Networking Platform for graduates and students

Design specification

2021.11.19.

Introduction to Software Engineering 41

TEAM 7

Team Leader	Jiyeol Park
Team Member	Shinhyeon Park
Team Member	Mujin Gwak
Team Member	Chaewon Jeong
Team Member	Hayoung Cho
Team Member	Bohyeon Mo
Team Member	Sojeong Um

CONTENTS

1. Preface

In this chapter, the summarization of this document is provided. Users can easily read this chapter and have a better understanding even though users are new for our software system. We expect users to have a precise overview of our software system so that they can freely use our software in a complementary environment.

Users are expected to give feedback on our system. For that, we will provide a precise explanation and compact information for better understanding. Diagrams, tables and pictures are our main tool to describe our service.

We will briefly introduce what we are aiming for. Quantitative objects will be mainly described in this section.

There will be a definition of readership and an overall description of the Design Specification. We will introduce terms that we are going to use for this document. There is some explanation about the recommended environment specification as well.

This document is for Software Design specification to provide much information and understanding for potential readers and users of this software application, especially group members of SungKyunKwan Univ. This software has been developed by Software Engineering Team 7 at SungKyunKwan Univ.

1.1 Objectives

In this section, what the software system is aiming for is introduced. Our software system is based on VRChat, which will be introduced in a later section. By using this, we are hoping to make connections between students and graduates.

In our school facilities, there is a Student Success Center, where make mentor group can give aphoristic advice, and better school life. However, we have come to know that there are a few students who utilize this functionality even though the benefit of this facility is way big.

Our project is to facilitate the role of the Student Success Center throughout this Software system. We are aiming to make a place for the undergraduates and graduates to meet, talk, and share their information. The overall functionality is to make counseling. Students are having trouble getting a job, so the graduates will give great advice as the incumbent.

For this, we are going to provide a virtual space where students and graduates can freely meet and talk together. Graduates may share their own information for undergraduates in this space. They can make appointments for counseling as well.

As our school is now on 600 years history of Sung Kyun Kwan, this project is planning to provide traditional virtual space. We will make or fetch fancy rendering on our own or from GitHub.

This project can be implemented using reuse-oriented development. We will find out reusable components and make things work by making interaction.

Our project is based on VRChat - Virutal Reality Chatting Platform. VRChat aims to enable anybody to create and share their own assets on an online community. People are making their social and virtual space and sharing. We are going to participate in this community and make our own space. There is a detailed description on this document, “Definition of Terms” section in the “Introduction” chapter. You can have some idea what VRChat is.

1.2 Readership

Expected reader of this document is every user of this software system and every people (System engineer) who commit to this project, Professor and TAs as well.

User readers are expected to have a brief vision of this project by reading this document. They can have information about what will be constructed through this project and what kind of methodology will be taken.

System Engineers are expected to have specified divisions of labor. This document explains

how the system works, but also the role of contracts to divide labor of team members. They can have precise objectives via this document as well as a contract.

Professor and TAs are expected to assess Software Engineering Team 7 by this document. This document is the product of team-work, thus every team member willingly commits to this document.

2. Introduction

In this chapter, the main concept of this software system is provided. Potential readers can have an overview of this software system by reading this section. The overview will be provided with definitions of terms.

This chapter also covers the description of software systems. The common environment for the system is explained in a literal way. The potential users are recommended to have such an environment to have better experience for this system.

2.1 Definition of Terms

In this section we define terms used in this project. Terms that are used are as follows.

2.1.1 VRChat

VRChat is an online virtual reality (VR)platform. This platform allows users to interact with others with their own character which is a 3D rendered avatar. They can speak to others, interact with other assets and play under a rule of the map.

figure 1 VRChat logo

VR platform is much extendable platform that many users can easily fertilize the asset and make events. It is now becoming a new playground for people in the era of COVID-19.

2.1.2 Asset

Asset corresponds to VRChat asset. VRChat platform utilize virtual object which is called "Asset". Assets have their own attributes that embody those assets in the VRChat platform.

2.1.3 Events

Events define certain operations of the VRChat assets. Such operations are implemented using script language (e.g. udon). Operations include interactions with users and changing features of other assets. Such interaction may occur automatically or manually by triggering.

2.1.4 Ganhtt Chart

There are some diagrams to describe our plan or flow of the scenario. Ganhtt chart is one of the diagram used in this document. Ganhtt char is bar-typed diagram to illustrate a project schedule.

In this chart, tasks are listed on the vertical axis and time interval on the horizontal axis. The width of horizontal bar represents the duration of the task, using start date and finish date.

2.1.5 Udon

Udon is a programming language used in VRChat platform. It is completely built in-house by

the VRChat Development Team. It is designed to be secure, performant, and easy to use. This can be represented in the form of Node Graph which is called VRChat Udon Node Graph.

VRChat Udon Node Graph is built-in visual programming interface using nodes and wires to represent the flow, inputs and outputs. By using this, we can replicate behavior of Triggers and Actions. We can create our own behavior and interactions as well. In this project, Udon is our main programming language.

2.2 Recommended Environment

Recommended Environment follows official system requirements of VRChat on Steam. Such requirements are as following.

System Requirements	
OS	Windows 8.1 or Windows 10
Processor	Intel® i5-4590 / AMD FX 8350 equivalent or greater
Memory	4 GB RAM
Graphics	NVIDIA GeForce® GTX 970 / AMD Radeon™ R9 290 equivalent or greater
DirectX	Version 11
Network	Broadband Internet connection
Storage	1GB Available space

table 1 system requirements

2.3 Document Structure

In this section, structure of this document and brief description of each chapter will be

introduced. All users are expected to build mainstream of this document so that they can easily understand what will be introduced next.

This document has 10 parts. Each description is followed.

2.3.1 Preface

In this section, overview of this document is provided. Objective and Readerships are described here.

2.3.2 Introduction

In this section, main idea of this software system is introduced. Definition of the term and components of the system will be briefly introduced for better understand afterward.

2.3.3 System Architecture – Overall & Tools and Language

In this section, overall system architecture is introduced. Overall features are described in this part. We are going to explain what kind of tools are used, what kind of language is used to develop our software system. The system architecture can be described in aspect of Asset and Events.

2.3.4 System Architecture – Asset

In this section, system architecture is described in terms of asset components. Objective of these assets are explained and the procurement of these components are introduced.

2.3.5 System Architecture – Events

In this section, system architecture is described in terms of events components. The way how the components interact is introduced. Some of the Events are described with diagram.

2.3.6 Protocol Design

In this section, a networking protocol of VRChat service is introduced. What kind of protocol is used and how the serviced are provided are described here.

2.3.7 Database Design

In this section, a database system of VRChat service is introduced. How they manage the metadata and user data for better procurement environment.

2.3.8 Testing Plan

In this section, a testing plans are introduced. The development tests are conducted in 3 ways: Development testing, Release testing and user testing. This testing plans act as a contract, so this chapter is as crucial as this document itself. Users and System Engineers should pay more attention for this chapter for better understand of this project.

2.3.9 Development Plan

In this section, a Development plan is introduced. Overall schedule is express in Gantt chart, so that we can check our plan in a chart.

2.3.10 Supporting Information

In this section, a supporting information of this project is introduced. Summarizing environment specification and tools that we are going to use are described.

3. System Architecture – Overall & Tools and Language

3.1 Overview (Tools and Language)

On building a metaverse service, three things should be considered. one is cost and labor.

metaverse is mostly covered by various 2D/3D assets. and each asset consumes much time to design and make. So if developers don't consider reducing the asset cost, the project will be very time-consuming. Second is platform familiarity. Most of the metaverse services target many kinds of devices and environments. If the development tool and platform we choose can't support PC or mobile, the range of our product can be delivered is very downsized. Last is procurement. In a metaverse system, many functions and objects must be implemented. But this is much time-consuming and the cost of development will become higher and higher. So, we have to consider how can we reduce the functions that we will implement on our hands by importing a precise platform or open source that already has a lot of useful functions. According to the three reasons, we choose Unity for development tools and VRChat for our VR platform.

3.2 About Unity

Unity is a game engine that provides a development environment for 3D and 2D video games and an integrated production tool for creating interactive content such as 3D animation, architectural visualization, and virtual reality (VR).

Unity can also create content available on 27 platforms, including Windows, MacOS, iOS, Android, PlayStation, Xbox, Nintendo Switch, and WebGL, while Unity Editor, a production tool, supports Windows and MacOS.

The engine itself is equipped with middleware such as light mapping and physical engine, and various facilities can be downloaded and used through the asset store built into the editor.

It is distributed in three editions: Personal, Plus, and Pro, and individuals can use personal editions for free. There is an official original character called Unitychan provided by Unity Technology Japan. It is distributed free of charge in accordance with the terms and conditions so that developers using the game engine Unity can freely set up characters.

3.3 Unity Features

- **Physics**

figure 2 Unity Physics

Unity provides two different kinds of physics engines. one is a built-in physics engine for Object-oriented-engine. It consists of built in 2D physics and 3D physics. like Nvidia PhysX engine integration and Box2D engine integration. and the other one is Physics engine packages for data-oriented projects. it is used when you use unity's data-oriented-tech-stack(DOTS). And by installing it to a local PC, users can use Unity physics package and Havok package.

- **Open source**

figure 3 Open source

Unity provides 2D/3D asset import function. And there are so many useful assets in unity asset stores. From fantasy assets to significantly modern assets, unity asset stores not only have

a lot of assets but also various kinds of things to use. Not all assets are free, but there are free assets and non-free assets too. So, users can buy or just download the asset they need, and by importing it to their Unity project, finally can make their own project easily.

- **Cross platform**

figure 4 Cross Platform

Unity's other interesting feature is cross-platform familiarity. Unity can access general PC OS like Windows, Linux, Mac and general phone OS like IOS, Android. It is a very important point because by this wide range of accessible platforms, game engineers can both make their game into PC game and mobile game.

And Unity's familiarity with various platforms is not limited to PC and mobile. it also provides console game platforms like WebGL, PS4, Xbox. So by using Unity, developers can make a range of people to use and enjoy their product.

3.4 Unity Architecture

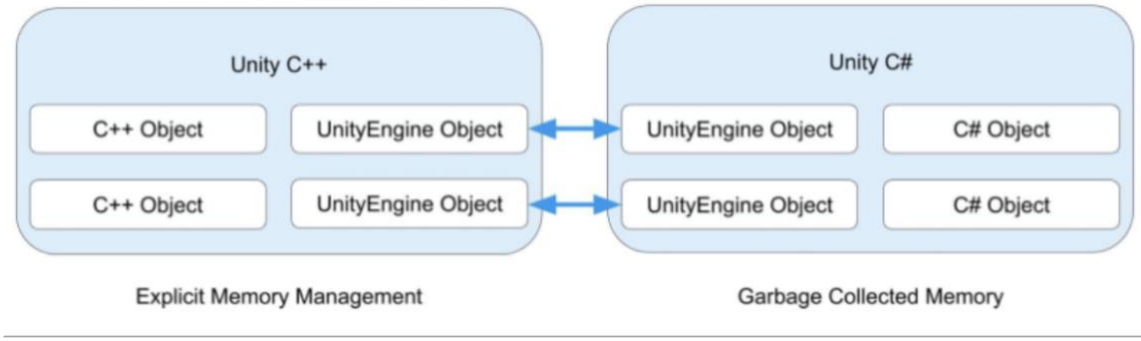


figure 5 Unity Architecture

Unity engine is built by native C/C++. But it uses C# reference to interactive with each object. this is because Unity has its own special type C# Unity engine object. That object type is connected to native C++ counterpart object. So, to interact with C++ object and C# object, Unity uses C# for scripting language.

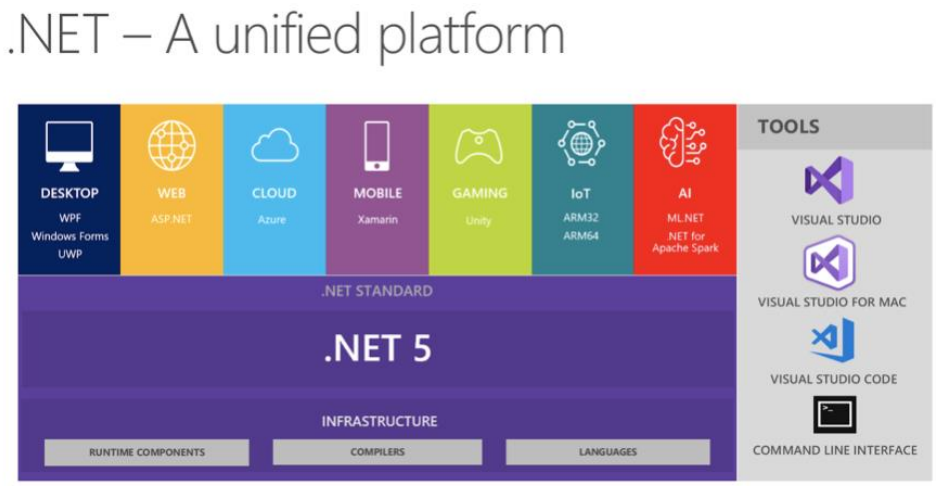


figure 6 .NET - a unified platform

And unity's network framework is .NET. An open source network framework .NET's most impressive pros is a wide range of device and environment familiarity. It supports basic devices like PC, phone, but also Web, console and so on. Because of this feature, .NET gets good synergy with Unity. And it is supported by two kinds of scripting backend. one is Mono and the other one is IL2CPP. Mono uses Just-in-time compiling method and IL2CPP uses ahead-of-time compiling method.

3.3 About VRChat

VRChat is a partially paid, large-scale online virtual reality social service for users scheduled to be officially released by Graham Gaylor and Jesse Joudrey. In this game, players can interact with other players implemented as 3D character models. The game was released exclusively for Microsoft Windows on February 1, 2017 as Steam's Early Access program. It supports Oculus Lift, HTC Vibe, and Windows MR Headset (en).

3.3.1 Overview

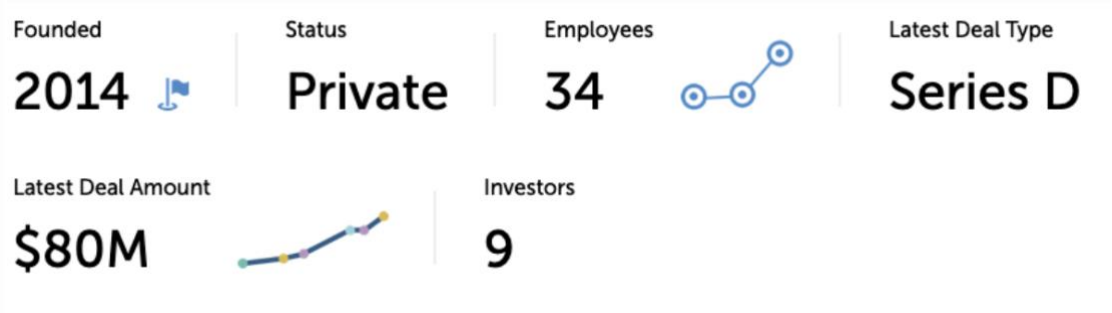


figure 7 VRChat's status

Developer of a virtual reality social platform designed to create and explore virtual worlds. The company's platform helps users to create, custom avatars, join communities, publish and explore virtual worlds with other people from around the world, enabling customers to experience social virtual reality by creating various 3D content.

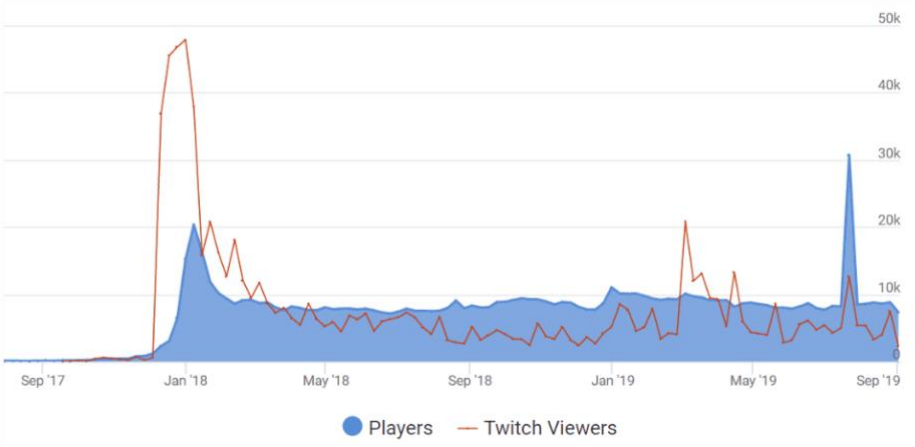


figure 8 VRChat's Popularity

Although the game was founded in 2014 and launched in 2017, VRChat just closed on an additional $10 million in Series C investment, bringing to total funding to just over $15 million in three rounds.

VRChat went viral a few months after it's launch, primarily due to Twitch live streaming. You can see that shortly after the spike of Twitch views, there was a corresponding influx of VRChat players. VRChat memes played a big part in its viral trend.

3.3.2 Features

As below in the above picture, VRCHAT has nine main characteristics.

A. Full Body Avatars: Avatars with lip sync, eye tracking/blinking, and complete range of motion.
B. Express Yourself: Express yourself with hand gestures, emotes, and emoji
C. 3-D Spatialized Audio: Our 3-D Spatialized audio helps you hear the conversations important to you
D. Play Games: Play Capture the Flag, Battle Discs and games built by our community.
E. Fun Things To Do: Chat, collaborate, draw, sculpt and more with your friends.
F. Avatar Creation: Create your own avatars and worlds with our Unity SDK.
G. Explore Worlds: Explore hundreds of worlds created by other community members.

H. Weekly Events: Get involved with official and community events.

I. Build Friendships: You never know if the next avatar you will meet will turn out to be your next best friend.

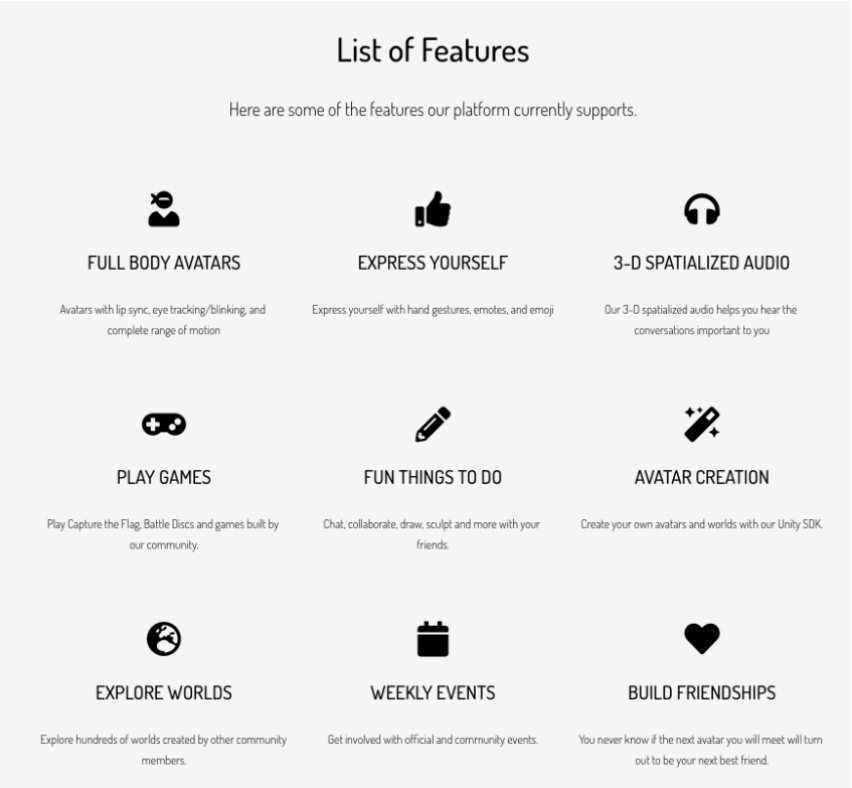


figure 9 List of Features

3.3.3 Play Method

VRChat's gameplay is similar to Second Life or Habbo Hotel. Players can create an instantized world that can interact with others through virtual avatars. Through the software development kit released with the game, players can create or call characters directly from various franchises to change characters. The player model supports "voice lip sync, eye tracking, blinking, and a full range of motion. It includes several mini games such as "en: Capture the Flag", "Roba bank in Steel 'n' Gold", and "Battle Disks."

figure 10 Similar Product (Left: Habbo Hotel / Right: Second Life)

figure 11 Interior Space of VRChat

Basically, VRChat uses VR headsets, but even without VR, it can be played limitedly using desktop versions. However, in this case, there is a limitation that the avatar cannot move freely.

3.3.4 Avatar

In VRChat, users can select and play numerous avatars. If there is no avatar, users can upload a 3D model using Unity and VRChat SDK to create each other's own avatar.

figure 12 Custom Avatare example from VRChat

- **Avatar 3.0**

Avatars 3.0 is a huge collection of features for avatars in VRChat. You can think of it as a new version of our avatar feature framework. Previously, you've been using Avatars 2.0 (in SDK2). Avatars 3.0 will be usable with VRCSDK3. Both can exist side-by-side in VRChat at the same time, but AV3 avatars will be far more powerful.

AV3's features are focused on improving expression, performance, and the abilities of avatars in VRChat. Moreover, we've learned a lot from years of watching users find ways to do cool stuff with Avatars 2.0. A lot of those methods are considered "hacky", and it is hard for us to support those. We want to formalize the process so you can do the things you want, access them more easily, and use them in a system that is officially supported.

3.3.5 SDK

- **VRChat SDK3 (VRCSDK3)**

The VRCSDK3-Avatars package comes with Avatars 3.0, the latest avatar framework we offer for creation of both basic and advanced avatars with full customization. VRCSDK3-Worlds comes pre-packaged with VRChat Udon for programming advanced actions.

3.4 Udon

VRChat Udon is a programming language built completely in-house by the VRChat Development Team. It is designed to be secure, performant, and easy to use via the VRChat Udon Node Graph, a built-in visual programming interface that uses nodes and wires (we call them “noodles”) to connect flow, inputs, and outputs. You can build complex behaviors with Udon-- far more complex and easier to understand than unwieldy chains of Triggers and Actions.

Not only can you replicate the full behavior of Triggers and Actions with VRChat Udon, but you can create your own behaviors, sync variables with others, interact with scenes, interact

with players, and more. In addition, Udon runs in both the VRChat client and the Unity Editor, allowing you to test and debug your creations with ease.

For the more technically inclined: VRChat Udon is a VM running bytecode compiled from Udon Assembly. You can generate Udon Assembly using the built-in VRChat Udon Node Graph UI, writing your own Udon Assembly, or even by writing your own compiler to generate Udon Assembly or bytecode programs directly.

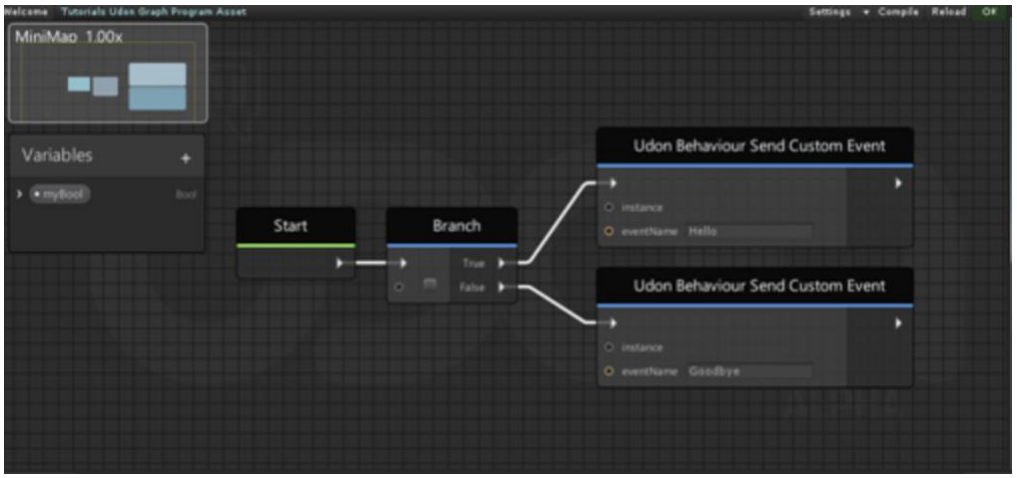


figure 13 Example of VRChat Udon

The VR Chat Udon Node Graph is a programming interface that enables visual programming through a connection between nodes and wires, and you can simply think of the blue print of the Unreal Engine. VR Chat Udon not only allows you to replicate all actions of triggers and actions, but also allows you to interact with your own actions, scenes, or players. Additionally, Udon works on both VRChat clients and Unity editors, making it easier to debug and see if it works properly.

4. System Architecture – Assets

4.1 Entrance Area

- **Concept**

Brick road entrance of SKKU Myeongyun Campus

- **Motivation**

It is essential to make user feel the feeling that they are entering to special space. At the same time, it is important to build familiarity because our space is not only just interesting space, but also community area for SKKU student and graduates. So we benchmark the brick road entrance of Myeongyun for familiarity, and place some features to make a special mood.

- **Outline**

- There is a long, vertical brick road from the entrance to the selection area.
- Right and left side of the brick road, there is Long guidance sign that introduce what is the objective of this place and how to use it
- Behind the guidance sign, there is some traditional houses on the left side that reproduce Myeongyun campus, and modern buildings on the right side that reproduce Yuljeon. campus.

- **Space Drawing & Specification**

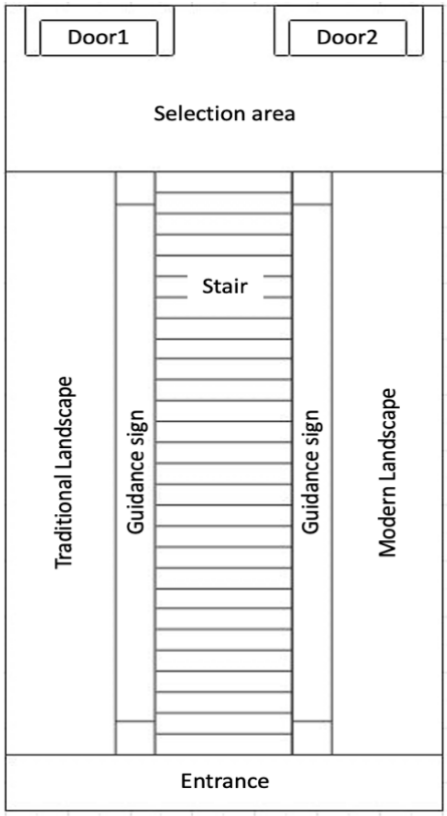


figure 14 Entrance Area

a. Entrance

: Area that users can see when they enter to our world through VRChat platform.

b. Stair & Guidance sign

: brick stairs and sign that introduce users what our world's objective is and how to use each sub-world.

c. Traditional landscape

: landscape that is consisted of several traditional features to symbolize long history of SKKU.

d. Modern landscape

: landscape that is consisted of several modern features to symbolize future of SKKU.

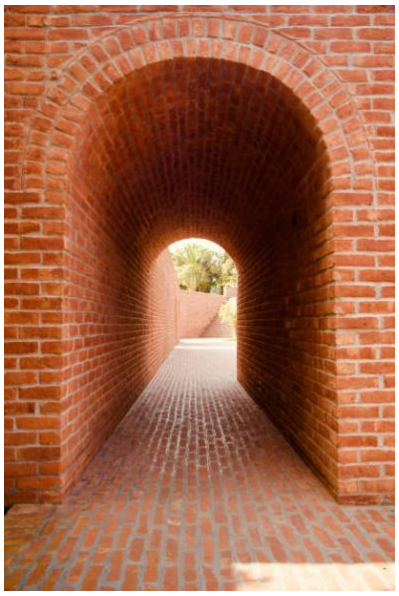
e. Selection area

: Area that can choose sub-world to go. and there is two doors that have transportation functions to each world.

- **Details (Space Architecture Introduction)**

a. Entrance

: short brick entrance that has shape of arch. before entering the entrance, users can't see the large view of our world because sight of user is blocked by bricks. But after they enter the short tunnel, they can have wide sight and interesting first feeling of our world.



- **Essential asset**

: arch brick entrance, SKKU mark

- User flow

1) enter to world and see brick tunnel with SKKU mark

2) enter the brick tunnel

3) enter to entrance stair

figure 15 Entrance details

b. Stair & Guidance sign

figure 16 Durumari and modern electric sign

figure 17 road

: Guidance about simple tutorial to our world and introduce each world (traditional world and modern world) and how to use it. there is traditional guidance sign on the left side and modern guidance sign on the right side.

- **Essential asset**

: Durumari, modern electric sign

figure 18 Traditional Landscapes

: show Korean traditional features and view on the left side of the brick stairs. this landscape design has objective about symbolizing long history of SKKU. So, by passing traditional landscape, users can see not only beautiful view, but also can feel like they are walking through the history of SKKU.

- **Essential asset**

: lake, trees and Hanok buildings

- **User flow**

1) enter to brick stairs and take eyes to the left side.

2) enjoy traditional buildings and landscape.

d. Modern landscape

figure 19 Modern landscape

: show Korean modern features and view on the right side of the brick stairs. this landscape design has objective about symbolizing present and future of SKKU. So, by passing modern landscape, users can see not only sophisticated view, but also can feel like they are walking with present and future of SKKU.

- **Essential asset**

: trees, modern buildings

- **User flow**

1) enter to brick stairs and take eyes to the left side.

2) enjoy traditional buildings and landscape.

e. Selection area

figure 20 Selection area

: At the end of brick stairs, there will be two doors. one is a door to private room that SKKU student and graduates can have private meeting and the other one is connected to public room that all SKKU people can communicate each other.

- **Essential asset**

: Traditional door, modern door

- **User flow**

1) when user get to the end of entrance, he/she can find two doors.

2) when user open the left door (traditional one), user will go to private room.

3) when user open the right door (modern one), user will go to public room.

4.2 Private Area

Area that SKKU students and graduates can have a private meeting.

- **Concept**

Brick road entrance of SKKU Myeongyun Campus

- **Motivation**

in entrance space, it is essential to make user feel the feeling that they are entering to special space. at the same time, it is important to build familiarity because our space is not only just interesting space, but also community area for SKKU student and graduates. So, we benchmark the brick road entrance of Myeongyun for familiarity, and place some interesing features to make a special mood.

- **Outline**

- The senior's private rooms are divided into several rooms, and between these rooms, there is a doorplate with a corridor and a senior's profile written on it.

- Senior's private room has a limited number of people. If there are more listeners than the limit on the number of people in private rooms, seniors can give lectures where there is a large screen (GLS).

- Next to the senior's private room and GLS, there is a hanok concept landscape. After the presentation and conversation, you can have additional questions and private conversations here.

- **Space Drawing & Specification**

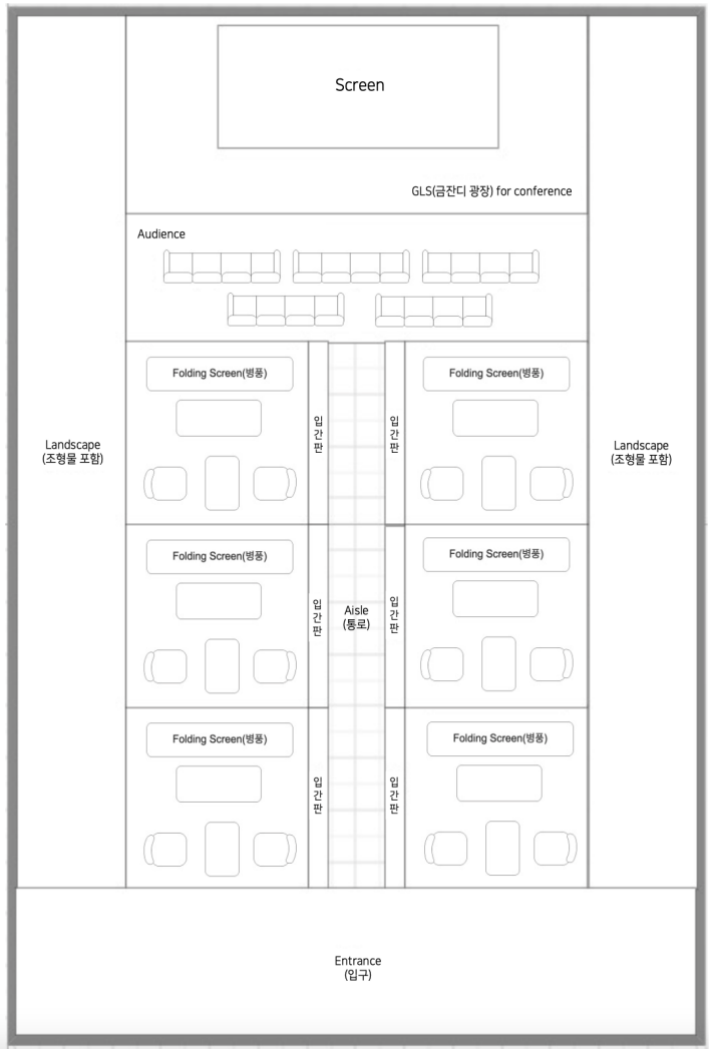


figure 21 Private area

- **Details (Space Architecture Introduction)**

a. Entrance

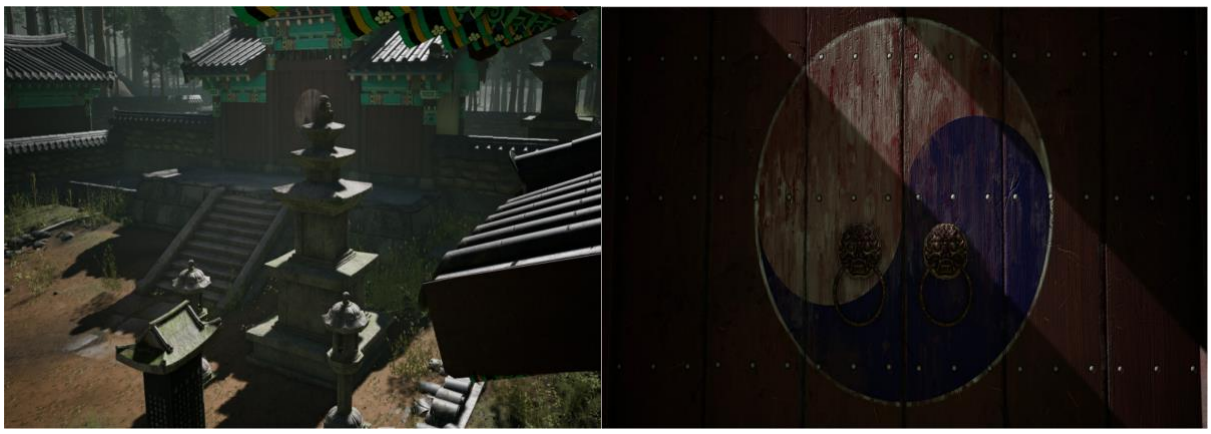


figure 22 Traditional Entrance

: Designed to actually open the entrance by arranging the asset of hanok building

- **Essential asset**

: Tile-roofed house, Main Entrance of the tile-roofed house, bottom of the soil floor.

- Additional asset

: pagoda, potted plant and tree, Jangdokdae, …

- **User flow**

1) Climb up the stairs

2) Approach the door by a certain distance, and the door asset will expand.

3) Opens the door

b. Aisle

figure 23 Traditional Aisle

: It is a space designed to run from the entrance to the GLS. This space was created between each senior's private rooms.

- **Essential asset**

: doorplate and standing signboard

- **Additional asset**

: Furniture that we can see in the hallway of Hanok like Hanok's door.

- **User flow**

1) As you pass through the passage, see the doorplate and standing signboard containing the senior's explanation

2-1) Enter the private room of the senior I want to hear.

2-2) Pass through senior's private rooms and go to the GLS & Audience Spectator.

c. Senior's room

figure 24 Senior's room

: Furniture is arranged to make the most of the feeling of hanok. The folding screen behind the senior's seat may become a screen sharing monitor or a board for writing in the future.

- **Essential asset**

: folding screen, sitting desk, sitting cushion

- Additional asset

: Traditional Korean-style room furniture like Jagaejang.

- **User flow**

1) Open the door and come in and sit in the seat you want.

2) In the case of seniors, they explain using a folding screen.

d. GLS & Audience sector

figure 25 GLS & audience sector

: The motif was GLS at Sungkyunkwan University's Myeongnyun Campus. It can be used if there are more listeners than the limited number of seniors' private rooms. Lectures can be given using the screen in the square. The Audience sector is designed so that characters can sit and watch conference with a sofa like a movie theater.

- **Essential asset**

: large screen, sofa like a movie theater,

- **Additional asset**

: lighting, …

- **User flow**

1) In the case of seniors, presentation is made using the screen.

2) In the case of juniors, sit in the audience sector and listen to the presentation.

e. Landscape

figure 26 Traditional Landscape

: Considering the geographical conditions of Myeongryun Campus and the hanok concept, we plan to deploy assets in this area that can graphically strengthen our concept through traditional sculptures.

- **Essential asset**

: potted plant and tree, gazebo,

- **Additional asset**

: pond, well, …

- **User flow**

1) Present and listen while enjoying the landscape in your senior's private room, andGLS.

4.3 Public Area

Area that SKKU students and graduates can communicate each other freely.

- **Concept**

The lawn in front of the Samsung library.

- **Motivation**

Everyone can meet the senior they want! It is important to give students feelings that they have various choices to get information. The wide lawn will help students to get this impression. Students can get along with the seniors or fellow students in this free zone.

- **Outline**

- There will be wide green lawn, where students can get along with each other.

- A big bulletin board will be floating in the front. It has information of the seniors.

- Students who want a private conversation with seniors can go to the private zone through doors.

- **Space Drawing & Specification**

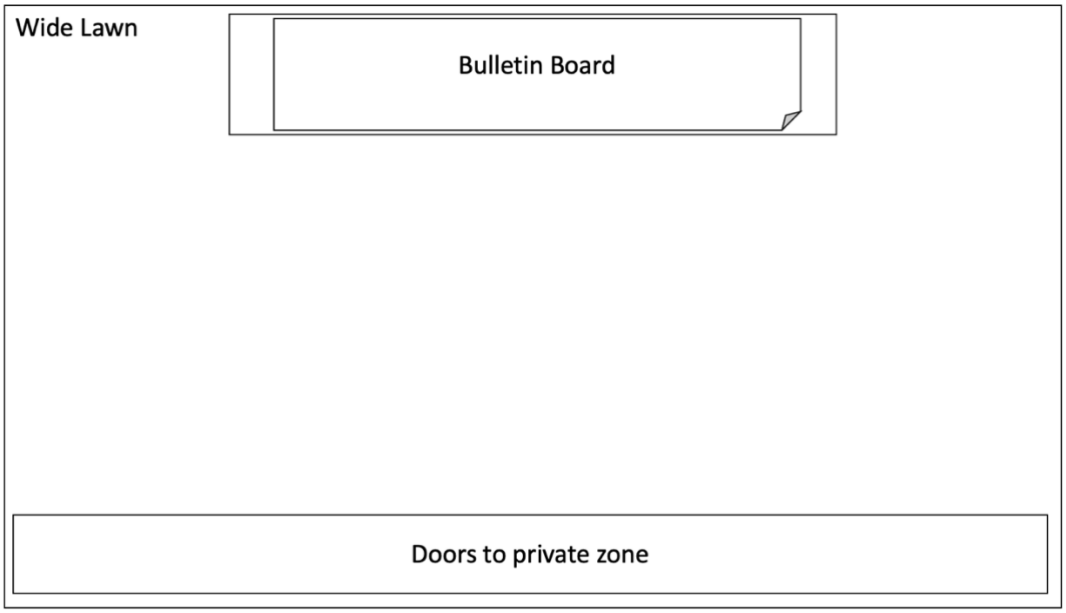


figure 27 Public Area

a. Wide lawn

: The wide lawn will imitate the lawn in the Yuljeon campus.

b. Bulletin Board (Notice board)

: Students can get information of seniors who is willing to help them. It has various information from their majors to jobs.

2) After the presentation, seniors and juniors can take a walk here and have private conversations.

c. Doors to private zone

: Students might want to have a private conversation with the seniors. Through this door, they can go to the private zone.

- **Details (Space Architecture Introduction)**

a. Wide lawn

figure 28 Wide Lawn

: From 12 AM to 3AM, and after the sunset, the landscape in the free zone will have night sightseeing of the Samsung library. In the day, it will have a bright atmosphere.

- **Essential asset**

: wide lawn, the sun, or the moon

- **User flow**

1) enter to the lawn

2) see the wide notice board

3) enter to the private zone

b. Bulletin board (Notice board)

figure 29 Bulletin Board

: Information about the seniors is written in the board. It will help students to set their questions or decide whom they want to talk to.

- **Essential asset**

: Big notice board

c. Doors to private zone

figure 30 Traditional Door

: Doors to the private zone looks traditional. It will lead the students to the private zone, where they can have personal questions, and conversation.

- **Essential asset**

: Traditional door

- **User flow**

1) whoever wants to have private conversation can access the door.

2) enter to the private zone.

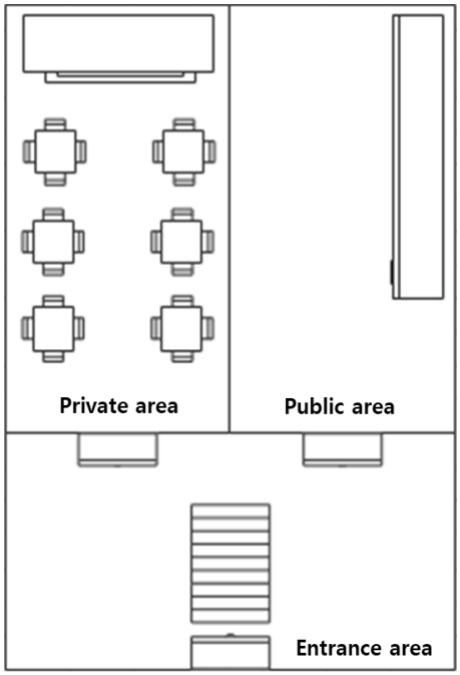
4.4 Procurement

By using Unity engine for our development tool, we can easily procure some essential assets form Unity asset store. There are free asset and non-free asset. Difference between two kinds of asset is quality. Free asset's pros are low cost of time and labor, but its quality is lower than non-free asset. So, it must be considered to choose asset type for each necessary asset.

On this respect, we will import non-free assets for essential assets and free assets for the others. Because users' satisfaction to essential is greatly depended on the quality of assets. On

the other hand, non-essential assets are on the role of essential asset supportation, paying a cost to that kind of asset is not reasonable.

4.5 The entire view



Our world is consisted of three different area. Entrance, private, and public area. Each world has its own function and objective for users

a. Entrance area

: area for user introduction. In this area, users can get a info about objective of each world and how to use it.

b. Public area

: area that SKKU student and graduates can communicate freely.

c. Private area

: area that SKKU students can have private meeting time with graduates by 1:1 or many:1.

5. System Architecture - Event

‘VRCHAT’ is a program written in unity, but it does not allow developers to C#. Instead, they create and provide a node-type programming language called 'Udon'.

VRChat Udon is a programming language built completely in-house by the VRChat Development Team. It is designed to be secure, performant, and easy to use via the VRChat Udon Node Graph, a built-in visual programming interface that uses nodes and wires (we call them “noodles”) to connect flow, inputs, and outputs. You can build complex behaviors with Udon-- far more complex and easier to understand than unwieldy chains of Triggers and

Actions.

Not only can you replicate the full behavior of Triggers and Actions with VRChat Udon, but you can create your own behaviors, sync variables with others, interact with scenes, interact with players, and more.

In addition, Udon runs in both the VRChat client and the Unity Editor, allowing you to test and debug your creations with ease.

For the more technically inclined: VRChat Udon is a VM running bytecode compiled from Udon Assembly. You can generate Udon Assembly using the built-in VRChat Udon Node Graph UI, writing your own Udon Assembly, or even by writing your own compiler to generate Udon Assembly or bytecode programs directly.

However, in the case of vrchat platform, it is not possible to build a database or access a server, so it is necessary to implement functions using various opensources and spks within a limited environment.

5.1 Event – Matching

5.1.1 Objectives

Students find the graduates they want. Students can find the best graduates based on the information provided by the graduates in advance. In this section, user stories will be presented, tasks will be classified, and operations will be introduced through a sequence diagram

5.1.2 Function description

We have prepared a space called a plaza for graduates. In there, the graduates have registered their information about themselves (YouTube, career, etc.) in advance for students to inquire it whenever students want. And graduates are grouped by major. There, students find the seniors they need.

Here is some expectable use scenario and task for developers.

5.1.3 User Story

Users spawn at the entrance. There are several portals for each major at the entrance, so you can move to the portal that suits your major. The user moves to a place where graduates with the same major as their major are gathered by the portal. There, they search for graduates' information and find the graduates they need.

5.1.4 Tasks for developers

Task 1 - Information of graduates is provided in advance. If it is a video, ‘VRC_SyncVideoPlayer’ is used, otherwise, ‘VRC_Panorama’ is used after converting to an image format. The character that represents the graduates is chosen by them. They are grouped by their major and use assets such as signposts so that users can easily distinguish them.

Task 2 - Users spawn at the entrance. At the entrance, there are installed portals divided by major. The portal name is the major of each group in the graduates, so users can easily identify it. Among the attributes of the portal, 'room id' allows you to move to the corresponding coordinates when you enter a destination within the same world. The attribute 'room id' has the coordinates of the place where graduates of each major are gathered.

5.1.5 Flow chart

The given chart is showing the flow of the task. It starts with user’s trying to find the portal. If the room name is not the same as user’s major, that means the portal is useless.

If the portal is found, the user is sent to the destination pointed by the attribute ‘rood id.’ This will only be used if the World parameter is set to "None".

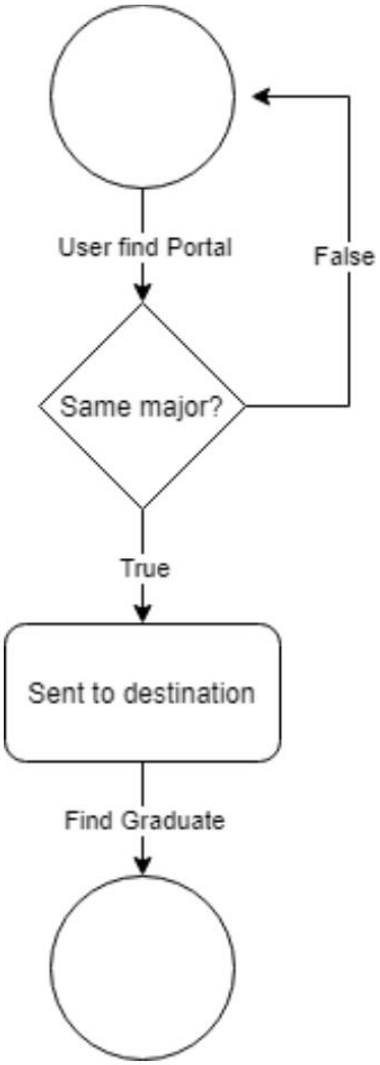


figure 31 Flow chart

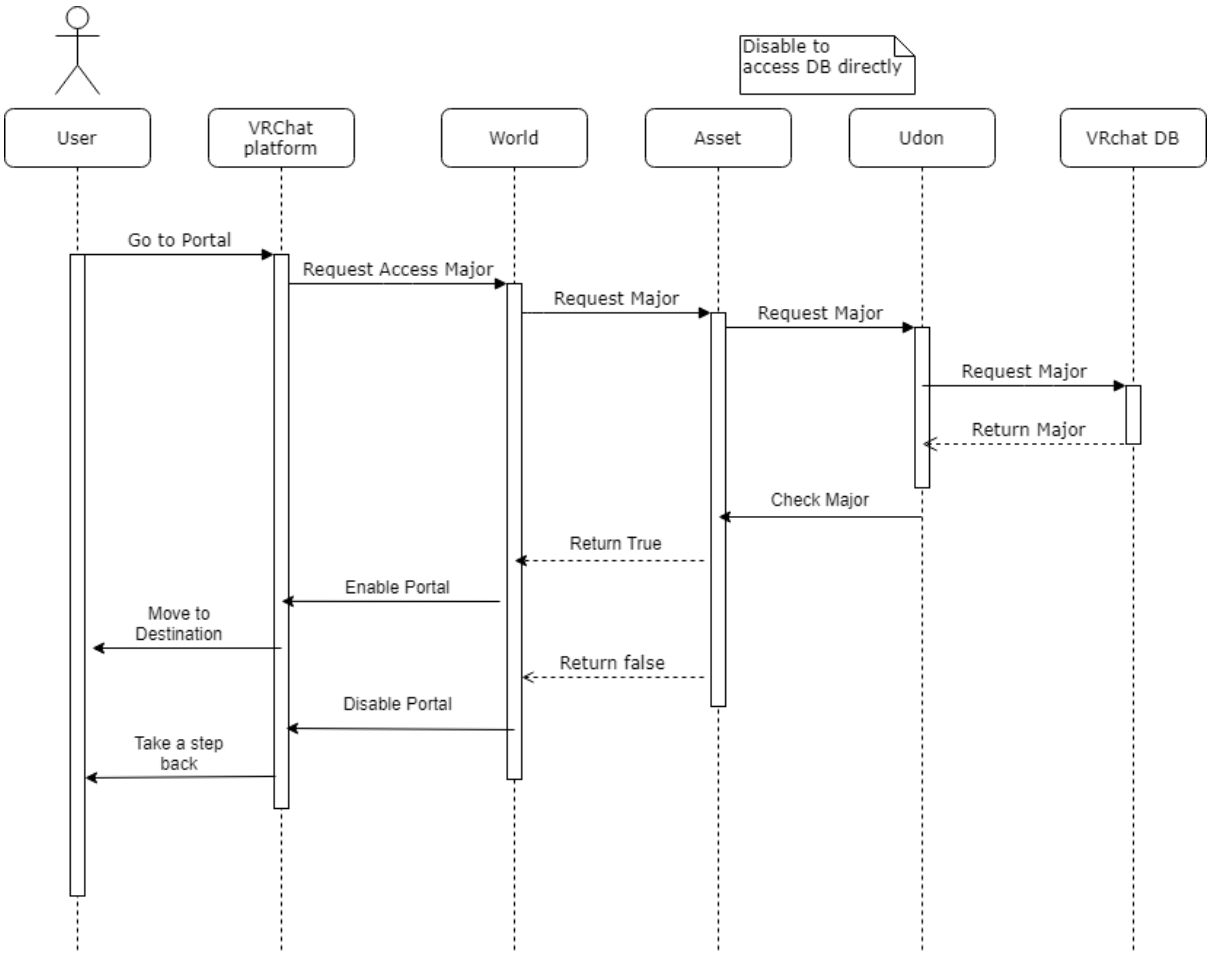


figure 32 Design process

5.2 Event – Entrance

5.2.1 Objective

The space for real-time communication provides an isolated personal place and the ability to play visual materials. In this section, user stories will be presented, tasks will be classified, and operations will be introduced through a sequence diagram.

5.2.2 Function description

A button is provided to notify outside personnel that a private room and a room are in use. By pressing the button, the person using the private room can notify that the private room is in use and block outside personnel from entering. Inside the room, there is a video player and a panorama for presentations. Using these, the registered video and PPT can be played.

5.2.3 User Story

The user story when entering the room is as follows.

The user is about to enter a private room. The user tries to enter the room, but is not entered and sees a message stating that the room is in use. It finds a room that is not in use, moves to it, and presses the usage status button to indicate that the room is in use. When the conversation is over, press the button again to make it unused and leave the room.

5.2.4 Tasks for developers

Task1 - Check the door in use mark or check the message that appears when entry is denied and find an empty room

Task2 - Restrict the access of people by pressing the usage status button inside the room.

Task3 - Press the usage status button again to enable entry and exit

5.2.5 Flow chart

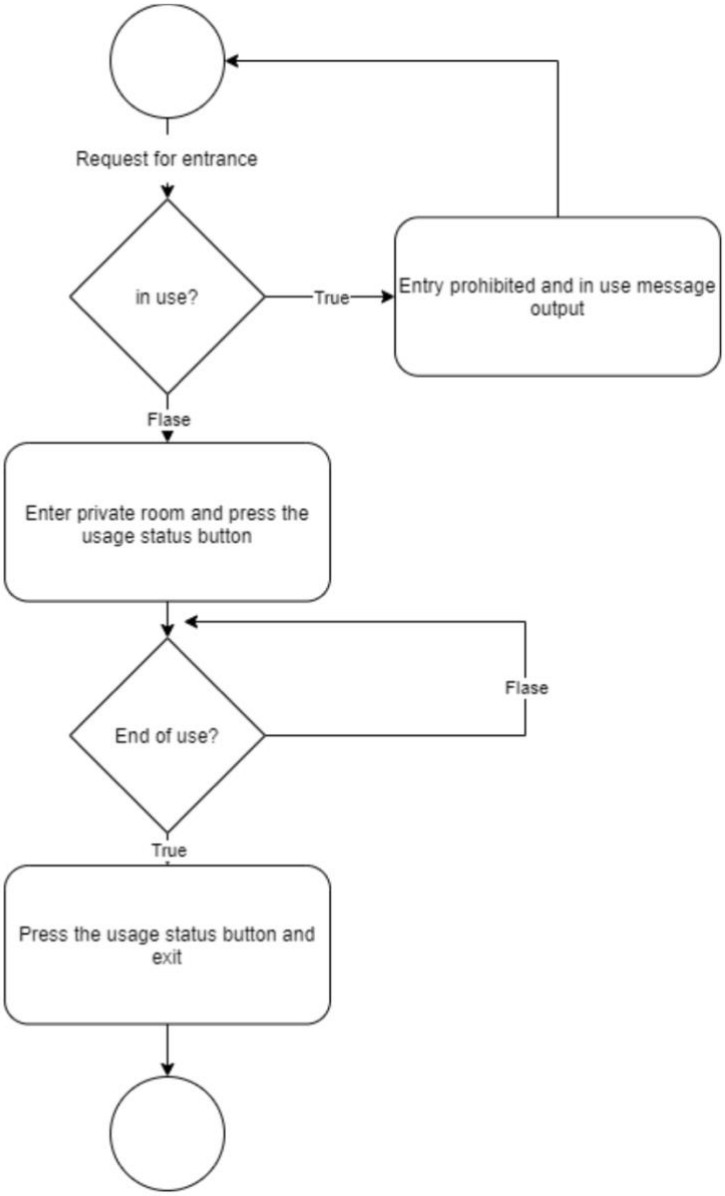


figure 33 Event driven diagram when entering a room

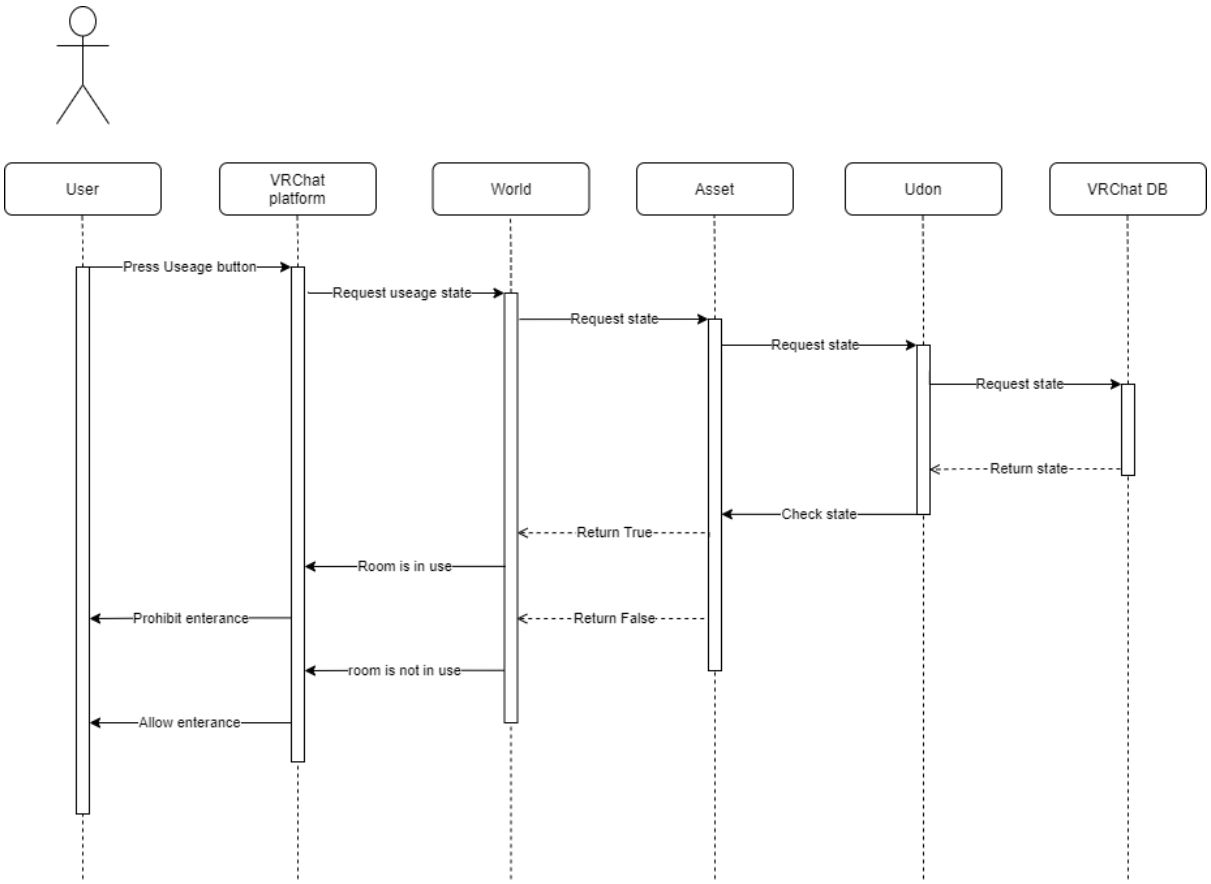


figure 34 Design process

5.2.6 Visual material

A user story when presenting a visual material is as follows.

When the user wants to play the video, the prepared video is played using the control panel. A control panel is used to stop and play the video, 5 seconds forward, 5 seconds backward functions.

When the user wants to use the PPT, the presentation is made using a panorama connected to Google drive. Turn the pages through the control panel.

The above story can be expressed as two tasks.

Task1 - Control the video via the control panel

Task2 - Proceed PPT by operating the control panel

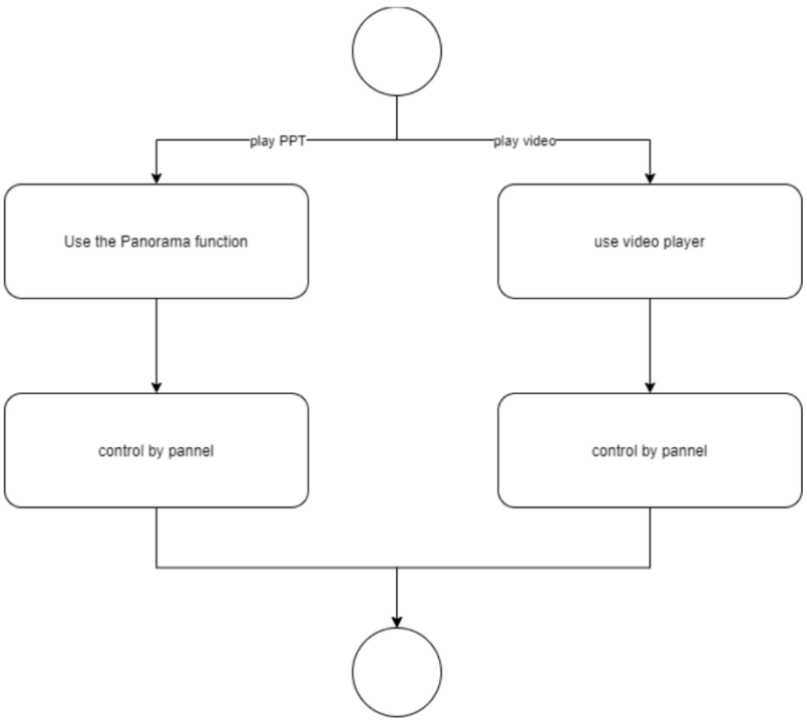


figure 35 Visual material event driven diagram

5.3 Event – Appointment and Counseling

5.3.1 Objective

In this section describes how to make an appointment with students and graduates.

5.3.2 Function description

We have a space to make communication and appointment. Graduates exist in the proper room so that they can make a communication with students. Communication is hold through vocal communication. They can talk each other as if they are in the same place. Here is some expectable use scenario and task for developers.

5.3.3 User Story

User select who the user wants to make an appointment with. User can find the list of the graduates in the virtual space. Once the user finds out who to make counseling with, user can enter the space of the graduates. As the user get in the room, the room is not available. Other users should wait for the user who have already got in there. After counseling finishes, the user came out, the room is now available. New user can get in the room, and this scenario repeats.

5.3.4 Tasks for developers

For developers, they should break down the task into pieces. Such pieces are tasks for developers. Here are some examples following.

Task 1 - Denote the room is whether it is available. When the room is available for a user, the room is denoted as an available status. Some triggers can change the status. Number of users in the room can be trigger.

Task 2 - Graduates can deactivate the room. By modifying the attribute of some asset, the graduates who counsel in the room can make a trigger manually. The status is saved as the attributes of the asset.

Task 3 - By using attribute of the asset, we can decide to activate or deactivate the portal for the room. Once the portal activated, user can enter the counseling room, while user can’t do with deactivated one. Activation and Deactivation can be triggered by manually. Both statuses can control “Entrance” of the room while “Exiting” the room is always possible.

5.3.5 Flow chart

The given chart is showing the flow of the task. It starts with user’s trying to enter the room. If the room is not available, that means the room is still not available. If the room is available, we should check the graduates available.

If the graduates are not available, the room is still blocked and deactivated. If the graduates

are available, that means the room is available, so the user can get in.

For every case, if the user gets out of the room, then the room is now available. Setting the attribute avail, the portal is activated.

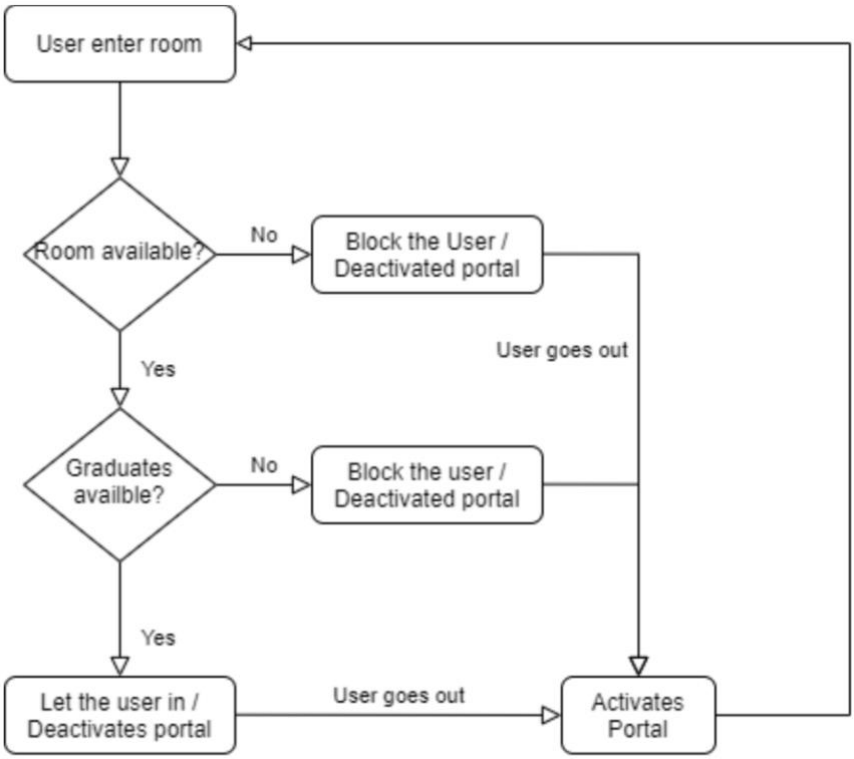


figure 36 Event driven diagram when entering room

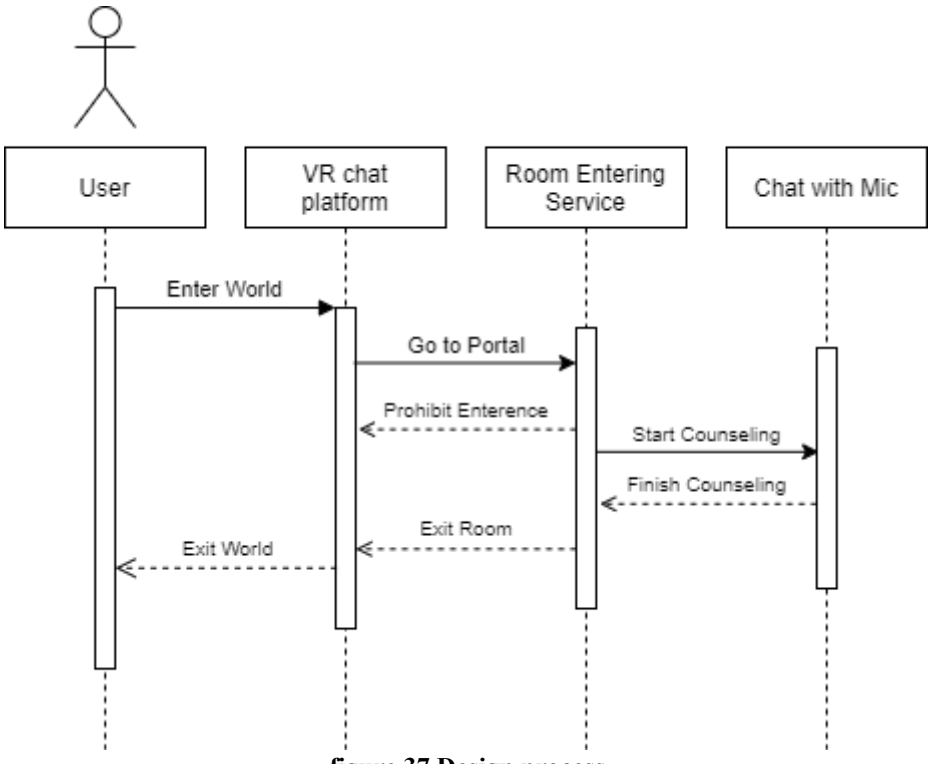


figure 37 Design process

6. Protocol Design

Originally, this chapter describes what structures are used for the protocols which are used for interaction between subsystem, application and server. However, based on VRCHAT platform, it is impossible to use the protocol.

Instead, it has been replaced with a brief description of the protocol definition and example.

6.1 Definition

Protocol is a term used by particular object-oriented programming languages with a variety of specific meanings, which other languages may term interface or trait.

Protocol when used otherwise is akin to a communication protocol, indicating the chain of interactions between the caller and the object.

Languages which use the term Protocol include: Clojure, Elixir, Java 8, Logtalk, Objective-C, Swift, Python. In these languages, a protocol is a common means for discrete objects to communicate with each other. These are definitions of methods and values which the objects agree upon, in order to co-operate, as part of an API.

The protocol/interface is a description of :

1. The messages that are understood by the object.
2. The arguments that these messages may be supplied with.
3. The types of results that these messages return.
4. The invariants that are preserved despite modifications to the state of an object
5. The exceptional situations that will be required to be handled by clients to the object.

   (For the communications-style usage only:)
6. The call sequence and decision points of the methods, such as would be represented in

UML interaction diagrams: Communication diagram, Sequence diagram, Interaction overview diagram/Activity diagram, Timing diagram.

If the objects are fully encapsulated then the protocol will describe the only way in which objects may be accessed by other objects. For example, in Java interfaces, the Comparable interface specifies a method compareTo() which implementing classes must implement. This means that a separate sorting method, for example, can sort any object which implements the Comparable interface, without having to know anything about the inner nature of the class (except that two of these objects can be compared by means of compareTo()).

Some programming languages provide explicit language support for protocols/interfaces (Ada, C#, D, Dart, Delphi, Go, Java, Logtalk, Object Pascal, Objective-C, PHP, Racket, Seed7, Swift). In C++ interfaces are known as abstract base classes, and are implemented using pure virtual functions. The object-oriented features in Perl also support interfaces.

6.2 JSON – Protocol Example

figure 38 JSON

JSON (JavaScript Object Notation) is an open standard file format and data interchange format that uses human-readable text to store and transmit data objects consisting of attribute–value pairs and arrays (or other serializable values). It is a common data format with diverse uses in electronic data interchange, including that of web applications with servers.

JSON is a language-independent data format. It was derived from JavaScript, but many modern programming languages include code to generate and parse JSON-format data. JSON

filenames use the extension. json.

7. Database Design

Originally, this chapter describes the system data structures and how these are to be represented in a database. However, based on VRCHAT platform, it is impossible to use the database.

Instead, it has been replaced with a brief description of the database definition and example.

7.1 Definition

In computing, a database is an organized collection of data stored and accessed electronically from a computer system. Where databases are more complex, they are often developed using formal design and modeling techniques.

The database management system (DBMS) is the software that interacts with end users, applications, and the database itself to capture and analyze the data. The DBMS software additionally encompasses the core facilities provided to administer the database. The sum total of the database, the DBMS and the associated applications can be referred to as a "database system". Often the term "database" is also used loosely to refer to any of the DBMS, the database system or an application associated with the database.

7.2 Mysql – Database Example

figure 39 MySQL

MySQL is an open-source relational database management system (RDBMS). A relational database organizes data into one or more data tables in which data types may be related to each other; these relations help structure the data. SQL is a language programmers use to create, modify and extract data from the relational database, as well as control user access to the database. In addition to relational databases and SQL, an RDBMS like MySQL works with an operating system to implement a relational database in a computer's storage system, manages users, allows for network access, and facilitates testing database integrity and creation of backups.

MySQL has stand-alone clients that allow users to interact directly with a MySQL database using SQL, but more often, MySQL is used with other programs to implement applications that need relational database capability. MySQL is a component of the LAMP web application software stack (and others), which is an acronym for Linux, Apache, MySQL, Perl/PHP/Python. MySQL is used by many database-driven web applications, including Drupal, Joomla, phpBB, and WordPress. MySQL is also used by many popular websites, including Facebook, Flickr, MediaWiki, Twitter, and YouTube.

8. Testing Plan

8.1 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 product and ensure flawless operation and stable release of a product to the market and customers.

8.2. Testing

8.2.1. Development testing

Development testing includes all testing activities that are carried out by the team developing the system. And this 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. The tester of the software is usually the programmer who developed that software. Some development processes use programmer/tester pairs where each programmer has an associated tester who develops tests and assists with the testing process. For critical systems, a more for- mal process may be used, with a separate testing group within the development team. This group is responsible for developing tests and maintaining detailed records of test results.

In this phase, since the software has not undergone enough tests, it can be unstable, and components can collide with each other. Therefore, it is divided into three stages as unit testing, component testing, system testing. Development testing is primarily a defect testing process, where the aim of testing is to discover bugs in the software. It is therefore usually interleaved with debugging—the process of locating problems with the code and changing the pro- gram to fix these problems.

8.2.1.1 Unit testing

Unit testing is the process of testing program components, such as methods or object classes. Individual functions or methods are the simplest type of component. When you are testing object classes, you should design your tests to provide coverage of all of the features of the

object. This means that you should test all operations associated with the object; set and check the value of all attributes associated with the object; and put the object into all possible states. This means that you should simulate all events that cause a state change.

The following advantages can be obtained through unit testing.

- Time and cost for testing can be saved.
- When adding new functions, it can be tested quickly from time to time to time.
- Stability can be secured during refactoring.
- It can be a document about the code.

8.2.1.2 Component testing

Software components are often made up of several interacting objects. For example, in the weather station system, the reconfiguration component includes objects that deal with each aspect of the reconfiguration. You access the functionality of these objects through component interfaces. Testing composite components should therefore focus on showing that the component interface or interfaces behave according to its specification. You can assume that unit tests on the individual objects within the component have been completed.

8.2.1.3 System testing

System testing during development involves integrating components to create a version of the system and then testing the integrated system. System testing checks that components are compatible, interact correctly, and transfer the right data at the right time across their interfaces. It obviously overlaps with component testing, but there are important differences.

System testing should focus on testing the interactions between the components and objects that make up a system. You may also test reusable components or systems to check that they work as expected when they are integrated with new components. This interaction testing

should discover those component bugs that are only revealed when a component is used by other components in the system. Interaction testing also helps find misunderstandings, made by component developers, about other components in the system.

Because of its focus on interactions, use case-based testing is an effective approach to system testing. Several components or objects normally implement each use case in the system. Testing the use case forces these interactions to occur. If you have developed a sequence diagram to model the use case implementation, you can see the objects or components that are involved in the interaction.

8.2.2 Release Testing

Release testing is the process of testing a particular release of a system that is intended for use outside of the development team. Normally, the system release is for customers and users. In a complex project, however, the release could be for other teams that are developing related systems. For software products, the release could be for product management who then prepare it for sale.

The primary goal of the release testing process is to convince the supplier of the system that it is good enough for use. If so, it can be released as a product or delivered to the customer. Release testing, therefore, has to show that the system delivers its specified functionality, performance, and dependability, and that it does not fail during normal use.

Release testing is usually a black-box testing process whereby tests are derived from the system specification. The system is treated as a black box whose behavior can only be determined by studying its inputs and the related outputs. Another name for this is functional testing, so-called because the tester is only concerned with functionality and not the implementation of the software.

8.2.3 User Testing

User or customer testing is a stage in the testing process in which users or customers provide input and advice on system testing. This may involve formally testing a system that has been commissioned from an external supplier. Alternatively, it may be an informal process where users experiment with a new software product to see if they like it and to check that it does what they need. User testing is essential, even when comprehensive system and release testing have been carried out. Influences from the user's working environment can have a major effect on the reliability, performance, usability, and robustness of a system.

It is practically impossible for a system developer to replicate the system's working environment, as tests in the developer's environment are inevitably artificial. For example, a system that is intended for use in a hospital is used in a clinical environment where other things are going on, such as patient emergencies and conversations with relatives. These all affect the use of a system, but developers cannot include them in their testing environment.

8.2.4 Testing Case

Testing case would be set according to 3 fundamental aspects of the application function(interaction), performance, and security. We would set 4 test cases from each aspect (12 cases in total) and proceed testing on application and would make an evaluation sheet

9. Development Plan

9.1 Objective

The Development Plan describes the development schedule of the project. The overall development sequence is easily viewed through the Gantt chart.

9.2 Gantt Chart

#	Contents		9/27~	10/4~	10/11~	10/18~	10/25~	11/1~	11/8~	11/15~
1	Requirement Specification		■	■	■					
2	Design					■	■			
3	Imple-ment	Component				■	■	■		
4		Integration					■	■	■	
5	Code Review								■	
6	Testing								■	■
7	Final Presentation									■

table 2 Gantt Chart

The developments so far are as follows.

Through specification, the goal, scope, expected users, development tools, development language, platform, and terminology of the project were defined, and the concept and procurement method of assets were determined. And the functions to be provided are defined and the operation is expressed as a diagram.

10. Supporting Information

10.1 Document History

Date	Description	Writer
2021/11/8	Addition of 3	정채원, 박신현
2021/11/9	Addition of 1,2	모보현
2021/11/10	Addition of 8	조하영
2021/11/11	Addition of 6,7	박지열
2021/11/11	Addition of 9, 10	곽무진
2021/11/13	Addition of 4	조하영, 엄소정, 정채원, 박신현
2021/11/15	Addition of 5	박지열, 곽무진, 모보현
2021/11/17	Revision of 4	조하영, 엄소정, 정채원, 박신현
2021/11/18	Revision of 5	박지열, 곽무진, 모보현
2021/11/19	Integration and Add Table of Contents	엄소정

table 3 Documentation History