Hall for Graduation Exhibition

Software Design Specification

2021.11. 21

Team 9

Jihee Kim

Bomin Namkoong

Jaehyeong Park

Jongin Park

Jaekwang Shin

Seokjun Chung

Gyumin Han

Contents

LIST OF TABLES

LIST OF FIGURES

1. Preface

This chapter contains the readership, scope, objective of this document and document structure of this Software Design Document for providing Metaverse hall for graduation exhibition service.

1.1. Readership

This Software Design Document is divided into 8 sections with various subsections. The structure of the Software Design Document can be found as listed below, in the Document Structure subsection of this SDD. In this document, Team 9 is the main reader. Additionally, professors, TAs, and team members in the Introduction to Software Engineering class can be the main readers.

1.2. Scope

This Design Specification is to be used by Software Engineering and Software Quality Engineering as a definition of the design to be used to implement the metaverse hall for graduation exhibition service that is meant to ease the come and go to graduation exhibition regardless of time and space.

1.3. Objective

The primary purpose of this Software Design Document is to provide a description of the technical design aspects for our exhibition hall service. It also provides a structural overview of the system that can describe various aspects of the system. It also specifies the structure and design of some modules discussed in the SRS document and displays several use cases converted into sequential and activity diagrams, including a class diagram showing how to implement a particular module in Team 9. The target of this document can be found in Readership part.

1.4. Document Structure

- 1. Preface: this chapter describes readership, scope of this document, object of this system, and structure of this document.
- 2. Introduction: this chapter describes several tools used for this document, several diagrams used in this document and the references, and object of this project.
- 3. System Architecture – Overall: this chapter describes overall architecture of the system using context diagram, sequence diagram, and use case diagram.
- 4. System Architecture – Frontend: this chapter describes architecture of the frontend system using class diagram and sequence diagram.
- 5. Human interface design:
- 6. Testing Plan: this chapter describes our team`s testing plan.
- 7. Development Plan: this chapter describes which tools to use to develop the system, constraints, assumption, and dependencies for developing this system.
- 8. Supporting Information: this chapter describes baseline of this document and history of this document.

2. Introduction

This project is to develop and design a virtual world that can be used as a hall for graduation work exhibitions. Once the world is completed, it should be able to be connected with other worlds. Also, the system should be able to run on the metaverse platform, Unity engine with VRChat SDK 3. This design document presents a collection of designs and resources used that cover all details used in implementing the project. The document covered here is written based on the requirement specification previously prepared.

2.1. Objectives

The purpose of this chapter includes a brief description of the diagram to be presented and a description of the tool used to create a diagram.

2.2. Applied Diagrams

2.2.1. UML

UML diagrams model a system visually by representing the system along with its main actors, roles, actions, artifacts, or classes. UML means Unified Modeling Language, which is a language that defines a method of expressing using symbols and diagrams, not using programming languages. The diagrams model not only the software area but also the system area using object-oriented concepts. With the purpose of converting the conceptual model in a visual graphical form, it can be used in diverse application domains to better understand, alter, maintain, or document information about the system. Types of UML diagrams include structural diagrams and behavioral diagrams. There are several diagrams, but to introduce only the diagrams to be described later, those belonging to the structural diagram include class diagrams, and those belonging to the behavioral diagram include youth case diagrams and sequence diagrams.

2.2.2. Use-case Diagram

Use-case diagrams model the behavior of a system and help to capture the requirements of the system. They describe the high-level functions and scope of a system. These diagrams also identify the interactions between the system and its actors. The use cases and actors in use-case diagrams describe what the system does and how the actors use it, but not how the system operates internally. They illustrate and define the context and requirements of either an entire system or the important parts of the system. We can model a world with a single use-case diagram or create many use-case diagrams to model the components of the world. Use-case diagrams can be developed in the early phases of a project and can be referred throughout the development process.

2.2.3. Sequence Diagram

Sequence diagrams, commonly used by developers, model the interactions between objects in a single use case. They illustrate how the different parts of a system interact with each other to carry out a function, and the order in which the interactions occur when a particular use case is executed. In simpler words, a sequence diagram shows different parts of a system work in a ‘sequence’ to get something done. A sequence diagram is structured in a way that it represents a timeline which begins at the top and descends gradually to mark the sequence of interactions.

Each object has a column and the messages exchanged between them are represented by arrows.

2.2.4. Class Diagram

The class diagram is a graphical notation used to construct and visualize object- oriented systems. A class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, methods, and the relationship among objects. Class is a description of a group of objects with similar roles in the system, while relationship represents an association with other classes.

2.2.5. Context Diagram

The Context Diagram shows the system under consideration as a single high-level process and then shows the relationship that the system has with other external entities. Context Diagrams are a specialized version of Data-Flow Diagrams, which is a graphical visualization of the movement of data through an information system. DFDs are one of the three essential components of the structured-systems analysis and design method. Context Diagrams and Data-Flow Diagrams were created for systems analysis and design. But like many analysis tools they have been leveraged for other purposes. They can also be leveraged to capture and communicate the interactions and flow of data between business processes. So, they can be applied to various system development phases.

2.3. Applied Tools

2.3.1. Microsoft PowerPoint

PowerPoint is a computer software created by Microsoft, which allows users to create slides with images, tables, recordings and other features in order to present information. UML diagrams can be easily drawn with PowerPoint, since it offers various kinds of polygons, lines, curves, tables and etc. There is another way to draw a diagram using dedicated software, but PowerPoint has the advantage of being able to create a schematic more intuitively and quickly.

2.4. Project Scope

This world is designed to allow “non-face-to-face" exhibitions in the current situation where it is impossible to hold graduation work exhibitions “face-to-face" due to the COVID-19 pandemic. The system is based on the metaverse platform, Unity and VRChat SDK, and aims to help smooth progress by inserting and creating the world’s components necessary for the

exhibition. As a result, participants and users can appreciate submitted works or papers and look at related contents, just as participating in an actual exhibition. To sum up, our goal is to imitate a real-life grad work exhibition, providing students with a comfortable user experience.

2.5. References

The details of the contents in this software design document can be found in following references.

- Team 1, 2020 Fall, Software Design Document, SKKU.
- IEEE Std 1016-1998

3. System Architecture – Overall

3.1. Objectives

This chapter describes the overall composition of the world, largely from components of the world to attributes of each component in detail.

3.2. System Organization

3.2.1. Context Diagram

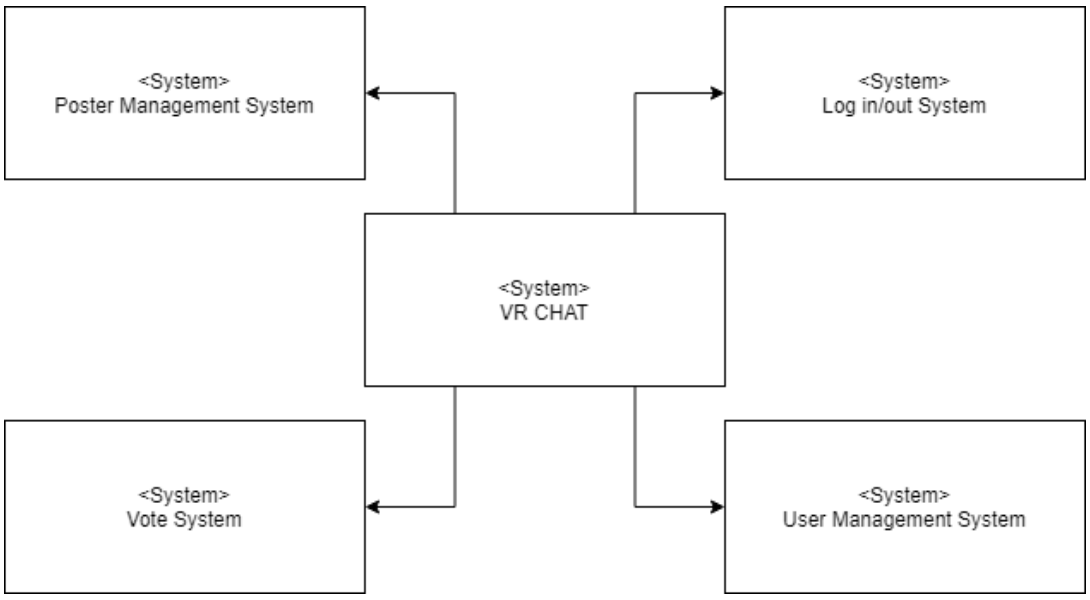

Figure 1. Context Diagram

3.2.2. Sequence Diagram

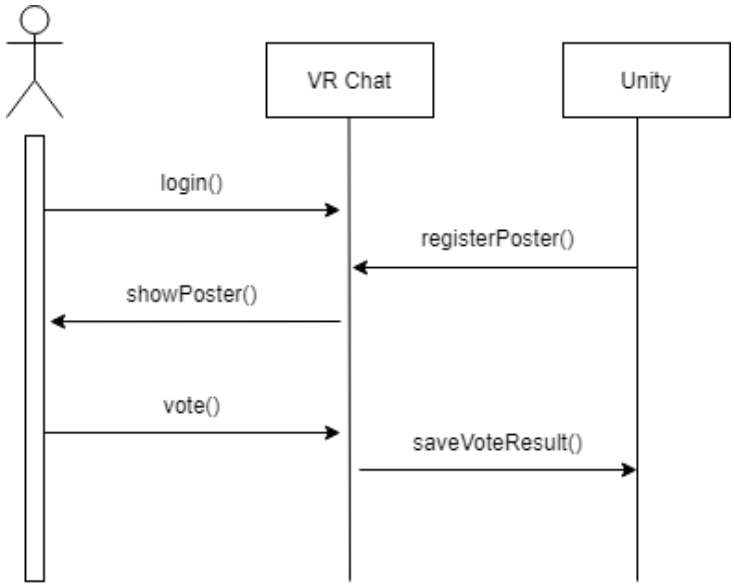

Figure 2. Sequence Diagram

3.2.3. Use-case Diagram

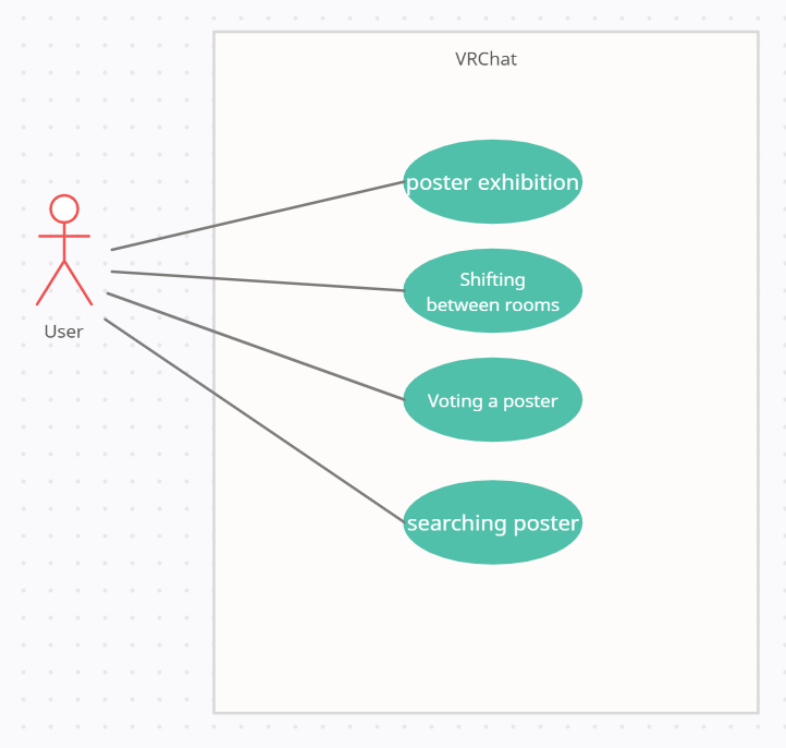

Figure 3. Use-case Diagram

4. System Architecture – Frontend

4.1. Objectives

This chapter describes the attributes and methods of the frontend components and describes the interface of each component.

4.2. Subcomponents

4.2.1. Move Processing

Input Processing class receive inputs from keyboards. If it receives a direction key or space key as an input, it changes the location of avatar. It is described because it is a component necessary for the system, although it does not have to be made essential because it is provided by VRchat.

4.2.1.1. Attributes

These are the attributes that input processing class has.

- Input: input from the keyboard
- Location: coordinates of the location where the avatar is located
- Velocity: the speed at which the avatar moves
- Gravity: the gravity value of where the avatar is standing

4.2.1.2. Methods

These are the methods that input processing class has.

- Identify ()
- Move ()
- Jump ()

4.2.1.3. Class Diagram

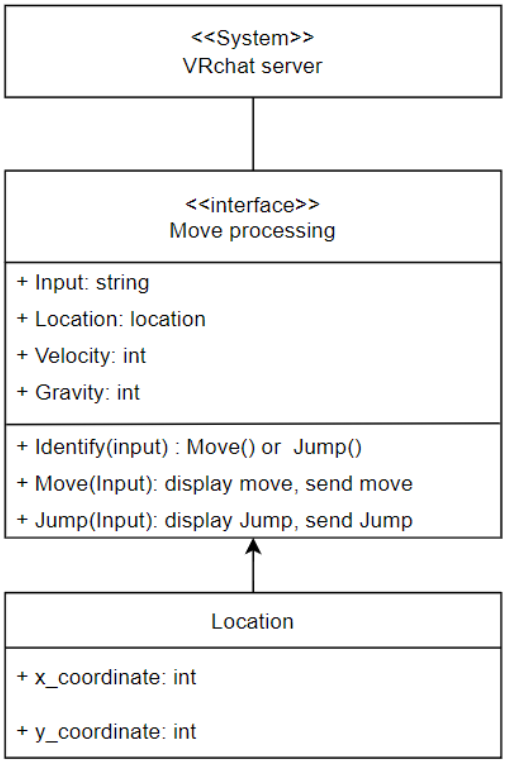

Figure 4. Class Diagram - Move Processing

4.2.1.4. Sequence Diagram

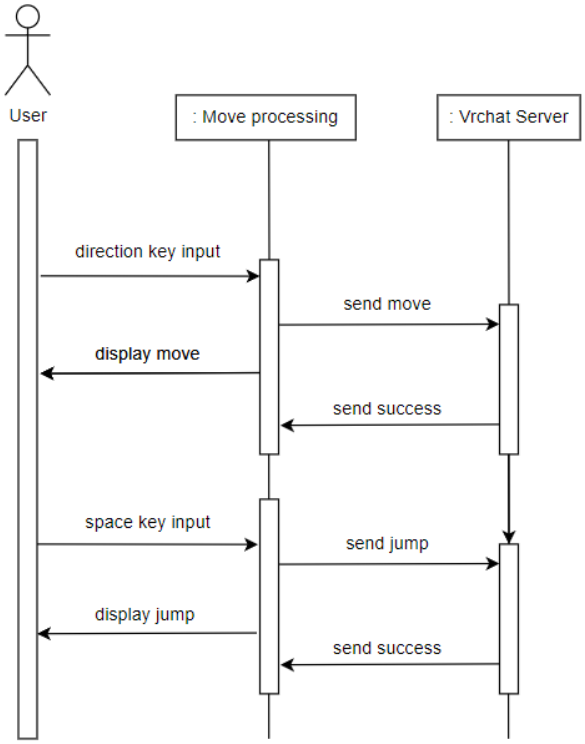

Figure 5. Sequence Diagram - Move Processing

4.2.2. Poster Exhibition

The poster exhibition class deals with files which users upload. Users can see the poster of the participant's graduation work and magnify the poster by clicking the poster.

4.2.2.1. Attributes

These are the attributes that poster exhibition object has.

- User: a name of a author
- User ID: a student ID of a author
- Subject: a subject of a graduation work/paper
- Work: a PDF file of a graduation work/paper
- Advisor: a name of an advisor
- Laboratory: a name of an advisor’s laboratory

4.2.2.2. Methods

By using VRC Book Builder, we can display a PDF file when we drag and drop text files into the prefab. There is no need for a method.

4.2.2.3. Class Diagram

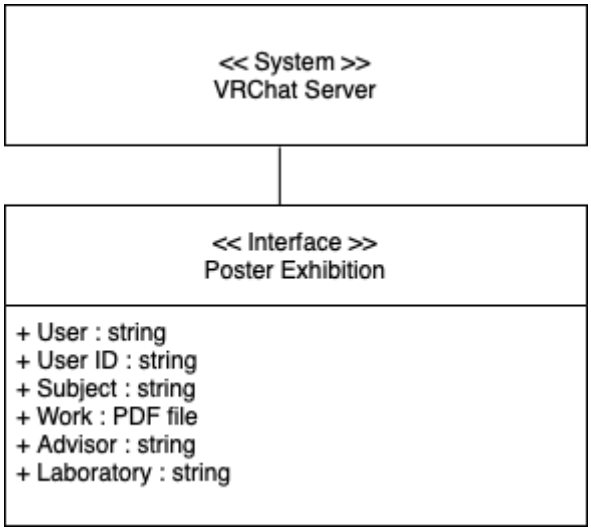

Figure 6. Class Diagram - Poster Exhibition

4.2.2.4. Sequence Diagram

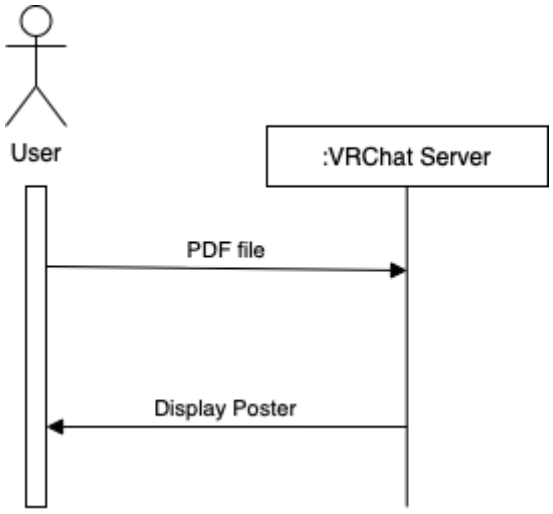

Figure 7. Sequence Diagram - Poster Exhibition

4.2.3. Playing Demo Videos

The playing demo video class deals with playing video for introduction graduation works. If user click the video, the video will be run.

4.2.3.1. Attributes

These are the attributes that video object has.

- UdonSyncPlayer
  - VRC Unity Video Player: if you want to play mp4, you have to write the file path to Video URL. Or you can write website video URL.
- Canvas: There is Input Field that player can write URL want to watch video.
- Audio Source: It use Audio Clip to play audio sound.
- Screen: It shows the video scene.

4.2.3.2. Methods

These are the methods that video class has.

- OnURLChanged() : When URL Field Changed, Become Owner and update synced URL
- OnVideoStart() : On Video Start, Update Offset on Owner, Resync on Others
- PlayUrl() : Play URL when it Changes
- OnOwnershipRequest() : Enable / Disable Guset Control.
- UpdateTimeAndOffset(): Update timeAndOffset variable, used to sync video time.

4.2.3.3. Class Diagram

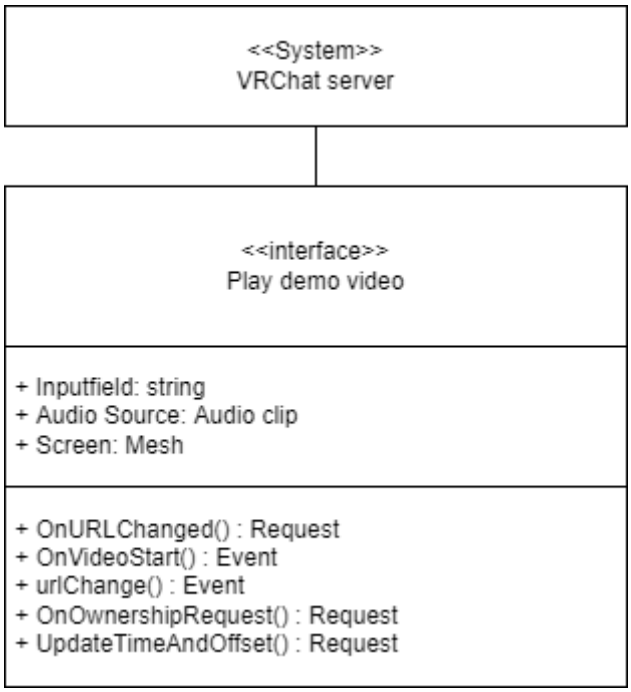

Figure 8. Class Diagram - Playing Demo Videos

4.2.3.4. Sequence Diagram

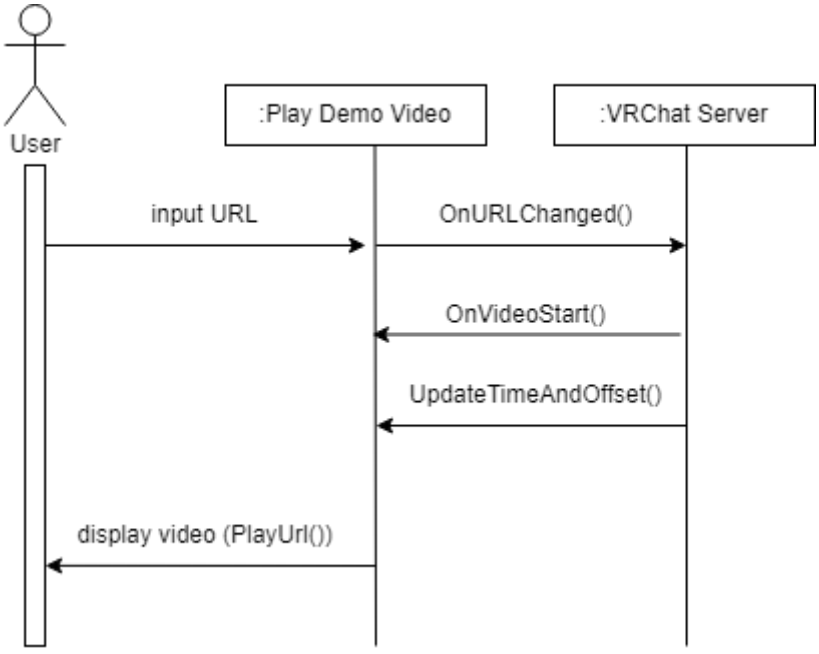

Figure 9. Sequence Diagram - Playing Demo Videos

4.2.4. Shifting Between Rooms

4.2.4.1. Attributes

- World: Specify the world destination of the portal. Hub / Home / None options available.
- Room Id: Specify the world destination of the portal by id. Input string starts with the “wrld_” prefix. If no string is specified, the world is determined by the search function below Sort Heading.
- Sort Heading: Specify what to search for. Input should be chosen among Featured / Trending / Updated / Created / Active / None options.
- Sort Order: The order where the search results are sorted. Ascending / Descending options available.
- Offset: A integer that specifies which of the search results to adopt. First with zero, next with 1.
- Search Term: Search results are narrowed down to worlds that include a specified string in their names.
- Tag: Specify tags.
- Use Default Presentation: If the checkbox is checked, the world uses the default presentation.
- Effect Prefab Name: Specify the name of prefabs to be affected.
- Room Name: Room name to display at the top of the portal. Input should be a string.

4.2.4.2. Methods

By using VRC_PortalMarker provided by VRChat Portal can be implemented without any methods

4.2.4.3. Class Diagram

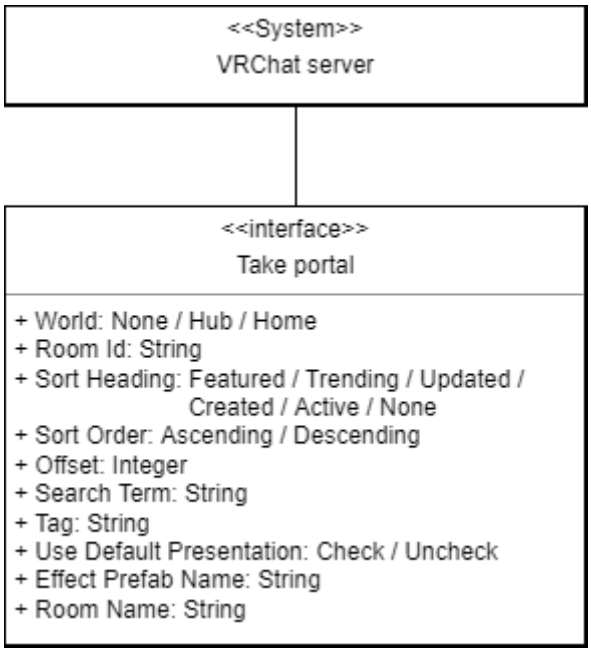

Figure 10. Class Diagram - Shifting Between Rooms

4.2.4.4. Sequence Diagram

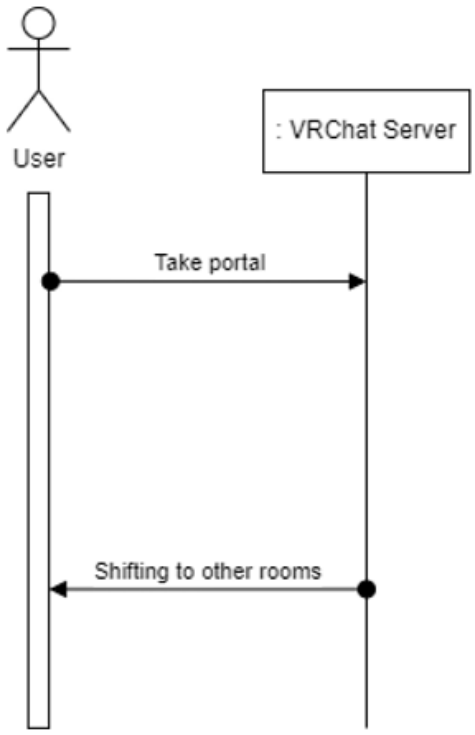

Figure 11. Sequence Diagram - Shifting Between Rooms

4.2.5. Voting a Poster

A ‘voting a poster’ class deals with the history of users’ choice. It is a process in which users can vote for the work they want. Users can vote for their favorite posters and each ach user can vote only once. Accordingly, it is possible to know posters that are popular with users.

4.2.5.1. Attributes

These are the attributes which voting object has.

- Board: a board for counting the vote
- Box: a box for voting, count would be increased when a user throws a ball into a box
- Ball: a ball for voting, every user would receive 1 ball when entering

4.2.5.2. Methods

These are the methods that voting class has.

- GameObject.Destroy() : If the ball hits the box, it disappears.
- Collider.OnTriggerEnter() : Detect that the ball is thrown into the box and perform the next move.

4.2.5.3. Class Diagram

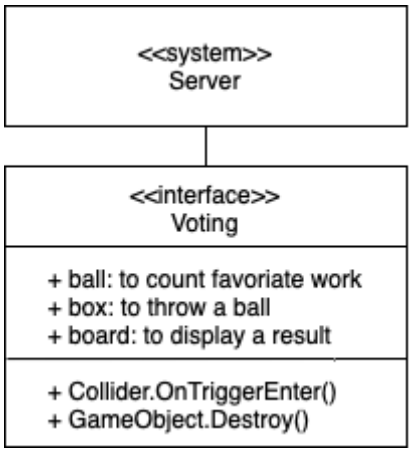

Figure 12. Class Diagram – Voting

4.2.5.4. Sequence Diagram

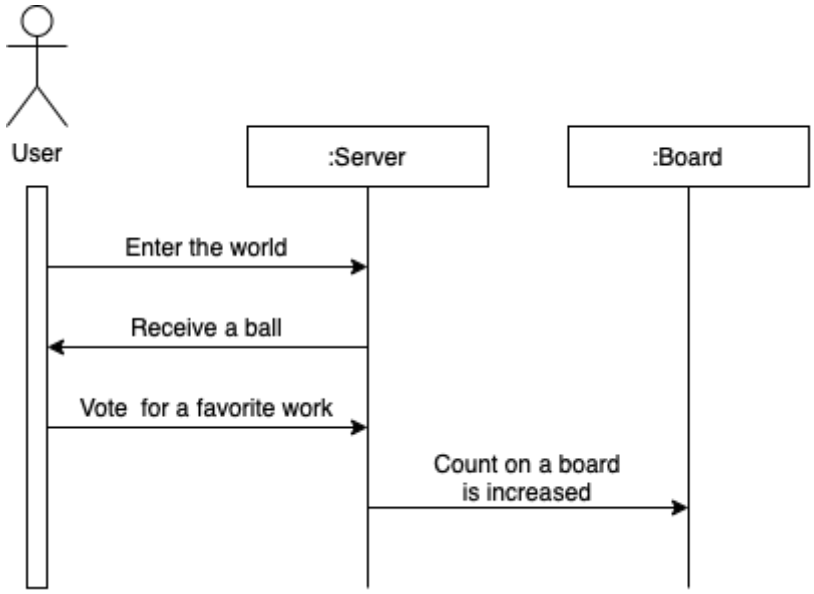

Figure 13. Sequence Diagram – Voting

5. Human Interface Design

This chapter describes the functionality of the system from the user's perspective and explain how the user will be able to use the system.

5.1. Overview of User Interface

5.1.1. Poster Exhibition

Users can see the poster of the participant's graduation work. Users can magnify the poster by clicking the poster. The posters of the participants are listed in a relatively small size in the room. When users click on the poster, the poster is magnified and displayed. Users can close the magnified window by clicking 'Done with the Poster' button.

5.1.2. Playing Demo Video

Users can watch the demo video of the work by clicking the 'play' button displayed on the poster whiled the poster is magnified. Then, a demo video that participant uploaded on Youtube will be played. Video will be closed when users click 'End video' button.

5.1.3. Shifting Between Rooms

Users can move to another space through the portal. This portal allows the user more convenient way to move to the other team's space and use services provided by them. After accessing the portal, the user can know the list of spaces that they can move from there through the portal click, and can select the space the user wants to move. If the user selects other space and click 'move' button, they can move to other space. When they click 'Exit' button, they can cancel the movement.

5.1.4. Voting a Poster

Users can vote for their favorite posters. Each user can vote only once. Accordingly, it is possible to know posters that are popular with users. The user receives one unique ball when entering the hall. Users can vote their favorite poster by putting the ball into ballot box in front of the poster. The ballot box shows how many users put the ball in the ballot box.

5.2. Screen Images

5.2.1. Poster Exhibition

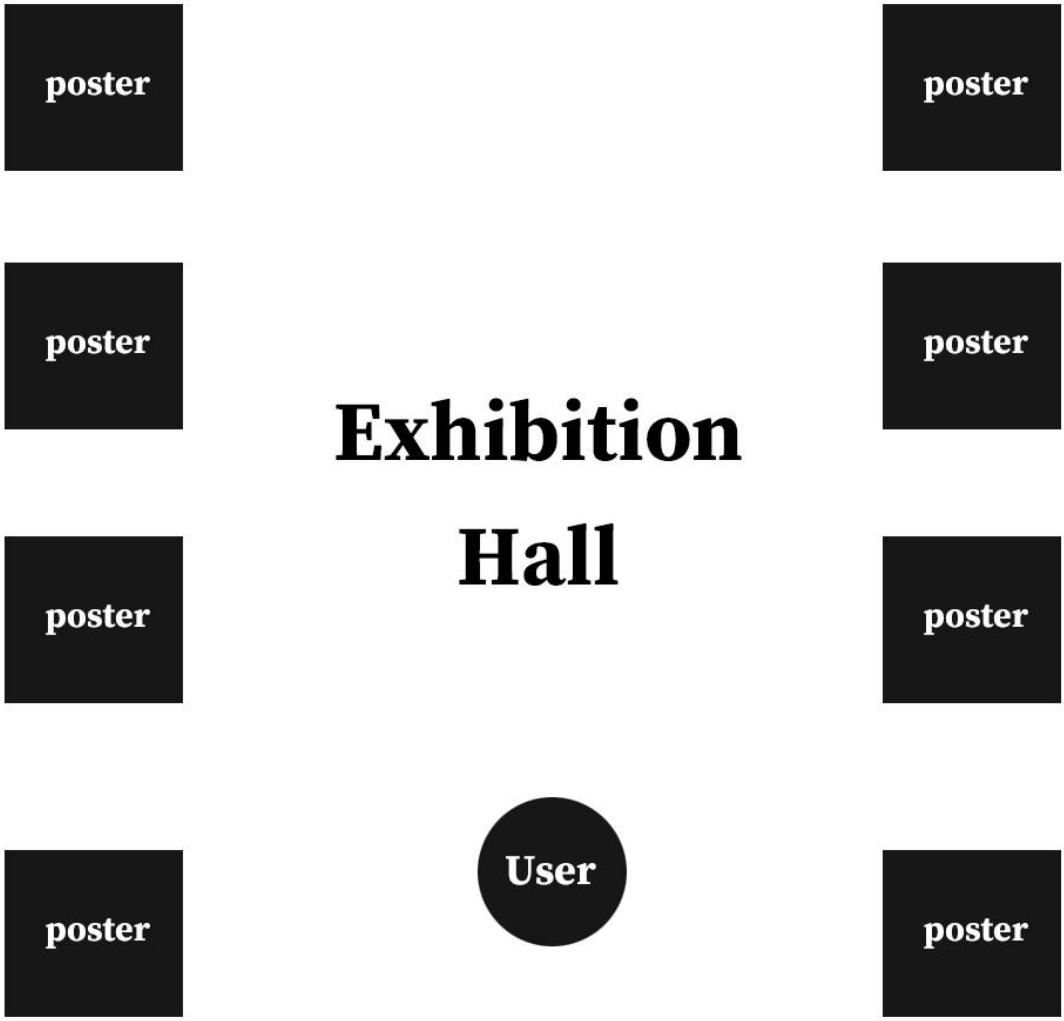

Figure 14. Exhibition Hall

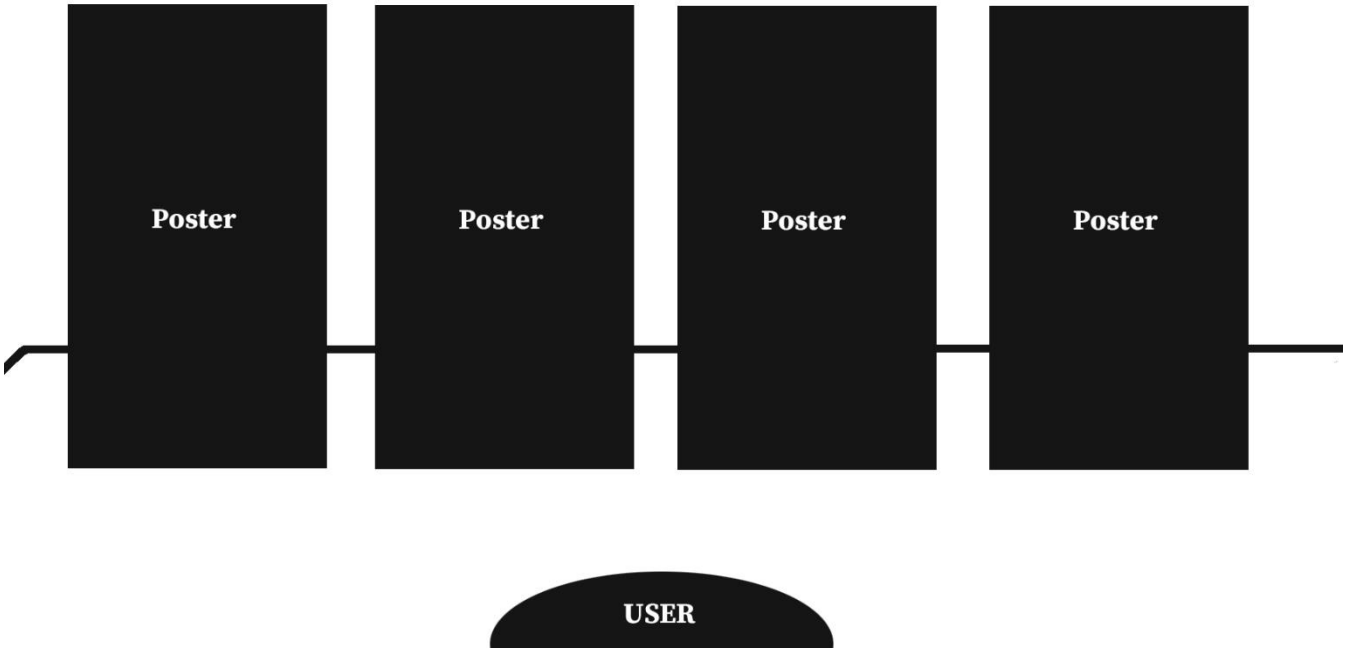

Figure 15. User's View in Front of Posters

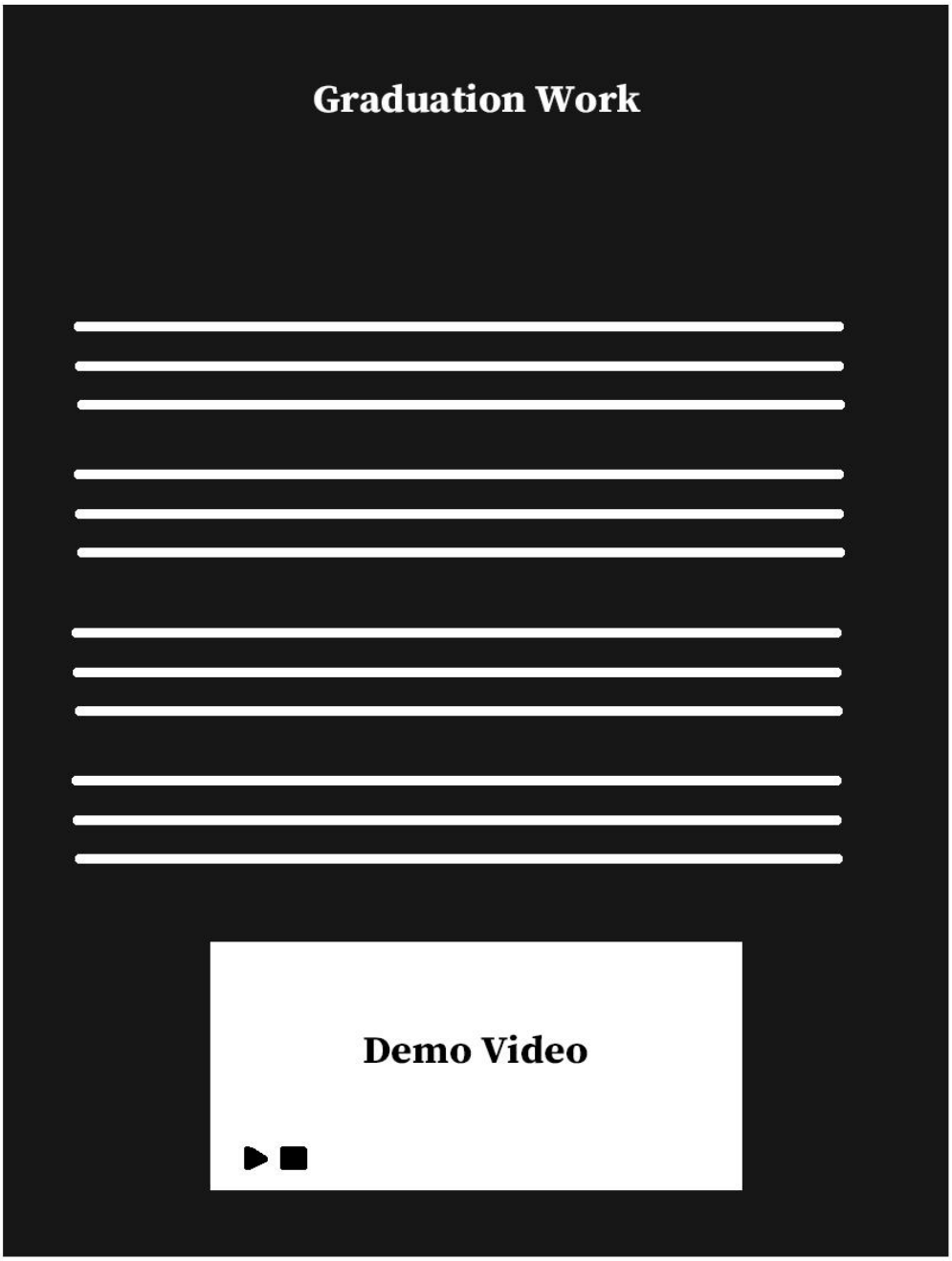

Figure 16. Magnified Poster

5.2.2. Playing Demo Video

Figure 17. Video Player of Demo Video

5.2.3. Shifting Between Rooms

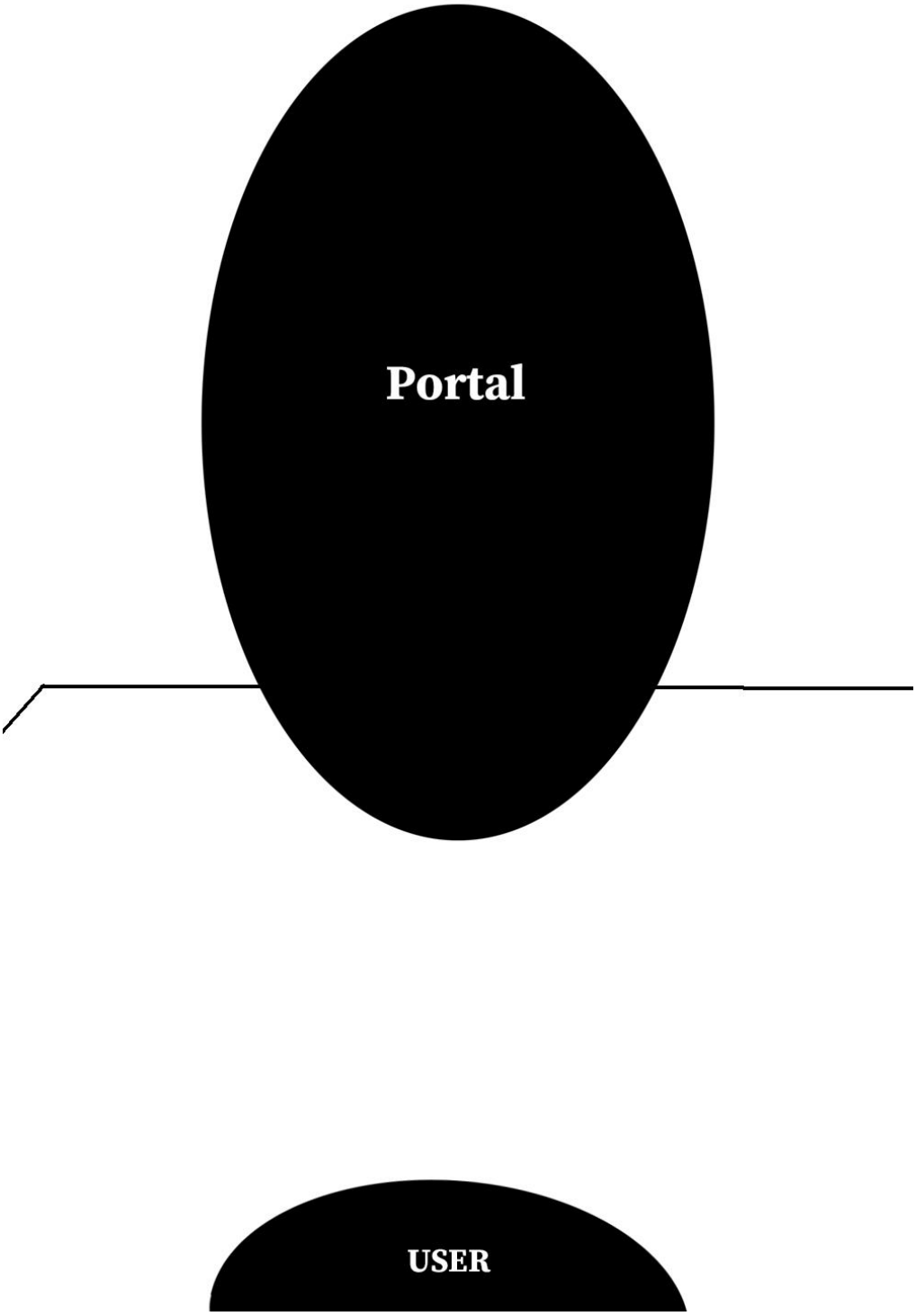

Figure 18. User's View in Front of the Portal

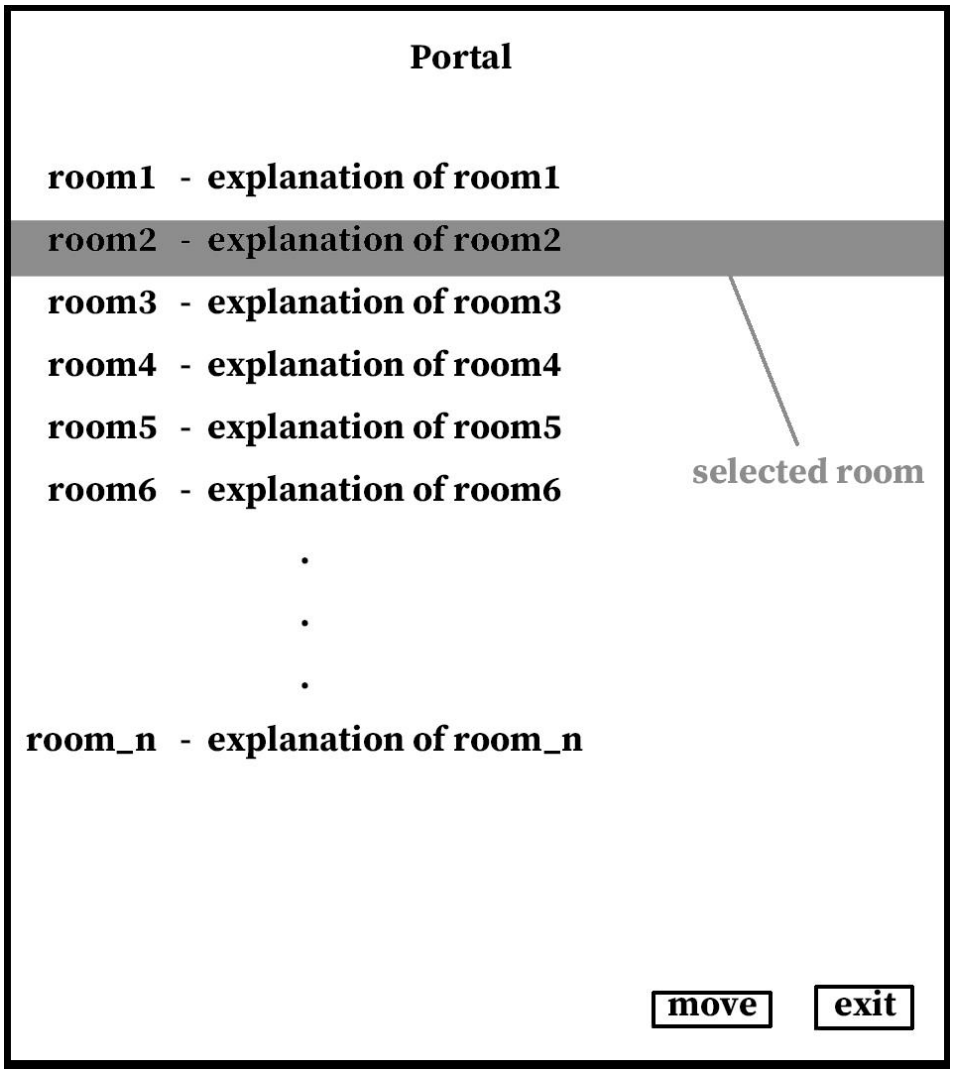

Figure 19. Window That Appears When User Click the Portal

5.2.4. Voting a Poster

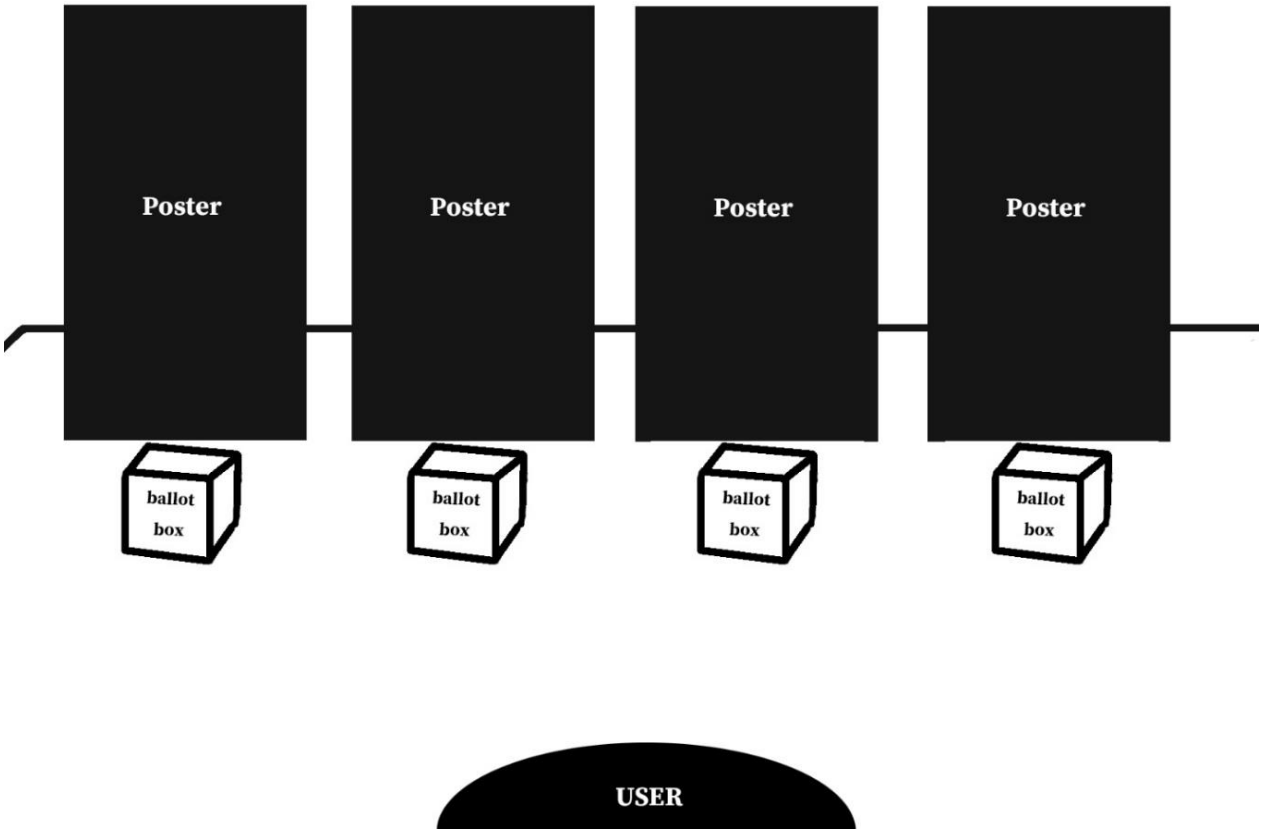

Figure 20. User's View in Front of Posters

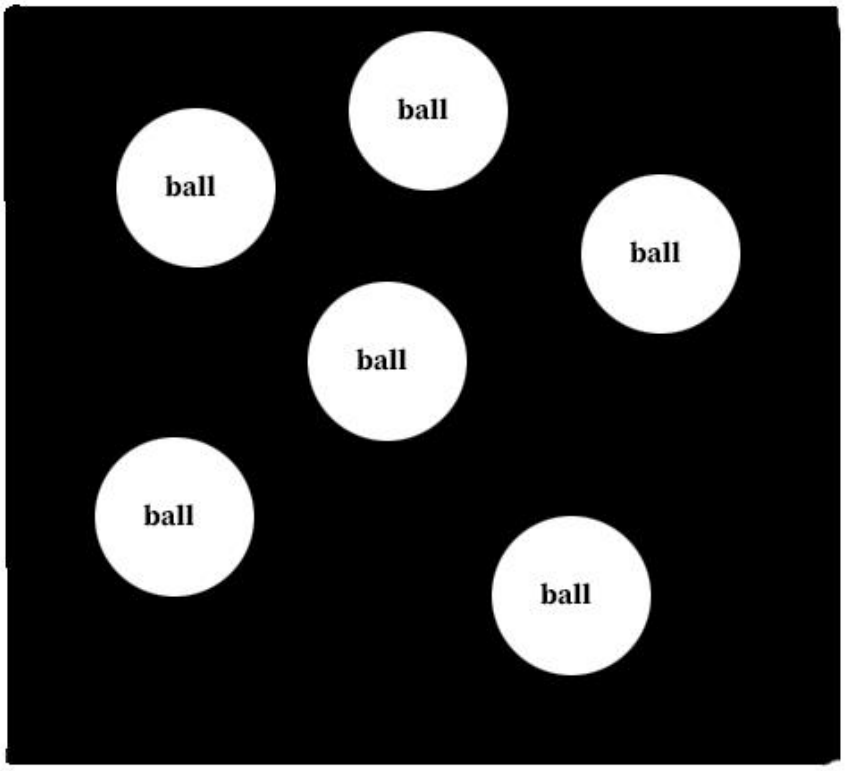

Figure 21. Example Figure of Inside the Ballot Box

6. Testing Plan

6.1. Objectives

This chapter illustrates plans on testing which has three major sub-groups of development testing, release testing, and user testing. These tests are crucial in that they detect potential errors and defects of the world and ensure flawless operation and stable release of the world to users.

6.2. Testing Policy

6.2.1. Development Testing

Development testing is carried out mainly for synchronized application of a broad spectrum of defect prevention and detection strategies in order to reduce potential risks of software development and save time and costs.

In this phase, since the software has not undergone enough tests, the world could malfunction. Therefore, static code analyzing, data flow analyzing, peer code review, and unit testing need to be carried out in this stage. Through these processes, we are mainly focusing on achieving 1) performance related with the functions we want to implement, 2) reliability for ensuring safe and failure-free operation, and 3) compatibility to check the stability of moving to another world.

6.2.1.1. Performance

Since the components we try to implement – PDF for external data-posters, YouTube links for demo videos - we need to make sure that these data are loaded well in a short time and that no errors occur during upload. We would prepare test cases on various data and evaluate whether errors occur from uploading to utilizing.

6.2.1.2. Reliability

In order for the system would operate safely without failure, the sub-components comprising the system should operate and be connected correctly first. Therefore, we need to undergo development testing from entering the world and check failure iteratively while users implement each component in the world.

6.2.1.3. Compatibility

Since this project is to create a huge smart campus, the world we create and the worlds of different teams should be connected through portals so that they can cross each other's worlds freely. Therefore, we will create a virtual dummy world and check whether the movement between the worlds takes place without problems.

6.2.2. Release Testing

One of the most critical parts of any software development project is to release the product (in our case, the world) to customers. A technically good software can go wrong due to a error during publishing. Therefore, release testing is inevitable for better connection between the product(world) and the user. Release testing is testing a new version of a software/application to verify that that the software can be released in a flawless, so it doesn’t have any defects on its working. It should be carried out before release.

Based on Software Release Life Cycle, testing is generally initiated from the ‘Alpha’ version, where a basic implementation of the software is completed. We would start development testing in Alpha version, and release Beta version for the further testing including user and release testing.

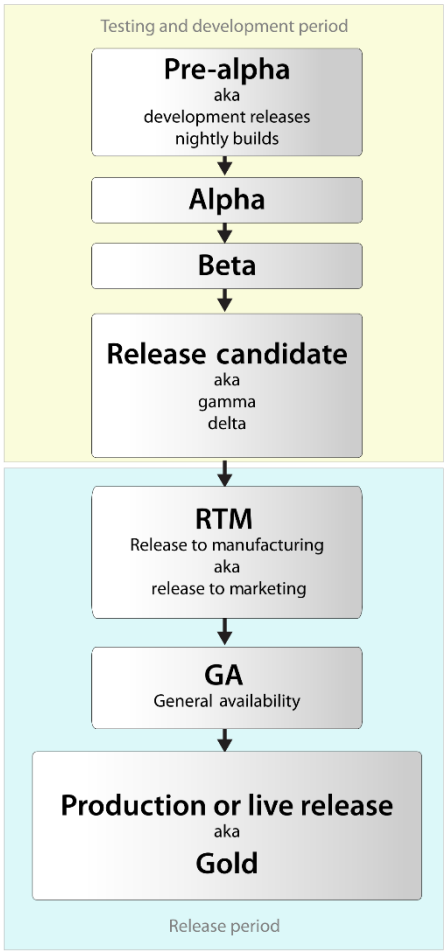

Figure 22. Software Release Life Cycle

After Beta version is released, we would get feedback from actual users as well as developers.

6.2.3. User Testing

We should set up possible scenario and realistic situation that can proceed necessary user tests. We assume that there would be more than 30 users accessing the world simultaneously. We would publish a test version so that random users could experience the world and we could collect user reviews.

6.2.4. Testing Case

Testing case would be set according to 3 fundamental aspects of the application– function(interaction), performance, and compatibility. We would proceed testing based on the components we implement and make an evaluation sheet for each component.

7. Development Plan

7.1. Objectives

This chapter illustrates the technologies and environment for the development of the world.

7.2. Environment

7.2.1. GitHub

Figure 23. GitHub Logo

It provides hosting for software development version control using Git. It enables teammates to develop a single project together, resulting in easy integration of components. We are now using GitHub for developing the world and controlling version of it.

7.2.2. VRChat

Figure 24. VRChat Logo

VRChat is an online virtual world platform and it helps to make an virtual world which anyone can join. Even if the physical distance of users is far away, it allows them to experience being in the same place. We use VRChat to create a virtual graduation exhibition hall and create a platform where you can enjoy graduation works and papers without having to visit the school in person.

7.3. Constraints

The system will be designed and implemented based on the contents mentioned in this document. Other details are designed and implemented by selecting the direction preferred by the developer, but the following items are observed.

- Use the technology that has already been widely proven.
- Users should be able to come and go hall anywhere VRChat is available, 24 hours.
- Use open source software whenever possible.
- Avoid using technology or software that requires a separate license or pays for royalty. (Exclude this provision if this is the only technology or software that the system must require.)
- Decide in the direction of seeking improvement of overall system performance.
- Decide in a more user-friendly and convenient direction.
- Consider integration to other VRChat projects.
- Consider future scalability and availability of the system.
- Optimize the source code to prevent waste of system resources.
- Consider future maintenance and add sufficient comments when writing the source code.
- Develop on Windows 10 environment and Unity 2019.4.30f1 that is compatible with VRChat.

7.4. Assumptions and Dependencies

All systems in this document are written on the assumption that they are designed and implemented based on VRChat and open source. Therefore, all contents are written based on VRChat with Unity 2019.4.31f1 and may not be applied to other operating systems or versions.

8. Supporting Information

8.1. Software Design Specification

This software design specification was written in accordance with the IEEE Recommendation (IEEE Recommended Practice for Software Design Description, IEEE-Std-1016).

8.2. Document History

Table 1. Document History

Date	Version	Description	Writer
11/12	0.1	contents and overview	Jaekwang Shin
11/13	1.0	Add 4.1, 4.2.1	Jihee Kim
11/13	1.1	Add 5.1, 5.2	Jongin Park
11/15	1.2	Add 1.1 ~ 1.4	Gyumin Han
11/16	1.3	Add 2.1 ~ 3.2	Seokjun Chung
11/18	1.4	Add 4.2.4	Gyumin Han Seokjun Chung
11/19	1.5	Add 6	Bomin Namkoong
11/19	1.6	Add 7	Jaekwang Shin
11/20	1.7	Add 4.2.2, 4.2.5	Bomin Namkoong
11/20	1.8	Add 4.2.3	Jaehyeong Park