the relationship between classes. One of the most fundamental reasons we are using class diagram is because it provides a clear distinction between each class and show the hierarchy and dependency between them. As far it goes for the inner structure of Class diagram, it consists of some fields indicating some variables, class methods and links or associations between classes.

C. State Diagram

State Diagram is a technique to represent different states of a system and all possible next states based on some particular stimuli which triggers the change of the state. This kind of diagram is very important to analyze different scenarios of the system as the states are represented as nodes and events as arcs which helps in identifying the behavior of the object classes defined in class diagram.

D. Sequence Diagram

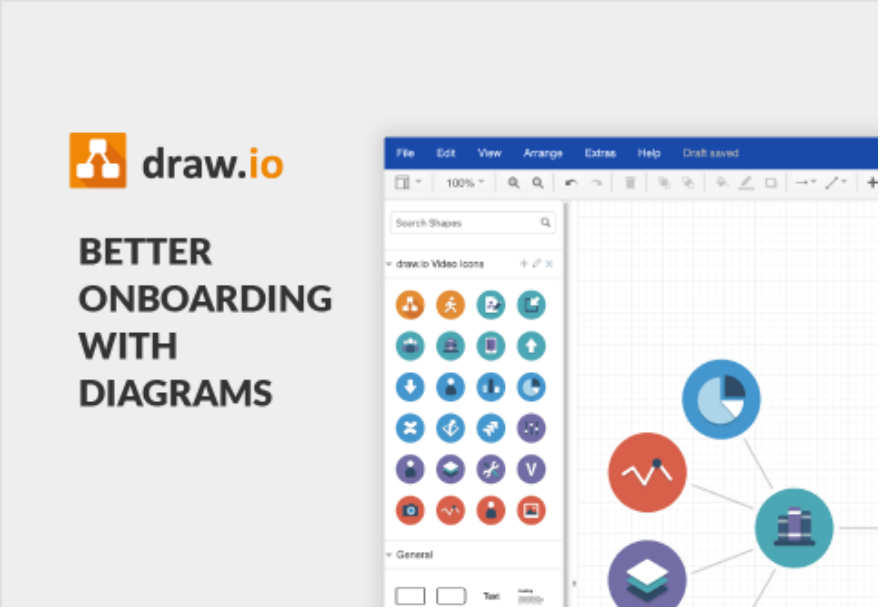
Sequence diagram is used to represent the interactions between the actors and objects of the system. Specifically, the goal of sequence diagram is present the sequence of interaction and processes that take place in a specific use case instance so that a result could be generated. It is important to notice that the direction of the arrows here is essential to indicate the correct flow of actions.

E. ER Diagram

ER diagram is used to represent relationships and attributes between entities. ER diagram is mostly used for designing database, and can generate relational schema, SQL DDL from this diagram.

2.3. Applied Tool

Draw.io



Draw.io is web-based modeling tool providing many templates and shapes that user can draw any kind of diagram easily. With draw.io, user do not need to redraw its shape for drawing diagram, especially for draw UML diagram. It also provides convenient canvas system that can easily align all shapes in grid. Most of diagrams from this document is drawn from draw.io.

2.4. Project Scope

The purpose of this project is giving better product recommendation system, especially the people who looking for food that has own taste tendency. Other previous services recommended products without customer's satisfactions. Our project is covering that part as labeling product from customers review, ask survey and categorize the customers that they can avoid the product that might be dissatisfied product from them, even can harm them.

First of all, we will cover the overall system architecture for our project, that can see some draft of overall application and service how works. And, we will describe about User management system, including Sign Up and Login process. After User management system, Product detail system will be followed for explain how detail of product can visualize and process for customers. And, Search system is described to provide search features for the customers who wants to search product with some categories, filter and order conditions, without any recommendation. Recommendation system, which is most important system in our project is after them, to explain how user categorized, and shows the process that what product should avoid and preferred to show. Lastly, Review and Rating system is described for how treat the user's review and rating of product.

3. System Architecture

3.1. Objectives

This chapter describes the overall structure of this system. It describes overall system architecture, each subsystem architecture and relation between subsystems.

3.2. System Organization

A. Overall Application Organization

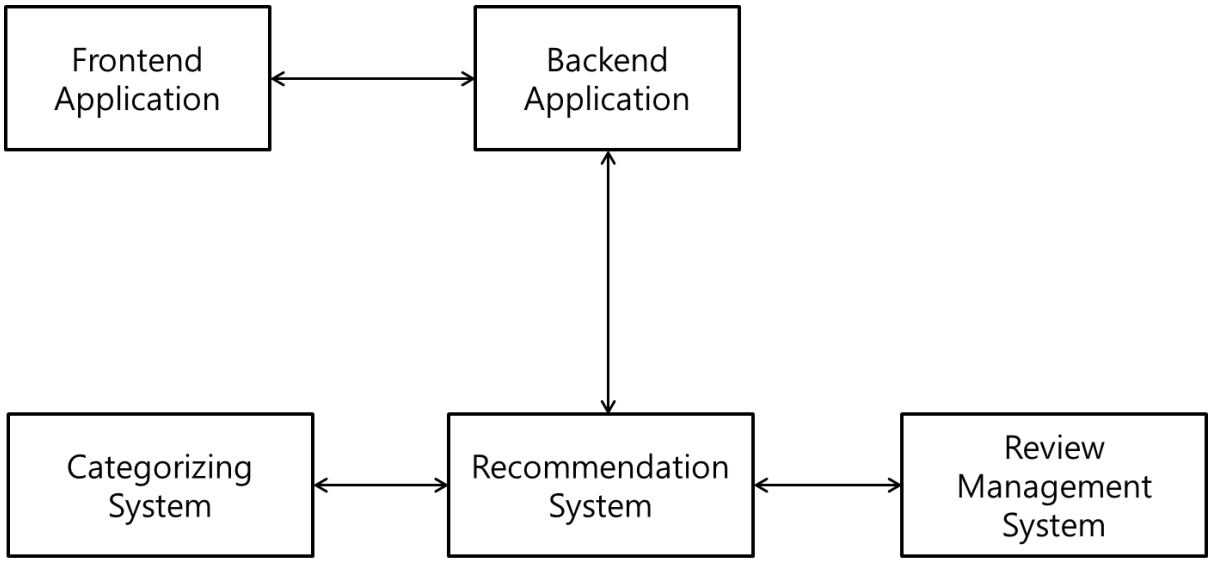


Figure 1. System Architecture - Overall

We design the system by applying client-server model. Frontend Application show user all features of this application and Backend Application manage all data which is used Frontend Application features. Frontend Application and Backend Application give and receive data based on JSON by using RTSP(Real Time Stream Protocol).

Recommendation system is executed when user click recommendation button. Recommendation system requires user preference category and filter to categorizing system. Then recommendation system sends this information to review management system in order to receive recommendation product list. If user changes preference, data is updated in real time by categorizing system and sent recommendation system.

B. Frontend Application Organization

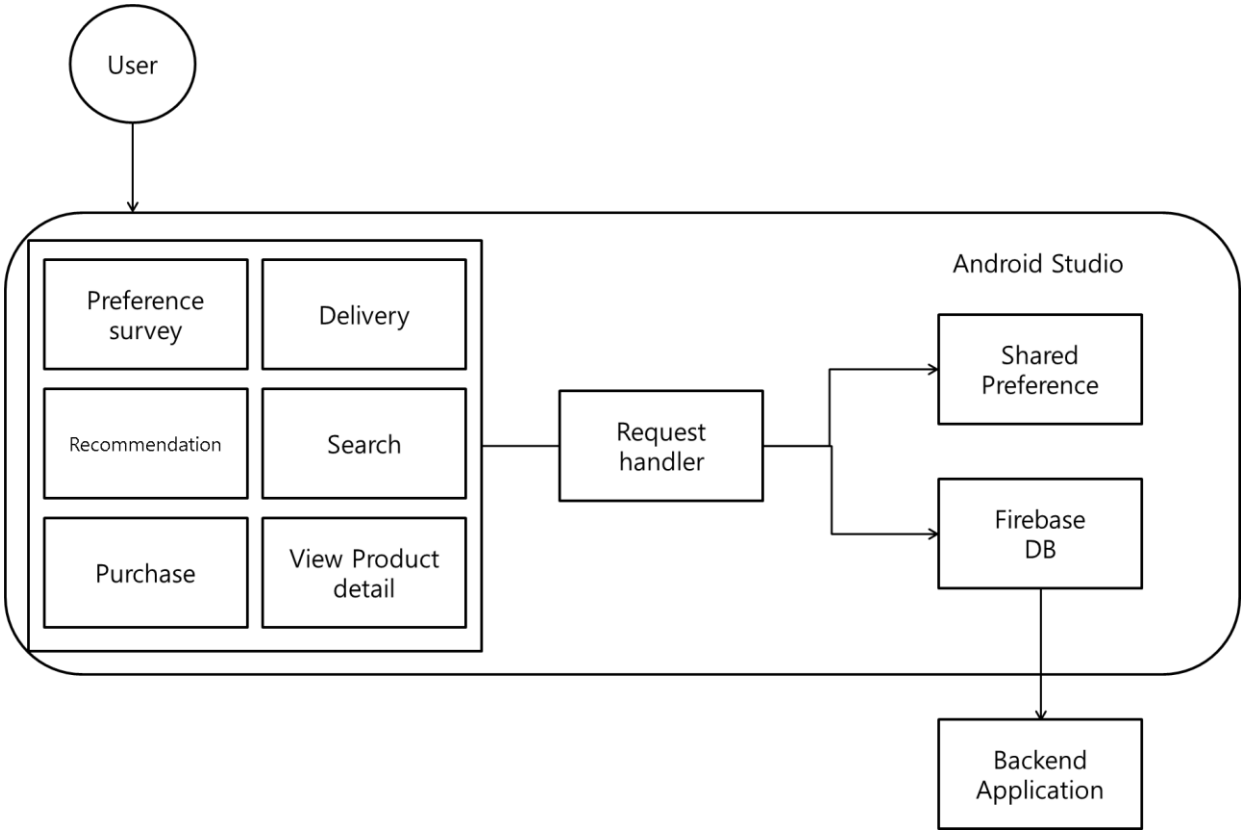


Figure 2. System Architecture - Frontend

Android studio is used frontend application framework. Each component requests data by using request handler. Request handler sends request message to backend application.

C. Backend Application Organization

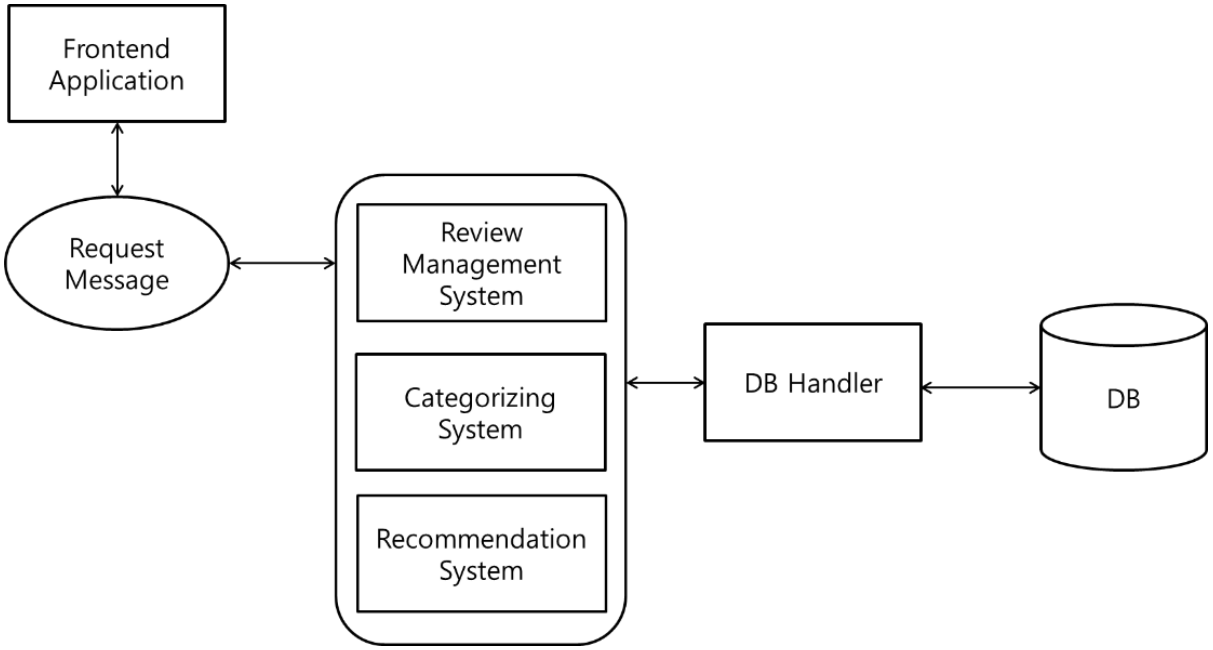


Figure 3. System Architecture - Backend

4. Sign up/Login System

4.1. Class Diagram

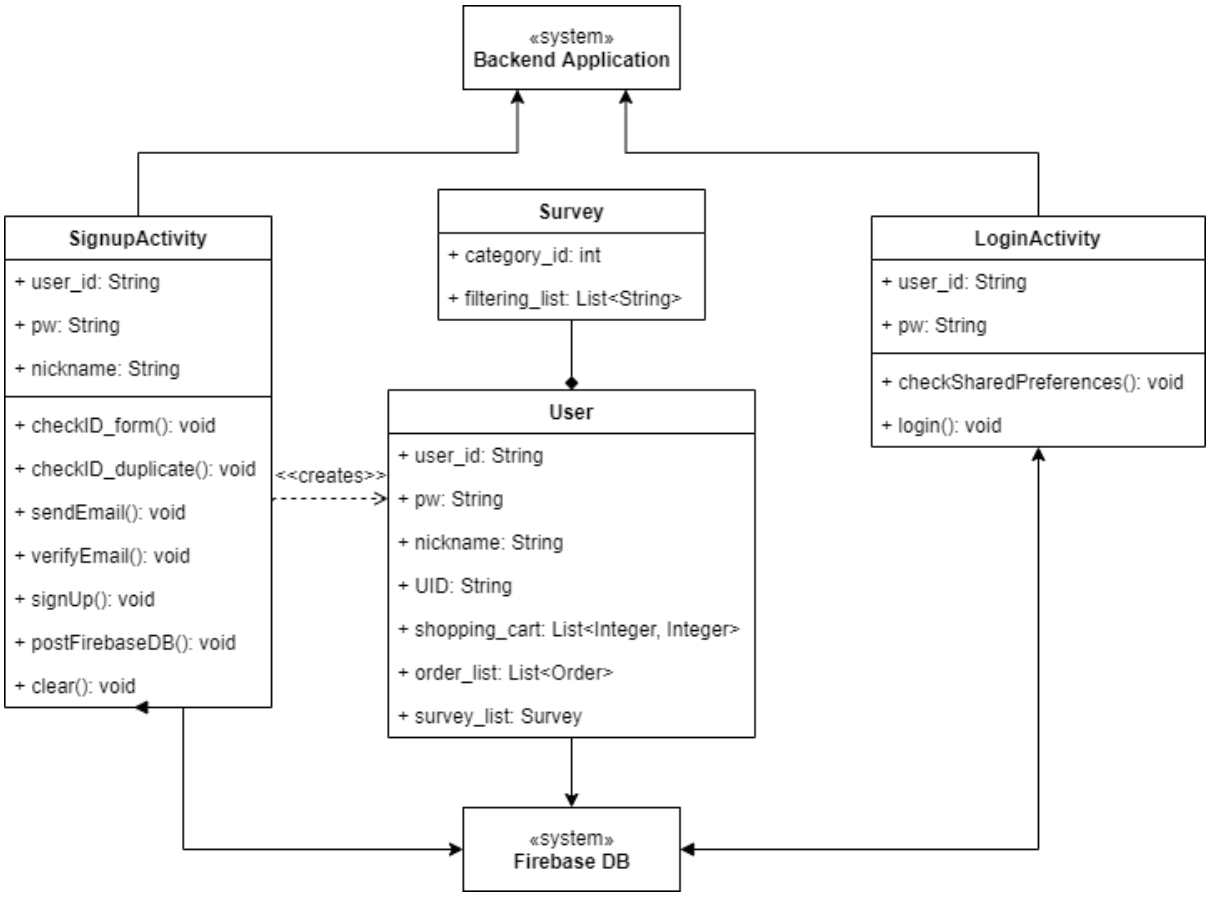


Figure 4. Sign up/Login System – Class Diagram

A. SignupActivity

(1) Attributes

+ user_id: user ID(email address)

+ pw: user password

+ nickname: user nickname

(2) Methods

+ checkID_form(): check if user_id(email address) form is correct

+ checkID_duplicate(): check if user_id is duplicate

+ sendEmail(): send verification email to user's email address

+ verifyEmail(): check if the email verification process is successfully done

+ signUp(): check if sign up process is successful and finish sign up process

+ postFirebaseDB(): save user information in DB

+ clear(): clear user input

B. LoginActivity

(1) Attributes

+ user_id: user ID(email address)

+ pw: user password

(2) Methods

+ checkSharedPreferences(): check if shared preferences is stored

+ login(): check if user input(user_id, pw pair) matches with the one in DB and finish login process

C. User

(1) Attributes

+ user_id: user ID(email address)

+ pw: user password

+ nickname: user nickname

+ UID: user ID(unique to the Firebase project)

+ shopping_cart: list of products in user's shopping cart

+ order_list: list of user's orders

+survey_list: user's preference survey response

D. Survey

(1) Attributes

+ category_id: user's preference about taste and main ingredient

+filtering_list: list of product filters(food that user hates, allergen, vegetarian)

4.2. Sequence Diagram

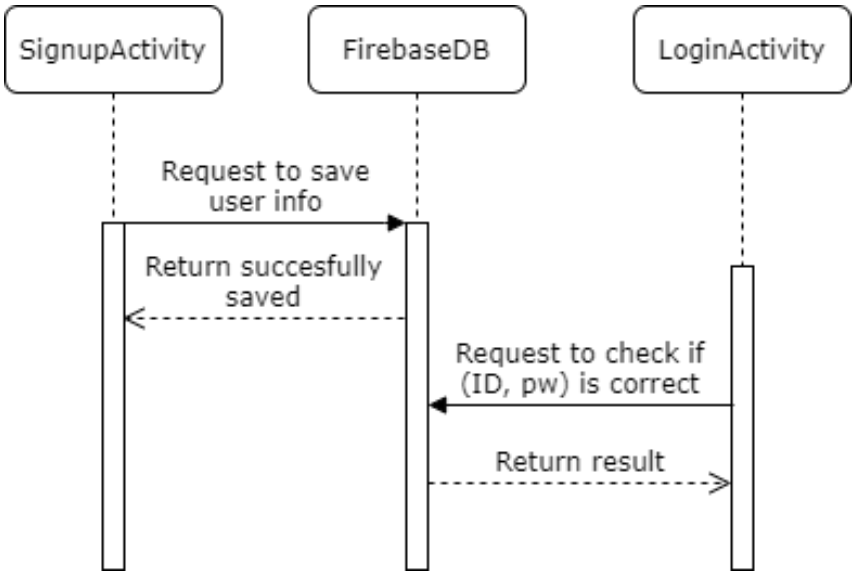


Figure 5. Sign up/Login – Sequence Diagram

5. Preference Survey System

5.1. Class Diagram

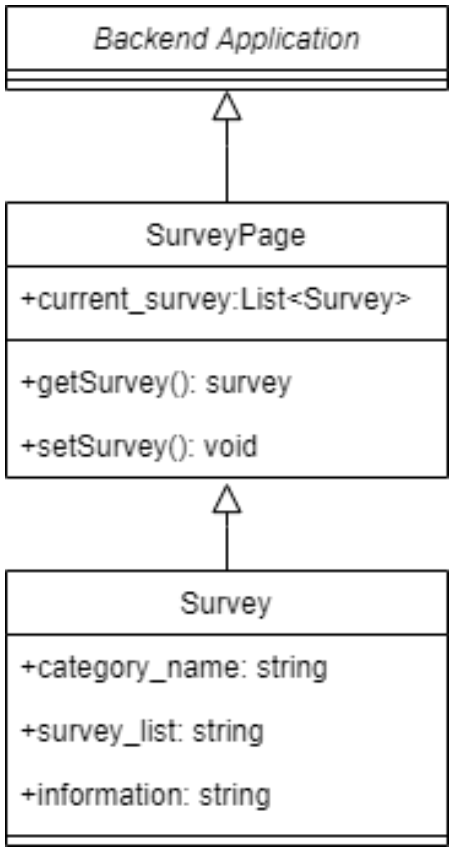


Figure 6. Preference Survey System – Class Diagram

A. SurveyPage

(1) Attribute

+current_survey : current survey

(2) Methods

+getSurvey(Survey): save the survey information

+setSurvey(Survey): bring the next survey information

B. Survey

(1) Attribute

+ category_name: type of survey

+ survey_list: survey result

+ information: detail description of survey

5.2. Sequence Diagram

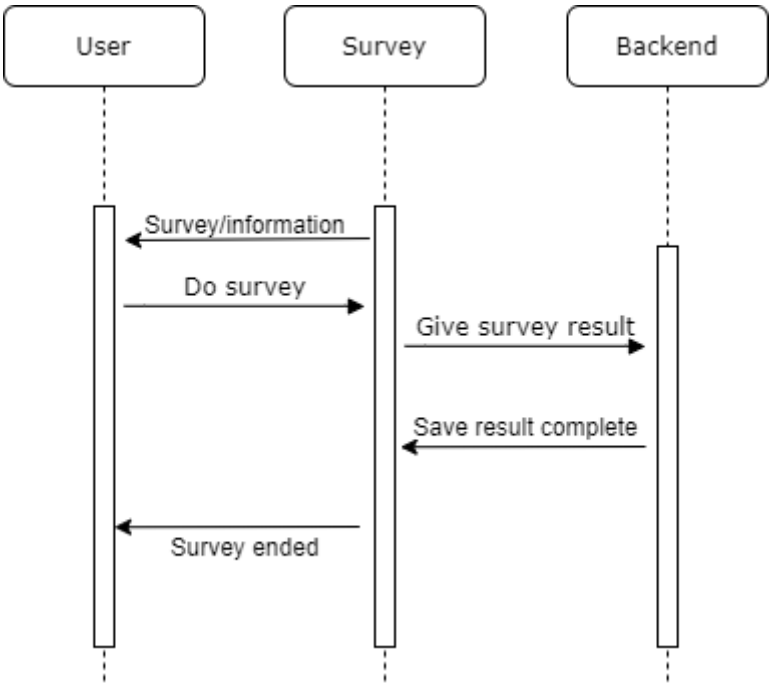


Figure 7. Preference Survey System – Sequence Diagram

6. Product Detail System

6.1. Class Diagram

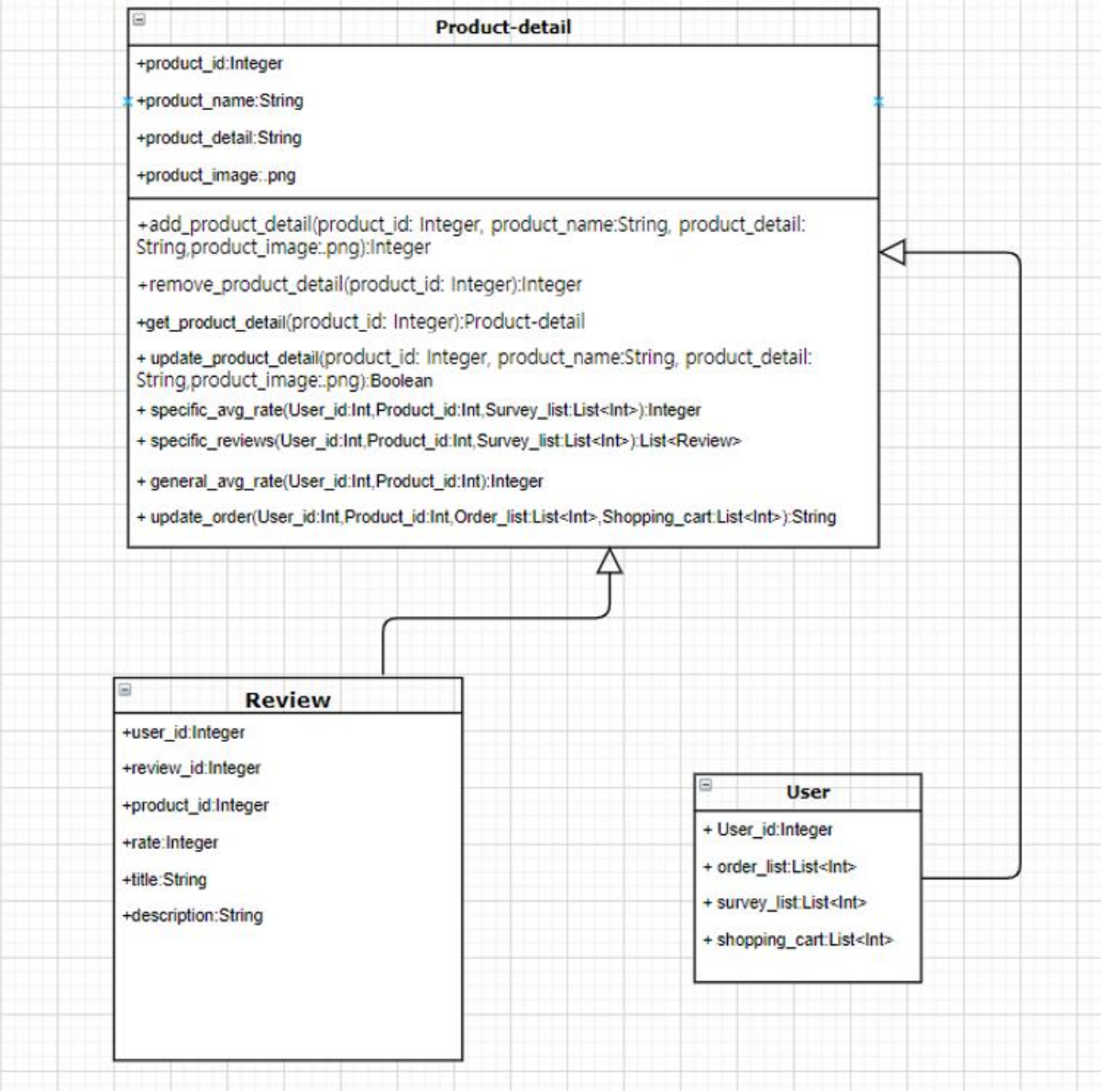


Figure 8. Product Detail System – Class Diagram

A. Product Detail

(1) Attributes

+ product_id: Product ID + product_name: Title of product

+ product_detail: Information of product + product_image: Photo of the product

(2) Methods

+add_product_detail (product_id: Integer, product_name: String, product_detail: String,product_image:.png): Add product detail

+remove_product_detail (product_id: Integer): Delete product detail

+get_product_detail (product_id: Integer): Get specific product from product id

+update_product_detail (product_id: Integer, product_name: String, product_detail: String,product_image:.png): Updating product detail

+specific_avg_rate(User_id:Int,Product_id:Int,Survey_list:List<int>): first of all it will get into reviews according to the user_id which will led the preference group of the user. Then,it will get the rates of each and calculate their average.

+specific_reviews(User_id:Int,Product_id:Int,Survey_list:List<int>): It will get the reviews according to the same preference group of user and will turn a list of Reviews.

+general_avg_rate(User_id:Int,Product_id:Int): This is similar with specific_avg_rate, the difference is that this one is not being calculated according to specific group of users but all the users who have been rated that product.

+update_order(user_id:Integer,product_id: Integer, Order_list:List<int>,Shoppingcart:List<Int>):It updates whether the product is directly purchased or in the chart or none of them.

6.2. Sequence Diagram

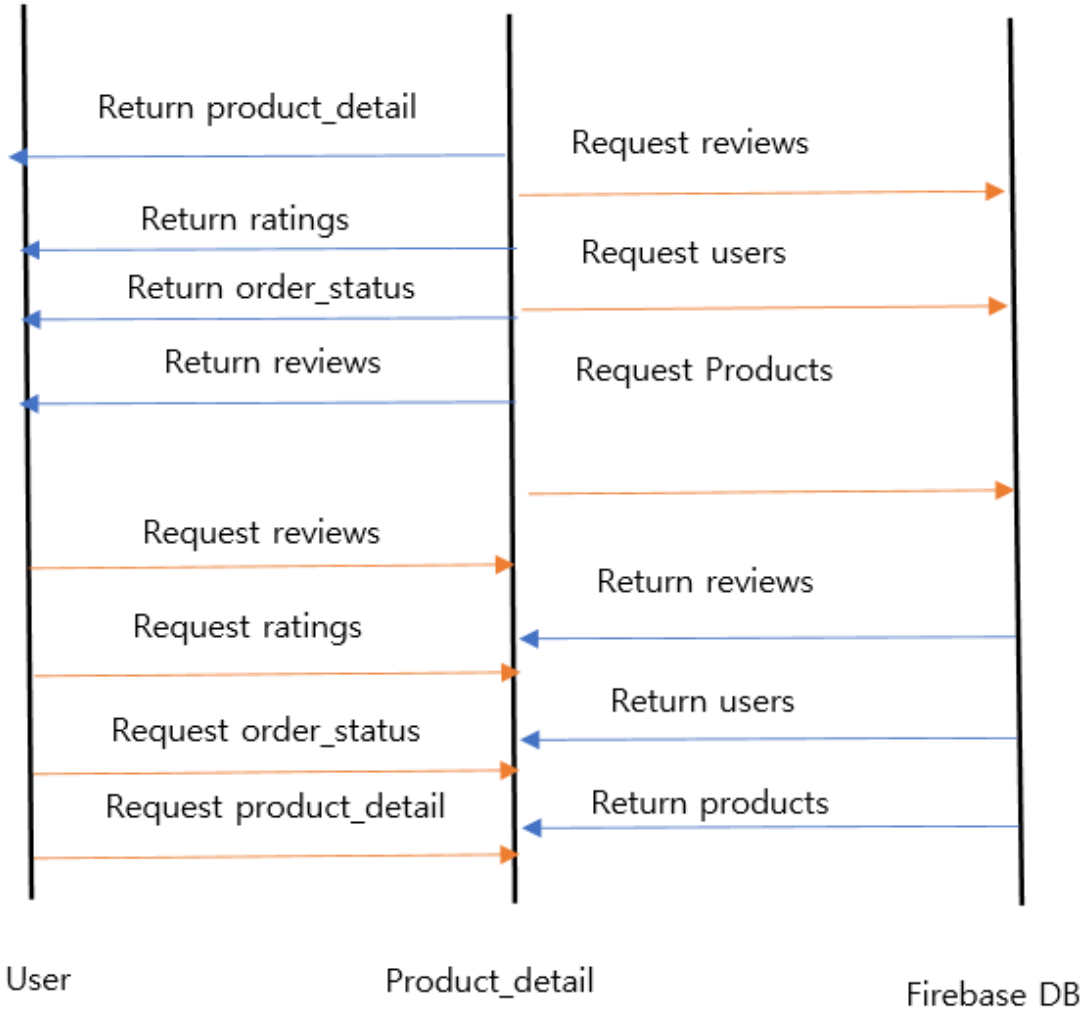


Figure 9. Product Detail System – Sequence Diagram

7. Search System

7.1. Class Diagram

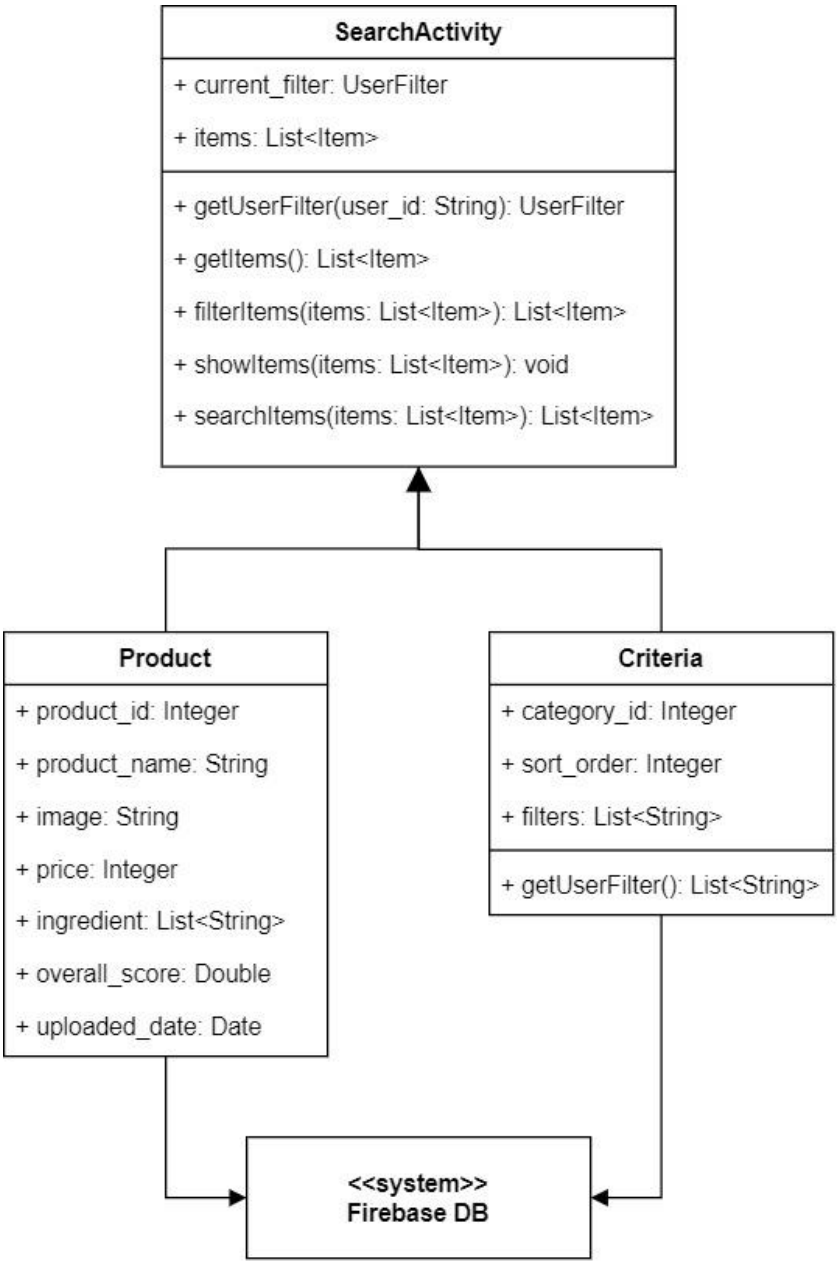


Figure 10. Search System – Class Diagram

A. SearchActivity

(1) Attributes

+ current_filter: filter which would be applied to search

+ items: item list to show

(2) Methods

+ getUserFilter(user_id: String): Get user filter from DB

+ getItems(): Get item list from DB

+ filterItems(items:List<Item>): Apply filter to item list

+ showItems(items: List<Item>): Show item list

+ searchItems(items: List<Item>): Search items with given filters.

B. Product

+ product_id: Id number of product

+ product_ name: name of the product

+ image: Image source of the product

+ price: Price of that item

+ ingredient: Possible allergens which are contained to the item or possible ingredients users prefer

+ overall_score: Score of the product which driven from review

+ uploaded_date: Product uploaded date

C. Criteria

(1) Attribute

+ category_id: Category ID based on user’s preference

+ sort_order: Sort order when showing list of products

+ filters: List of filtering

7.3. Sequence Diagram

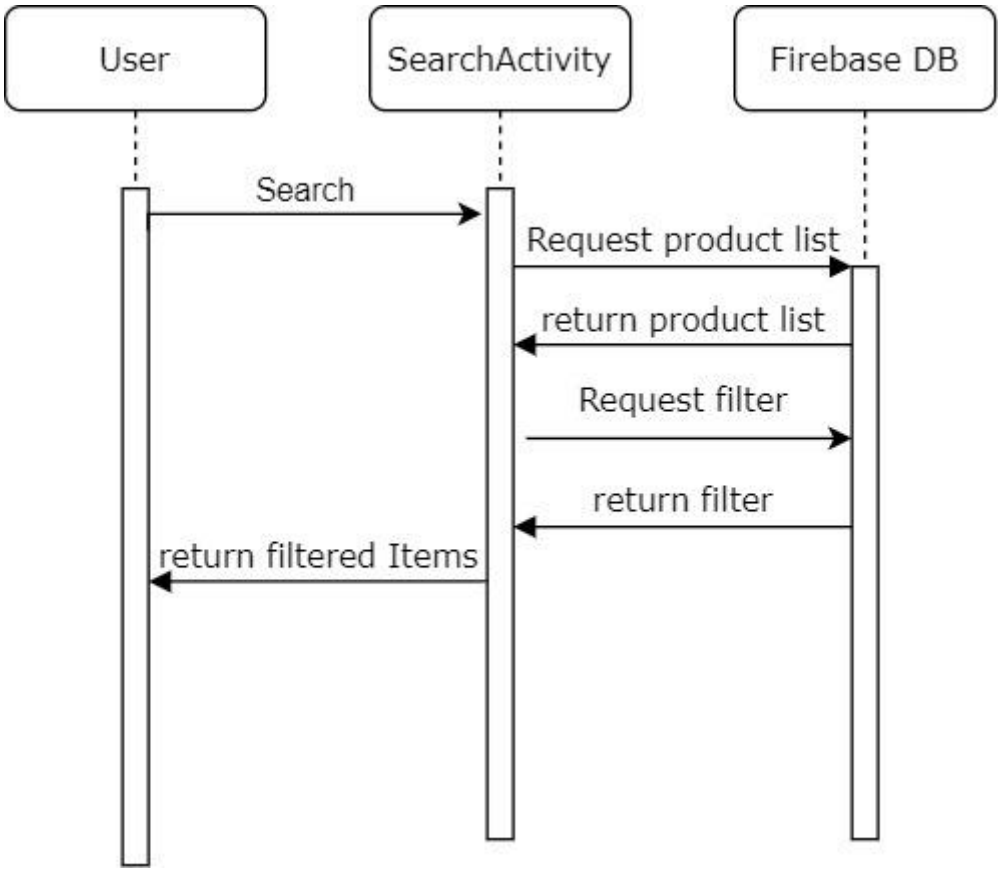


Figure 11. Search System – Sequence Diagram

8. Recommendation System

8.1. Class Diagram - Frontend

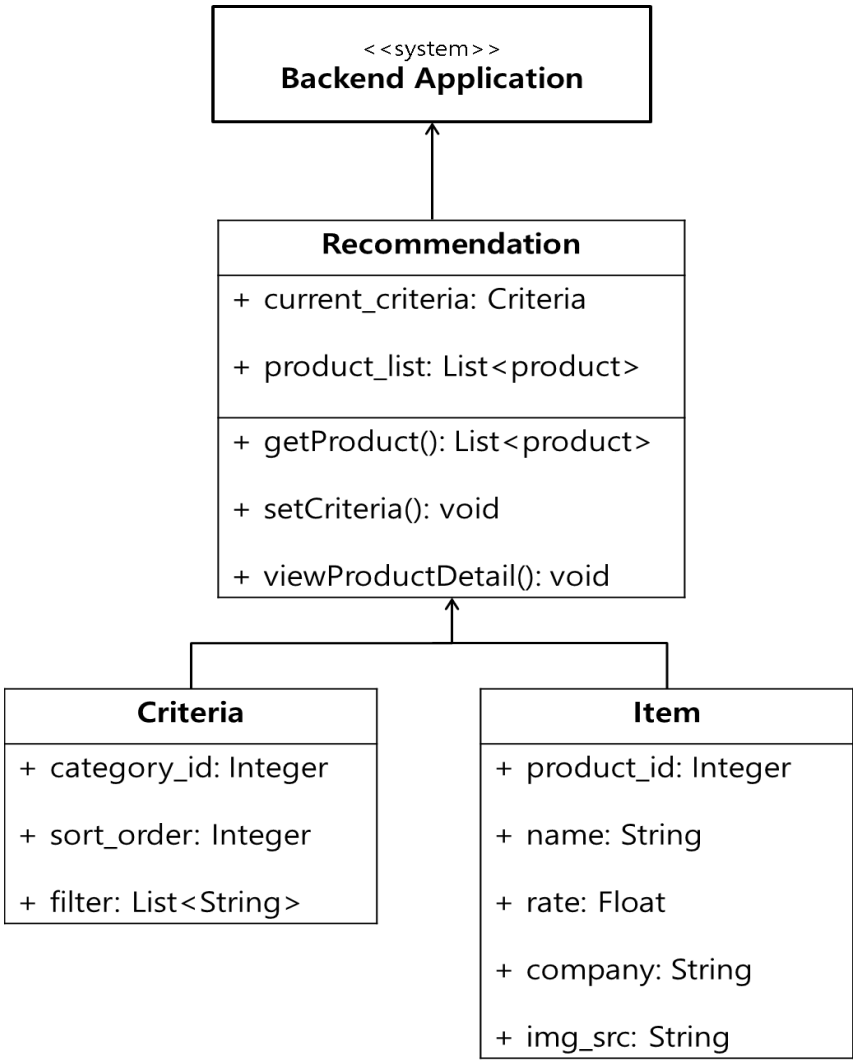


Figure 12. Recommendation System – Class Diagram - Frontend

A. Recommendation

(1) Attributes

+ current_criteria: Search criteria which is used printing item list

+ product_list: recommended product list

(2) Methods

+ getproduct(criteria: Criteria): Get item list which is matched search criteria from backend server

+ setCriteria(criteria: Criteria): Set criteria

+ viewproductDetail(product_id: Integer): Move to product detail page

B. Criteria

(1) Attributes

+ category_id: Category ID based on user’s preference

+ sort_order: sort order

+ filter: filter not to print based on user’s preference

C. Item

(1) Attributes

+ product_id

+ name

+ rate: rating matched category ID

+ company

+ img_src

8.2. Sequence Diagram – Frontend

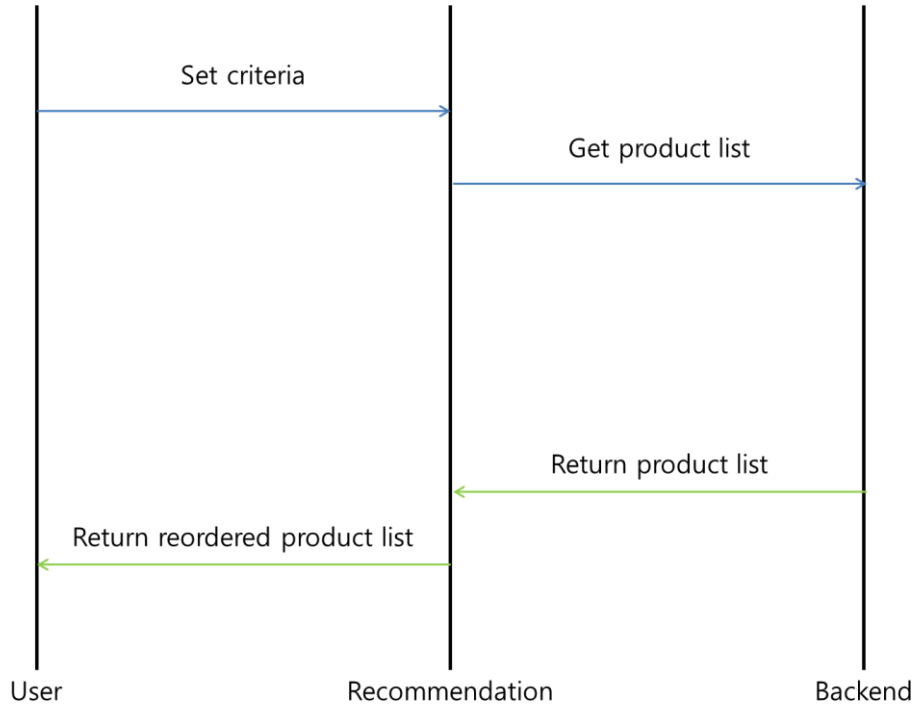


Figure 13. Recommendation System – Sequence Diagram - Frontend

8.3. Class Diagram – Backend

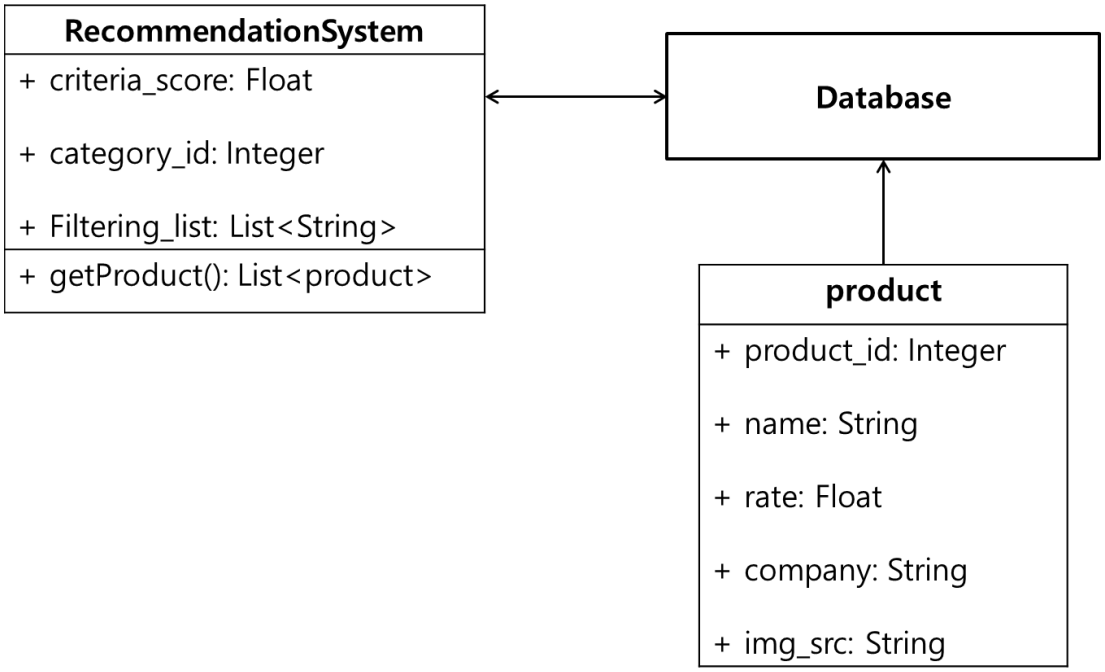


Figure 14. Recommendation System – Class Diagram - Backend

8.4. Sequence Diagram - Backend

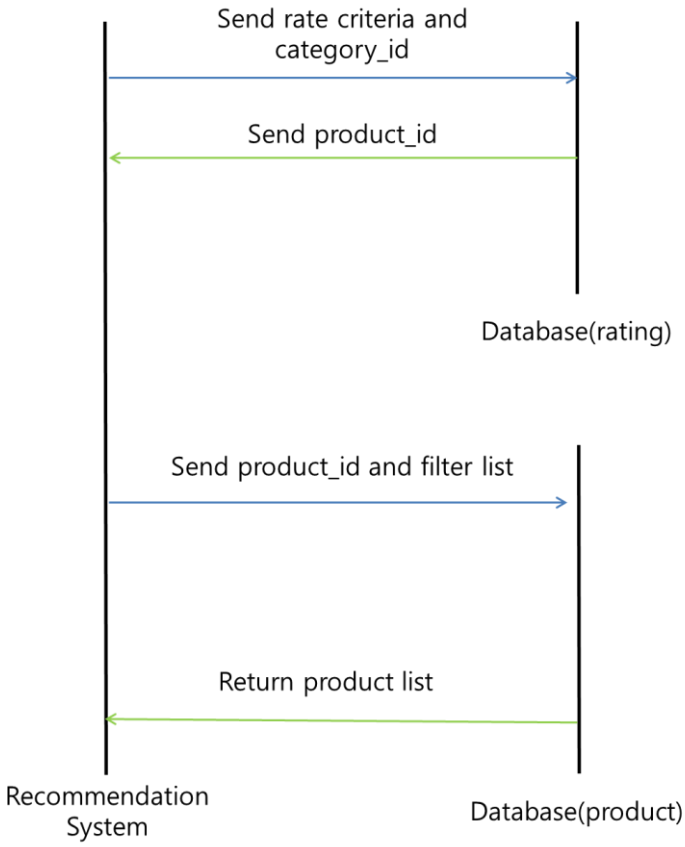


Figure 15. Recommendation System – Sequence Diagram - Backend

9. Review & Rating System

9.1. Class Diagram - Review

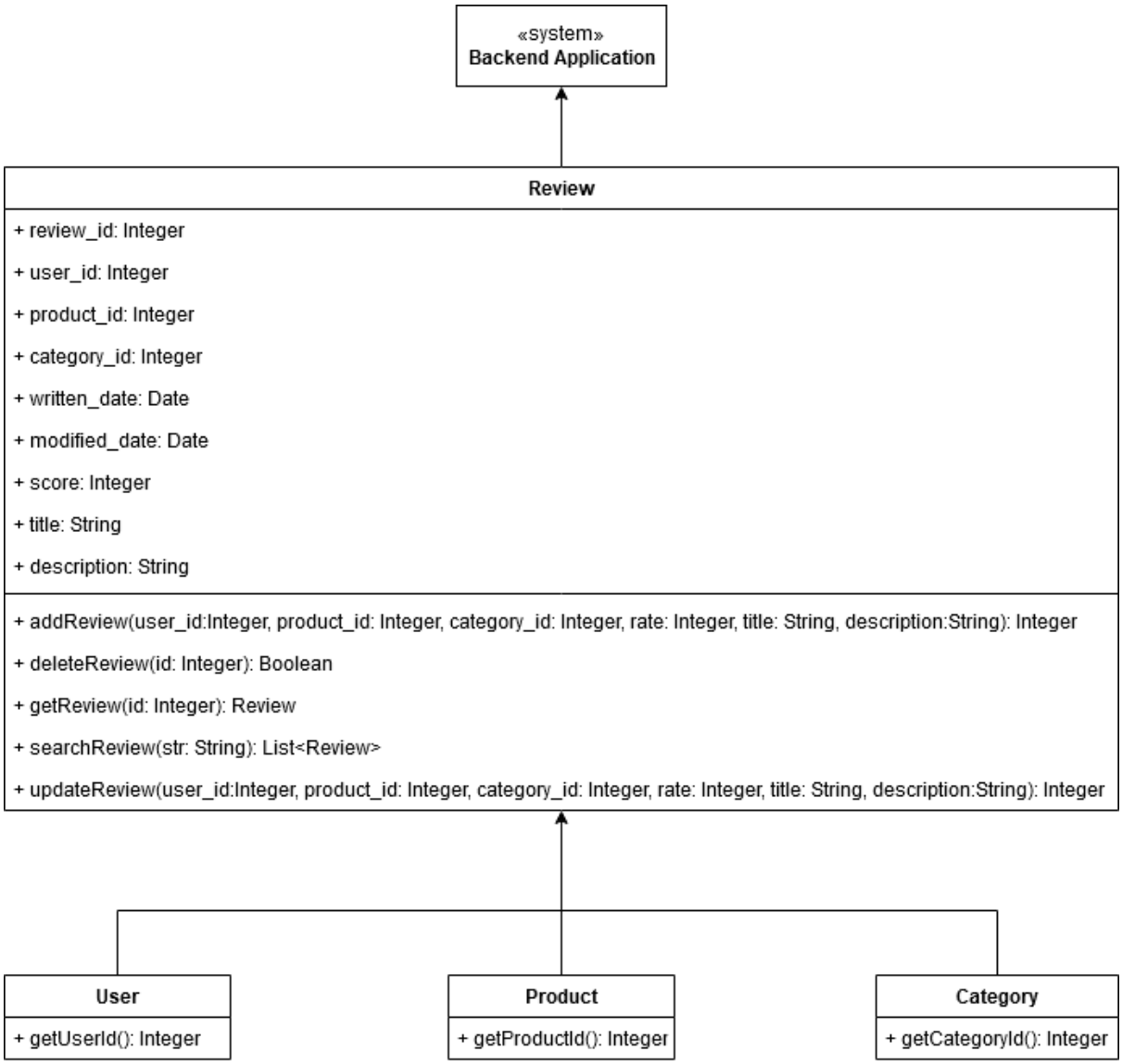


Figure 16. Review & Rating System – Class Diagram - Review

A. Review

(1) Attributes

+ review_id: Review ID

+ user_id: Review author ID

+ product_id: Review product ID

+ category_id: Category of Review

+ written_date: Review written Date

+ modified_date: Review modified Date

+ score: Score of product

+ title: Title of review

+ description: Content of review

(2) Methods

+ addReview(author_id: Integer, product_id: Integer, rate: Integer, title: String, description: String): Add review.
+ removeReview(id: Integer): Delete review.
+ getReview(id: Integer): Get specific review from review id.
+ searchReview(str: String): Search review list from keyword.
+ updateReview(id: Integer, rate: Integer, title: String, description: String): Update review contents.

B. User

(1) Methods

+ getUserId(): Retrieve User ID

C. Product

(1) Methods

getProductId(): Retrieve Product ID

D. Category

(1) Methods

getCategoryId(): Retrieve Category ID

9.2. Class Diagram – Review Handling System

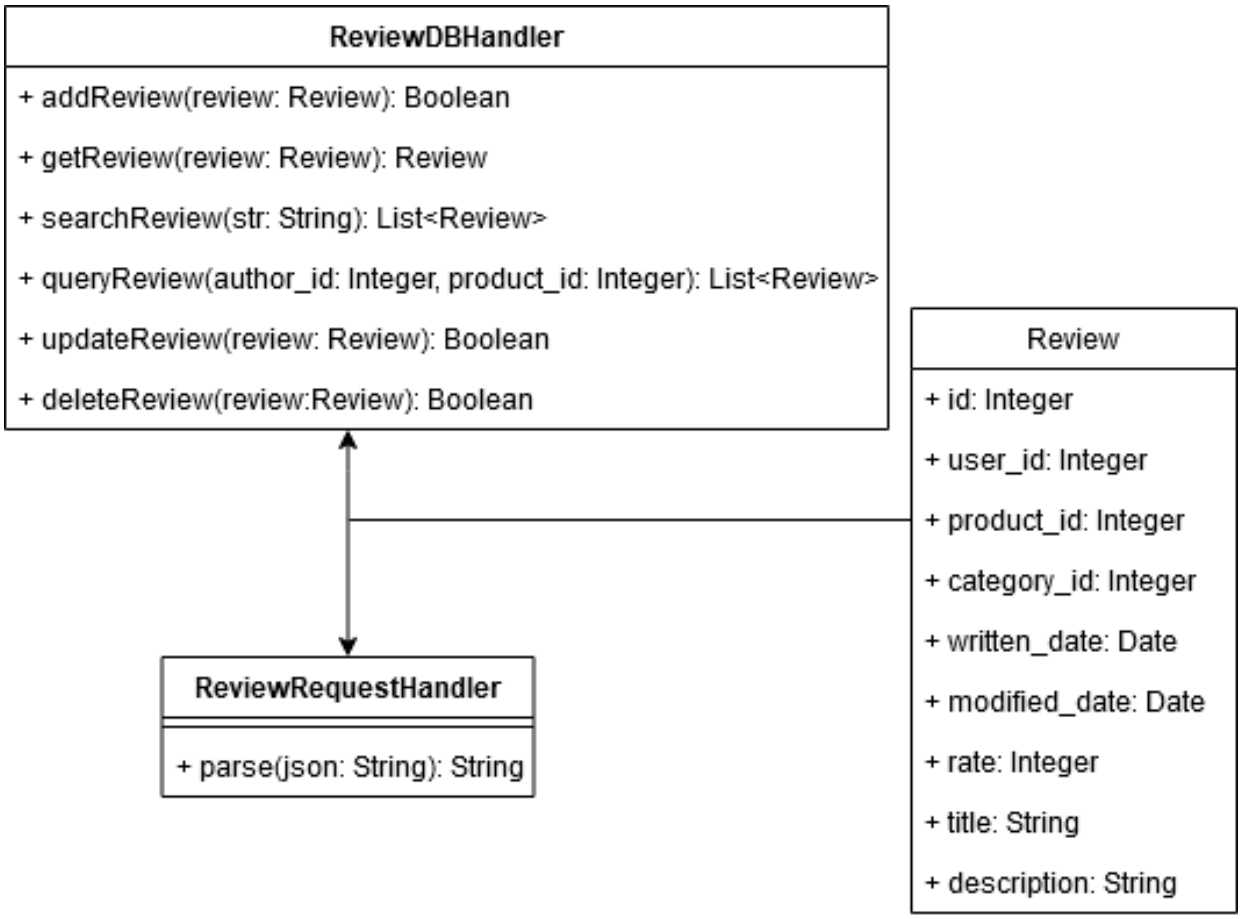


Figure 17. Review & Rating System – Class Diagram – Review Handling System

A. ReviewDBHandler

(1) Methods

addReview: Add specific reviews from given review.

getReview: Get a review that matches all conditions. Raise exception when found 2 or more reviews or none. Some review fields can be optional in Review.

searchReview: Search reviews from given keywords. Result should be List<Review> for multiple reviews.

searchReview: Search reviews from given keywords. Result should be List<Review> for multiple reviews.

queryReview: Query reviews from given ids. Both author_id and product_id is optional, but Either of them should be provided. Result should be List<Review> for multiple reviews.

updateReview: Update specific review's content.

deleteReview: Delete specific review.

B. ReviewRequestHandler

(1) Methods

Parse: parse json request and return json result.

9.3. Sequence Diagram

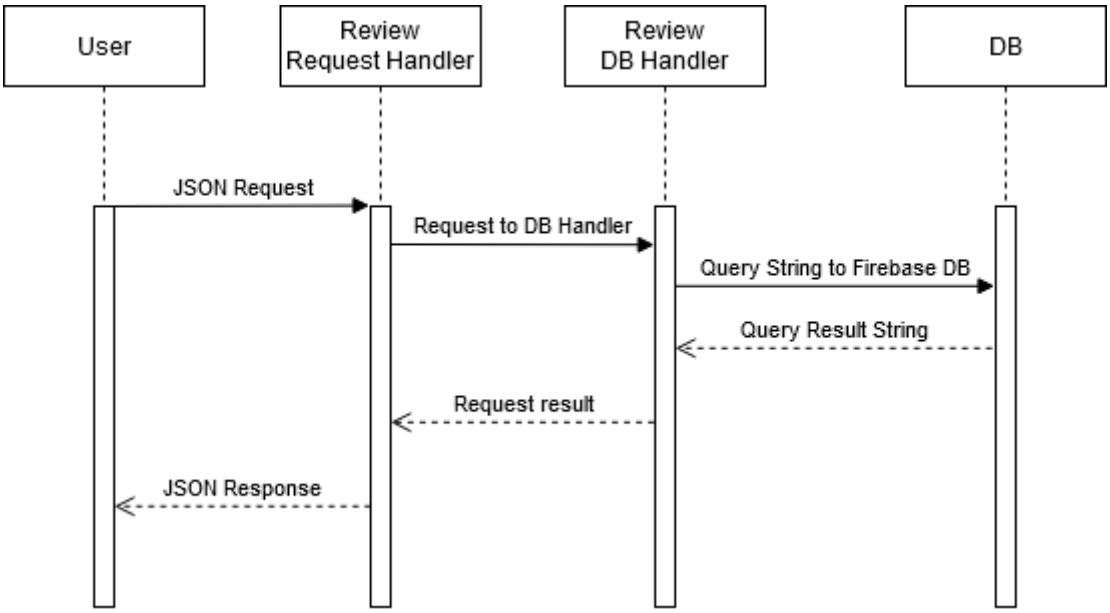


Figure 18. Review & Rating System – Sequence Diagram

10. Protocol Design

10.1. Objectives

At protocol design, there's description how communication is performed between subsystems. Protocol's basic format is JSON.

10.2. Protocol Format

JSON

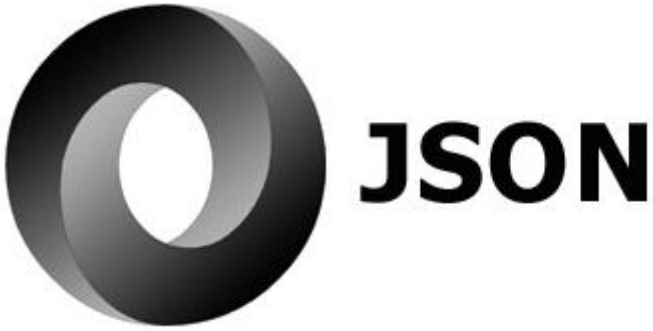


Figure 19. JSON icon

JSON (JavaScript Object Notation) is a lightweight data-interchange format. It is easy for humans to read and write. It is easy for machines to parse and generate. JSON is a text format that is completely language independent but uses conventions that are familiar to programmers of the C-family of languages, including C, C++, C#, Java, JavaScript, Perl, Python, and many others. These properties make JSON an ideal data-interchange language (Json,n.d.).

JSON is built on basically name/value pairs.

10.3. Protocol Description

A. Overview

This part describes messages used in application ‘Fooding’. Each message has attribute/value pairs.

Messages are divided into two types, ‘Request’ and ‘Response’. ‘Request’ means message is sending from client to server. ‘Response’ is message sending from server to client.

B. Sign up/Login System

(1) Sign up

- Request

Attribute	Value
user_id	user ID(email address)
pw	user password
nickname	user nickname

Table 1. Sign Up - Request

- Response

Attribute	Value
signup_success	true/false

Table 2. Sign up - Response

(2) Login

- Request

Attribute	Value
user_id	user ID(email address)
pw	user password

Table 3. Login - Request

- Response

Attribute	Value
login_success	true/false

Table 4. Login - Response

C. Preference Survey System

- Request

Attribute	Value
recommend_item	category_id(int)
filtering_item	filter list(List<string>)

Table 5. Preference Survey - Request

- Response

Attribute	Value
survey_list	survey list(List<category_id, filter_list>)

Table 6. Preference Survey - Response

D. Product Detail System

(1) Add product detail

- Request

Attribute	Value
Product_id	Id of product
Product_name	Name of product
Product_detail	Information about product
Product_image	Photo of product

Table 7. Add Product Detail - Request

- Response

Attribute	Value
Status_code	Success or not

Table 8. Add Product Detail - Response

(2) Remove product detail

- Request

Attribute	Value
Product_id	Id of product

Table 9. Remove Product Detail - Request

- Response

Attribute	Value
Status_code	Success or not

Table 10. Remove Product Detail - Response

(3) Get product detail

- Request

Attribute	Value
Product_id	Id of product

Table 11. Get Product Detail - Request

- Response

Attribute	Value
Product_id	Id of product
Product_name	Name of product
Product_detail	Information about product
Product_image	Photo of product

Table 12. Get Product Detail - Request

(4) Update product detail

- Request

Attribute	Value
Product_id	Id of product
Product_name	Name of product
Product_detail	Information about product
Product_image	Photo of product

Table 13. Update Product Detail - Request

- Response

Attribute	Value
Status_code	Success or not

Table 14. Update Product Detail - Response

(5) Specific avg rate

- Request

Attribute	Value
User_id	Id of user
Product_id	Id of product
Survey_list	List of matched people

Table 15. Specific Avg Rate - Request

- Response

Attribute	Value
Avg_rate	Integer

Table 16. Specific Avg Rate - Response

(6) Specific reviews

- Request

Attribute	Value
User_id	Id of user
Product_id	Id of product
Survey_list	List of matched people

Table 17. Specific Reviews - Request

- Response

Attribute	Value
Specific_reviews	List<Reviews>

Table 18. Specific Reviews - Response

(7) General avg rate

- Request

Attribute	Value
User_id	Id of user
Product_id	Id of product

Table 19. General Avg Rate - Request

- Response

Attribute	Value
General_Avg_rate	Integer

Table 20. General Avg Rate - Response

(8) Update order

- Request

Attribute	Value
User_id	Id of user
Product_id	Id of product
Order_list	List of ordered items
Shoppingcart_list	List of items that are in shopping cart

Table 21. Update Order - Request

- Response

Attribute	Value
Order_status	An integer value indicating which state

Table 22. Update Order - Response

E. Search System

(1) Get Product List

- Request

Attribute	Value
User_id	Id of user

Table 23. Get Product List - Request

- Response

Attribute	Value
product_list	product list(List<product>)

Table 24. Get Product List - Response

(2) Get survey_list

- Request

Attribute	Value
User_id	Id of user

Table 25. Get Survey List - Request

- Response

Attribute	Value
Survey_list	List of matched people

Table 26. Get Survey List - Response

F. Recommendation System

- Request

Attribute	Value
category_id	user preference(int)
rate_criteria	rate criteria(float)
filtering_list	filter list(List<string>)

Table 27. Recommendation - Request

- Response

Attribute	Value
product_list	product list(List<product>)

Table 28. Recommendation - Response

G. Review & Rating System

(1) Add review

- Request

Attribute	Value
Product_id	Product ID
Rate	Rate of review
Title	Title of review
Description	Description of review

Table 29. Add Review - Request

- Response

Attribute	Value
Review_id	Id of review added.
Status_code	Success? Not Logged in? No Perm? Not found?

Table 30. Add Review - Response

(2) Delete review

- Request

Attribute	Value
Review_id	Review ID

Table 31. Delete Review - Request

- Response

Attribute	Value
Status_code	Success? Not Logged in? No Perm? Not found?

Table 32. Delete Review - Response

(3) Get review

- Request

Attribute	Value
Review_id	Id of review added.

Table 33. Get Review - Request

- Response

Attribute	Value
Status_code	Success? Not Logged in? No Perm? Not found?
Review_id	Id of review
Author_id	Id of review author
Product_id	Id of review product
rate	Rate of review
title	Title of review
description	Description of review

Table 34. Get Review - Response

(4) Search review

- Request

Attribute	Value
author_id	Id of review author
product_id	Id of review product
rate_lower	Lower bound of rate
rate_upper	Upper bound of rate
keyword	Search keyword in all field
title	Search keyword in title
description	Search keyword in description

Table 35. Search Review - Request

- Response

Attribute	Value
Status_code	Success? Not Logged in? No Perm? Not found?
Reviews	List of reviews

Table 36. Search Review - Response

(5) Update review

- Request

Attribute	Value
Review_id	Review ID
Product_id	Product ID
Rate	Rate of review
Title	Title of review
Description	Description of review

Table 37. Update Review - Request

- Response

Attribute	Value
Review_id	Id of review added.
Status_code	Success? Not Logged in? No Perm? Not found?

Table 38. Update Review - Response

11. Database Design

11.1. Objectives

Detailed descriptions based on requirement specification about database, using ER Diagram to show the relationship between entities. Followed by Relational Schema, Json tree.

11.2. ER Diagram

There are user, review, product, and order entities, which are represented at the rectangular box below. And the attributes about entities will be expressed using oval shape. The relationship is described inside rhombus box. The primary key of each entity is underlined inside the corresponding attribute.

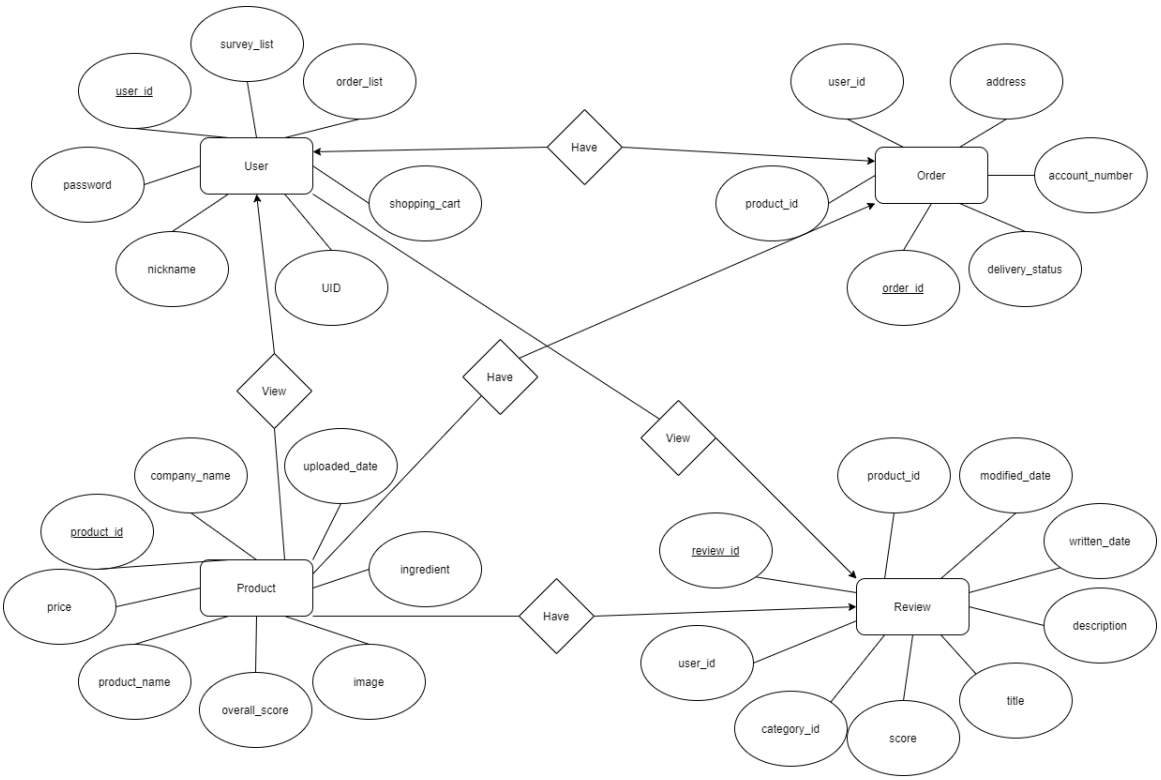


Figure 20. ER Diagram - Overall

A. ER Diagram - User

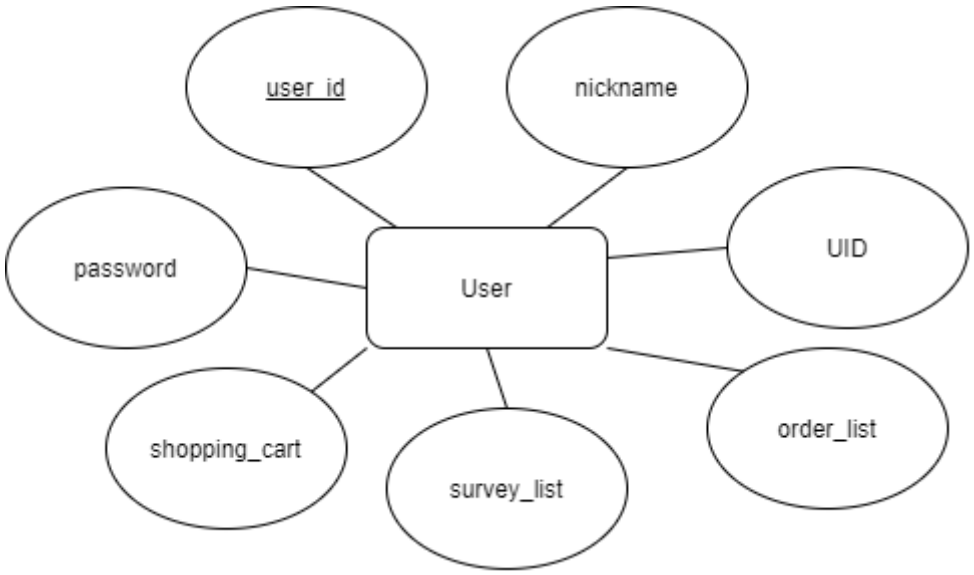


Figure 21. ER Diagram - User

User entity includes the information such as nickname, survey preference result, password, UID, product order list, and the primary key is user_id.

B. ER Diagram – Product

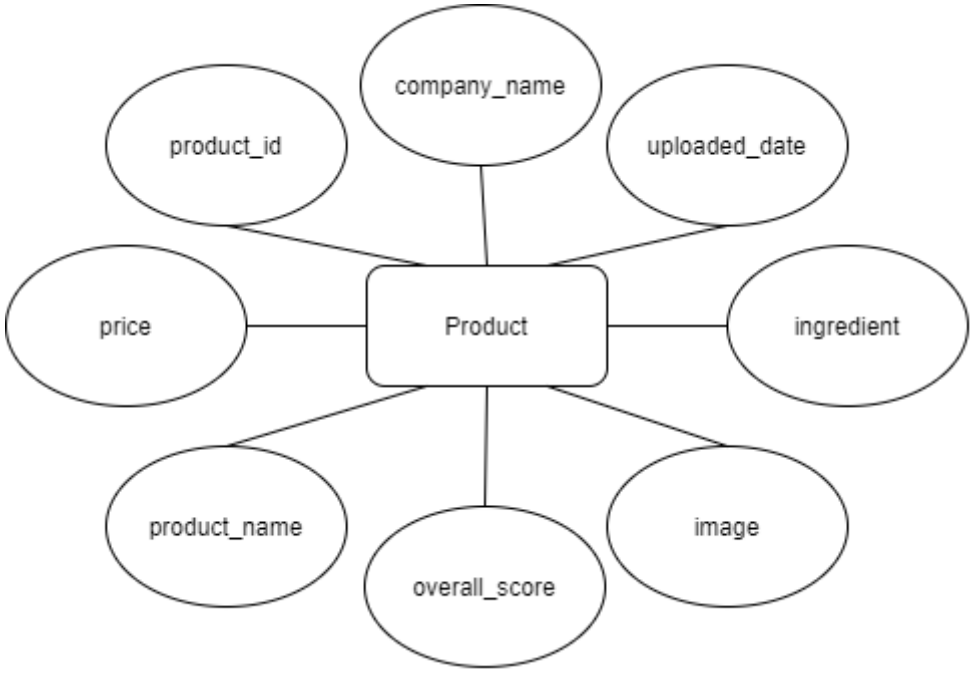


Figure 22. ER Diagram - Product

Product entity includes the information about product, such as product's name and its company name, uploaded date to our application, ingredient, image, and prize of the product, overall_score is based on the score that user who bought the product gave. The primary key is product_id.

C. ER Diagram – Order

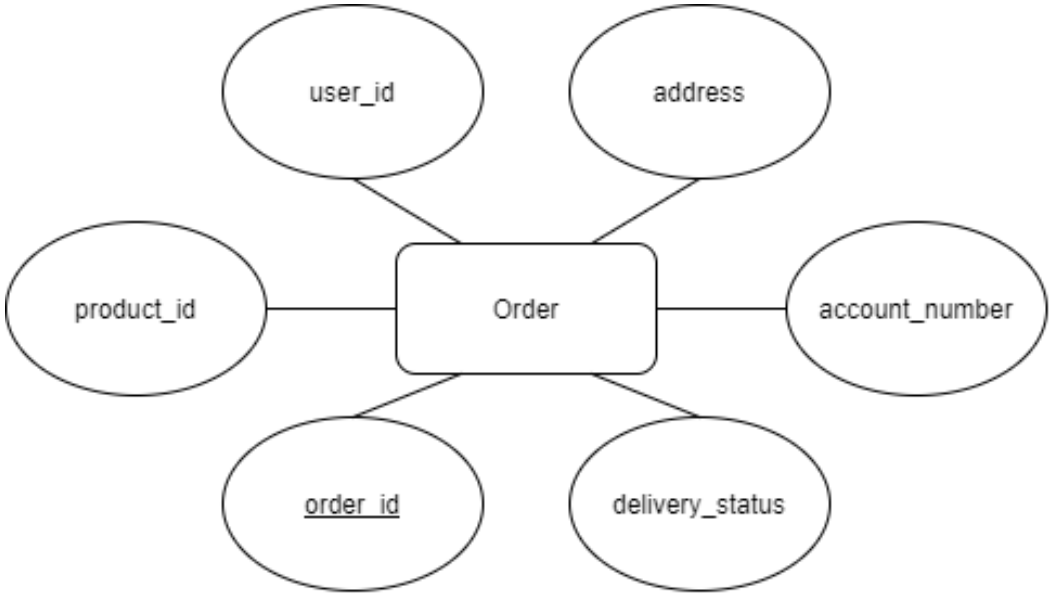


Figure 23. ER Diagram - Order

Order entity includes the information about product's order, such as ordered product_id, user_id, account_number, delivery_status and address of the user. The primary key is order_id.

D. ER Diagram - Review

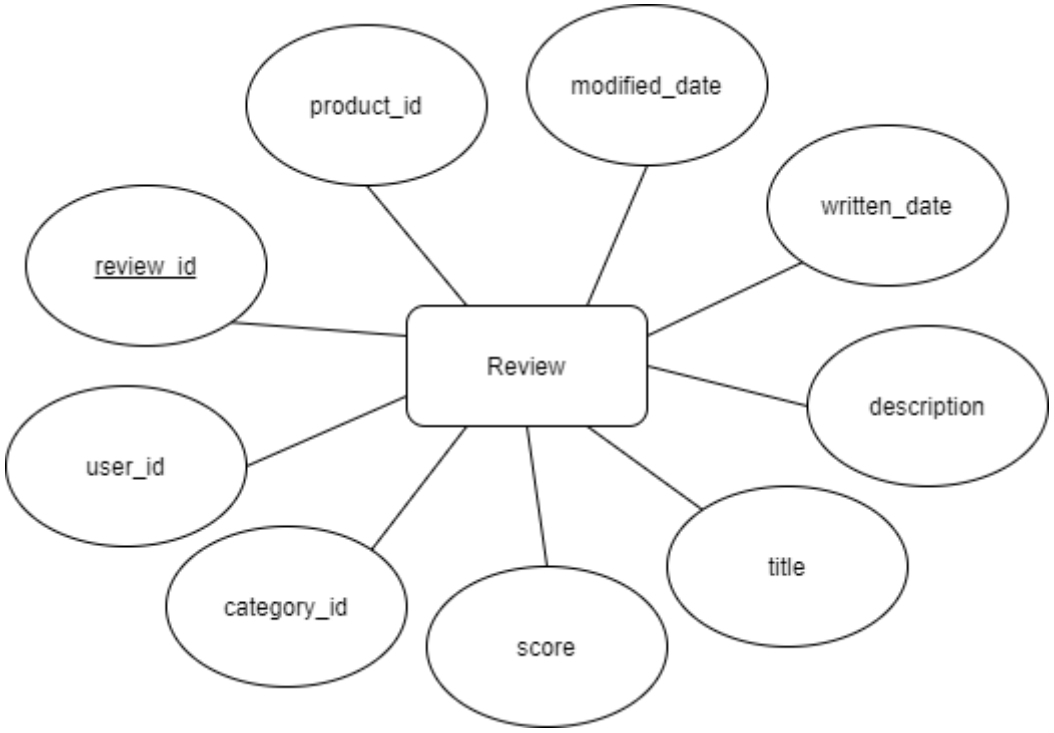


Figure 24. ER Diagram - Review

Review entity includes the information about user's review, such as ordered product_id, category_id, score, title, description, written_date. And modified_date. The primary key is review_id.

11.3. Relational Schema

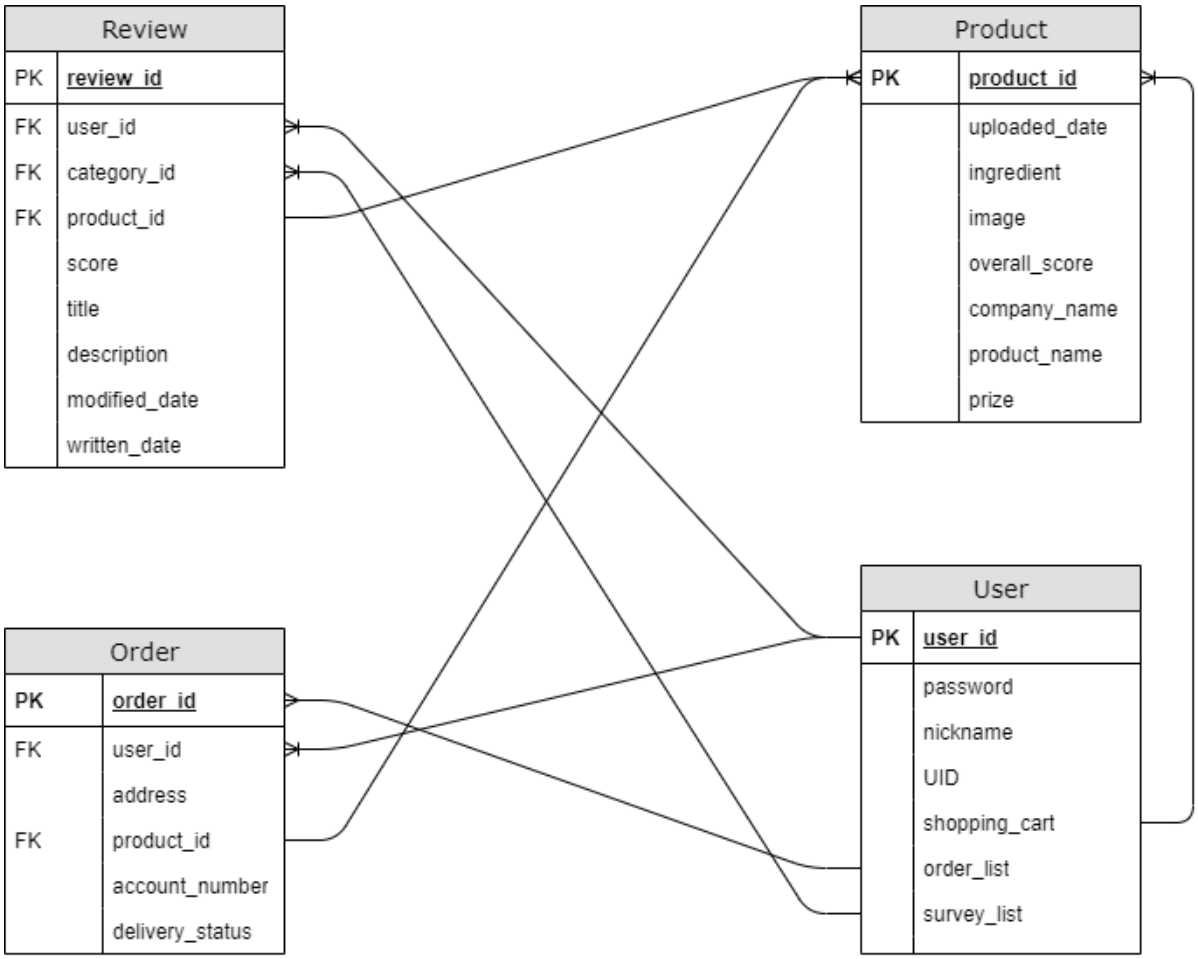


Figure 25. Database Relational Schema

11.4. JSON Tree

A. User

User

```
{
  “user”: {
        “user1”: {
              “user_id”: .....
              “pw”: .....
              “nickname”: .....
              “UID”: .....
              “shopping_cart”: .....
              “order_list”: .....
              “survey_list”: .....
        }
  },
}
```

B. Order

```
{
    “order”: {
        “order1”: {
            “user_id”: .....
            “product_id”: .....
            “address”: .....
            “account_numver”: .....
            “delivery_status”: .....
        }
    },
}
```

C. Product

```
{
    "product": {
        "product1": {
            "product_id": .....
            "uploaded_date": .....
            "ingredient": .....
            "image": .....
            "prize": .....
            "overall_score": .....
            "product_name": .....
            "company_name": .....
        }
    },
}
```

D. Review

{

“review”: {

“review1”: {

“review_id”: .....

“user_id”: .....

“title”: .....

“description”: .....

“shopping_cart”: .....

“order_list”: .....

“survey_list”: .....

}

},

}

12 Testing Plan

12.1. Objectives

This chapter describes about testing policy that will be used to test our system and check whether the system has an error.

12.2. Testing Policy

A. Development Testing

Development testing is performed during the construction stage of software development process. Development testing aims to eliminate construction errors before performing other tests. Development testing includes 4 testing process; component testing, integrating testing, system testing and acceptance testing.

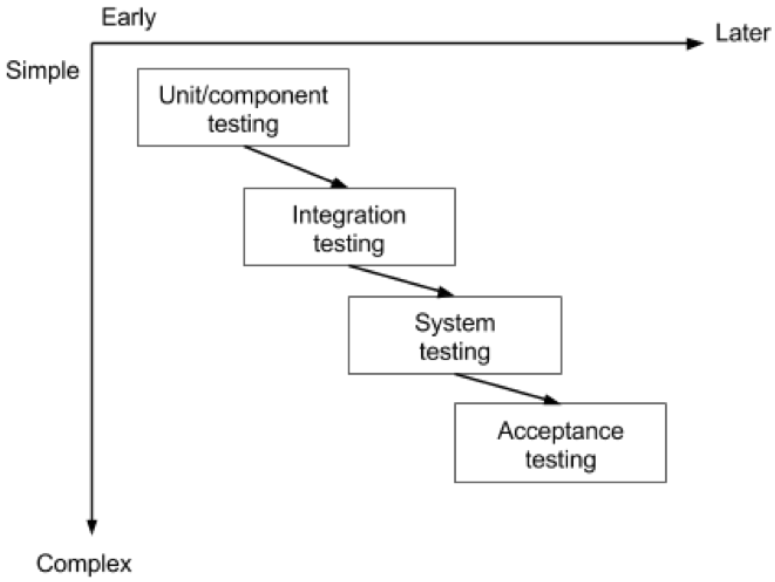


Figure 26. Development Testing Process

(1) Component Testing

Component testing independently verifies the functionality of each component. Component may be functions, objects or coherent groupings of these entities.

(2) Integration Testing

Integration testing is performed to ensure that the system is functioning properly while integrating the subsystems.

(3) System Testing

System testing tests the system as a whole to check whether the entire system is functioning properly.

(4) Acceptance Testing

Acceptance testing uses customer’s data to check whether the system meets the customer’s needs.

B. Release Testing

Release testing tests a new version of the software to verify that the software can be released. It tests the entire functionality of the software and check if there is any issue or error.

C. User Testing

(1) Usability Testing

Usability testing checks if the user interface is easy to use and understand.

(2) Closed Beta Testing

Beta version of the software is released to restricted group of people. Beta testing is performed in real conditions because it aims to identify problems in real conditions.

D. Test cases

Test cases are generated based on use case of the system. A test case consists of input and expected output. To make sure that the system is working properly, expected output of the test case should be same as the actual output.

13. Development Plan

13.1. Objectives

Describes the programming language and the IDE that is going to be used. In addition to create a workflow for efficient development.

13.2. Programming Language & IDE

A. JAVA

JAVA is used as the development programming language. JAVA is an object-oriented language and has the advantage of running on any platform. It is widely used in Android application development language and is suitable for our project.

B. Firebase

For the backend database development side, we used Firebase. Firebase is a technology that allows you to create web applications without server-side programming, making development faster and easier. It supports Web, iOS, OS X and Android clients. Apps that use Firebase can use and control data without thinking about how data is stored and synchronized across different instances of the application in real-time. That would ease our development process.

C. Android Studio

The IDE used in the project development process is Android Studio. Android Studio is the official integrated development environment for Android and Android applications. Java and Kotlin are the programming languages used in Android Studio.

13.3. Version Management Tool

GitHub

GitHub will be used for code management and more efficient development. GitHub is a web hosting service that provides extended project management support to Git, a version management system. It is currently one of the most popular web hosting services for open source project management around the world.

13.4. Workflow dates

Requirements specification 13th May

Design specification 24th May

Code and test result 7th June

Final presentation 14th June

Code review 22nd June - 26th June

14. Index

14.1. Figure

14.2. Table

14. Reference

Json. (n.d.). Introducing JSON. Retrieved May 24, 2020, from https://www.json.org/json-en.html