Design

Specification

METAVERSE LIBRARY SYSTEM

Introduction to Software Engineering 41 - Group 11

|  |  |
| --- | --- |
| 2016312818 | Changmin Park |
| 2018311925 | Dajung Choi |
| 2018312257 | Youngsuh Chin |
| 2019313857 | Amirah Binti Ahmad Nadzri |
| 2016314576 | Jaehyuk Choi |
| 2016314494 | Wonjae Lee |
| 2019313897 | Angel Fitri Sari |

Contents

[1. Preface 7](#_Toc88432616)

[1.1. Readership 7](#_Toc88432617)

[1.2. Scope 7](#_Toc88432618)

[1.3. Objective 7](#_Toc88432619)

[1.4. Document Structure 8](#_Toc88432620)

[2. Introduction 10](#_Toc88432621)

[2.1. Objectives 10](#_Toc88432622)

[2.2. Applied Diagrams 10](#_Toc88432623)

[2.2.1. UML 10](#_Toc88432624)

[2.2.2. Use case Diagram 11](#_Toc88432625)

[2.2.3. Sequence Diagram 11](#_Toc88432626)

[2.2.4. Class Diagram 12](#_Toc88432627)

[2.2.5. Context Diagram 12](#_Toc88432628)

[2.2.6. Entity Relationship Diagram 13](#_Toc88432629)

[2.3. Applied Tools 13](#_Toc88432630)

[2.3.1. Microsoft PowerPoint 13](#_Toc88432631)

[2.3.2. ERD Plus 13](#_Toc88432632)

[2.4. Project scope 14](#_Toc88432633)

[2.5. References 14](#_Toc88432634)

[3. System Architecture - Overall 15](#_Toc88432635)

[3.1. Objectives 15](#_Toc88432636)

[3.2. System Organization 16](#_Toc88432637)

[3.2.1. Context Diagram 16](#_Toc88432638)

[3.2.2. Sequence Diagram 17](#_Toc88432639)

[3.2.3. Use Case Diagram 18](#_Toc88432640)

[4. System Architecture 18](#_Toc88432641)

[4.1. Objectives 18](#_Toc88432642)

[4.2. Subcomponents 19](#_Toc88432643)

[4.2.1. Profile 19](#_Toc88432644)

[4.2.2. Book Search 21](#_Toc88432645)

[4.2.3. Read Book 23](#_Toc88432646)

[4.2.4. Book Translation 25](#_Toc88432647)

[4.2.5. Book Memo 28](#_Toc88432648)

[4.2.6. Meeting with friends 30](#_Toc88432649)

[5. Protocol Design 31](#_Toc88432650)

[5.1. Objectives 31](#_Toc88432651)

[5.2. JSON 31](#_Toc88432652)

[5.3. Authentication 32](#_Toc88432653)

[5.3.1. Register 32](#_Toc88432654)

[5.3.2. Log In 32](#_Toc88432655)

[5.4. Profile 33](#_Toc88432656)

[5.4.1. Set Profile 33](#_Toc88432657)

[5.4.2. Get Profile 33](#_Toc88432658)

[5.5. Book 33](#_Toc88432659)

[5.5.1. Search Book 33](#_Toc88432660)

[5.5.2. Read Book 34](#_Toc88432661)

[5.5.3. Book translation 34](#_Toc88432662)

[5.5.4. Book Memo 35](#_Toc88432663)

[6. Database Design 35](#_Toc88432664)

[6.1. Objectives 35](#_Toc88432665)

[6.2. ER Diagram 36](#_Toc88432666)

[6.2.1. Entities 37](#_Toc88432667)

[6.2.2. User 37](#_Toc88432668)

[6.2.3. Book 38](#_Toc88432669)

[6.3. Relational Schema 39](#_Toc88432670)

[6.4. SQL DDL 39](#_Toc88432671)

[6.4.1. User 39](#_Toc88432672)

[6.4.2. Book 40](#_Toc88432673)

[7. Testing Plan 40](#_Toc88432674)

[7.1. Objectives 40](#_Toc88432675)

[7.2. Testing Policy 41](#_Toc88432676)

[7.2.1. Development Testing 41](#_Toc88432677)

[7.2.2. Release Testing 42](#_Toc88432678)

[7.2.3. User Testing 43](#_Toc88432679)

[7.2.4. Testing Case 44](#_Toc88432680)

[8. Development Plan 44](#_Toc88432681)

[8.1. Objectives 44](#_Toc88432682)

[8.2. Frontend Environment 45](#_Toc88432683)

[8.2.1. Unity 45](#_Toc88432684)

[8.2.2. VRChat 45](#_Toc88432685)

[8.3. Backend Environment 46](#_Toc88432686)

[8.3.1. Github 46](#_Toc88432687)

[8.3.2. Unity 46](#_Toc88432688)

[8.4. Constraints 47](#_Toc88432689)

[8.5. Assumptions and Dependencies 48](#_Toc88432690)

[9. Supporting Information 48](#_Toc88432691)

[9.1. Software Design Specification 48](#_Toc88432692)

[9.2. Document History 49](#_Toc88432693)

List of Figures

[Figure 1 Context Diagram 16](#_Toc88432512)

[Figure 2 Sequence Diagram 17](#_Toc88432513)

[Figure 3 Use Case Diagram 18](#_Toc88432514)

[Figure 4 Profile - Class Diagram 20](#_Toc88432515)

[Figure 5 Profile – Sequence Diagram 21](#_Toc88432516)

[Figure 6 Book Search – Class Diagram 22](#_Toc88432517)

[Figure 7 Book Search – Sequence Diagram 23](#_Toc88432518)

[Figure 8 Read Book – Class Diagram 24](#_Toc88432519)

[Figure 9 Read Book – Sequence Diagram 25](#_Toc88432520)

[Figure 10 Book Translation – class Diagram 27](#_Toc88432521)

[Figure 11 Book Translation – Sequence Diagram 28](#_Toc88432522)

[Figure 12 Book Memo – Class Diagram 29](#_Toc88432523)

[Figure 13 Book Memo – Sequence Diagram 29](#_Toc88432524)

[Figure 14 Meeting – Class Diagram 30](#_Toc88432525)

[Figure 15 Meeting – Sequence Diagram 31](#_Toc88432526)

[Figure 16 ER Diagram 36](#_Toc88432527)

[Figure 17 ER Diagram - user 37](#_Toc88432528)

[Figure 18 ER Diagram - book 38](#_Toc88432529)

[Figure 19 Relational Schema 39](#_Toc88432530)

[Figure 20 DDL - user 39](#_Toc88432531)

[Figure 21 DDL - book 40](#_Toc88432532)

[Figure 22 Software Release Life Cycle 43](#_Toc88432533)

[Figure 23 Unity 45](#_Toc88432534)

[Figure 24 VRChat 45](#_Toc88432535)

[Figure 25 Github 46](#_Toc88432536)

[Figure 26 Unity 46](#_Toc88432537)

List of Tables

[Table 1 Register Authentication 32](#_Toc88432703)

[Table 2 Log In - Authentication 33](#_Toc88432704)

[Table 3 Set Profile 33](#_Toc88432705)

[Table 4 Get Profile 33](#_Toc88432706)

[Table 5 Search Book 34](#_Toc88432707)

[Table 6 Read Book 34](#_Toc88432708)

[Table 7 Book Translation 35](#_Toc88432709)

[Table 8 Book Memo 35](#_Toc88432710)

[Table 9 Document History 49](#_Toc88432711)

# Preface

## Readership

This document is about Group 11’s Software Design Specification (SDS) that is explained in detail and organized into 10 sections with various subsections. The structure of this document can be referred to below, in the Document Structure subsection of this SDS. For this document, Group 11, professor, TAs, and members of the class Introduction to Software Engineering are the main readers.

## Scope

This describes the software architecture and software design decisions for the implementation of Metaverse Library application.

## Objective

The aim of the software design specification document is to document and track the information needed to properly describe architecture and system design, as well as to provide guidance to the development team on the architecture of the system being created. During the project's planning phase, this document is created. Some parts of this document, such as the user interface (UI), may be shared with the client/user and other stakeholders who need to give input or approval on the UI.

## Document Structure

1.4.1. Preface

The description of the readership, scope, objective and structure of this document.

1.4.2. Introduction

Various diagrams and expression tools used to describe this document are described, and the scope of the system covered by this software project is described.

1.4.3. System Architecture

Definition of the technical terms used throughout the document and descriptions in what context the terms are used in. Since this document includes not only system developers but also readers such as users and stakeholders, the terms will be described in as much detail as possible.

1.4.4. Protocol Design

The functional and non-functional requirements of the system are described from the user's point of view. Along with a brief schematic, describe the document in natural language to make it easier for users of the system to understand it.

1.4.5. Database Design

The structure of the system is outlined, and how the functions of the system are allocated and distributed in each subsystem is described.

1.4.6. Testing Plan

Based on the requirements briefly described in User Requirements Definition, functional and non-functional requirements, and other requirements are described in detail. This section is used throughout the development process, including the system design stage and the implementation stage, so it is systematically expressed using diagrams.

1.4.7. Development Plan

The relationship between each component of the system and the relationship with the external environment surrounding the system are expressed in diagrams.

1.4.8. Development Plan

The description of which tools to use to develop the system, constraints, assumptions and dependencies for developing this system.

1.4.9. Index

Indices such as pictures, tables, and diagrams used in this document are described.

# Introduction

The project is to develop a Metaverse Library System that allows students of SKKU

to use and experience the facilities that libraries provide virtually as a way to overcome limitations due to compulsory social distancing from the COVID-19 pandemic whilst saving architectural cost simultaneously. This design document will provide details of the designs that are used and applied on the system for this project.

## Objectives

In this section of the Software Design Specification, we aim to provide and elaborate in detail the tools and diagrams that have been used for this project.

## Applied Diagrams

### UML

UML stands for Unified Modeling Language and it is a standardized modeling language that consists of a set of diagrams that work together. This is a very essential part of the software as it helps the software developers to assist with defining, visualizing, building, and documenting software system artifacts. The UML diagrams are designed for business customers, developers, ordinary people, or anyone else who wants to learn about a system, whether it's software or non-software. The Unified Modeling Language (UML) has been utilized in the following domains; Information systems for businesses, financial services and banking, defense, transportation and many others. We can better comprehend potential weaknesses or problems in the software by utilizing these visual representations from the UML diagrams.

### Use case Diagram

Use case diagrams are also known as behavior diagrams, and they are used to represent a collection of activities or use cases that a system or systems should do in a partnership with external factors. This is usually used for an underdeveloped software as it effectively communicates the system behavior in the user's terms that can help design the system based on the user’s perspective.

### Sequence Diagram

A sequence diagram is a form of interaction diagram that shows how a group of items interacts and its chronological order. Software engineers and business experts use these diagrams to understand the requirements for a new system or to describe an existing process. The sequence diagram helps with creating a visual representation of how messages and tasks are passed between objects or components in a system. Not only that, it is also used to comprehend the intricacies of current or future systems.

### Class Diagram

A class diagram is a static structural diagram that depicts the structure of a system by displaying the system's classes, attributes, actions and relationships among objects. The class diagram helps with displaying the system's static structure of classifiers, and it also serves as a starting point for other UML-required structure diagrams. Using a class diagram is beneficial for software developers and can also be used by business analysts to model systems from a business standpoint.

### Context Diagram

The Context Diagram is a common tool among Business Analysts who use it to comprehend the specifics and limitations of a project's system to be created. It highlights the information flow between the system and its external components. A context diagram is included in a project's requirements document. Unlike other project diagrams, the Context Diagram is geared for project stakeholders instead of engineers/technicians. As a result, it should be written in clear, easy-to-understand language so that stakeholders may grasp the items when they study it. A Context Diagram is also known as a Context-Level Data Flow Diagram or a Level-0 Data Flow Diagram.

### Entity Relationship Diagram

Entity Relationship Diagram is a software engineering data modeling method for creating a conceptual data model of an information system. During the process of creating the ERD, the database analyst obtains a deeper grasp of the data that will be stored in the database. The ERD is used to link the database's logical structure to users. The ERD, in particular, efficiently communicates the database's logic to users.

## Applied Tools

### Microsoft PowerPoint

Microsoft Powerpoint is a software that is practical when it comes to creating diagrams and figures to properly depict the structure of the system and the relationships. This is very comfortable to use as it also has collaborative features that makes it easier to work with other members or partners because it can be used virtually everywhere. Not only that, Microsoft Powerpoint has existing templates that users can easily choose from and efficient tools to help edit them.

### ERD Plus

ERDPlus is a web-based database modeling tool that allows you to construct databases quickly and efficiently. It helps easily create; Entity Relationship Diagrams (ERDs), Relational Schemas (Relational Diagrams) and Star Schemas (Dimensional Models). More additional features include; automatically converting ER Diagrams into Relational Schemas, exporting SQL, exporting diagrams as a PNG and saving diagrams safely on the server.

## Project scope

Metaverse Library system provides a library system in a virtual environment to students who cannot enjoy school life in COVID-19. This system provides virtual library facilities and provides not only reading books but also the functions to translate into other languages or take notes on interesting phrases. In addition, it provides users with an environment where they can share opinions and communicate with various people.

## References

- Team 1. “Software Design Specification”. SKKU, Last Modified: May. 24, 2020. Accessed: Oct 24 2021, <https://github.com/skkuse/2020spring_41class_team1/blob/master/docs/SRS_TEAM1.pdf>

- VRCPrefabs, “VRCPrefabs”, 2021, Accessed: Oct 27 2021,<https://vrcprefabs.com/>

- iVRy, Steam Community, “How to use VRChat with a smartphone and Google Cardboard viewer”, Last Modified: Jan 14 2018, Accessed: Oct 27 2021,<https://steamcommunity.com/app/438100/discussions/0/2595630410184136626/>

# System Architecture - Overall

## Objectives

This section describes the overall system organization of the Metaverse Library System. The structure of the system is represented by context diagram, sequence diagram, and use case diagram.

## System Organization

### Context Diagram

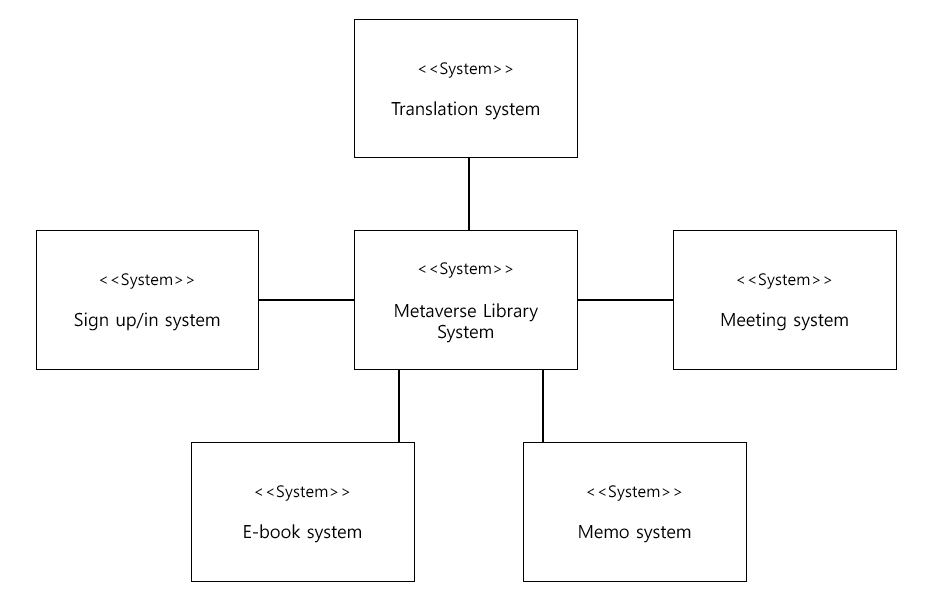


Figure 1 Context Diagram

### Sequence Diagram

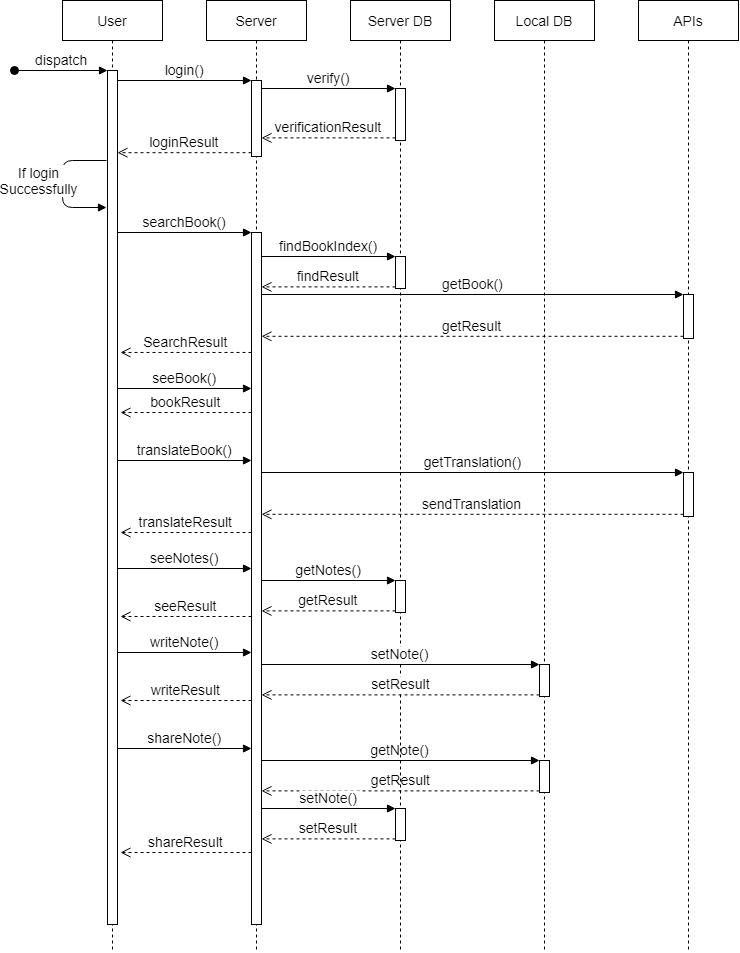


Figure 2 Sequence Diagram

### Use Case Diagram

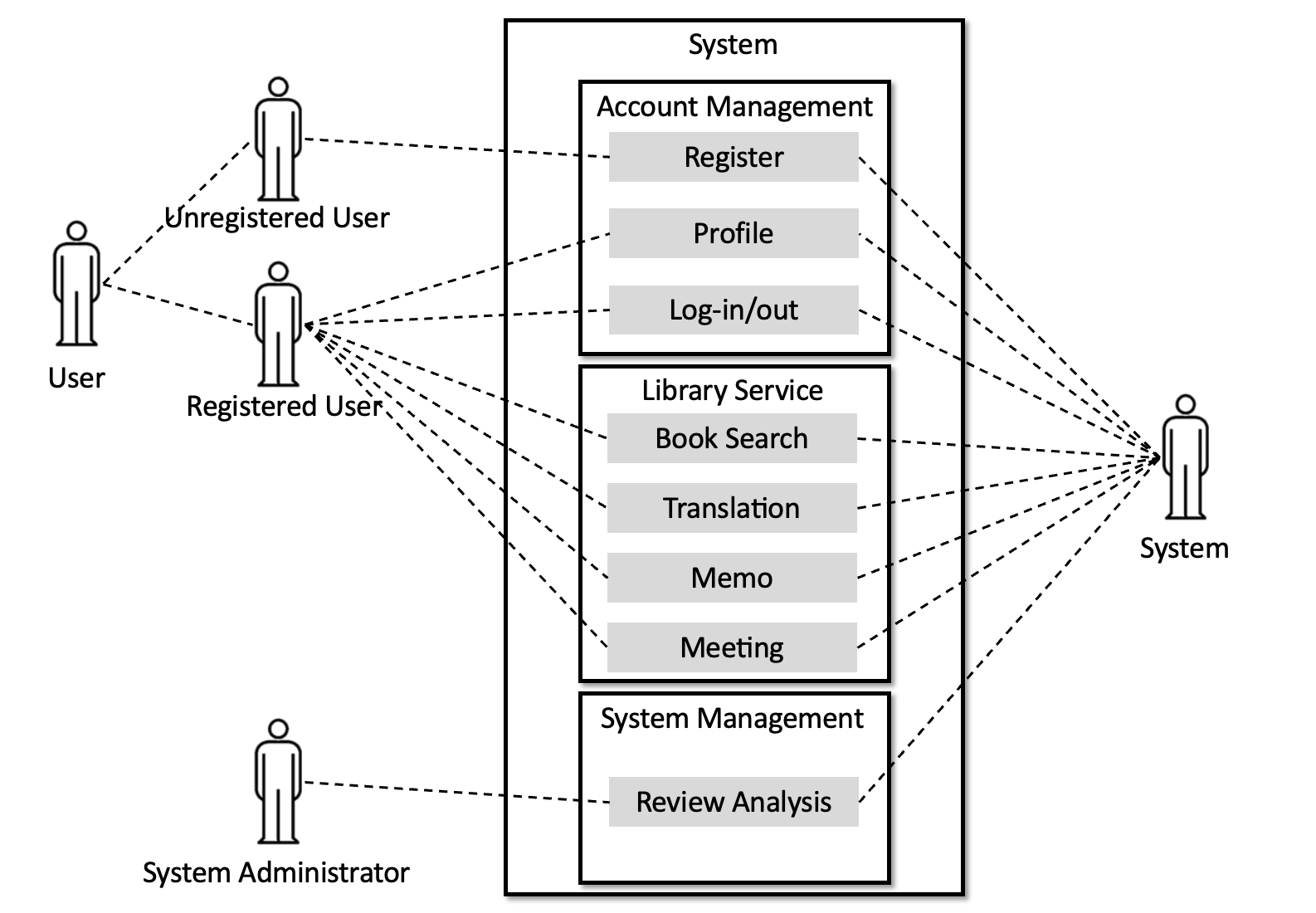


Figure 3 Use Case Diagram

# System Architecture

## Objectives

This section describes the structure of the frontend system and attributes, method of each component.

## Subcomponents

### Profile

The profile class is the class that deals with user's information. When a registered user logs in, the user can check and modify his or her profile.

#### Attributes

These are the attributes that profile class has,

* user\_id : ID of the user
* user\_name : name of the user
* age : age of the user
* gender : gender of the user

#### Methods

These are the methods that profile class has,

* getProfile()
* setProfile()

#### Class Diagram

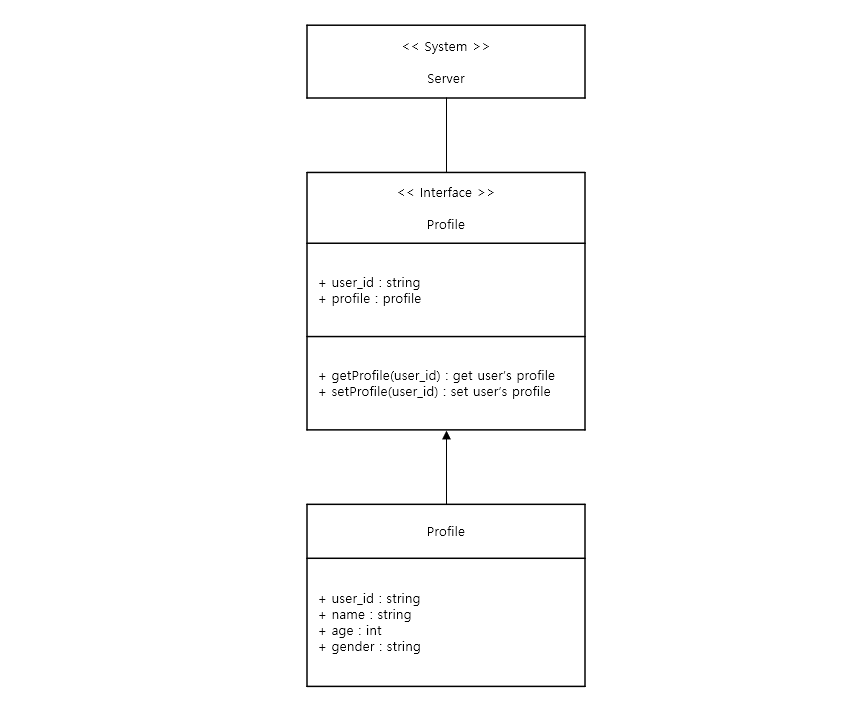


Figure 4 Profile - Class Diagram

#### Sequence Diagram

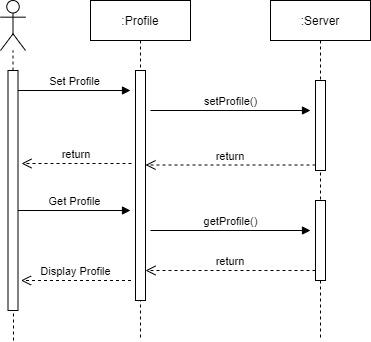


Figure 5 Profile – Sequence Diagram

### Book Search

#### Attribute

These are the attributes that book object has,

* book\_id : ID of book
* title : title of book
* author : author of book
* tag : category or tag name

#### Methods

These are the methods that Book Search class has,

* searchBook() : search book with some information that entered by user
* viewResult() : show the list of searched book

#### Class Diagram

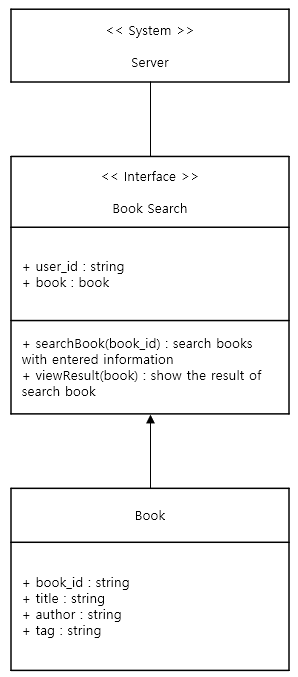


Figure 6 Book Search – Class Diagram

#### Sequence Diagram

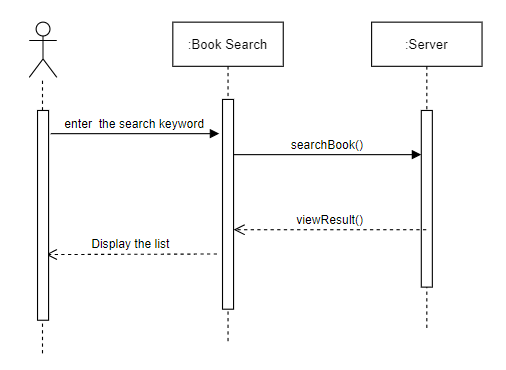


Figure 7 Book Search – Sequence Diagram

### Read Book

#### Attributes

These are the attributes that the book object has,

* book\_id : ID of book
* title : title of book
* author : author of book
* tag : category or tag name
* pages\_num : total number of pages
* page : each pages that reader read
* text : String that extracted from pages

#### Methods

These are the methods that Read Book class has,

* seeBook() : find the book contents which user requested
* bookResult() : show the book contents
* seeNextpage() : move to the next page
* seePreviouspage() : move to the previous page
* closeBook() : close the book when user finished reading

#### Class Diagram

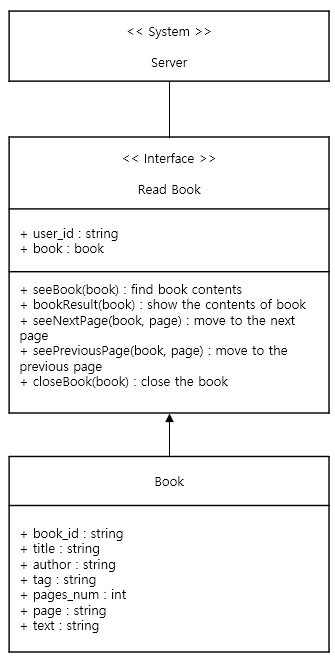


Figure 8 Read Book – Class Diagram

#### Sequence Diagram

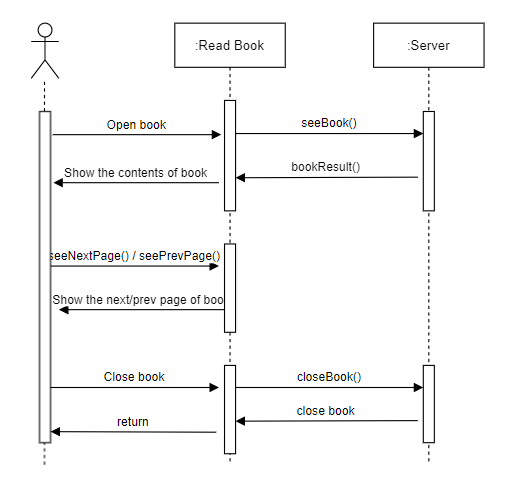


Figure 9 Read Book – Sequence Diagram

### Book Translation

#### Attributes

These are the attributes that book translation class has,

target\_file : Target pdf file that will be translated

reader : pdf reader that already created

pages\_num : total number of pages

page : each pages that reader read

text : String that extracted from pages

translator : translator made with using google api

#### Methods

These are the methods that book translation class has,

\_openPdf() : open the target file

\_translate() : translate the text that extracted by reader

\_writePdf() : create a new Pdf file which is translated

#### Class Diagram

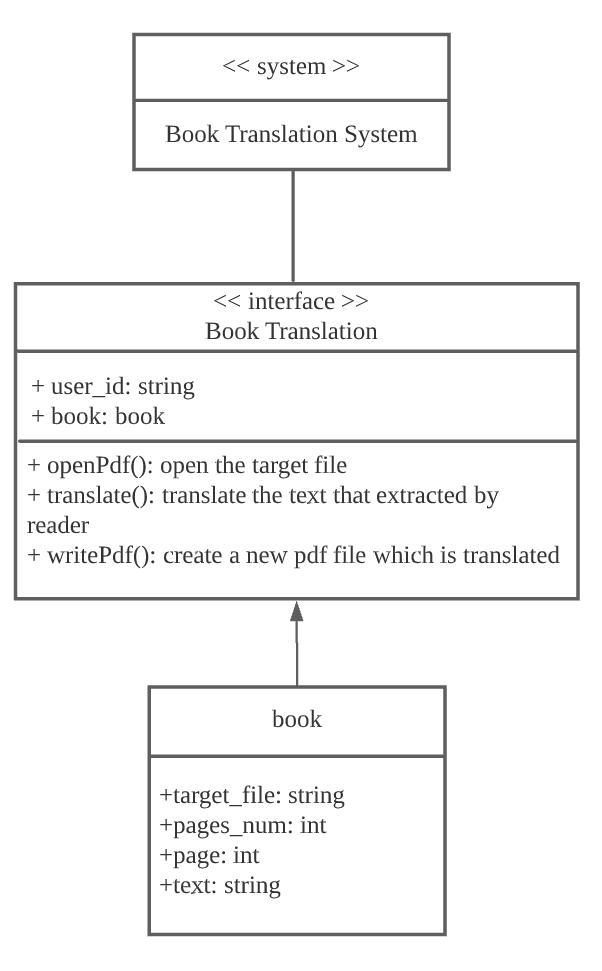


Figure 10 Book Translation – class Diagram

#### Sequence Diagram

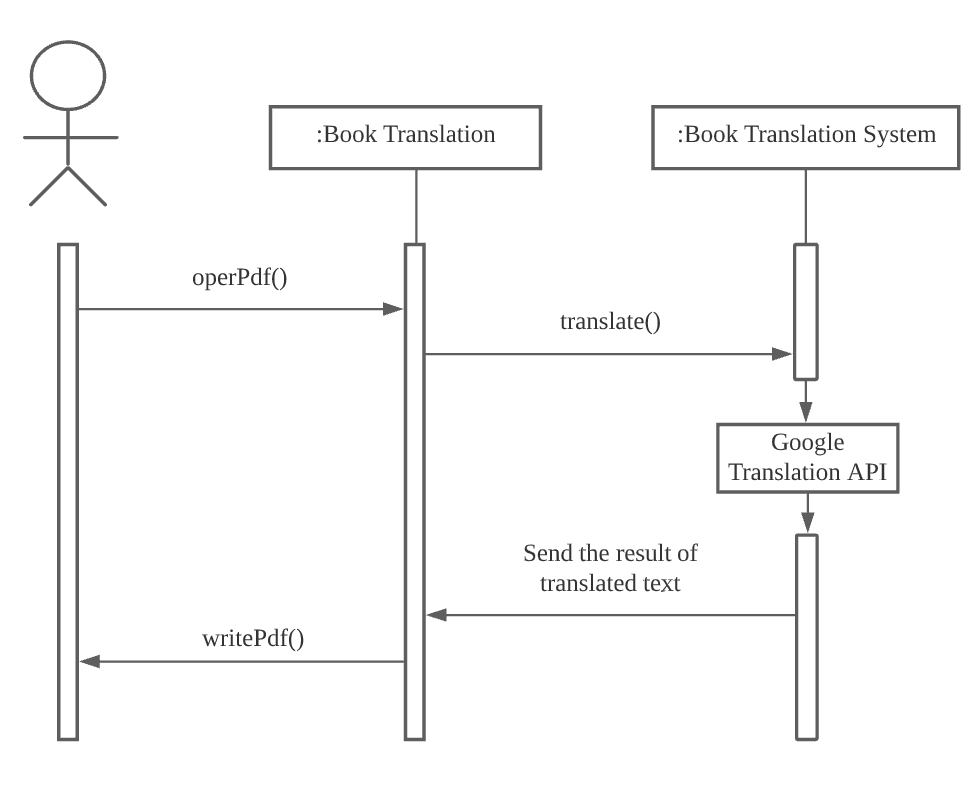


Figure 11 Book Translation – Sequence Diagram

### Book Memo

#### Attributes

These are the attributes that the Book Memo class has,

* target\_file: target book or PDF file
* page: the page where the user is currently writing

#### Methods

These are the methods that the Book Memo class has,

* \_openPDF(): open the target file
* \_writeMemo(): write on the target page of the book

#### Class Diagram

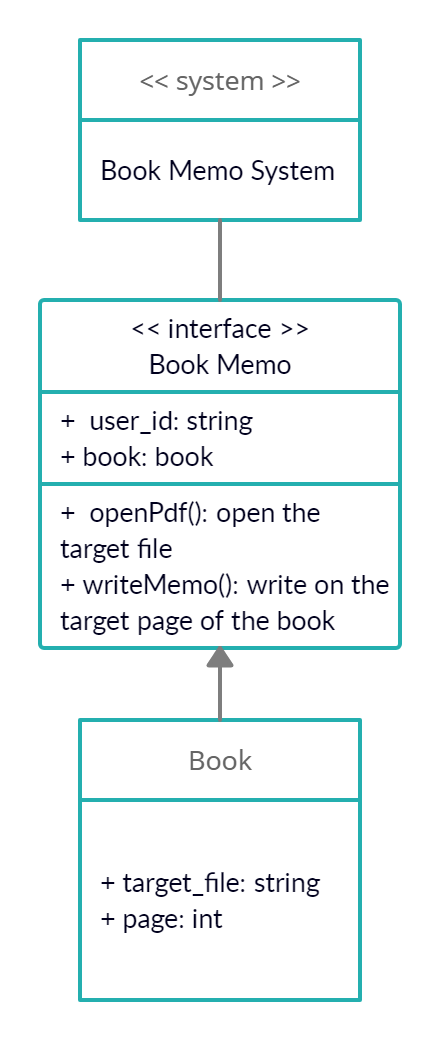


Figure 12 Book Memo – Class Diagram

#### Sequence Diagram

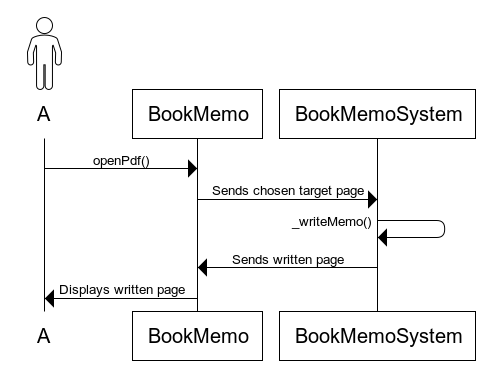


Figure 13 Book Memo – Sequence Diagram

### Meeting with friends

#### Attributes

These are the attributes that the Book Memo class has,

* user\_id: personal user id and user id of friends

#### Methods

These are the methods that the Book Memo class has,

* findUserID(): find the user ID
* addFriends(): adding friends feature

#### Class Diagram

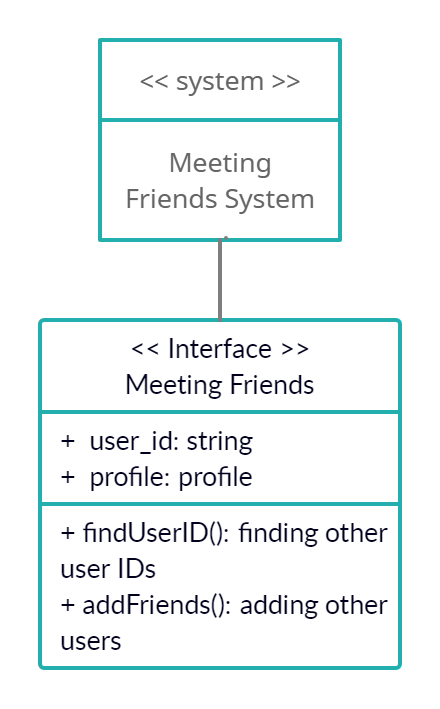


Figure 14 Meeting – Class Diagram

#### Sequence Diagram

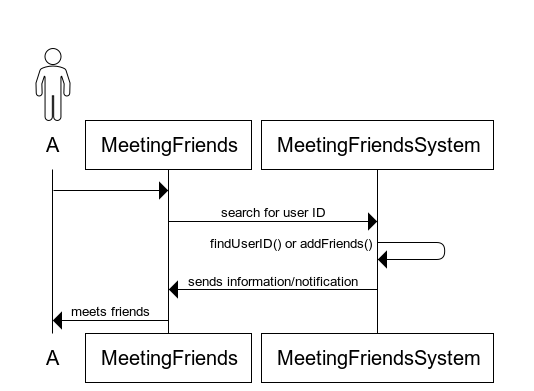


Figure 15 Meeting – Sequence Diagram

# Protocol Design

## Objectives

This section describes what structures are used for the protocols which are used for interaction between each subsystem. Also, this part describes how each interface is defined.

## JSON

JavaScript Object Notation (JSON) is a human-readable text format to convey data objects that consist of attribute-value pairs and array data types (or any other serializable value) or "key-value pairs". It is an open standard format used. For asynchronous browser/server communication (AJAX), it is broadly the primary data format replacing XML (used by AJAX). We chose JSON over XML for protocol communication. When using Google Translate API, use json format. For example, as a request, set the sentence to be translated in "q", the language to be translated in "source", target language to translate is in "target", and the format to be returned in "format". This will return a corresponding JSON response.

## Authentication

### Register

|  |  |
| --- | --- |
| **Attributes** | **Details** |
| Method | POST |
| HTTP Method | /authentication/signup |
| JSON Request Body | user id, name, age, gender |
| JSON Response | success/fail to sign up message |

Table 1 Register Authentication

### Log In

|  |  |
| --- | --- |
| **Attributes** | **Details** |
| Method | POST |
| HTTP Method | /authentication/login |
| JSON Request Body | user id, password |
| JSON Response | success/fail to log in message |

Table 2 Log In - Authentication

## Profile

### Set Profile

|  |  |
| --- | --- |
| **Attributes** | **Details** |
| Method | POST |
| URI | /user/:id/profile |
| JSON Request Body | user id and informations |
| JSON Response | success/fail message |

Table 3 Set Profile

### Get Profile

|  |  |
| --- | --- |
| **Attributes** | **Details** |
| Method | POST |
| HTTP Method | /user/:id |
| JSON Request Body | user id |
| JSON Response | success/fail message |

Table 4 Get Profile

## Book

### Search Book

|  |  |
| --- | --- |
| **Attributes** | **Details** |
| Method | Get |
| HTTP Method | https://www.googleapis.com/books/v1/volumes?q=isbn=9788950984526 |
| JSON Request Body | book id, title, author |
| JSON Response | list of searched book |

Table 5 Search Book

### Read Book

|  |  |
| --- | --- |
| **Attributes** | **Details** |
| Method | Request |
| HTTP Method | https://www.googleapis.com/books/v1/volumes?q=isbn=9788950984526 |
| JSON Request Body | book\_id, title, author |
| JSON Response | book contents |

Table 6 Read Book

### Book translation

|  |  |
| --- | --- |
| **Attributes** | **Details** |
| Method | Request |
| Http method | https://translation.googleapis.com/language/translate/v2 |
| Json request body | q, source, target, format |
| Json response | translations |

Table 7 Book Translation

### Book Memo

|  |  |
| --- | --- |
| **Attributes** | **Details** |
| Method | PATCH |
| Http method | /openfile/page/write |
| Json request body | file, pdf, page |
| Json response | write memo |

Table 8 Book Memo

# Database Design

## Objectives

This section describes the system data structures and how these are to be represented in a database. Entities and their relationships are identified through ER-diagram (Entity Relationship diagram). And then, it generates Relational Schema and SQL DDL(Data Description Language) specification.

## ER Diagram

The metaverse library system contains two entities: User and Book. Each entity is represented as rectangles. If the entity had two or more relationships with other entities, the end was connected by a line that splits into three(shape of trident). If the entity had only one relationship with other entities, the end of line marked in a cross shape. Attribute is represented as an oval shape. The only key attribute that distinguishes each entity is underlined. Multivated attributes, in which several attribute values may exist, are expressed as double ellipses. The relationship between each entity is expressed as diamonds.

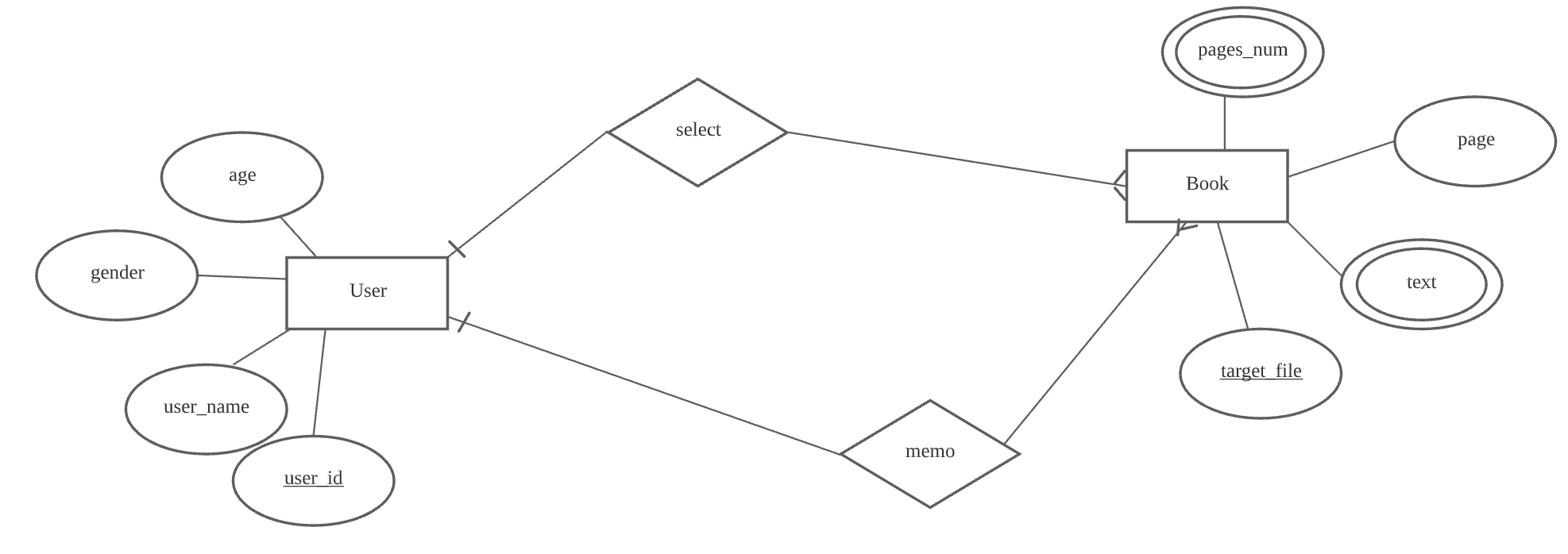


Figure 16 ER Diagram

### Entities

### User

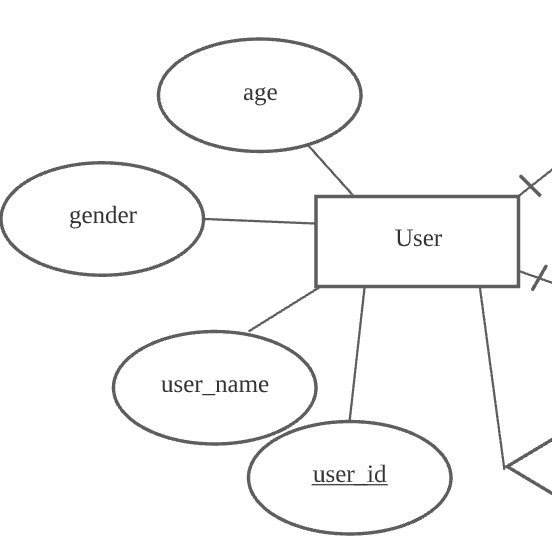


Figure 17 ER Diagram - user

User entity consists of age, gender, user\_name, and user\_id. User\_id is the key attribute of a user entity which distinguishes other user entities. A user entity has one-to-many relationships with book entities.

### Book

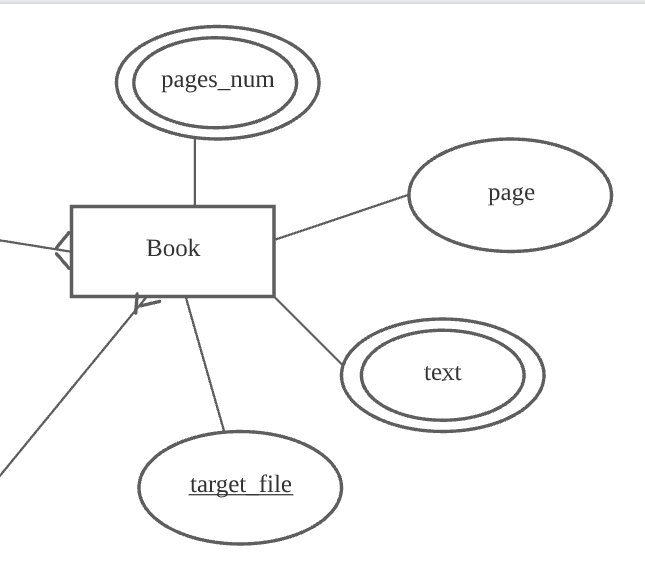


Figure 18 ER Diagram - book

A book entity consists of page\_num, page, text and target\_file attributes. Target\_file attribute is the key attribute of a book entity. Pages\_num and target\_file are multivalued attributes. Book entity has a many-to-one relationship with a user entity.

## Relational Schema

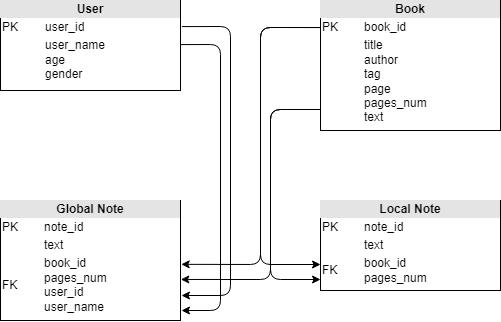


Figure 19 Relational Schema

## SQL DDL

### User

CREATE TABLE User

(

user\_id STR NOT NULL,

user\_name STR NOT NULL,

age INT NOT NULL,

gender INT NOT NULL,

PRIMARY KEY(user\_id)

)

Figure 20 DDL - user

### Book

CREATE TABLE Book

(

book\_id STR NOT NULL,

title STR NOT NULL,

author STR NOT NULL,

tag STR NOT NULL,

pages\_num INT NOT NULL,

page INT NOT NULL

text STR NOT NULL

PRIMARY KEY(book\_id)

)

Figure 21 DDL - book

# Testing Plan

## Objectives

This section describes the test plan of this system. There are three tests described here. Development testing, release testing, user testing. These tests can detect various defects that may occur in actual operation in advance.

## Testing Policy

### Development Testing

It is a means of regularly implementing testing practices across the software development life cycle. This testing guarantees the timely detection of defects or errors, as well as the avoidance of any risk in terms of time and money. In this section we will focus on three essential ways, included in development testing which are; performance, reliability and security.

#### Performance

Testing the performance is one method of determining how a system performs in terms of responsiveness and stability when subjected to a specific workload. Other quality features of the system, such as scalability, dependability, and resource utilization, can also be investigated, measured, validated, or verified using it. The benefits of testing for performance are; it helps validate the basic features of the software, helps identify problems in the system and maintain the speed, accuracy and stability of the software itself.

#### Reliability

Testing for reliability is a critical approach used by the team to ensure that the program performs and functions consistently in a variety of environments and over a set length of time. It assures that the product is free of flaws and suitable for its intended use and keeps customers happy. The aim of testing for reliability can be summarized into; to determine the number of failures that occur in a certain period of time, identifying the cause of failure and find out how to break down the framework of recurring failures.

#### Security

Security testing is very important as it is a sort of software testing that identifies system vulnerabilities and ensures that the system's data and resources are safe from potential invaders. It assures that the software system and application are free of dangers or risks that could result in a data loss.

### Release Testing

The most important thing in software development projects is to release products to markets and consumers. The release testing is essential to prevent bugs that may occur in this process and reduce great damage. It is a process of checking whether there are any defects in advance or whether it works as intended in the release of a new version. Therefore, It should be carried out before release.

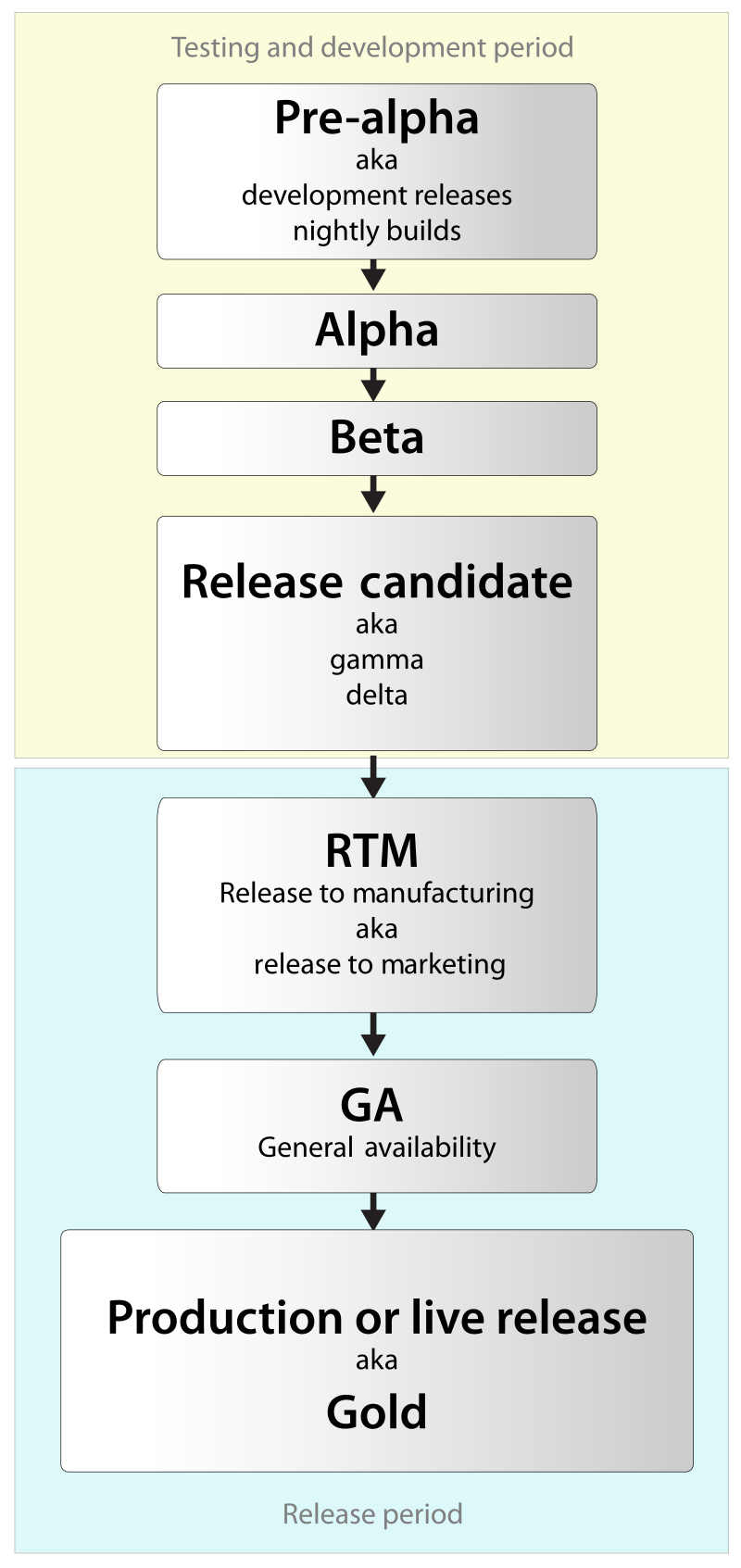


Figure 22 Software Release Life Cycle

Based on Software Release Life Cycle, we should test the software by alpha version first by developer and then release beta version to proceed final checking. By beta version testing, we can get feedback from actual users as well as developers.

### User Testing

We will test our system to users in advance for possible scenarios and various possible situations. Before deployment, we will distribute preVersion to approximately 15 users and collect their reviews. Through the data, a partial defect can be found in the system, and if it is found, the importance of the defect should be evaluated and corrected.

### Testing Case

Testing case is to be performed on a system to determine if the metaverse library project satisfies software requirements and functions correctly. Its purpose is to determine whether or not various features of a system are performing as expected and to confirm that the system satisfies all related standards, guidelines, and customer requirements. The process of writing a test case can also aid in the discovery of errors or defects in the system. Test case is to be done by checking the quality assurance and can be used as step-by-step instructions for each system test.

# Development Plan

## Objectives

This section illustrates the technologies and environment for the development of the system.

## Frontend Environment

### Unity

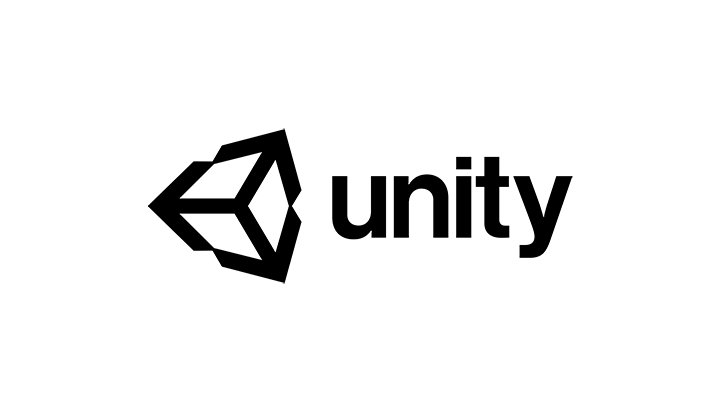


Figure 23 Unity

Basic world design is done through Unity. The world looks like a simple library, and interacts with the book to open a pdf file and read it. When designing worlds, use several SDKs from the unity asset store. A typical example is Probuilder.

### VRChat



Figure 24 VRChat

Conversation, which is the most important part of the metaverse, uses the functions that vrChat provides by default. The chat function of vrchat allows users to have a conversation about a topic related to a book. In addition, the emoji feature in vrchat will help convey the user's emotions more effectively.

## Backend Environment

### Github



Figure 25 Github

GitHub is a code hosting platform for version control and collaboration. It lets us as a team work together on projects from anywhere. The code that is done or modified for this metaverse library project will be committed through this platform.

### Unity

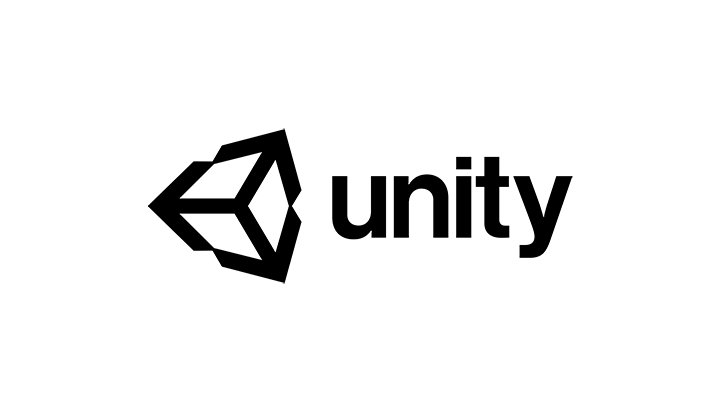


Figure 26 Unity

Unity also acts as a backend environment for the development of this metaverse library project. In Unity’s Integrated Development Environment (IDE), not only it can run the game and show graphical previews, it also serves as a platform for developers to write code, which is a purely linguistic / logical / sequential universe and also direct the assets from code, in a way that is practical, flexible, and as safe as possible.

## Constraints

In developing the Metaverse library system, the following matters should be considered. Constraints are considered in a number of ways, including policy aspects, equipment aspects, and market research aspects.

● Cost and reliability must be considered in using the Open source API

● Copyright aspects should be considered in the use of e-book APIs

● The cost of system maintenance must be considered

● The service should be provided only to users who have access to Sungkyunkwan University's library service

● Natural 3D avatar modeling that is less heterogeneous is needed

● It should be compatible with the Windows 10 operating system, the basic operating system of library computers

● The maximum number of concurrent users of the metaverse system should be considered

● Detailed specification work is required to minimize the cost of system improvement

● It should be easy for users to use and be able to correct their own wrong choices

## Assumptions and Dependencies

All systems in this document are written on the assumption that they are designed and implemented based on Vrchat. Therefore, all contents are written based on the Vrchat with Unity 2019.4.30f1 and may not be applied to other operating systems or versions.

# Supporting Information

## Software Design Specification

This software requirements specification was written in accordance with the IEEE Recommendation (IEEE Recommended Practice for Software Requirements Specifications, IEEE-Std-830).

## Document History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Writer** |
| 2021/11/12 | 0.1 | Style and overview | Youngsuh Chin |
| 2021/11/15 | 1.0 | Addition of Preface (1.1,1.2,1.3,1.4) | Amirah |
| 2021/11/15 | 1.1 | Addition of 2. Introduction, 2.1, 2.2, 2.3 | Angel Fitri Sari |
| 2021/11/19 | 1.2 | Addition of 2.4, 3.1, 4.1, 4.2.1 | Dajung Choi |
| 2021/11/21 | 1.3 | Addition of 3.2, 10.1, 9.1, 9.4, 9.5, 7.1 | Youngsuh Chin |
| 2021/11/21 | 1.4 | Addition 4.2.4 6.2 9.2 8.1 8.2.3 | Wonjae Lee |
| 2021/11/21 | 1.5 | Addition of 9.3.1, 9.3.2, 6.5.4, 8.2.4 | Amirah |
| 2021/11/21 | 1.6 | Addition of 4.2.5, 4.2.6, 6.3, 8.2.1 | Angel Fitri Sari |
| 2021/11/21 | 1.7 | Addition of 7.3 | Changmin Park |
| 2021/11/21 | 1.8 | Addition of 4.2.4.3, 4.2.4.4, 7.2, 7.2.2, 7.2.3 | Jaehyuk Choi |
| 2021/11/21 | 1.9 | Addition of 2.5, 4.2.3.1, 4.2.3.2, 6.1, 7.4, 8.2.2, 6.5 | Youngsuh Chin |
| 2021/11/21 | 1.10 | Addition of 4.2.1.3, 4.2.2, 4.2.3.3, 4.2.3.4, 6.4, 6.5.1 | Dajung Choi |

Table 9 Document History