Laboratory Safety Education System

Software Design Specification

2021.11.13.

Introduction to Software Engineering 41

TEAM 5

Team Leader 한영진

Team Member 강승목

Team Member 김재윤

Team Member 김진성

Team Member 김예준

Team Member 손병호

CONTENTS

1. Preface 10

1.1. Readership 10
1.2. Scope 10
1.3. Objective 10
1.4. Document Structure 11

2. Introduction 12

2.1. Objectives 12
2.2. Applied Diagrams 12
2.2.1. UML 12
2.2.2. Use case Diagram 13
2.2.3. Sequence Diagram 13
2.2.4. Class Diagram 13
2.2.5. Context Diagram 14
2.3. Applied Tools 14
2.4.1 Diagrams.net(draw.io) 14
2.4.1 Microsoft PowerPoint 15
2.4. Project Scope. 15
2.5. References 15

3. System Architecture – Overall 16

3.1. Objectives 16
3.2. System Organization 16
3.2.1. Context Diagram 16
3.2.2. Sequence Diagram 17
3.2.3. Use Case Diagram 18

4. System Architecture – VRChat World 19

4.1. Objectives 19
4.2. 공통 Subcomponents 19
4.2.1. User 19
4.2.1.1. Attributes 19
4.2.1.2. Methods 19
4.2.1.3. Class Diagram 20
4.2.1.4. Sequence Diagram 21

4.2.2. Fire 21
4.2.2.1. Attributes 21
4.2.2.2. Methods 22
4.2.2.3. Class Diagram 22
4.2.2.4. Sequence Diagram 22
4.2.3. FireExtinguisher 23
4.2.3.1. Attributes 23
4.2.3.2. Methods 23
4.2.3.3. Class Diagram 23
4.2.3.4. Sequence Diagram 24
4.2.4. EvacuationMap 24
4.2.4.1. Attributes 24
4.2.4.2. Methods 24
4.2.4.3. Class Diagram 24
4.2.4.4. Sequence Diagram 25
4.2.5. Obstacle 25
4.2.5.1. Attributes 25
4.2.5.2. Methods 25
4.2.5.3. Class Diagram 26
4.2.5.4. Sequence Diagram 26
4.2.6. Assistant 26
4.2.6.1. Attributes 27
4.2.6.2. Methods 27
4.2.6.3. Class Diagram 27
4.2.6.4. Sequence Diagram 27
4.2.7. PPE(Personal Protective Equipment) 28
4.2.7.1. Attributes 28
4.2.7.2. Methods 28
4.2.7.3. Class Diagram 28
4.2.7.4. Sequence Diagram 29
4.3. 화학 실험실 Subcomponents 29
4.3.1. ChemistryUser 29
4.3.1.1. Attributes 29
4.3.1.2. Methods 30
4.3.1.3. Class Diagram 30
4.3.1.4. Sequence Diagram 31
4.3.2. ChemistryPPE 31
4.3.2.1. Attributes 31

4.3.2.2. Methods 31
4.3.2.3. Class Diagram 32
4.3.2.4. Sequence Diagram 32
4.3.3. WashingTool 33
4.3.3.1. Attributes 33
4.3.3.2. Methods 33
4.3.3.3. Class Diagram 33
4.3.3.4. Sequence Diagram 33
4.3.4. Beaker 34
4.3.4.1. Attributes 34
4.3.4.2. Methods 34
4.3.4.3. Class Diagram 34
4.3.4.4. Sequence Diagram 35
4.3.5. LiquidWaste 35
4.3.5.1. Attributes 35
4.3.5.2. Methods 35
4.3.5.3. Class Diagram 35
4.3.5.4. Sequence Diagram 36
4.3.6. Sink 36
4.3.6.1. Attributes 36
4.3.6.2. Methods 36
4.3.6.3. Class Diagram 37
4.3.6.4. Sequence Diagram 37
4.4. 전기 실험실 Subcomponents 37
4.4.1. ElectricityUser 37
4.4.1.1. Attributes 38
4.4.1.2. Methods 38
4.4.1.3. Class Diagram 38
4.4.1.4. Sequence Diagram 39
4.4.2. Multitap 40
4.4.2.1. Attributes 40
4.4.2.2. Methods 41
4.4.2.3. Class Diagram 41
4.4.2.4. Sequence Diagram 41
4.4.3. AED 42
4.4.3.1. Attributes 42
4.4.3.2. Methods 42

4.4.3.3. Class Diagram 42
4.4.3.4. Sequence Diagram 43
4.4.4. Insulator 44
4.4.4.1. Attributes 44
4.4.4.2. Methods 44
4.4.4.3. Class Diagram 44
4.4.4.4. Sequence Diagram 45
4.4.5. InsulatingGloves 45
4.4.5.1. Attributes 45
4.4.5.2. Methods 45
4.4.5.3. Class Diagram 46
4.4.5.4. Sequence Diagram 46
4.4.6. DirectCurrentGenerator 47
4.4.6.1. Attributes 47
4.4.6.2. Methods 47
4.4.6.3. Class Diagram 48
4.4.6.4. Sequence Diagram 48
4.4.7. Breadboard 49
4.4.7.1. Attributes 49
4.4.7.2. Methods 50
4.4.7.3. Class Diagram 50
4.4.7.4. Sequence Diagram 50
4.4.8. Resistor 51
4.4.8.1. Attributes 51
4.4.8.2. Methods 51
4.4.8.3. Class Diagram 51
4.4.8.4. Sequence Diagram 52
4.4.9. ConductingWire 52
4.4.9.1. Attributes 52
4.4.9.2. Methods 52
4.4.9.3. Class Diagram 53
4.4.9.4. Sequence Diagram 53
4.4.10. Voltmeter 54
4.4.10.1. Attributes 54
4.4.10.2. Methods 54
4.4.10.3. Class Diagram 54
4.4.10.4. Sequence Diagram 55

5. System Architecture – Backend 56

5.1. Objectives 56
5.2. Overall Architecture 56
5.3. Subcomponents 57
5.3.1. Cloud Function 57
5.3.1.1. Endpoint Handler Class 57
5.3.1.2. FirebaseUI 57
5.3.1.3. DB_Handler Class 57
5.3.2. VRChat World Management System 58
5.3.2.1. Class Diagram 58
5.3.2.2. Sequence Diagram 59
5.3.3. VRChat World Allocation System 60
5.3.3.1. Class Diagram 60
5.3.3.2. Sequence Diagram 61
5.3.4. Course Tracking System 62
5.3.4.1. Class Diagram 62
5.3.4.2. Sequence Diagram 63

6. Protocol Design....64

6.1. Objectives 64
6.2. Interaction between User and Back-end Server 64
6.2.1. Register 64
6.2.2. Log In 65
6.2.3. Get Profile 65
6.3. Interaction between VRChat and Back-end Server 66
6.3.1. Complete request 66

7. Database Design 68

7.1. Objectives 68
7.2. ER Diagram 68
7.2.1. Entities 68
7.2.1.1. User data 68
7.2.1.2. VRChat 69
7.3. JSON 69
7.3.1. Objective 69

8. Testing Plan 71

8.1. Objectives 71
8.2. Testing Policy 71

8.2.1. Development Testing 71
8.2.1.1. Usability 71
8.2.1.2. Reliability 72
8.2.1.3. Security 72
8.2.2. Release Testing 72
8.2.3. User Testing 72
8.2.4. Testing Case 73

9. Development Plan 73

9.1. Objectives 73
9.2. Frontend Environment 73
9.2.1. Unity 73
9.2.2. VRChat 74
9.2.3. Udon 74
9.3. Backend Environment 75
9.3.1. Github 75
9.3.2. Node.js + Express.js 75
9.3.3. Firebase 76
9.4. Constraints 77
9.5. Assumptions and Dependencies 78

10. Supporting Information 78

10.1. Software Design Specification 78
10.2. Document History 78

LIST OF FIGURES

Figure 1 Context Diagram - Overall 16
Figure 2 Sequence Diagram - Overall 17
Figure 3 Use Case Diagram 18
Figure 4 User Class Diagram 20
Figure 5 User Sequence Diagram 21
Figure 6 Fire Class Diagram 22
Figure 7 Fire Sequence Diagram 22
Figure 8 FireExtinguisher Class Diagram 23
Figure 9 FireExtinguisher Sequence Diagram 24
Figure 10 Evacuation Class Diagram 24
Figure 11 Evacuation Map Sequence Diagram 25
Figure 12 Obstacle Class Diagram 26
Figure 13 Obstacle Sequence Diagram 26
Figure 14 Assistant Class Diagram 27
Figure 15 Assistant Sequence Diagram 27
Figure 16 PPE Class Diagram 28
Figure 17 PPE Sequence Diagram 29
Figure 18 ChemistryUser Class Diagram 30
Figure 19 ChemistryUser Sequence Diagram 31
Figure 20 ChemistryPPE Class Diagram 32
Figure 21 ChemistryPPE Class Diagram 32
Figure 22 WashingTool Class Diagram 33
Figure 23 WashingTool Sequence Diagram 33
Figure 24 Beaker Class Diagram 34
Figure 25 Beaker Sequence Diagram 35
Figure 26 LiquidWaste Class Diagram 35
Figure 27 LiquidWaste Sequence Diagram 36
Figure 28 Sink Class Diagram 37
Figure 29 Sink Sequence Diagram 37
Figure 30 ElectricityUser Class Diagram 39
Figure 31 ElectricityUser Sequence Diagram 40
Figure 32 Multitap Class Diagram 41
Figure 33 Multitap Sequence Diagram 42
Figure 34 AED Class Diagram 43
Figure 35 AED Sequence Diagram 43
Figure 36 Insulator Class Diagram 44
Figure 37 Insulator Sequence Diagram 45
Figure 38 InsulatingGloves Class Diagram 46
Figure 39 InsulatingGloves Sequence Diagram 47
Figure 40 DirectCurrentGenerator Class Diagram 48
Figure 41 DirectCurrentGenerator Sequence Diagram 49
Figure 42 Breadboard Class Diagram 50
Figure 43 Breadboard Sequence Diagram 50
Figure 44 Resistor Class Diagram 51
Figure 45 Resistor Sequence Diagram 52
Figure 46 ConductingWire Class Diagram 53

Figure 47 ConductingWire Sequence Diagram 53
Figure 48 Voltmeter Class Diagram 54
Figure 49 Voltmeter Sequence Diagram 55
Figure 50 Overall Architecture - Backend 56
Figure 51 Cloud Function - Backend 57
Figure 52 VRChat World Management System - Class Diagram 58
Figure 53 VRChat World Management System - Sequence Diagram 59
Figure 54 VRChat World Allocation System - Class Diagram 60
Figure 55 VRChat World Allocation System - Sequence Diagram 61
Figure 56 Course Tracking System - Class Diagram 62
Figure 57 Course Tracking System - Sequence Diagram 63
Figure 58 ER Diagram - User data 68
Figure 59 ER Diagram - VRChar 69
Figure 60 JSON - Format 70
Figure 61 Unity - Logo 73
Figure 62 VRChat - Logo 74
Figure 63 Github - Logo 75
Figure 64 Node.js - Logo 75
Figure 65 Express.js - Logo 76
Figure 66 Firebase - Logo 76

Table 1 Table of register request 64
Table 2 Table of register response 64
Table 3 Table of log-in request 65
Table 4 Table of log-in response 65
Table 5 Table of get profile request 65
Table 6 Table of get profile response 66
Table 7 Table of complete request 66
Table 8 Table of complete response 66
Table 9 Document History 78

1. Preface

이 장에는 Laboratory Safety Education System용 소프트웨어 설계 문서의 구독자 정보, 범위, 목표 및 문서 구조가 포함되어 있습니다.

1.1. Readership

이 문서는 각각 하위 부문이 있는 10개의 부문으로 구성되어 있습니다. Document Structure는 부문 1.4에서 확인할 수 있습니다. 이 문서의 메인 독자는 5팀이지만 Introduction to Software Engineering의 구성원들도 읽을 수 있습니다.

1.2. Scope

이 문서는 VRChat으로 안전교육을 이수하는 시스템을 구현하는 데 사용되는 설계에 대한 설명을 제공하는 데 사용됩니다.

1.3. Objective

이 Software Design Specification 문서의 주요 목적은 Laboratory Safety Education System의 설계를 설명하는 것입니다. 이 문서에서는 software architecture 및 design decisions에 대해 설명합니다. 또한 소프트웨어 Software Requirement Specification에 언급된 기능의 구조와 설계를 사용 사례와 다이어그램과 함께 묘사합니다.

1.4. Document Structure

- **1. Preface**:
  현재 읽고 있는 장입니다. 이 장에는 본 프로젝트에 사용되는 Software Design의 readership, scope, objective 및 Document Structure에 대한 정보가 포함되어 있습니다

- **2. Introduction**:
  이 장에서는 이 프로젝트에 사용되는 Diagram과 도구에 대한 정의와 설명을 제공하며, 이 문서를 이해하는 데 필요한 참고 자료도 제공합니다.

- **3. Overall System Architecture**:
  이 장에서는 시스템 구성 방법에 대한 정보를 제공합니다.

- **4. System Architecture – VRChat World**:
  이 장에서는 VRChat World 내의 structure, behavior, 및 objcet를 정의하는 conceptual model을 제공합니다.

- **5. System Architecture - Backend**:
  이 장은 backend system의 structure, behavior, 및 views를 정의하는 conceptual model을 제공합니다.

- **6. Protocol Design**:
  이 장에서는 subsystem의 상호작용에 사용되는 protocol과 인터페이스를 정의하는 protocol의 구조에 대해 설명합니다.

- **7. Database Design**:
  이 장에서는 시스템 데이터 구조와 데이터베이스에서 데이터 구조를 표현하는 방법에 대해 설명합니다.

- **8. Testing Plan**:
  이 장에서는 development testing, release testing, 및 user testing에 대한 계획을 제공합니다.

- **9. Development Plan**:
  이 장에서는 개발 환경 및 개발 도구에 대한 정보를 제공합니다.

- **10. Supporting Information**:
  이 장에서는 Document History를 제공합니다.

2. Introduction

본 프로젝트는 실험실 안전교육을 개선하여 학습 효율성을 증가시키는 것을 목표로 하고 있습니다. 기존의 안전교육 시스템은 제3자의 입장에서 강의와 test를 수행하였기 때문에 학습자에게 피동적인 자세를 유지하게 하여 안전불감증과 시험을 통과하기 위한 공부를 하게 했습니다. 하지만 본 프로젝트에서는 test단계에서 메타버스 환경을 사용해 이번 학기에 수강 예정인 실험강의에서 일어날 수 있는 위기상황을 극복하는 방식으로 계획되어 있기 때문에 더 몰입해서 학습을 진행할 수 있다는 기대효과를 가지고 있습니다.

본 문서에서는 다음과 같은 내용에 대한 설명을 담고 있습니다. 사용자의 학습 진척도를 서버와 통신 저장하고 평가하여 이용자의 안전교육 이수 여부를 확인, 인증서를 발급하는 기능의 서버에 대한 설명이 들어가 있습니다. 그리고 test 진행을 위해 위기상황을 발생시키거나 위기상황을 극복하는데 필요한 object들에 대한 설명이 들어가 있습니다.

2.1. Objectives

이 장은 설계 단계에서 구체화된 본 프로젝트에 사용될 도구와 구조를 설명하는 다이어그램에 대한 설명을 담고 있습니다.

2.2. Applied Diagrams

2.2.1. UML

UML (Unified Modeling Language)을 통해 해당 프로젝트의 모델링, 및 소프트웨어 분석을 실행하였습니다. UML은 프로젝트를 모델링 하는 단계에서 통일성을 위해 정의된 언어 규격입니다. UML을 사용해 모델링한 자료를 통해 stakeholder와 원활한 소통에 사용할 수 있을 뿐만 아니라 실제 개발에 참여하는 인원들 간에도 프로젝트를 가시화하고 진행에 앞서 문제점을 미리 파악할 수 있는 등의 장점이 존재합니다. UML은 많은 개발 방식 그리고 implementation과 호환되어 사용할 수 있습니다. 본 프로젝트에서는 UML을 기반으로 작성하되 domain에 생소한 독자를 위해 도식화

위주로 문서를 작성하였습니다.

2.2.2. Use case Diagram

어떠한 방식으로 사용자와 시스템이 interaction하여 사용자의 요구사항을 충족시키는지는 시스템의 기능을 설명할 때 가장 중요한 부분입니다. use case diagram은 시스템이 user의 high-level 요구사항을 충족시키는지 여부를 판단하는데 사용됩니다. 이러한 요구사항은 크게 3가지로 나눠 분석할 수 있습니다. 첫째 functional requirements에서는 use case의 경우들이 나열됩니다. 둘째 actor에서는 actor가 어떠한 방식으로 시스템과 상호작용하는지가 나열됩니다. Actor는 사용자, 조직, 혹은 다른 프로그램이 될 수 있습니다. 셋째 actor와 use case사이의 관계에서는 화살표를 사용하여 requirement를 달성하기 위해actor가 어떠한 방식으로 system과 상호작용하는지를 나타냅니다.

2.2.3. Sequence Diagram

Sequence diagram은 프로그래밍 이외의 다른 산업의 system개발 과정에 있어서도 많이 사용되는 도식입니다. Sequence diagram에서는 actor와 object간에 발생하는 메시지 및 상호작용을 sequential하게 담고 있습니다. Actor와 object는 묘사에 필요한 부분만 따와 Sequence diagram에 도식화될 수 있습니다. 따라서 sequence diagram에 보이는 actor나 object 부분이 전체를 설명한다고 할 수 없습니다. Actor나 object자체에 집중하기 보다 그 상호작용에 집중해서 살펴보아야 하는 diagram입니다. Sequence diagram에 나타나는 actor, object 상호작용(통신)은 시간순서대로 표현됩니다.

2.2.4. Class Diagram

Class Diagram은 시스템에 사용되는 object class(클래스)와 class를 구성하는 attributes(요소), operations(연산/상호작용), method(메서드) 들의 관계를 표시하며 시스템 구조를 설명하는 도식입니다. OOP적 관점에서 class를 통해 시스템에 사용되는 object를 구현하게 되므로 class별로 class diagram이 묘사되게 됩니다. Class diagram을 통해 프로그래밍될 클래스의 종류, 각 class에 포함될 object들에 대한 정보, 그리고

class-class 혹은 class내에서 attributes간의 상호작용을 visualization할 수 있습니다. Class diagram의 형식은 다음과 같습니다. 각 class는 3개의 행으로 이루어져 있습니다. 가장 위에 class의 name정보를 담고 있습니다. 가운데에는 class의 attributes를 담고 있습니다. 가장 아래에는 class가 사용할 수 있는 method와 operation을 담고 있습니다.

2.2.5. Context Diagram

Context diagram에서는 가장 high-level의 data flow를 표현합니다. 표현하고자 하는 system에서 진행되는 하나의 process에 집중해 그 과정을 묘사하는 방식을 통해 시스템을 표현하게 됩니다. 표현하고자 하는 system과 사용자 사이의 정보 흐름에 집중해 시스템을 묘사합니다. Context diagram을 작성할 때는 시스템 요구사항에 따라 외부에서 상호작용을 시도하는 경우에 발생하는 event와 접근에 집중해서 작성해야 합니다. System context diagram은 system과 상호작용할 수 있는 모든 외부 entities를 포함해야 하며, 시스템 자체는 세분화되지 않은 하나의 개체로 표현되어야 합니다.

2.3. Applied Tools

아래에는 보고서 작성에 사용된 외부 자료들에 대한 내용이 있습니다.

2.4.1 Diagrams.net(draw.io)

Draw.io는 그 자리에서 다이어그램을 만들고 싶은 사람이라면 누구나 쉽게 사용할 수 있도록 특별히 설계되었습니다. 다이어그램에 도형을 빠르게 삽입할 수 있는 다양한 도구가 제공됩니다. 이들은 타원형과 직사각형과 같은 기본 도형부터 스틱 그림과 화살표와 같은 보다 구체적인 디자인, 맞춤형 플로차트와 같은 고급 요소에 이르기까지 다양합니다. 요컨대 눈길을 사로잡는 다이어그램으로 가능한 한 빨리 아이디어를 얻으려면 여기에 도구를 사용하십시오.

2.4.1 Microsoft PowerPoint

해당도구는 그림과 텍스트 작성을 지원하는 도구입니다. 다양한 도형, 도표, 표, 그래프 등을 간단하게 그릴 수 있습니다. 다양한 워드 에디터와 호환이 되며 텍스트 포맷팅을 지원함으로 범용성 또한 뛰어납니다.

2.4. Project Scope.

메타버스 안전교육 test system은 기존 안전교육 학습 시스템의 문제점이었던 “시험을 통과하기 위한 공부”와 “안전불감증” 그리고 앞의 이유로 발생하는 학습 능률 저하에 대한 문제를 해결하기 위해 만들어졌습니다. 사용자들은 메타버스 환경속에서 자신을 대변하는 캐릭터에 몰입해 실제 실험실에서 벌어질 수 있는 위기상황을 극복하며 test를 하게 됩니다. 이런 새로운 방식은 향상된 몰입감을 통한 안전불감증 해소와, 학습효과 향상을 목표로 하고 있습니다.

본 system은 독립적인 시나리오 형식을 가지고 있기 때문에 새로운 시나리오를 추가하기 용이하며 이러한 확장성을 통해 화학, 전기 실험 외에도 다른 실험들을 구현하여 사용할 수 있습니다. 실험실 안전교육은 성균관대학교 뿐만 아닌 타 대학이나 산업현장에서도 진행되므로 다른 기관에서도 본 시스템을 도입해 사용할 수 있을 것이라고 생각됩니다.

2.5. References

본 문서를 작성하는데 다음 문서들의 도움이 있었습니다.

- IEEE Recommended Practice for Software Requirements Specifications
- Ian Sommerville, ‘Software Engineering 10th Edition’
- 2020 SKKUSE Team 1, ‘https://github.com/skkuse/2020spring_41class_team1’
- Welcome To UML Web Site!, ‘https://www.uml.org/’
- Node.js 공식 사이트, ‘https://nodejs.org’
- Express.js 공식 사이트, ‘https://expressjs.com/’

- Firebase 공식 사이트, ‘https://firebase.google.com/’

3. System Architecture – Overall

3.1. Objectives

이 장에서는 VRChat 내부 시스템과 백엔드 설계의 시스템 구성을 설명합니다.

3.2. System Organization

이 서비스는 VRChat 플랫폼과 이를 활용하기 위한 백엔드 서버로 구성됩니다. 사용자는 VRChat 시스템과 상호작용하며, VRChat의 데이터 정보는 HTTP 요청을 통해 백엔드 서버로 전송됩니다. 백엔드 서버는 VRChat에서 발생하는 이벤트를 처리 및 기록하며 데이터 처리를 위한 여러 내부 및 외부 API를 사용합니다.

3.2.1. Context Diagram

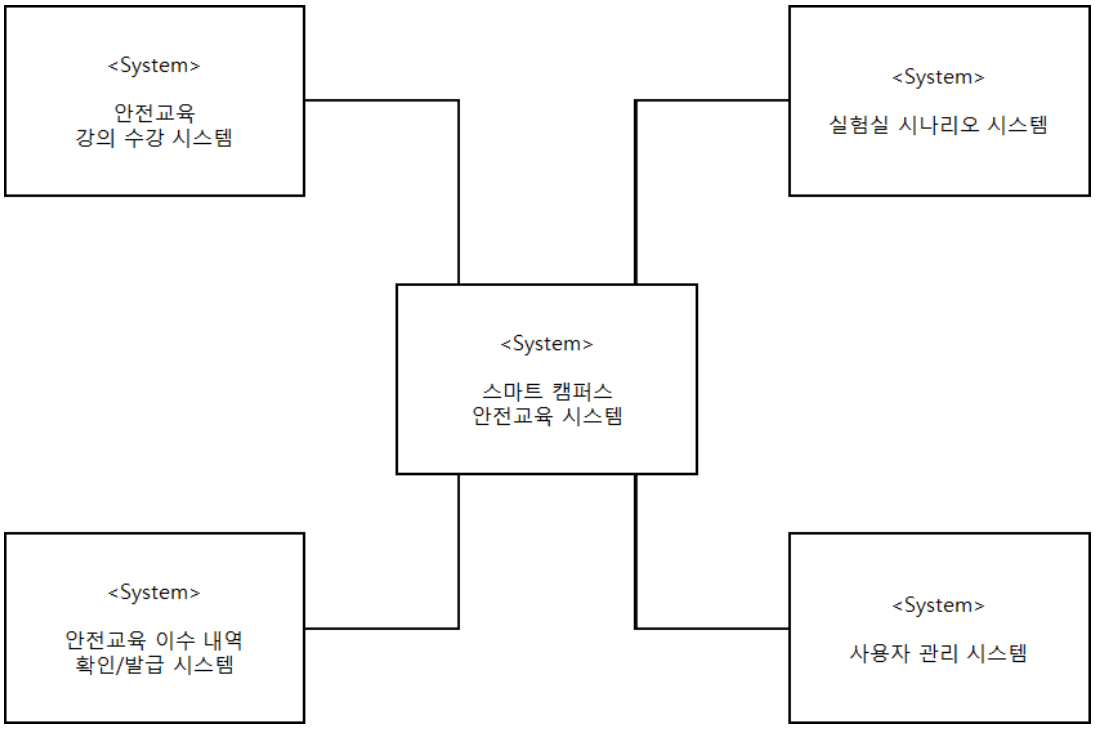

Figure 1 Context Diagram - Overall

3.2.2. Sequence Diagram

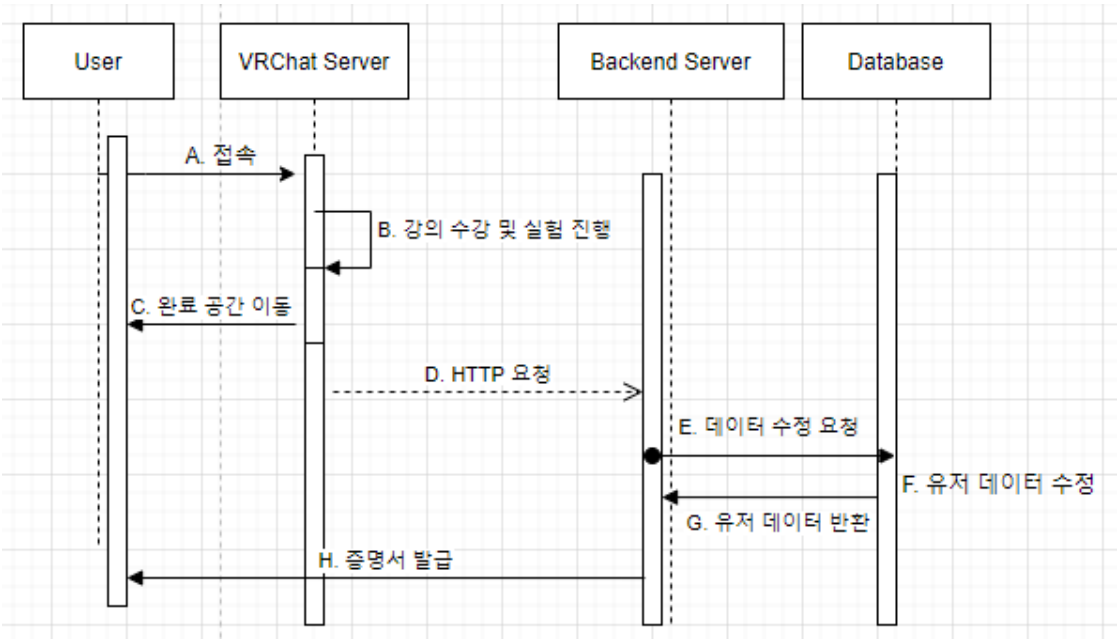

Figure 2 Sequence Diagram - Overall

3.2.3. Use Case Diagram

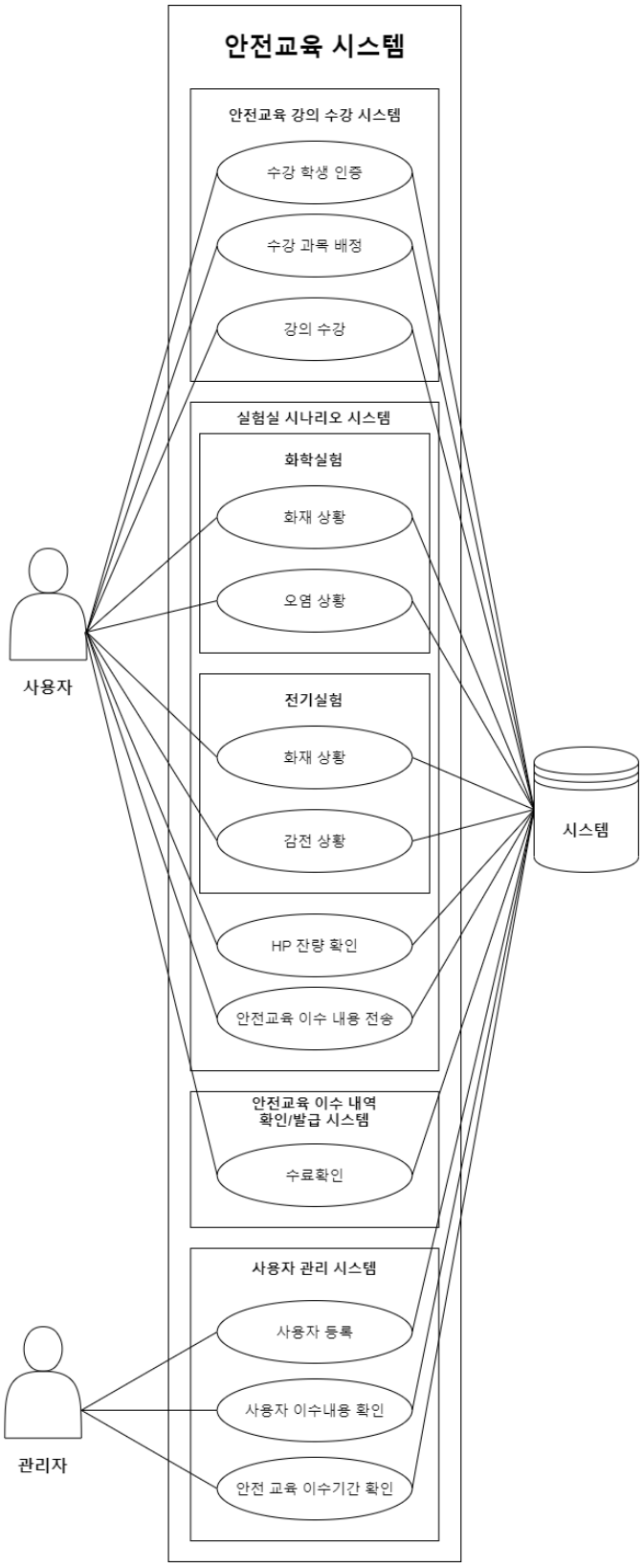

Figure 3 Use Case Diagram

4. System Architecture – VRChat World

4.1. Objectives

이 장에서는 VRChat 내부에서 구현되어야 하는 object와 그 속성, 그리고 상호작용에 대해 설명합니다.

4.2. 공통 Subcomponents

4.2.1. User

User 클래스는 사용자의 현재 체력(Hp)와 interaction 중인 object, 다른 objects로 인해 변화하는 Hp를 표시합니다.

4.2.1.1. Attributes

User 클래스가 갖고 있는 특성들은 다음과 같습니다.

- Hp: 사용자의 현재 체력
- Interaction_objects: 사용자와 현재 상호작용 중인 object(들)
- Position: 사용자의 위치
-

4.2.1.2. Methods

User 클래스가 가지는 메소드는 다음과 같습니다.

- ChangeHp: object와의 상호작용으로 인한 Hp의 변화
- Move: 사용자 캐릭터 위치 이동
- Click: 마우스 클릭했을 시
- Drag: 마우스 드래그 시
- Mousedown: 마우스 누르고 있을 시

4.2.1.3. Class Diagram

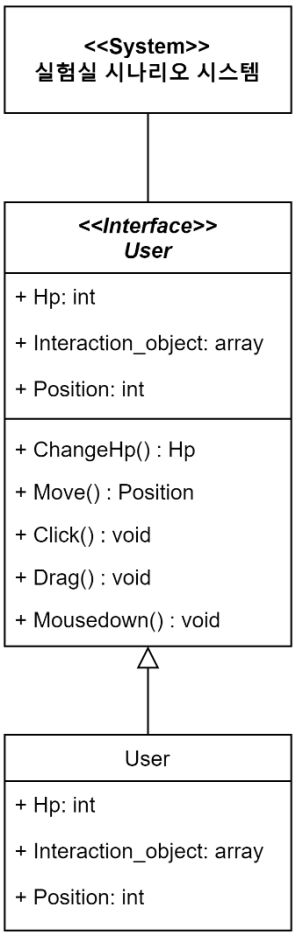

Figure 4 User Class Diagram

4.2.1.4. Sequence Diagram

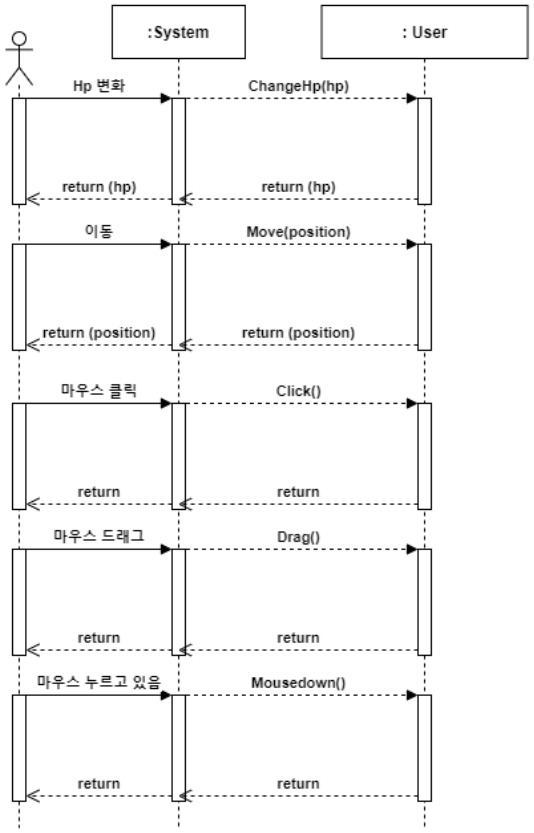

Figure 5 User Sequence Diagram

4.2.2. Fire

Fire 클래스는 화재와 관련된 요소들을 나타내는 클래스입니다. 화재는 소형, 중형, 대형의 세 가지 단계를 가지고, 소형, 중형은 사용가능한 소화기를 통해 제거할 수 있습니다. 대형은 연기를 발생시키며, 제거할 수 없습니다.

4.2.2.1. Attributes

Fire 클래스가 갖는 특성은 다음과 같습니다.

- Fire_grade: 화재의 단계. 1~60은 소형, 61~120은 중형, 121이상은 대형
- Smoke: 연기의 존재

4.2.2.2. Methods

Fire클래스가 갖는 메소드는 다음과 같습니다.

- FireGrow: 화재의 단계 값을 설정합니다.

4.2.2.3. Class Diagram

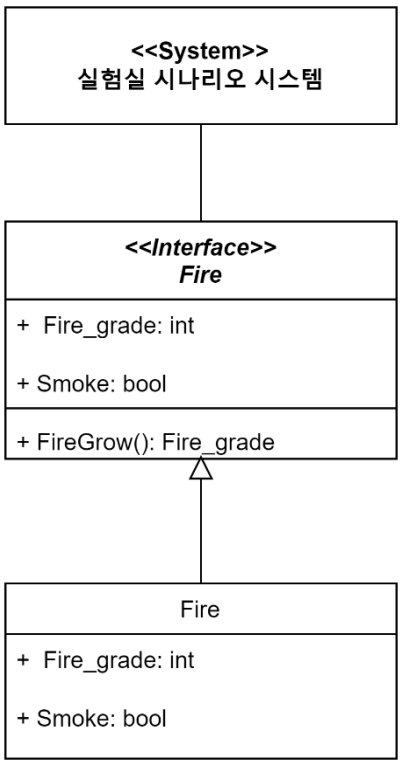

Figure 6 Fire Class Diagram

4.2.2.4. Sequence Diagram

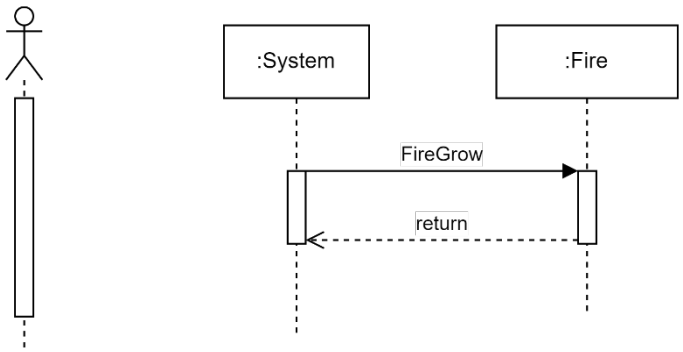

Figure 7 Fire Sequence Diagram

4.2.3. FireExtinguisher

FireExtinguisher 클래스는 소화기와 소화기의 교체, 사용을 의미하는 클래스입니다.

4.2.3.1. Attributes

FireExtinguisher 클래스가 갖는 특성은 다음과 같습니다.

- Fire_extinguihser_available: 소화기의 사용 가능 여부

4.2.3.2. Methods

FireExtinguisher클래스가 갖는 메소드는 다음과 같습니다.

- ChangeFireExtinguihser: 소화기 교체
- UsingFireExtinguisher: 소화기 사용(대형이 아닌 Fire의 Fire_grade가 초당 3씩 감소)

4.2.3.3. Class Diagram

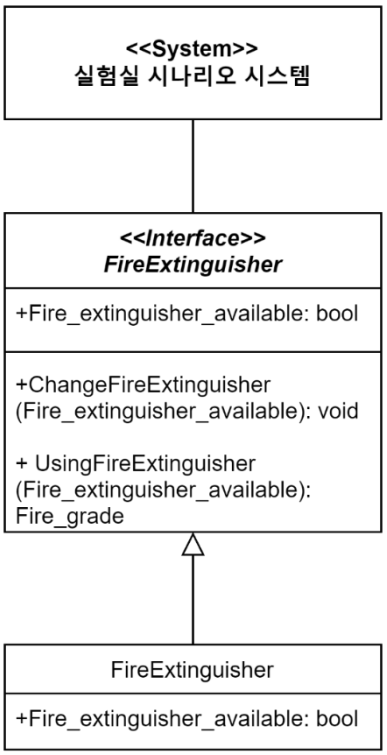

Figure 8 FireExtinguisher Class Diagram

4.2.3.4. Sequence Diagram

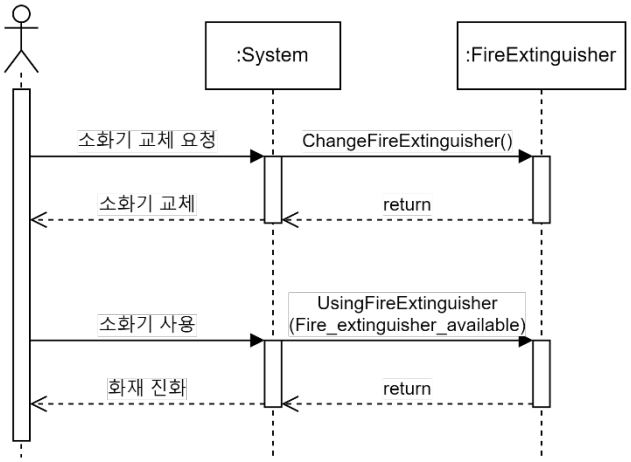

Figure 9 FireExtinguisher Sequence Diagram

4.2.4. EvacuationMap

EvacuationMap 클래스는 대피도를 보여줄 수 있도록 하는 클래스입니다.

4.2.4.1. Attributes

4.2.4.2. Methods

EvacuationMap이 갖는 메소드는 다음과 같습니다.

OpenEvacuationMap: 대피도를 펼칩니다.

4.2.4.3. Class Diagram

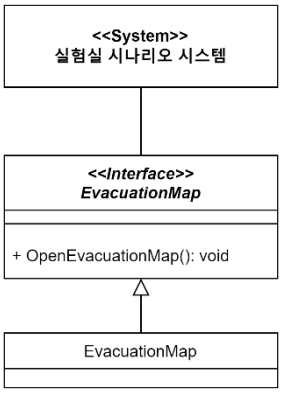

Figure 10 Evacuation Class Diagram

4.2.4.4. Sequence Diagram

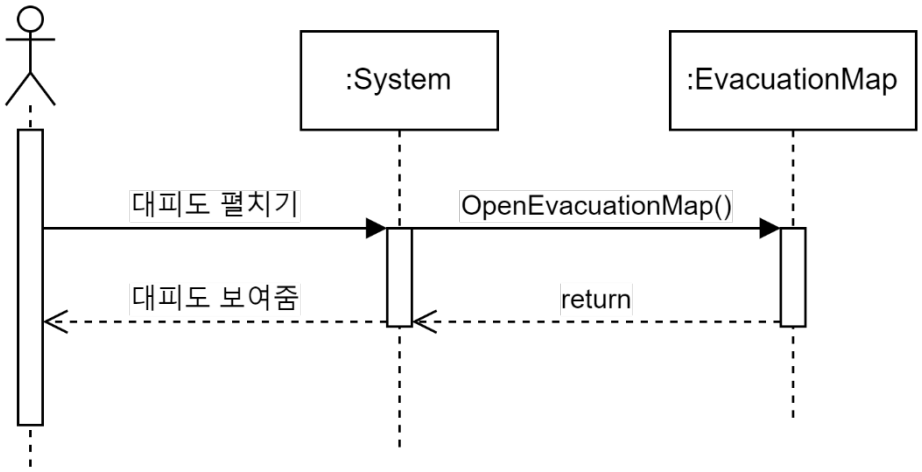

Figure 11 Evacuation Map Sequence Diagram

4.2.5. Obstacle

Obstacle 클래스는 사용자 캐릭터에게 '넘어짐'을 유발할 수 있는 장애물에 관한 클래스입니다.

4.2.5.1. Attributes

Obstacle 클래스가 갖는 특성은 다음과 같습니다.

- Obstacle_Position: 장애물의 위치를 나타냅니다.
-

4.2.5.2. Methods

Obstacle 클래스가 갖는 메소드는 다음과 같습니다.

- Arrangement: 장애물을 정리(제거)합니다.
- ObstacleBlock: 장애물의 위치로 이동한 User를 '넘어짐' 상태로 만듦니다.

4.2.5.3. Class Diagram

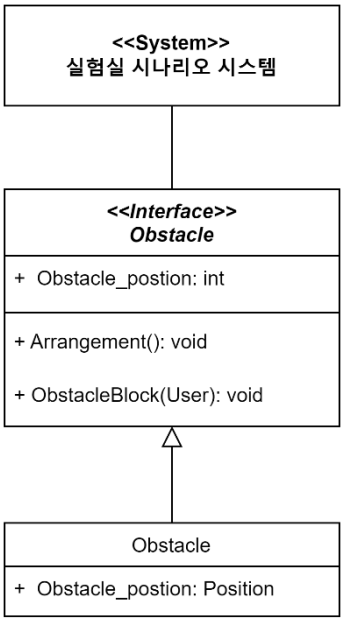

Figure 12 Obstacle Class Diagram

4.2.5.4. Sequence Diagram

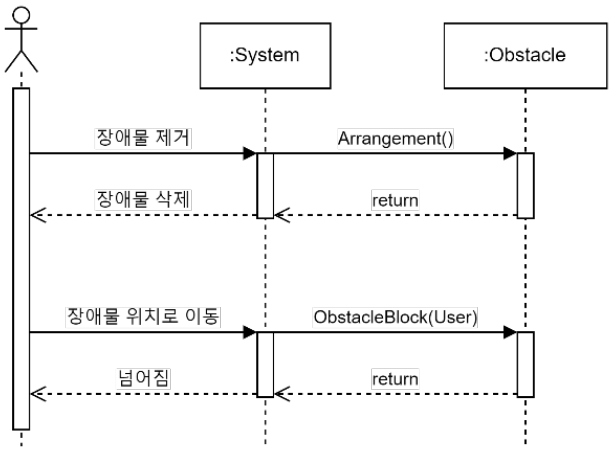

Figure 13 Obstacle Sequence Diagram

4.2.6. Assistant

Assistant 클래스는 실험 안전 수칙에 관한 설명을 영상 자료 형태로 보여주어 조교가 설명을 하는 것과 같도록 합니다.

4.2.6.1. Attributes

Assistant가 갖는 attributes는 없습니다.

4.2.6.2. Methods

Assitant가 갖는 메소드는 다음과 같습니다.

- AssitantExplain: 조교가 설명을 시작하도록 합니다.
-

4.2.6.3. Class Diagram

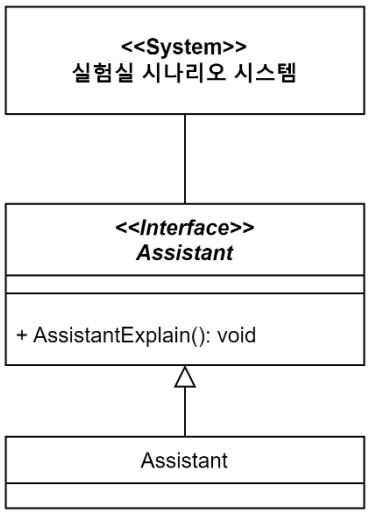

Figure 14 Assistant Class Diagram

4.2.6.4. Sequence Diagram

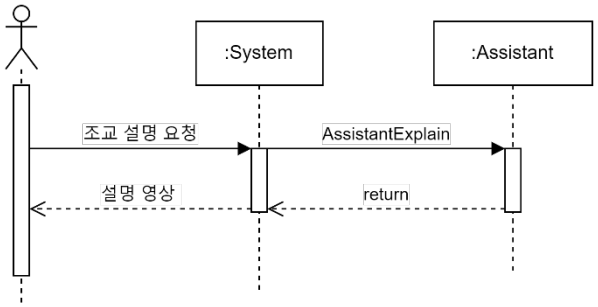

Figure 15 Assistant Sequence Diagram

4.2.7. PPE(Personal Protective Equipment)

PPE 클래스는 사용자가 착용하는 보호구를 나타내는 클래스입니다. 보호구를 착용한 채로 신체 오염, 안구 오염이 발생하면 해당하는 부위의 보호구가 대신 오염됩니다. 보호구가 오염된 상태로 5초가 지나면 다시 신체 오염, 안구 오염으로 전이됩니다.

4.2.7.1. Attributes

PPE 클래스가 갖는 특성은 다음과 같습니다.

- Is_worn: 보호구의 착용 여부
-

4.2.7.2. Methods

PPE 클래스가 갖는 메소드는 다음과 같습니다.

- SetIsWorn: 보호구를 착용합니다.
-

4.2.7.3. Class Diagram

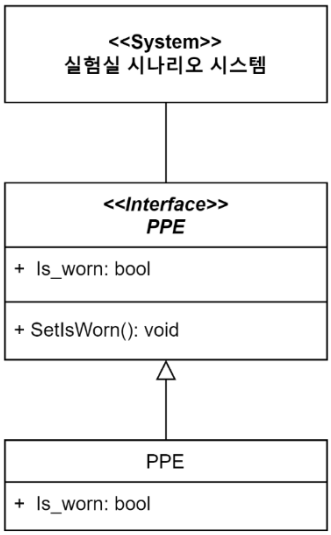

Figure 16 PPE Class Diagram

4.2.7.4. Sequence Diagram

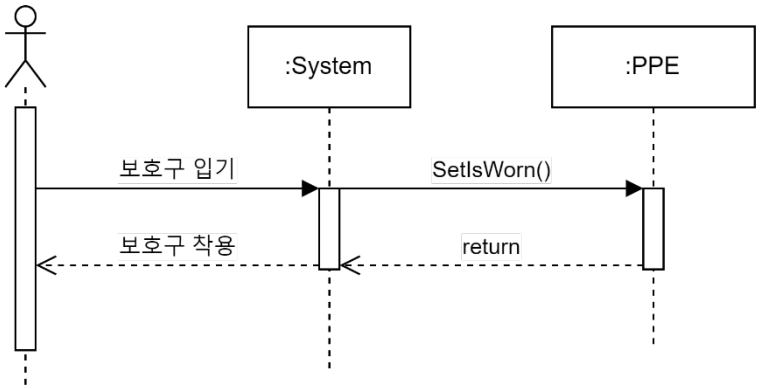

Figure 17 PPE Sequence Diagram

4.3. 화학 실험실 Subcomponents

4.3.1. ChemistryUser

ChemistryUser 클래스는 User 클래스를 상속하는 클래스입니다. 보호구를 착용한 채로 신체 오염, 안구 오염이 발생하면 해당하는 부위의 보호구가 대신 오염됩니다. 보호구가 오염된 상태로 5초가 지나면 다시 신체 오염, 안구 오염으로 전이됩니다.

4.3.1.1. Attributes

ChemistryUser 클래스가 추가로 갖는 특성은 다음과 같습니다.

- Used_beaker: 실험 중 사용한 비커의 목록
- Lab_coat: 실험복
- Lab_goggle: 보안경
-

4.3.1.2. Methods

ChemistryUser 클래스가 추가로 갖는 메소드는 다음과 같습니다.

- Pour()

4.3.1.3. Class Diagram

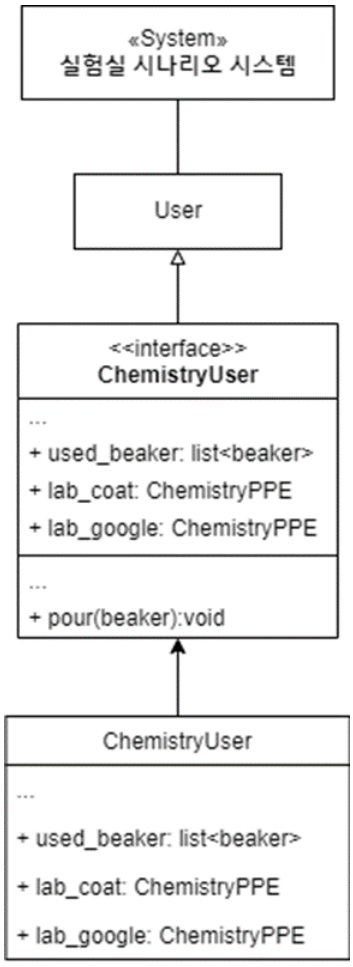

Figure 18 ChemistryUser Class Diagram

4.3.1.4. Sequence Diagram

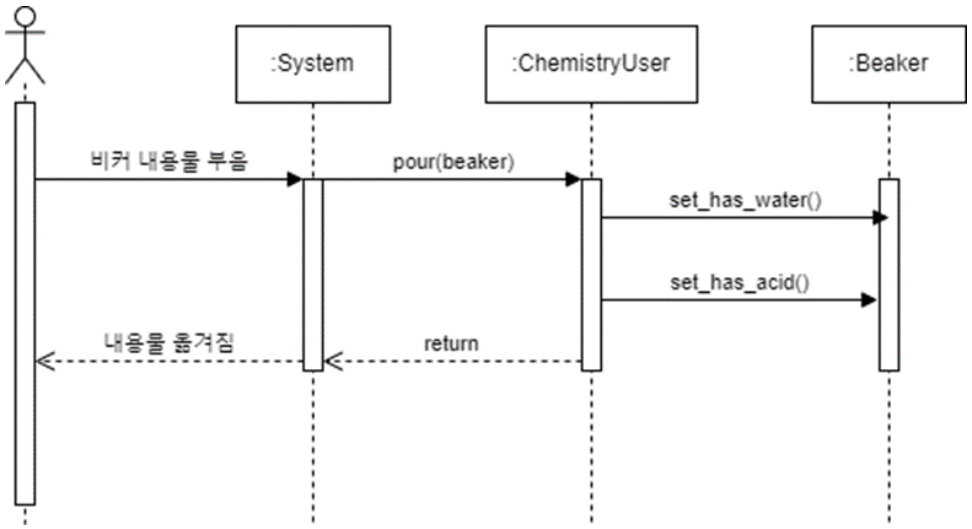

Figure 19 ChemistryUser Sequence Diagram

4.3.2. ChemistryPPE

ChemistryPPE 클래스는 화학 실험실에서 사용자가 착용하는 보호구를 나타내는 클래스입니다. 이 클래스는 실험복 혹은 보안경을 나타냅니다. 보호구를 착용한 채로 신체 오염, 안구 오염이 발생하면 해당하는 부위의 보호구가 대신 오염됩니다. 보호구가 오염된 상태로 5초가 지나면 다시 신체 오염, 안구 오염으로 전이됩니다.

4.3.2.1. Attributes

ChemistryPPE 클래스가 추가로 갖는 특성은 다음과 같습니다.

- Is_clean: 보호구의 오염 여부
- Time_dirty: 보호구가 오염된 채로 지난 시간

4.3.2.2. Methods

ChemistryPPE 클래스가 추가로 갖는 메소드는 다음과 같습니다.

- Spread_dirty()

4.3.2.3. Class Diagram

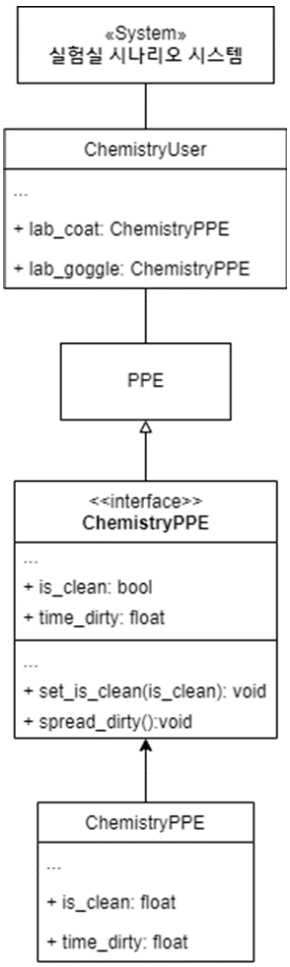

Figure 20 ChemistryPPE Class Diagram

4.3.2.4. Sequence Diagram

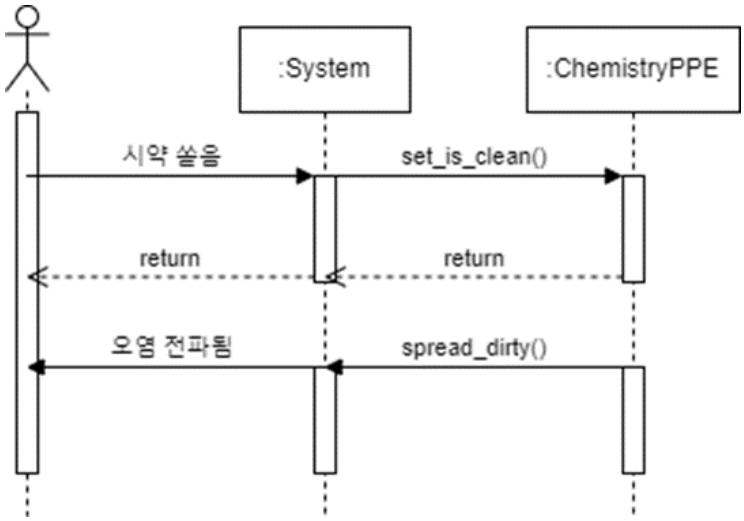

Figure 21 ChemistryPPE Class Diagram

4.3.3. WashingTool

WashingTool 클래스는 오염을 씻기 위해 사용되는 긴급 샤워, 안구 세척을 포함하는 클래스입니다.

4.3.3.1. Attributes

세척 도구 클래스가 갖는 특성은 다음과 같습니다.

- Part: 세척 부위

4.3.3.2. Methods

세척 도구 클래스가 갖는 메소드는 다음과 같습니다.

- Clean()

4.3.3.3. Class Diagram

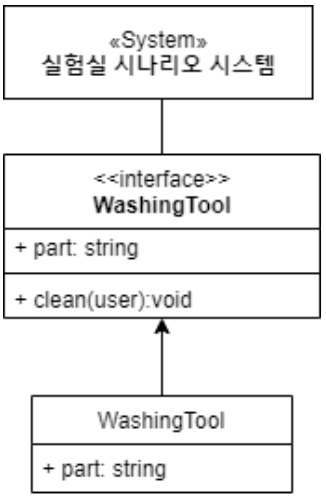

Figure 22 WashingTool Class Diagram

4.3.3.4. Sequence Diagram

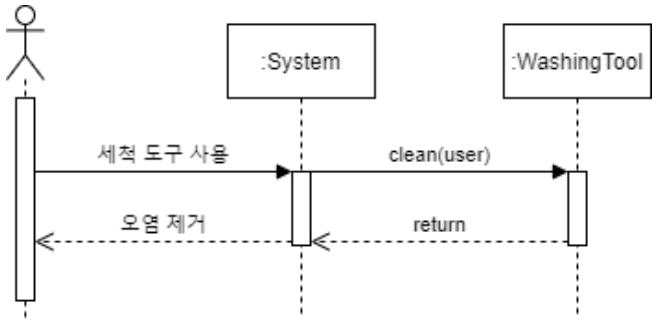

Figure 23 WashingTool Sequence Diagram

4.3.4. Beaker

Beaker 클래스는 실험 중 사용되는 비커를 나타내는 클래스입니다. 비커의 내용물을 다른 비커로 옮기거나 비커가 깨질 수 있습니다.

4.3.4.1. Attributes

Beaker 클래스가 갖는 특성은 다음과 같습니다.

- Has_water: 비커에 물이 있는지 여부
- Has_acid: 비커에 황산이 있는지 여부
- Is_used: 비커를 사용했는지 여부

4.3.4.2. Methods

Beaker 클래스가 갖는 메소드는 다음과 같습니다.

- Break()

4.3.4.3. Class Diagram

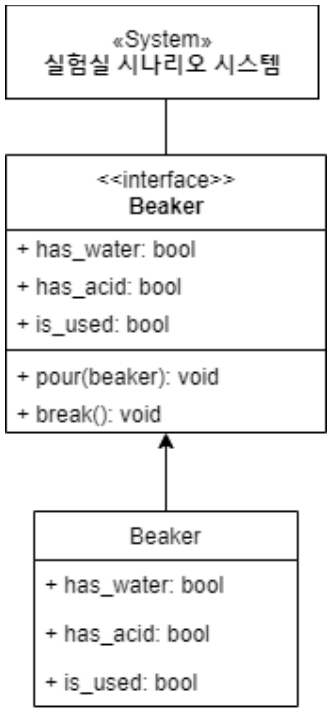

Figure 24 Beaker Class Diagram

4.3.4.4. Sequence Diagram

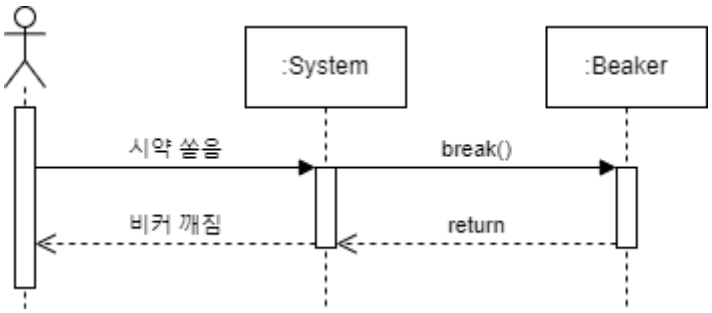

Figure 25 Beaker Sequence Diagram

4.3.5. LiquidWaste

LiquidWaste 클래스는 폐액통을 나타내는 클래스입니다. 실험 중 사용된 액체를 폐액통에 버리지 않으면 페널티가 주어집니다.

4.3.5.1. Attributes

LiquidWaste 클래스가 갖는 특성은 없습니다.

4.3.5.2. Methods

LiquidWaste클래스가 갖는 메소드는 다음과 같습니다.

- Dispose()

4.3.5.3. Class Diagram

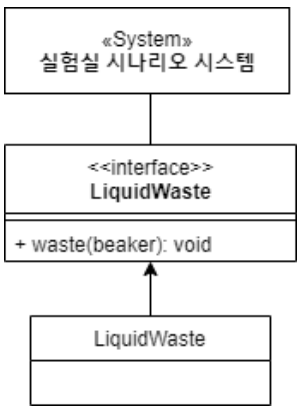

Figure 26 LiquidWaste Class Diagram

4.3.5.4. Sequence Diagram

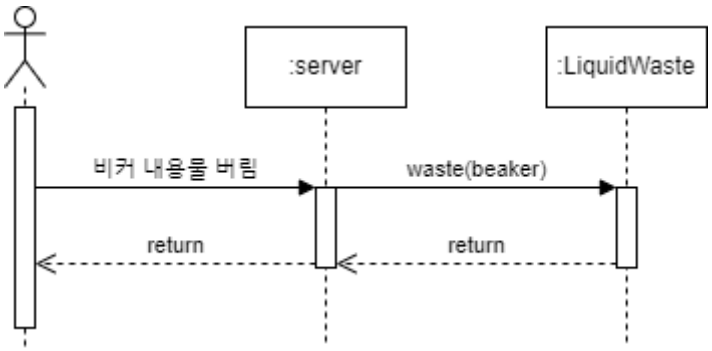

Figure 27 LiquidWaste Sequence Diagram

4.3.6. Sink

Sink 클래스는 세면대를 나타내는 클래스입니다. 실험 중 사용된 비커를 세면대에서 씻지 않으면 페널티가 주어집니다.

4.3.6.1. Attributes

Sink 클래스가 갖는 특성은 없습니다.

4.3.6.2. Methods

Sink 클래스가 갖는 메소드는 다음과 같습니다.

- Clean()

4.3.6.3. Class Diagram

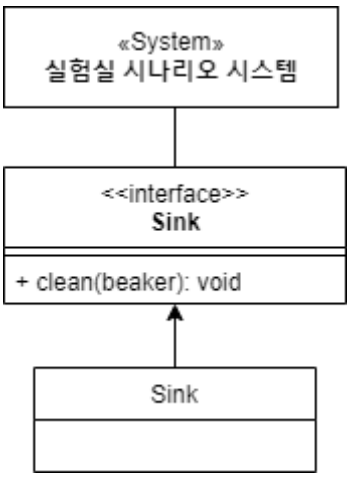

Figure 28 Sink Class Diagram

4.3.6.4. Sequence Diagram

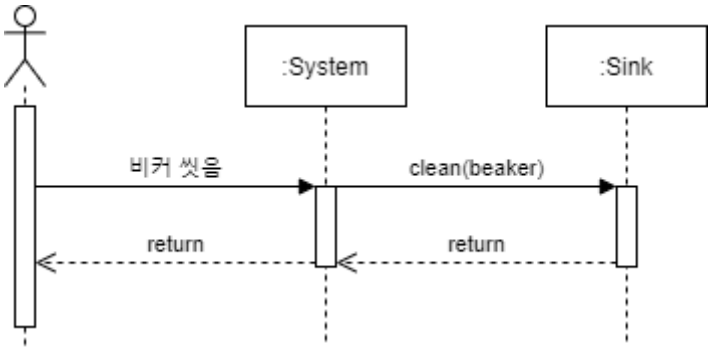

Figure 29 Sink Sequence Diagram

4.4. 전기 실험실 Subcomponents

4.4.1. ElectricityUser

전기실험을 하는 유저입니다. ElectricityUser 클래스는 사용자의 현재 체력(Hp)와 interaction 중인 object, 다른 objects로 인해 변화하는 Hp를 표시합니다.

4.4.1.1. Attributes

다음은 ElectricityUser object가 가지고 있는 변수입니다.

- int hp (사용자의 현재 체력)
- array interaction_objects (사용자와 현재 상호작용 중인 object)
- int position (사용자의 위치)
- bool electrified (감전 여부)
- bool contaminated (오염 여부)

4.4.1.2. Methods

다음은 DirectCurrentGenerator object가 가지고 있는 메소드입니다.

- ChangeHp(Hp) return (Hp)
  object와의 상호작용으로 인한 Hp의 변화
- Move(position) return (position)
  사용자 캐릭터 위치 이동
- Click() return ()
  마우스 클릭했을 시
- Drag() return ()
  마우스 드래그 시
- Mousedown() return ()
  마우스 누르고 있을 시

4.4.1.3. Class Diagram

System을 구성하는 현재 부분을 Attributes위주로 도식화

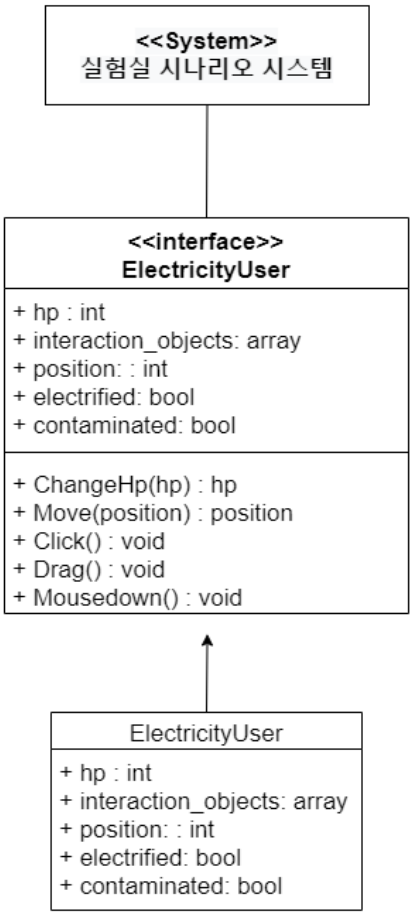

Figure 30 ElectricityUser Class Diagram

4.4.1.4. Sequence Diagram

ElectricityUser가 system의 현재 해당 부분과 소통하게 되는 절차(method 위주로)를 도식화

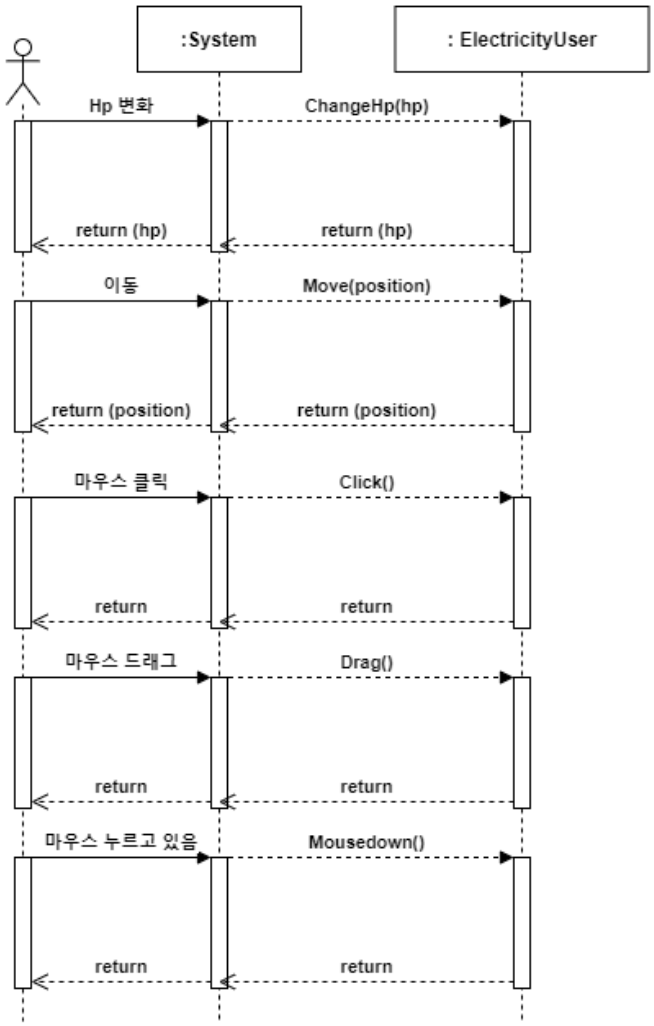

Figure 31 ElectricityUser Sequence Diagram

4.4.2. Multitap

DirectCurrentGenerator나 다른 Multitap을 연결할 수 있습니다.

4.4.2.1. Attributes

다음은 Multitap object가 가지고 있는 변수입니다.

- int connect_num (하위에 연결된 전자기기 수)
- int overload_num (과부하 수치)
- bool overload (과부하 여부)

4.4.2.2. Methods

다음은 DirectCurrentGenerator object가 가지고 있는 메소드입니다.

- connect(overload_num) return (overload)
  해당 Multitap에 연결된 하위 항목의 수가 과부하 수치를 넘긴 상태로 1분이 경과될 경우 화재가 발생하게 됩니다.
  오브젝트 상호작용이 어려울 수 있어 퀴즈로 대체할 수 있습니다.

4.4.2.3. Class Diagram

System을 구성하는 현재 부분을 Attributes위주로 도식화

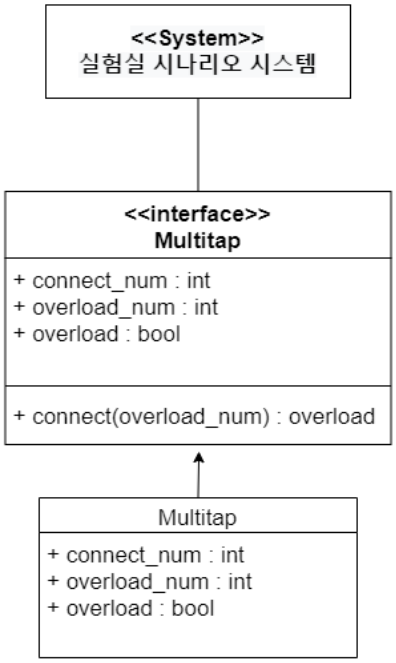

Figure 32 Multitap Class Diagram

4.4.2.4. Sequence Diagram

ElectricityUser가 system의 현재 해당 부분과 소통하게 되는 절차(method 위주로)를 도식화

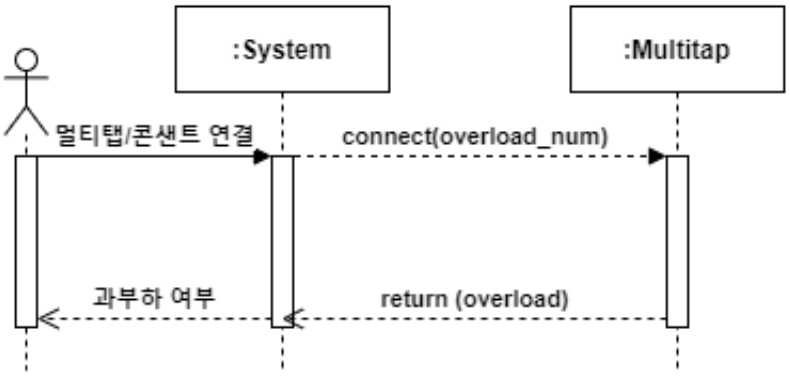

Figure 33 Multitap Sequence Diagram

4.4.3. AED

재세동을 위한 장치입니다. 심정지 상태의 사용자를 회복시키는 장비입니다.

4.4.3.1. Attributes

다음은 AED object가 가지고 있는 변수입니다.

- int q_point (퀴즈 점수)
- bool q_result (퀴즈 통과여부)

4.4.3.2. Methods

다음은 AED object가 가지고 있는 메소드입니다.

- defibrillation (electrified) return(q_result)
  제세동으로 상호작용이 어려울 수 있어서 퀴즈를 해결하는 방식으로 제세동 기능을 사용할 수 있습니다. 퀴즈를 맞추면 제세동 성공으로 간주하고 +10Hp를 얻습니다.

4.4.3.3. Class Diagram

System을 구성하는 현재 부분을 Attributes위주로 도식화

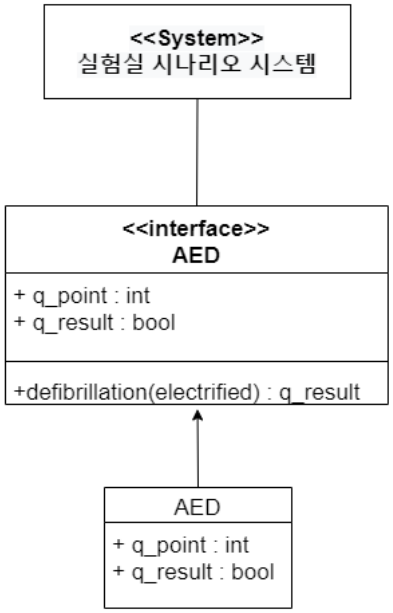

Figure 34 AED Class Diagram

4.4.3.4. Sequence Diagram

ElectricityUser가 system의 현재 해당 부분과 소통하게 되는 절차(method 위주로)를 도식화

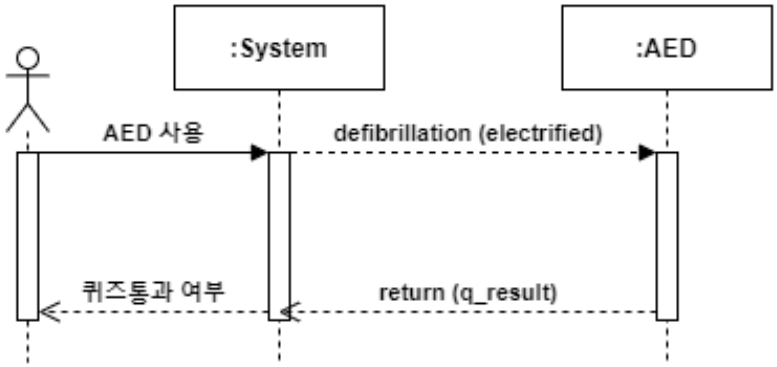

Figure 35 AED Sequence Diagram

4.4.4. Insulator

전선과 사람을 분리 조치를 할 수 있게 하는 것입니다.

4.4.4.1. Attributes

다음은 Insulator object가 가지고 있는 변수입니다.

- int q_point (퀴즈 점수)
- bool q_result (퀴즈 통과여부)

4.4.4.2. Methods

다음은 Insulator object가 가지고 있는 메소드입니다.

- separation(electrified) return(q_result)
  분리조치가 상호작용이 어려울 수 있어서 퀴즈를 해결하는 방식으로 판단합니다. 퀴즈를 맞출 시 성공으로 간주하고 +10Hp를 얻습니다.

4.4.4.3. Class Diagram

System을 구성하는 현재 부분을 Attributes위주로 도식화

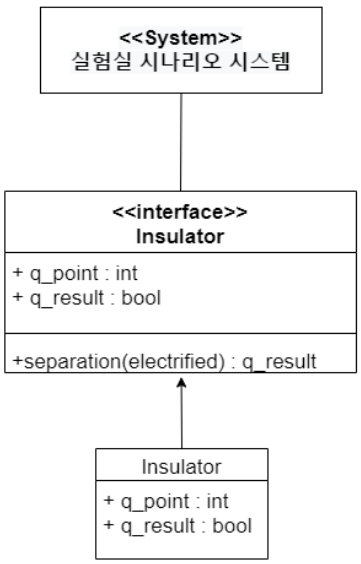

Figure 36 Insulator Class Diagram

4.4.4.4. Sequence Diagram

ElectricityUser가 system의 현재 해당 부분과 소통하게 되는 절차(method 위주로)를 도식화

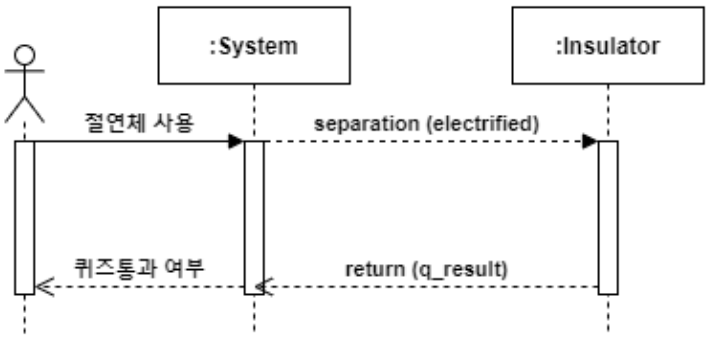

Figure 37 Insulator Sequence Diagram

4.4.5. InsulatingGloves

InsulatingGloves는 PPE 클래스를 상속하며, 전기 실험실에서 사용자가 착용하는 사용자가 감전 당하지 않게 하는 보호 장비를 나타내는 클래스입니다. 보호구를 착용한 채로 신체 오염이 발생하면 해당하는 부위의 보호구가 대신 오염됩니다. 보호구가 오염된 상태로 3초가 지나면 다시 사용자에게 오염상태가 전이됩니다.

4.4.5.1. Attributes

다음은 InsulatingGloves object가 가지고 있는 특성입니다. 추가되는 특성은 다음과 같습니다.

- bool safe (감전 / 안전 여부)

4.4.5.2. Methods

다음은 InsulatingGloves object가 가지고 있는 메소드입니다.

- putON_OFF (contaminated) return(contamination)
  보호장비에 오염이 발생하였을 경우 3초안에 "벗기"를 사용해 오염된 장

비를 벗지 않는다면 3초가 지난 뒤 사용자에게 오염상태가 전이되며 그 페널티를 받게 됩니다.

- protect (electrified) return (penalty)
  보호 장비를 "착용"중인 상태에서 "감전"상태를 받게 되었을 경우 사용자는 3초간 "감전"페널티(마비 + 도트 데미지)로부터 보호됩니다. "감전"상태가 3초동안 지속될 경우 보호장비가 파괴되며 "감전"페널티(마비 + 도트 데미지)가 적용됩니다. "감전"후 3초내에 전류에서 손을 땔 경우 문제상황이 해결됩니다.
  감전 페널티: 마비 + 도트 데미지

4.4.5.3. Class Diagram

System을 구성하는 현재 부분을 Attributes위주로 도식화

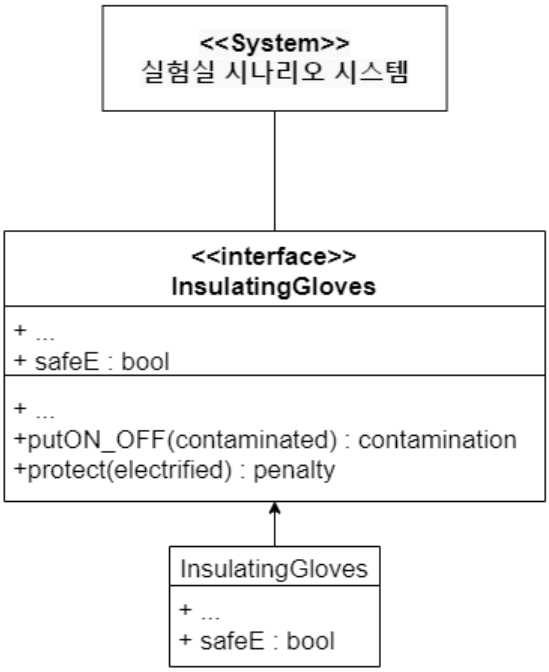

Figure 38 InsulatingGloves Class Diagram

4.4.5.4. Sequence Diagram

ElectricityUser가 system의 현재 해당 부분과 소통하게 되는 절차(method 위주로)를

도식화

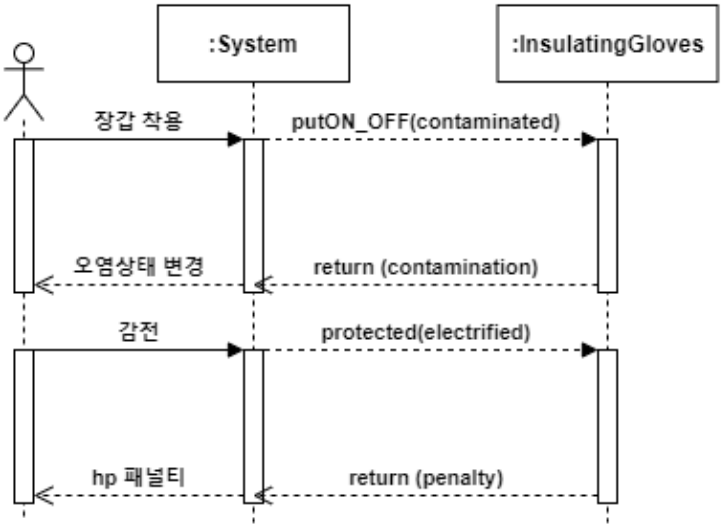

Figure 39 InsulatingGloves Sequence Diagram

4.4.6. DirectCurrentGenerator

구성된 회로에 전기 신호를 공급합니다

4.4.6.1. Attributes

다음은 DirectCurrentGenerator object가 가지고 있는 변수입니다.

- int v_gain (전원 공급중인 V / I)
- bool power (전기 on/off 여부)

4.4.6.2. Methods

다음은 DirectCurrentGenerator object가 가지고 있는 메소드입니다.

- connect () return()
  끼우기/뽑기 기능을 통해 Breadboard과 연결합니다.
- conPower(Multitap) return ()

Multitap에 연결을 합니다. 연결 시 Multitap에 과부하가 걸리지 않았는지 확인해주는 절차가 필요합니다

- power_In(circuit) return (electrified)
  전원을 공급합니다. 회로를 구성하고 전원을 공급할 경우 감전이 일어나지 않지만 전원을 공급한 다음 회로를 만지려고 한다면 감전이 발생하게 됩니다.

4.4.6.3. Class Diagram

System을 구성하는 현재 부분을 Attributes위주로 도식화

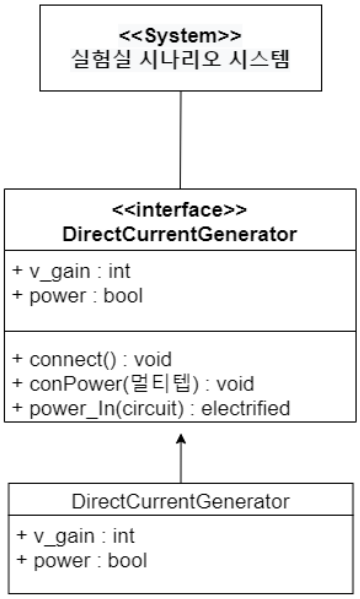

Figure 40 DirectCurrentGenerator Class Diagram

4.4.6.4. Sequence Diagram

ElectricityUser가 system의 현재 해당 부분과 소통하게 되는 절차(method 위주로)를 도식화

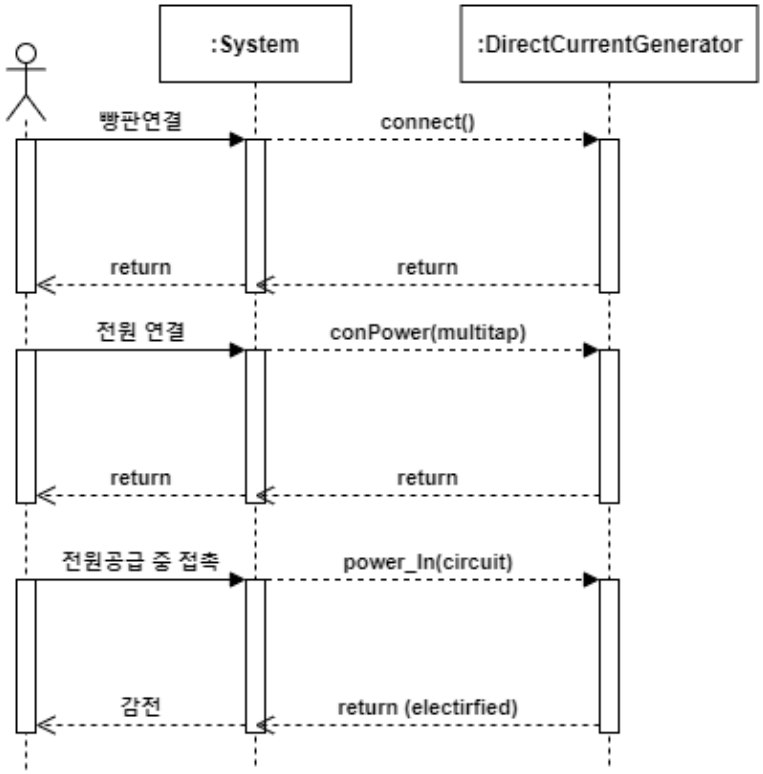

Figure 41 DirectCurrentGenerator Sequence Diagram

4.4.7. Breadboard

전원(DirectCurrentGenerator), Resistor, ConductingWire을 끼워서 회로를 구성할 수 있게 해주는 부품입니다.

4.4.7.1. Attributes

다음은 Breadboard object가 가지고 있는 변수입니다.

- int v_input (parent(DirectCurrentGenerator)로부터 받아오는 전류량)
- array connect_Res (Resistor(child) 연결)
- array connect_Wire (ConductingWire(child) 연결)
- bool circuit_info (회로가 단선 없이 구성되었는지)

4.4.7.2. Methods

Breadboard는 Method가 없습니다.

4.4.7.3. Class Diagram

System을 구성하는 현재 부분을 Attributes위주로 도식화

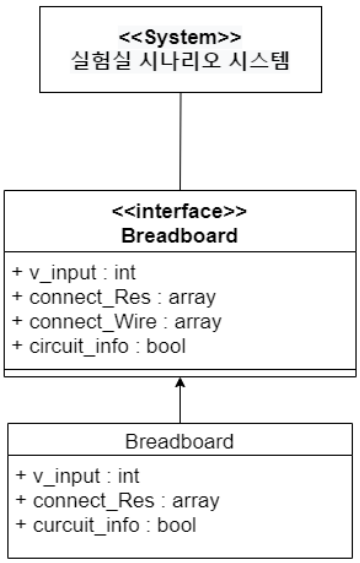

Figure 42 Breadboard Class Diagram

4.4.7.4. Sequence Diagram

ElectricityUser가 system의 현재 해당 부분과 소통하게 되는 절차(method 위주로)를 도식화

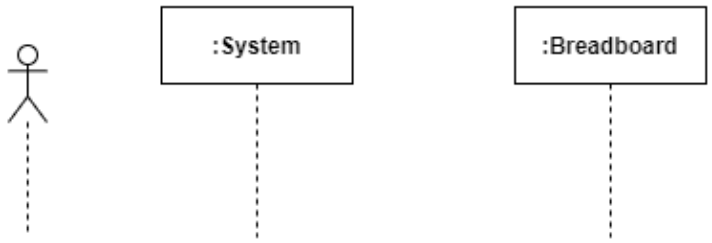

Figure 43 Breadboard Sequence Diagram

4.4.8. Resistor

실험에서 V_drop을 발생시키는 회로 소자입니다.

4.4.8.1. Attributes

다음은 Resistor object가 가지고 있는 변수입니다.

- Bool circuit_info (회로가 단선없이 구성되었는지)
- int v_drop (전압 감소량)
- int resist (저항값)

4.4.8.2. Methods

다음은 DirectCurrentGenerator object가 가지고 있는 메소드입니다.

- connect () return()
  끼우기/뽑기 기능을 통해 Breadboard과 연결합니다.

4.4.8.3. Class Diagram

System을 구성하는 현재 부분을 Attributes위주로 도식화

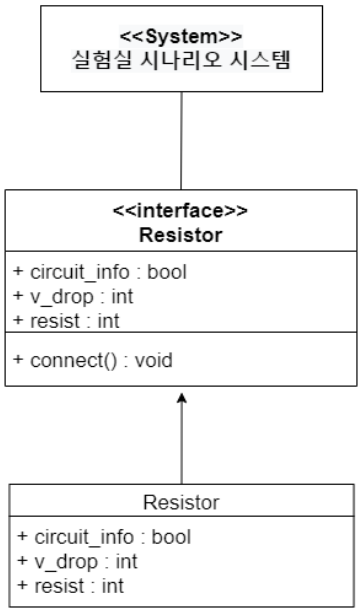

Figure 44 Resistor Class Diagram

4.4.8.4. Sequence Diagram

ElectricityUser가 system의 현재 해당 부분과 소통하게 되는 절차(method 위주로)를 도식화

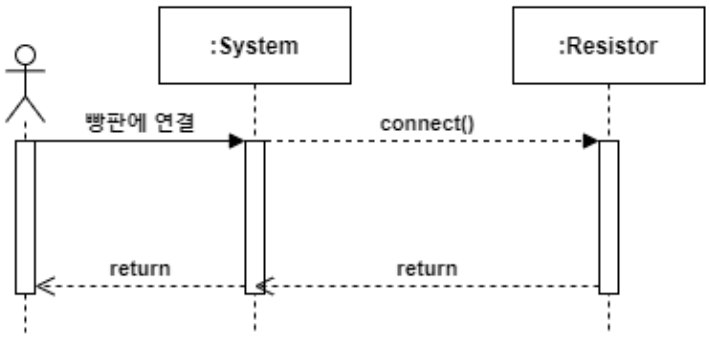

Figure 45 Resistor Sequence Diagram

4.4.9. ConductingWire

실험에서 Resistor과 Resistor 사이를 이어주는 역할을 합니다.

4.4.9.1. Attributes

다음은 ConductingWire object가 가지고 있는 변수입니다.

- Bool circuit_info (회로가 단선없이 구성되었는지)
- int v_drop (전압 감소량)
- int resist (저항값)

4.4.9.2. Methods

보호 다음은 DirectCurrentGenerator object가 가지고 있는 메소드입니다.

- connect() return()
  끼우기/뽑기 기능을 통해 Breadboard과 연결합니다.

4.4.9.3. Class Diagram

System을 구성하는 현재 부분을 Attributes위주로 도식화

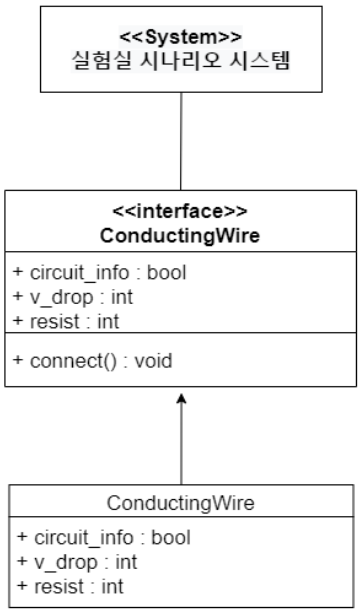

Figure 46 ConductingWire Class Diagram

4.4.9.4. Sequence Diagram

ElectricityUser가 system의 현재 해당 부분과 소통하게 되는 절차(method 위주로)를 도식화

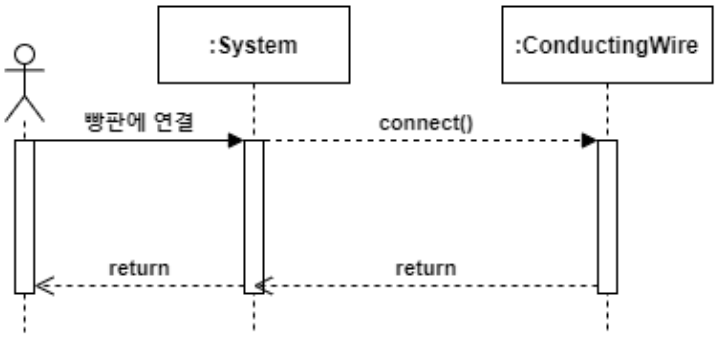

Figure 47 ConductingWire Sequence Diagram

4.4.10.Voltmeter

Resistor의 소자들의 전압변화를 측정하는 회로 소자입니다.

4.4.10.1.Attributes

다음은 Resistor object가 가지고 있는 변수입니다.

- Bool connect_Res (Resistor에 연결 중인지)
- int v_drop (연결중인 Resistor의 V_drop)

4.4.10.2.Methods

다음은 DirectCurrentGenerator object가 가지고 있는 메소드입니다.

- measure (v_gain) return(v_drop_curr)
  소자의 전압변화를 측정합니다. Loop의 전압변화의 대수적 합이 0이 되는지 확인합니다.
  v_drop_curr = V_DirectCurrentGenerator * R_curr / R_sum

4.4.10.3.Class Diagram

System을 구성하는 현재 부분을 Attributes위주로 도식화

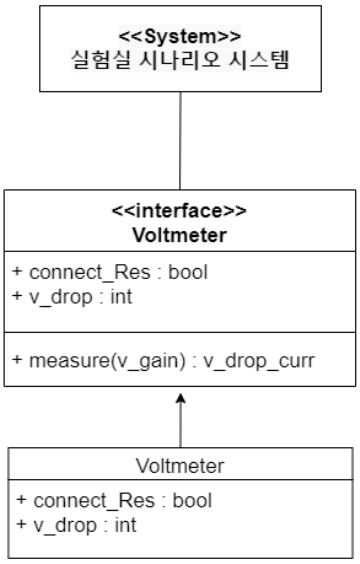

Figure 48 Voltmeter Class Diagram

4.4.10.4.Sequence Diagram

ElectricityUser가 system의 현재 해당 부분과 소통하게 되는 절차(method 위주로)를 도식화

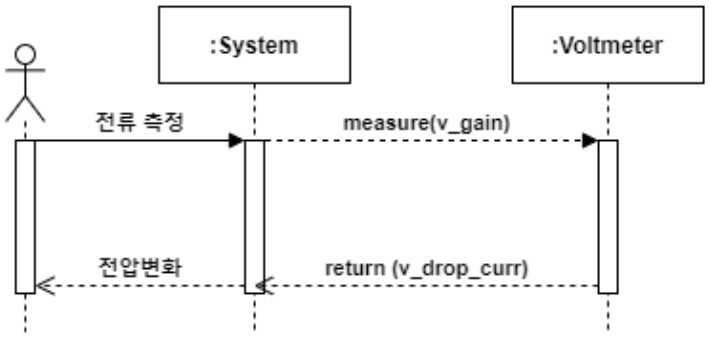

Figure 49 Voltmeter Sequence Diagram

5. System Architecture – Backend

5.1. Objectives

이번 장에서는 데이터베이스와 cloud API를 포함한 백 엔드 시스템의 구조를 설명합니다.

5.2. Overall Architecture

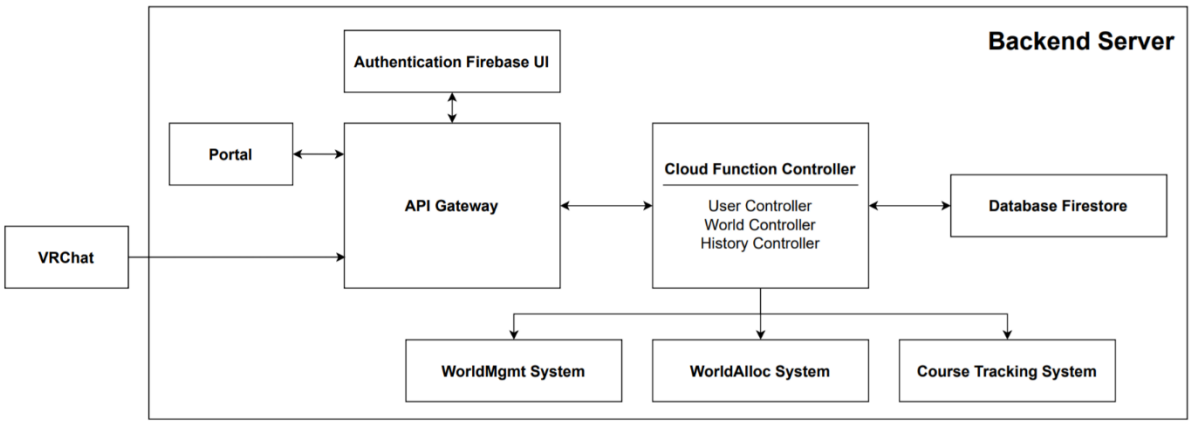

Figure 50 Overall Architecture - Backend

백엔드 시스템의 전체 구조는 위와 같습니다. 포탈은 사용자 인증과 사용자의 강의실 입장, 그리고 관리자를 위한 도구를 제공합니다. API 게이트웨이(Request Handler)는 VRChat 강의실 혹은 실험실에서 받은 요청이나 포탈로부터 받은 요청을 수신하여 적합한 cloud function (Controller / Manager)으로 할당합니다. 이 과정에서 사용자 인증과 같이 외부 API가 사용되는 기능들이 처리됩니다. 내부적으로 처리되는 요청들은 적합한 시스템의 컨트롤러로 할당되어 내부 데이터베이스 혹은 서버와 직접적인 상호작용을 통해 처리됩니다.

5.3. Subcomponents

5.3.1. Cloud Function

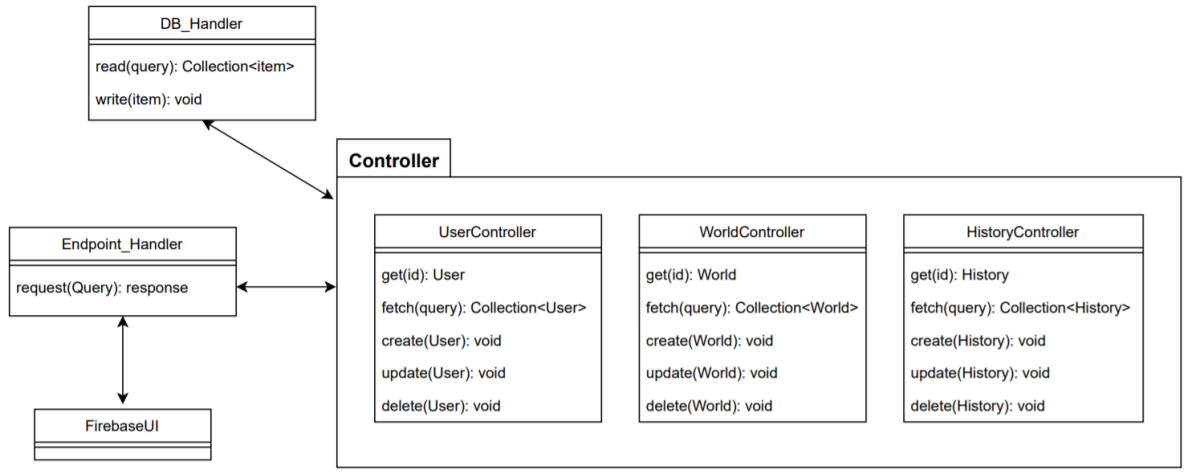

Figure 51 Cloud Function - Backend

5.3.1.1. Endpoint Handler Class

시스템 API gateway 혹은 router 역할을 수행하는 클래스로 VRChat월드나 포탈에서 전달받은 요청을 적합한 컨트롤러 혹은 API에 전달합니다.

5.3.1.2. FirebaseUI

Firebase 인증 SDK를 바탕으로 구현된 라이브러리로 로그인과 같은 사용자 인증 처리를 수행합니다.

5.3.1.3. DB_Handler Class

데이터베이스인 firestore과 통신하는 인터페이스를 제공하는 클래스로 사용자 정보 혹은 사용자 안전교육 이수 관련 정보를 가져오고 작성하는 역할을 수행합니다. 사용자 별로 하나의 문서를 가지고 있으며 해당 사용자 별로 이수 내역 컬렉션을 통해 관리됩니다.

5.3.2. VRChat World Management System

5.3.2.1. Class Diagram

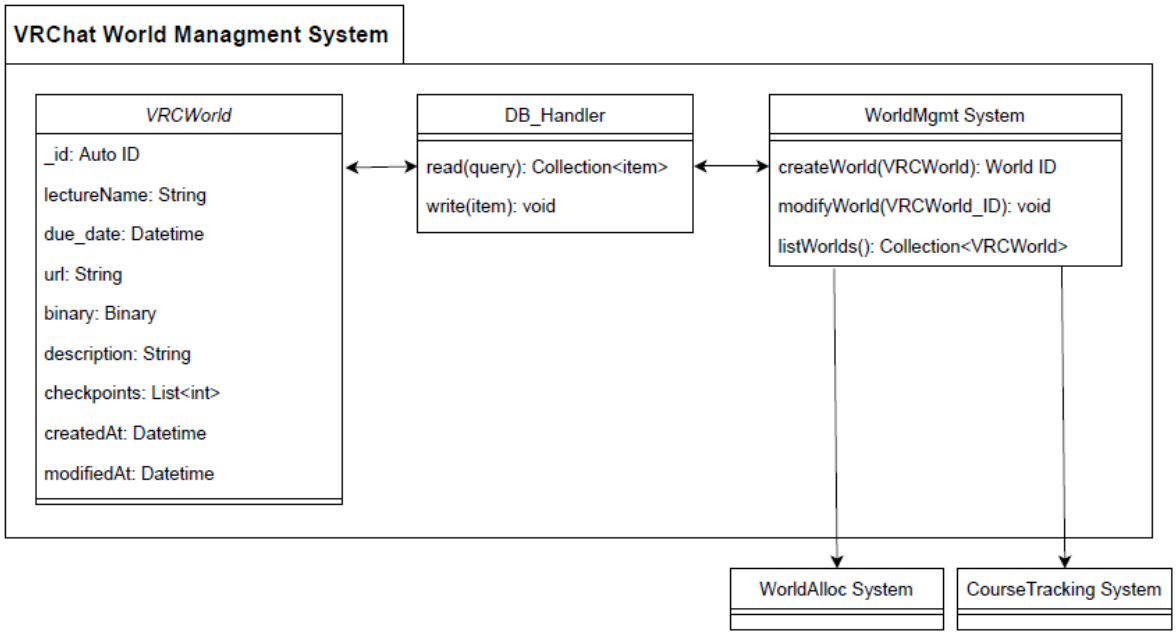

Figure 52 VRChat World Management System - Class Diagram

- Class Description

  ✓ **VRChat World Management System**: 해당 시스템은 시스템에 존재하는 VRChat 월드를 관리하는 시스템입니다. 해당 시스템의 사용자는 시스템 관리자입니다. 관리자가 새로운 VRChat 월드 즉, 새로운 강의를 추가하는 기능을 제공합니다. 주어진 새로운 월드에 대한 정보를 데이터베이스에 저장합니다. 저장과 동시에 해당 월드가 생성됨을 다른 시스템에게 알립니다. 해당 시스템은 기존에 존재하는 월드를 수정하는 기능을 제공합니다. 수정된 기존 월드에 대한 내용을 데이터베이스에 저장합니다. 저장과 동시에 해당 월드가 수정됨을 다른 시스템에게 알립니다. 해당 시스템은 데이터베이스에 저장된 모든 VRChat 월드의 목록을 보여주는 기능을 제공합니다. 데이터베이스로부터 데이터를 읽어와 파싱하여 반환합니다.

5.3.2.2. Sequence Diagram

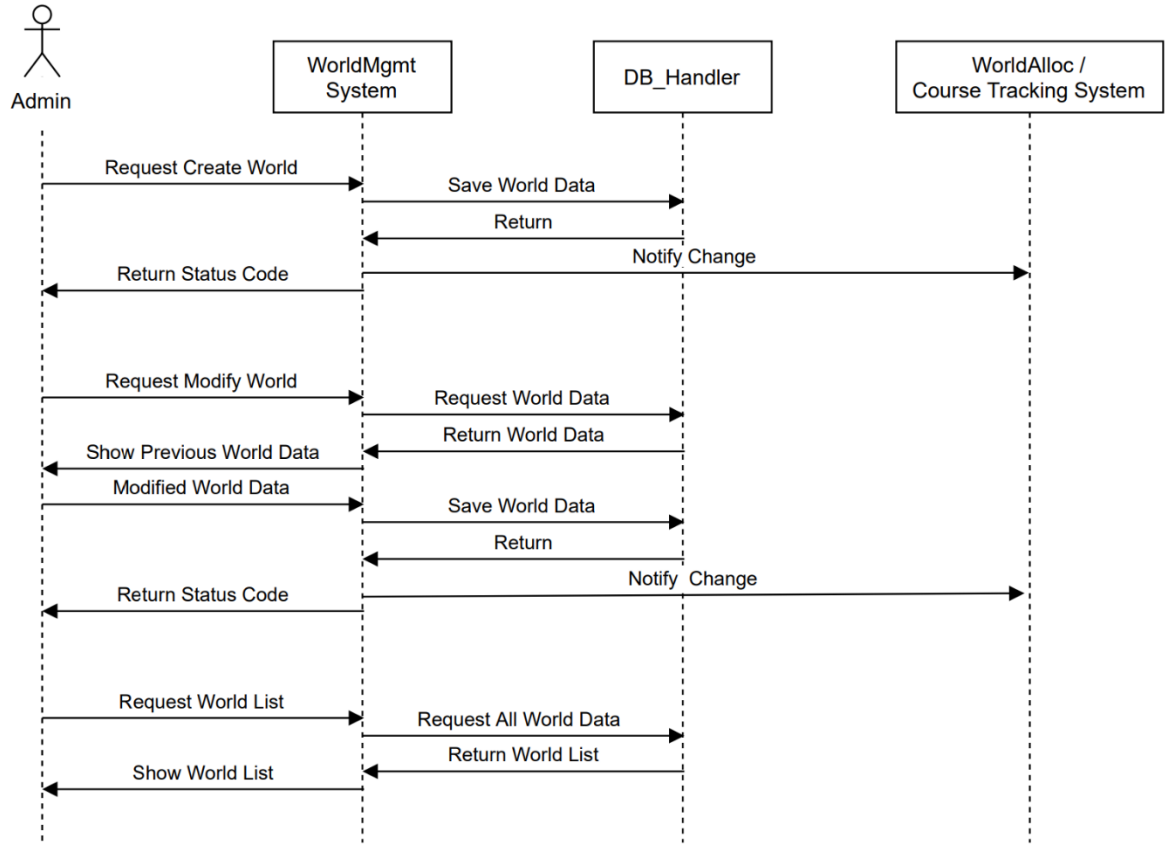

Figure 53 VRChat World Management System - Sequence Diagram

5.3.3. VRChat World Allocation System

5.3.3.1. Class Diagram

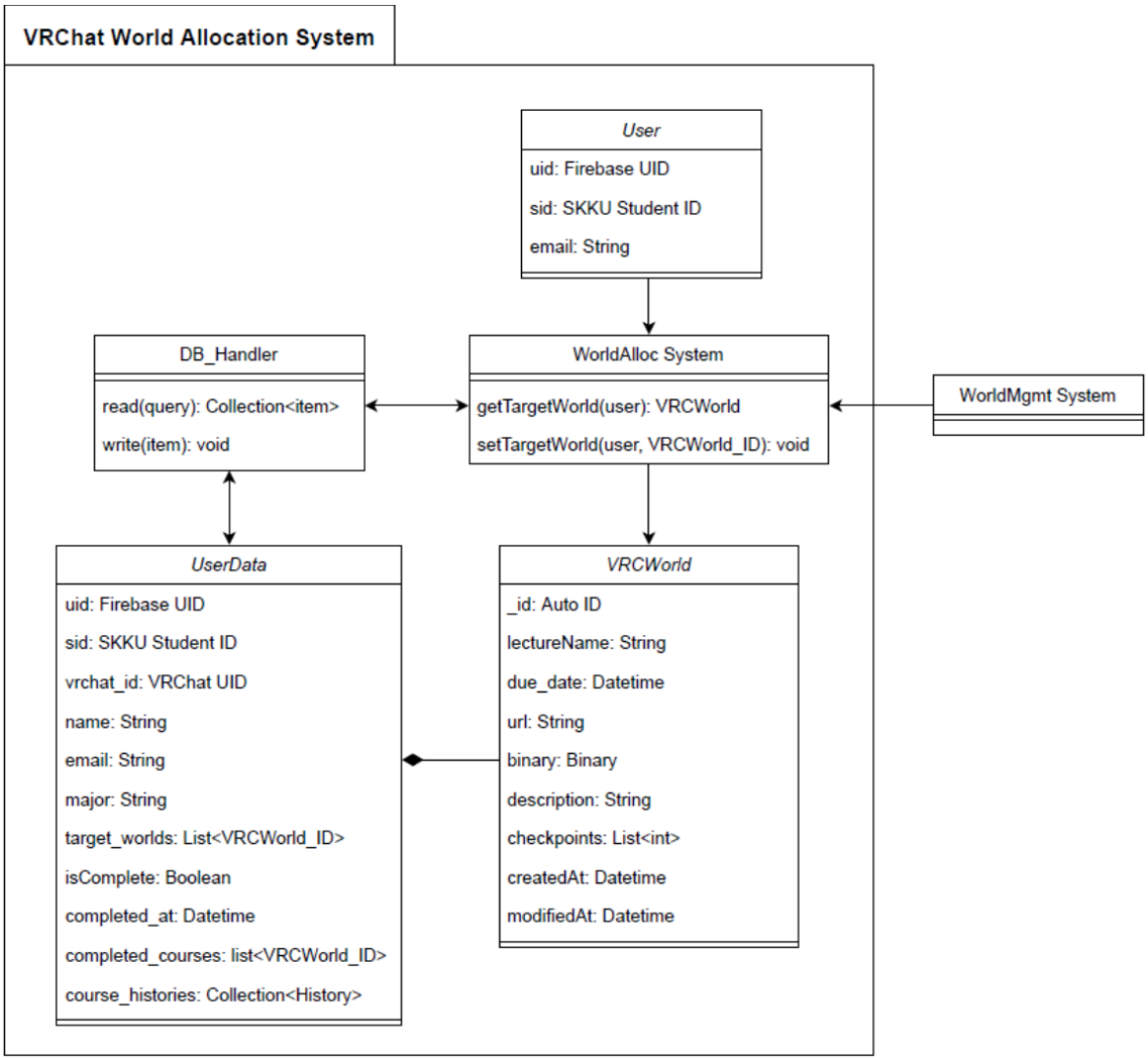

Figure 54 VRChat World Allocation System - Class Diagram

- Class Description

  ✓ **VRChat World Allocation System**: 해당 시스템은 사용자 강의실 제공 및 배정 기능을 수행하는 시스템입니다. 사용자가 수강하여야 하는 강의실이 담긴 VRChat 월드 URL을 제공합니다. 데이터베이스로부터 유저정보와 유저정보에 상응하는 VRChat 월드의 정보를 불러옵니다. VRChat World Management 시스템으로부터 월드의 변경 여부를 확인 후 VRChat 월드의

URL을 반환합니다. 해당 시스템은 시스템 관리자가 각 사용자 별로 수강하여야 하는 안전교육의 VRChat 월드를 할당하는 기능을 제공합니다. VRChat World Management 시스템으로부터 월드의 변경 여부를 확인 후 데이터베이스의 사용자 정보를 주어진 VRChat 월드 값으로 변경하여 저장합니다.

5.3.3.2. Sequence Diagram

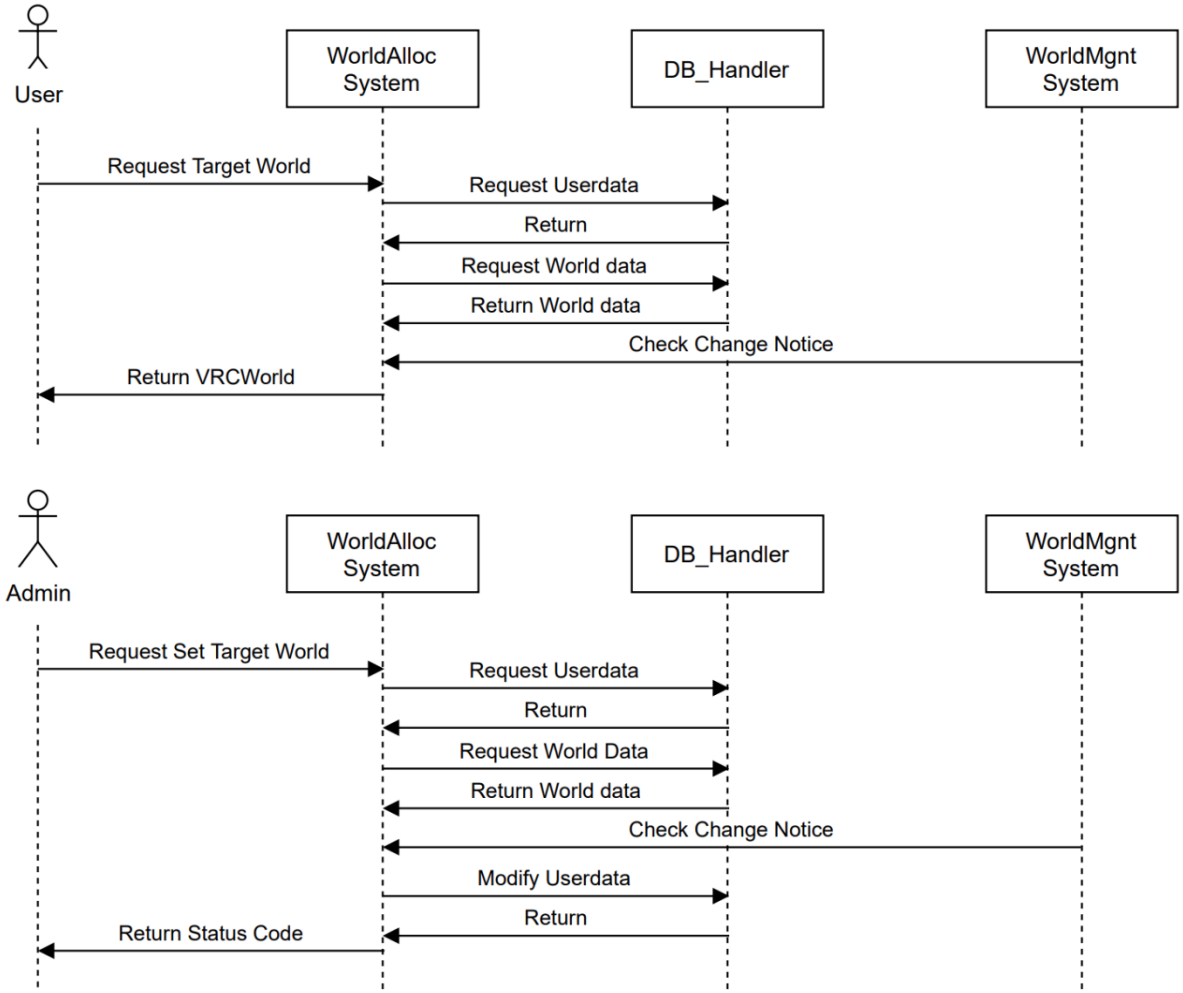

Figure 55 VRChat World Allocation System - Sequence Diagram

5.3.4. Course Tracking System

5.3.4.1. Class Diagram

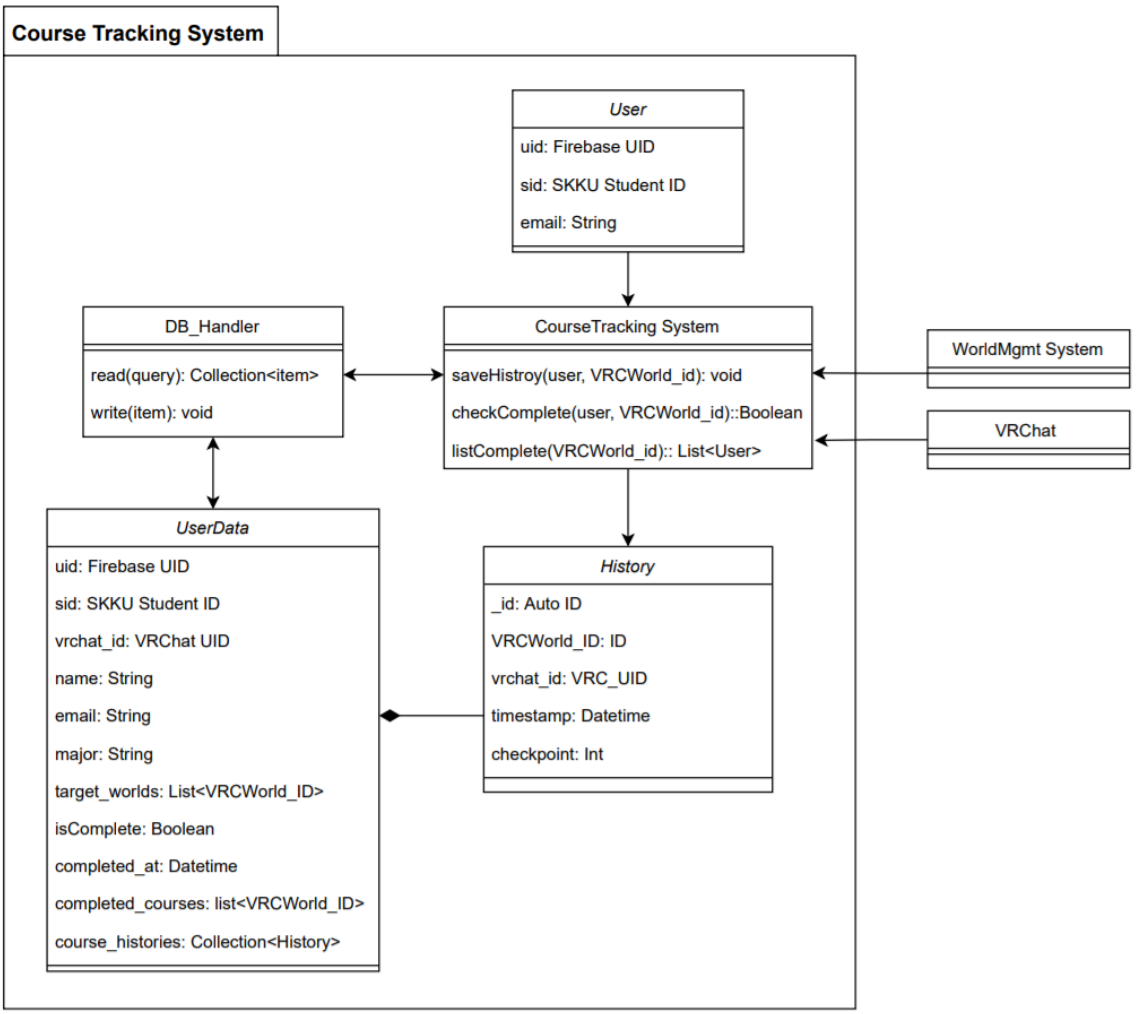

Figure 56 Course Tracking System - Class Diagram

- Class Description

  ✓ **Course Tracking System**: 해당 시스템은 사용자의 안전 교육 수강이력 기록과 시스템 관리자의 사용자 수강 이력을 확인하는 기능을 제공합니다. 사용자가 VRChat 월드에서의 강의실과 실험실에서 이수 완료시에 전송되는 HTTP 요청을 받습니다. VRChat World Management 시스템으로부터 월드의 변경 여부를 확인 후 사용자 정보에 기록합니다. 해당 시스템은 관리

자의 사용자 혹은 전체 사용자의 수강 이력을 확인하는 기능을 제공합니다. 사용자 정보의 수강정보와 VRChat World Management 시스템으로부터 월드의 변경 여부를 확인 후 이수 완료 내역을 반환합니다.

5.3.4.2. Sequence Diagram

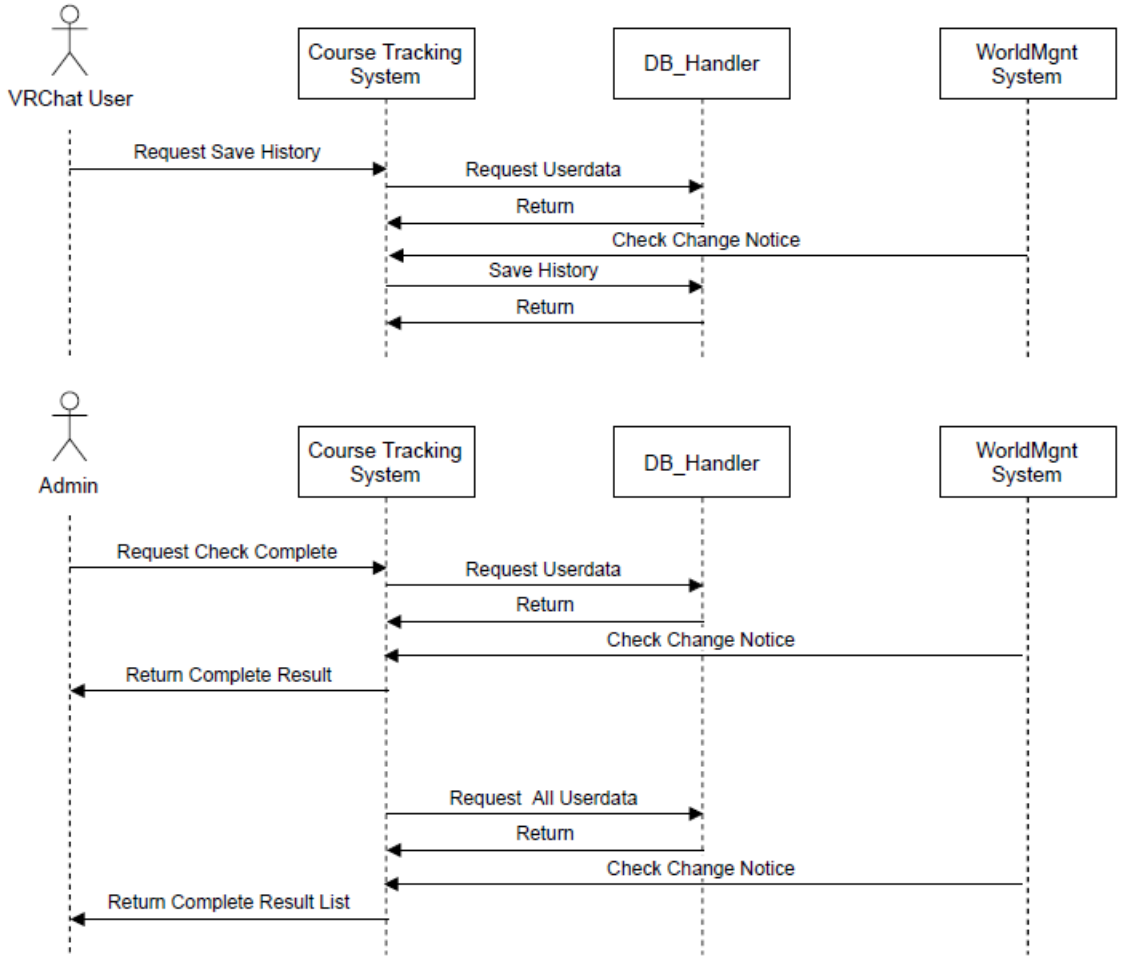

Figure 57 Course Tracking System - Sequence Diagram

6. Protocol Design

6.1. Objectives

이 장에서는 각 서브시스템, 특히 VRChat Worlds와 백엔드 서버 간의 상호 작용에 사용되는 프로토콜에 사용되는 구조를 설명합니다. 또한 이 장에서는 백엔드 서버에 대한 인터페이스에 대해 설명합니다.

6.2. Interaction between User and Back-end Server

6.2.1. Register

- Request

Table 1 Table of register request

Attribute	Detail	
HTTP Request Method	POST	
Request Body	ID	학번 or 학생 고유번호
	password	비밀번호 설정
	major	학과 정보

- Response

Table 2 Table of register response

Attribute	Detail	
Success Code	200 OK	
Failure Code	HTTP error code = 400	
Success Response Body	Message	계정 등록 성공 메시지
Failure Response Body	Message	계정 등록 실패 메시지

6.2.2. Log In

- Request

Table 3 Table of log-in request

Attribute	Detail	
HTTP Request Method	POST	
Request Body	ID	학번 or 학생 고유번호
	Password	비밀번호

- Response

Table 4 Table of log-in response

Attribute	Detail	
Success Code	200 OK	
Failure Code	HTTP error code = 400	
Success Response Body	Message	로그인 성공 메시지
	return	계정 정보 반환 및 로그인
Failure Response Body	Message	로그인 실패 메시지

6.2.3. Get Profile

- Request

Table 5 Table of get profile request

Attribute	Detail
Method	GET
URI	/user/:id/user

Attribute	Detail
Header	Host, Accept, Origin

- Response

Table 6 Table of get profile response

Attribute	Detail	
Success Code	200 OK	
Failure Code	HTTP error code = 404 (Not Found)	
Success Response Body	userInfo	사용자 학번 or 학생 고유번호
	completeCheck	실험실 별 안전교육 이수 여부
Failure Response Body	Message	Empty

6.3. Interaction between VRChat and Back-end Server

6.3.1. Complete request

- Request

Table 7 Table of complete request

Attribute	Detail	
Method	GET	
URI	/user_id/complete	
Request Body	type	실험실 종류
	time	현재 시간

- Store

Table 8 Table of complete response

~~Attribute~~	Detail	
Electric_experiment	Complete_flag	1 (안전교육 이수 완료)

Attribute	Detail	
	Complete_time	완료 시각
Chemical_experiment	Complete_flag	1 (안전교육 이수 완료)
	Complete_time	완료 시각

7. Database Design

7.1. Objectives

이 절에서는 데이터 구조가 데이터베이스에 표시되는 방법에 대해 설명합니다. 먼저 ER-다이어그램(Entity Relationship diagram)을 통해 entity와 그들의 관계를 식별합니다. 그리고 파이어베이스에 저장하기 위해 Json 포맷으로 데이터를 구성합니다.

7.2. ER Diagram

7.2.1. Entities

7.2.1.1. User data

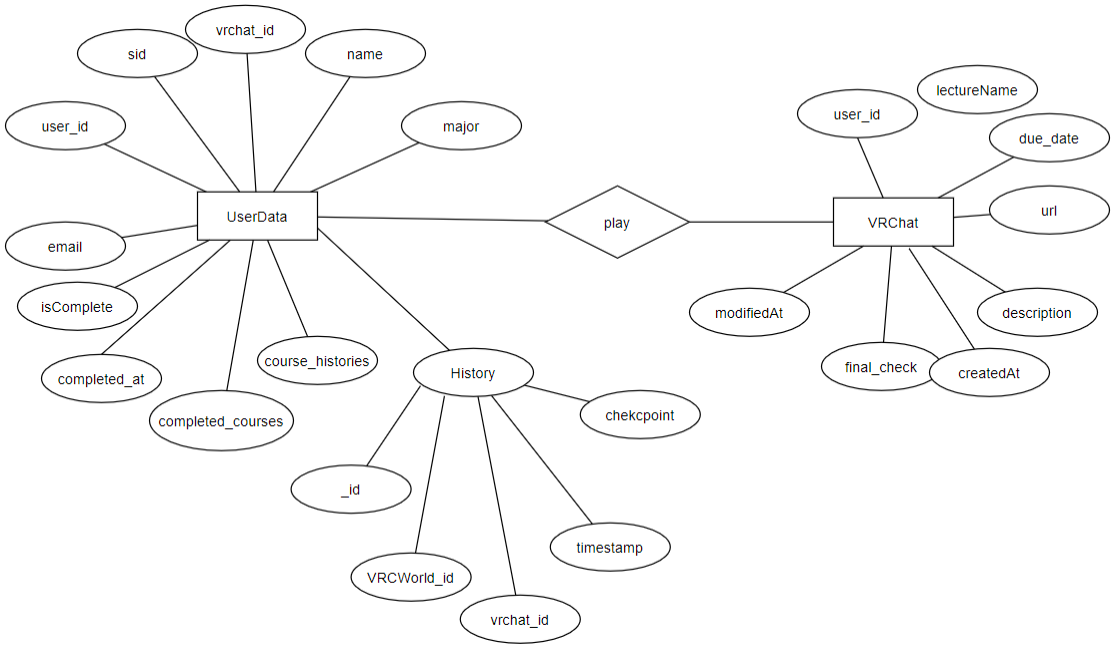

Figure 58 ER Diagram - User data

userData entity는 안전교육 시스템 사용자의 정보를 나타냅니다. user_id, sid, vrchat_id, name, email, major, isComplete, completed_at, completed_courses, course_histories, history로 구성되며 user_id가 primary key입니다.

7.2.1.2. VRChat

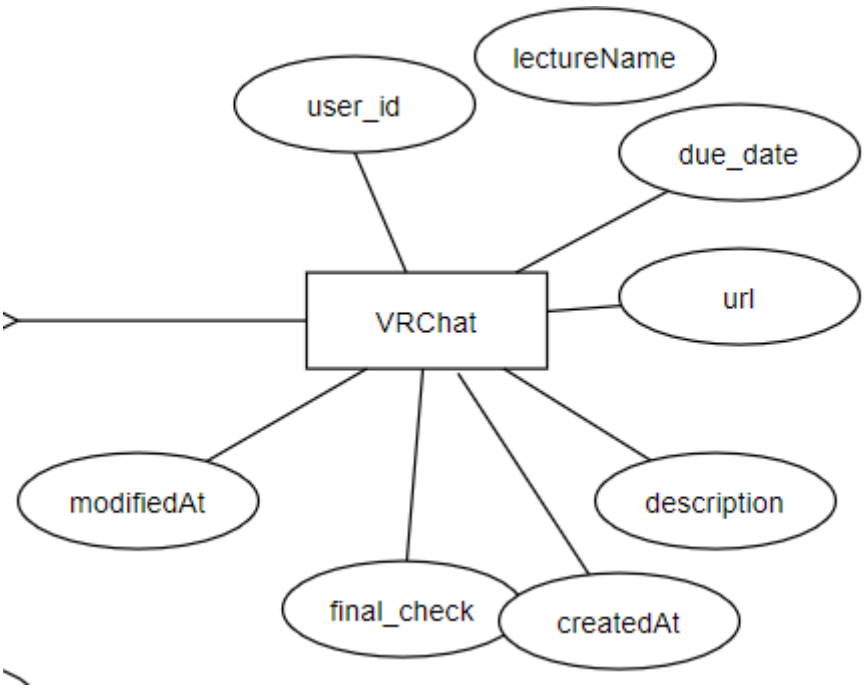

Figure 59 ER Diagram - VRChar

VRChat Entity는 VRChat 내에서 사용되는 정보들을 나타냅니다. User_id, lectureName, due_date, url, description, createdAt, final_check, modifiedAt 으로 구성됩니다.

7.3. JSON

7.3.1. Objective

JSON(JavaScript Object Notification)은 사람이 읽을 수 있는 텍스트를 사용하여 속성-값 쌍과 배열 데이터 형식(또는 기타 직렬화 가능한 값)으로 구성된 데이터 객체를 저장하고 전송하는 개방형 표준 파일 형식이다. AJAX 시스템에서 XML을 대체하는 등 다양한 응용 프로그램을 갖춘 매우 일반적인 데이터 형식이다.

7.3.2 JSON Format

Figure 60 JSON - Format

8. Testing Plan

8.1. Objectives

해당 섹션에서는 시스템에 대한 테스트 계획을 development testing, release testing, user testing으로 크게 3개로 나누어 설명합니다. 해당 테스트들을 통해 시스템이 내포할 수 있는 잠재적인 에러와 결함을 사전에 탐지 및 수정하여 시스템이 출시되었을 때 문제 없이 안정적으로 작동할 수 있도록 하고자 합니다.

8.2. Testing Policy

8.2.1. Development Testing

Development testing은 시스템 전반의 잠재된 에러나 결함을 예방하기 위해 수행됩니다. 이를 통해 소프트웨어 시스템 개발 비용과 시간 절약을 도모할 수 있습니다. 이 단계에서는 개발된 시스템이 충분한 테스트 단계를 거치지 않았음으로 시스템 전체가 불안정하고 컴포넌트간의 충돌이 발생할 수 있습니다. 이 단계에서는 static code analyzing, data flow analyzing, peer code review, unit testing 등이 수행됩니다. 이러한 일련의 과정을 통해 시스템의 useability, reliability, security 등을 확인하고자 합니다.

8.2.1.1. Usability

교육 시스템인 만큼 시스템에서 손쉬운 사용성이 중요한 특징 중 하나입니다. 사용자에게 보다 직관적이고 사용자 친화적인 경험을 제공하여 수업과 실험참여에 집중할 수 있도록 하여야 합니다. 이를 위해 개발 단계 전반에서 보다 나은 사용자 경험을 제공하기 위해 고민하고 사용자 인터페이스가 생성된 시점에서 다양한 사용자 집단에 설문조사를 진행하여 수정 및 반영합니다

8.2.1.2. Reliability

시스템이 오작동하지 않고 작동하기 위해서는 각각의 시스템 구성요소들이 올바르게 작동하고 적절하게 상호작용하여야 합니다. 따라서 development testing을 유닛 개발 단계부터 각 유닛이 시스템에 통합되는 동안 반복적으로 누적하여 수행해야 합니다..

8.2.1.3. Security

정보보안은 시스템 전반에서 중요한 emergent property입니다. 사용자가 아닌 비정상적인 외부 접속 제한과 관리자 권한 접근을 제제하고자 노력하여야 합니다. 높은 수준의 보안성을 유지하기 위해 유닛 단계에서부터 각 유닛의 보안 취약점을 분석합니다. 완성된 시스템에서도 정적 코드 리뷰, 각종 penetration test들을 통해 보안 테스트를 진행합니다.

8.2.2. Release Testing

소프트웨어가 원하는 방향으로 정상적으로 작동하는지 확인하기 위해서는 테스트가 필요합니다. Release testing은 소프트웨어를 작동 가능한 상태로 만든 후, 해당 소프트웨어에 대해서 의도했던 대로 요구사항에 맞게 작동하는지 확인하는 것, 즉 validation하는 것을 의미합니다. 소프트웨어를 사용자들이 사용하도록 하기 전에, 프로토타입으로부터 시작하여 테스트를 진행합니다. 만약 소프트웨어가 요구사항을 충족시키지 못하는 부분이 있다면, 프로토타입부터 점차 수정하여 확인된 충족하지 못한 요구사항을 충족하도록 합니다.

8.2.3. User Testing

유저 테스트는 실제 사용자가 소프트웨어를 이용하도록 하여 피드백을 얻습니다. 실제 사용자가 본 소프트웨어를 이용했을 때, 소프트웨어의 실행결과, 즉 통과 여부와 플레이 도중 일어난 오류나 불편사항 등의 리뷰 데이터를 수집합니다.

8.2.4. Testing Case

테스트 케이스는 위에서 제시한 3가지 주요 특성인 사용성, 안전성, 보안성을 측정하기 적합한 테스트 케이스로 진행합니다. 각 특성 별로 5개의 테스트 케이스를 준비하여 총합 15개의 테스트 케이스를 준비합니다. 테스트를 진행하고 수정하면서 테스트 내용과 결과를 문서화합니다

9. Development Plan

9.1. Objectives

이 장에서는 시스템의 개발 환경과 기술에 대해 설명합니다.

9.2. Frontend Environment

9.2.1. Unity

Figure 61 Unity - Logo

유니티(Unity)는 3D 및 2D 비디오 게임의 개발 환경을 제공하는 게임 엔진이자, 3D 애니메이션과 건축 시각화, 가상현실 등 인터랙티브 콘텐츠 제작을 위한 통합 제작 도구입니다. 엔진 자체에 라이트 매핑, 물리 엔진 등 미들웨어를 탑재했으며, 에디터에 내장된 애셋스토어를 통해 다양한 기능의 애셋을 다운로드하여 사용할 수 있습니다.

9.2.2. VRChat

Figure 62 VRChat - Logo

VRChat은 Graham Gaylor와 Jesse Joudrey가 만든 정식 출시 예정인 부분 유료화 대규모 다중 사용자 온라인 가상 현실 소셜 서비스입니다. 이 게임에서 플레이어들은 3D 캐릭터 모델로 구현된 다른 플레이어들과 상호작용할 수 있습니다. VRChat는 유니티 엔진을 통해 직접 자신만의 아바타나 월드를 등록할 수 있습니다. 적당한 모델을 구할 수 있다면 아바타를 제작해 업로드하는 것으로 VRChat 내에서 사용할 수 있습니다. 월드의 경우, 유니티에서 지원하는 모델 외에도 월드 제작자가 직접 필요한 콘텐츠를 제작할 수도 있습니다.

9.2.3. Udon

Udon은 VRChat 개발팀에서 만든 프로그래밍 언어입니다. Udon은 시각적 프로그래밍 인터페이스인 VRChat Udon 노드 그래프를 안전하고 빠르고 쉽게 만들 수 있는 언어입니다. VRChat Udon 노드 그래프는 노드와 와이어를 이용하여 시스템의 흐름, 입력, 출력들을 연결합니다. Udon은 VRChat과 Unity 모두에서 실행되므로 쉽게 제작물을 테스트하고 디버그할 수 있습니다.

9.3. Backend Environment

9.3.1. Github

Figure 63 Github - Logo

Github은 분산 버전 관리 툴인 git 저장소 호스팅을 지원하는 웹 서비스입니다. 이를 통해 개발 버전 관리를 손쉽게 할 수 있고, 시스템 구성 요소들을 하나의 완성된 시스템으로 쉽게 통합할 수 있습니다.

9.3.2. Node.js + Express.js

Figure 64 Node.js - Logo

Node.js는 확장성 있는 네트워크 애플리케이션(특히 서버 사이드) 개발에 사용되는 소프트웨어 플랫폼입니다. 작성 언어로 자바스크립트를 활용하며 논블로킹

(Non-blocking) I/O와 단일 스레드 이벤트 루프를 통한 높은 처리 성능을 가지고 있습니다.

내장 HTTP 서버 라이브러리를 포함하고 있어 웹 서버에서 아파치 등의 별도의 소프트웨어 없이 동작하는 것이 가능하며 이를 통해 웹 서버의 동작에 있어 더 많은 통제를 가능케 합니다.

Figure 65 Express.js - Logo

Express.js는 웹 및 모바일 애플리케이션을 위한 일련의 강력한 기능을 제공하는 간결하고 유연한 Node.js 웹 애플리케이션 프레임워크입니다. 자유롭게 활용할 수 있는 수많은 HTTP 유틸리티 메소드 및 미들웨어를 통해 쉽고 빠르게 강력한 API를 작성할 수 있습니다.

Node.js와 Express.js를 이용해 백엔드 서버를 구축할 것입니다.

9.3.3. Firebase

Figure 66 Firebase - Logo

클라우드 스토리지, 실시간 데이터베이스, 머신러닝 키트 등 다양한 기능을 제공하여 모바일 및 웹 애플리케이션 개발을 지원합니다. 그 중 실시간 데이터베이스

기능을 사용하여 사용자의 안전교육 영상 수강 여부, VRChat 실험실 이수 여부 등의 데이터를 관리할 것입니다. 실시간 데이터베이스를 사용하기 때문에 백엔드 서버에 바로 데이터가 동기화됩니다. 사용자는 강의 수강 내역이나 실험실 이수 여부를 완료 즉시 백엔드 서버에서 확인할 수 있습니다.

9.4. Constraints

이 시스템은 본 문서에 언급된 내용을 바탕으로 설계 및 구현될 것입니다. 그 외 세부사항은 개발자가 선호하는 방향을 선정하여 설계 및 구현하되, 다음과 같은 사항을 준수합니다.

- 안전교육 실험실 시나리오는 처음 접속한 세션으로 종료될 때까지 변화 없이 수행하여야 정상적으로 이수처리 되며, 재접속하여 세션이 변경되는 경우 처음부터 다시 수행하여야 합니다.
- VRChat world에 접속할 수 있는 최대 동시 접속자수는 40명이며, 활동 사용자수는 30명입니다.
- 월드의 로딩은 1분 이내로 완료되어야 합니다.
- 안전교육 시나리오 진행 중 각 단계의 이수 완료 메시지는 10초 이내로 전송되어야 합니다.
- 이수 완료 메시지는 백엔드에서 10초 이내로 처리되어 저장되어야 합니다.
- VRChat SDK 이외의 다른 컴포넌트를 사용할 수 있으며 VRChat 환경 안에서 작동되어야 합니다.
- 모든 패키지와 개발툴은 안정화된 것을 사용해야 합니다.
- 실험실 위험 상황은 시나리오를 따라 개발하며, 보다 명확한 내용으로 구현합니다.
- 백엔드 서버와 데이터베이스 서버는 시스템 비용과 유지비용을 고려하여 설계합니다.

- 시스템의 향후 확장성과 가용성을 고려하여 설계합니다.
- 소스코드를 최적화하여 시스템 자원 낭비를 방지합니다.
- 사용자가 부정으로 시스템을 사용하지 못하도록 보안성을 갖추어 설계합니다.
- 오브젝트와 트리거는 원래 의도한 대로 작동되도록 설계합니다.

9.5. Assumptions and Dependencies

실험실 시나리오는 Unity와 VRChat SDK를 사용하여 VRChat 안에서 구동될 수 있게 구현합니다. Unity는 2019.4.30f1 이후 버전을 기준으로 개발합니다. VRChat SDK는 SDK3 Avatar 2021.9.30.16.19 이후 버전, SDK3 Worlds (2021.9.30.16.18 이후 버전을 사용합니다.

백엔드 서버는 16.13.0 LTS 버전을 기준으로 설계하며 LTS 버전 이후 버전을 사용할 경우 일부 모듈을 사용할 때 불안정할 수 있습니다.

10.Supporting Information

10.1. Software Design Specification

이 소프트웨어 설계 사양은 IEEE 권장사항(IEEE Recommended Practice for Software Design Description, IEEE-Std-1016)에 따라 작성되었습니다.

10.2. Document History

Table 9 Document History

Date	Version	Description	Writer
2021/11/13	0.1	형식 정리	한영진
2021/11/14	1.0	6.1, 6.2, 6.3 추가	한영진

Date	Version	Description	Writer
2021/11/14	1.1	5.1, 5.2	김재윤
2021/11/14	1.2	1	김예준
2021/11/15	1.3	2	강승목
2021/11/15	1.4	3	손병호
2021/11/16	1.5	3.2.3	김예준
2021/11/16	1.6	4 layout 및 역할 분배	강승목
2021/11/16	1.7	4.3	손병호
2021/11/17	1.8	4.4	김예준
2021/11/18	1.9	2.4.1, 5.3	김재윤
2021/11/18	1.10	7.1, 7.2, 7.3 추가	한영진
2021/11/18	1.11	7.1, 7.2 수정	한영진
2021/11/19	1.12	4.4.1 수정	김예준
2021/11/19	1.13	4.2,8.2.2,8.2.3,8.2.4	김진성
2021/11/19	1.14	4.4.5 수정	김예준
21/11/20	1.15	8.1, 8.2.1, 8.2.4	김재윤
21/11/20	1.16	9.1, 9.2	손병호
21/11/20	1.17	9.3, 9.4, 9.5, 10.1 추가	한영진