Software Design Specification

스마트 약 디스펜서

소프트웨어공학
Team 8

2017314474 강창우

2018311782 김하늘

2018314653 김민종

2020310548 원현선

1. Preface

이 챕터에서는 문서의 Readership information, Readership, Scope, Objective를 포함하며, 스마트 약 디스펜서의 Software Design Specification의 문서 구조를 설계한다.

1.1 Readership

이 문서는 10개의 섹션으로 구성되어 있으며, 각각은 더욱 상세한 sub-section으로 이루어져 있다. 각각의 섹션에 대한 설명은 아래의 1.4(Document Structure)에 기재되어 있으며, 이 문서의 주된 Reader는 소프트웨어공학 Team 8이다. 하지만, '소프트웨어공학' 수업의 학생, 교수, TA, 혹은 기타 관심이 있는 모든 사람이 독자가 될 수 있다.

1.2 Scope

특정 사람들에게 약을 복용하는 것은 매우 중요하거나 어려운 일이 될 수 있다. 스마트 약 디스펜서는 그 과정을 정확하고 편리하게 만들고자 한다. 이 문서는 이를 구현하기 위한 디자인을 묘사하고 있다.

1.3 Objective

스마트 약 디스펜서의 목표는 노인과 시각장애인을 주 타겟층으로 하여 약 복용이 원활하게 이루어질 수 있고자 하는 것이다. 모든 약은 처방받은 후 약을 등록하여 스토리지에 알약을 종류별로 구분하여 저장할 수 있다. 이후에 미리 설정한 시간에 알림이 울리어 시간에 맞게 적정량을 복용할 수 있다. 또한, 약을 복용할 때 음성 인식 기능을 사용하기 때문에 노인과 시각 장애인이 특별한 보호자의 도움 필요없이 이용할 수 있다.

특히 이 문서(Software Design Document)의 주된 목적은 스마트 약 디스펜서, 그리고 이와 연동되는 애플리케이션의 기술적 디자인을 묘사하기 위함이다. 또한, 이 문서는 시스템의 다른 측면을 묘사하기 위한 구조적인 Overview를 제공한다. 또한, SRS 문서에서 언급한 모듈을 보다 상세하게 기술하였으며, Use cases를 프로그래밍 팀이 어떻게 특정 모듈을 구현할지를 보여주는 Sequential diagrams와 Activity diagrams(Class diagram을 포함하여)로 변환하여 기술하였다. 이것의 주된 대상은 이해관계자, 개발자, 디자이너, 그리고 모바일 애플리케이션의 소프트웨어 테스터이다.

1.4 Document Structure

1. Preface
   : 현재 읽고 있는 챕터로, 이 장에서는 이 문서의 독자 수, 범위, 목적 및 구조에 대한 정보를 제공한다.

2. Introduction
   : 이 챕터에서는 프로젝트에 사용된 다이어그램 및 도구에 대한 정의와 설명을 제공한다. 또한 이 문서를 이해하는 데 필요할 수 있는 Scope 와 Reference 를 제공한다.

3. System Architecture - Overall
   : 이 챕터에서는 어떻게 시스템이 구성되는지에 대한 정보를 제공한다.

4. System Architecture - Frontend
   : 이 챕터에서는 시스템 Frontend 의 구조, 동작 및 뷰를 정의하는 개념 모델을 설명한다.

5. System Architecture - Backend
   : 이 챕터에서는 시스템 Backend 의 구조, 동작 및 뷰를 정의하는 개념 모델을 설명한다.

6. Protocol Design
   : 이 챕터에서는 하위 시스템의 상호 작용에 사용되는 프로토콜과 인터페이스를 정의하는 프로토콜에 대한 설명을 제공한다.

7. Database Design
   : 이 챕터에서는 시스템 데이터 구조에 대한 설명과 해당 데이터 구조가 데이터베이스에 표현되는 방법을 제공합니다.

8. Testing Plan
   : 이 챕터에서는 개발 테스트, 릴리스 테스트 및 사용자 테스트에 대한 계획을 기술한다.

9. Development Plan
   : 이 챕터에서는 개발 환경 및 개발 시 사용되는 도구에 대한 정보를 제공합니다.

10. Supporting Information: 이 챕터에서는 본 문서의 기록에 대해 기술한다.

2. Introduction

2.1 Objectives

국내에서 약을 만성적으로 복용하는 노인의 비율은 매우 높지만, 약을 복용하는 것을 잊거나 중복 복용하는 문제는 계속되고 있다. 또한, 의약품 점자 표기 의무화를 이행하지 않음으로 인하여 시각 장애인은 약을 구별할 수 없는 불편함을 겪고 있다. 이와 같이 약 복용 중 일상에서 겪는 문제점을 해소하기 위하여 스마트 약 디스펜서를 개발하기로 하였다.

스마트 약 디스펜서의 목표는 노인과 시각장애인을 주 타겟층으로 하여 약 복용이 원활하게 이루어질 수 있고자 하는 것이다. 모든 약은 처방받은 후 약을 등록하여 스토리지에 알약을 종류별로 구분하여 저장할 수 있다. 이후에 미리 설정한 시간에 알림이 울리어 시간에 맞게 적정량을 복용할 수 있다. 또한, 약을 복용할 때 음성 인식 기능을 사용하기 때문에 노인과 시각 장애인이 특별한 보호자의 도움 필요없이 이용할 수 있다. 본 문서에는 스마트 약 디스펜서에 직/간접적으로 사용되는 여러 디자인 구조가 묘사되어 있다. 묘사된 디자인 구조는 이전에 작성된 SRS에서 명시된 요구사항들에 따라 설계되어 있다.

2.2 Applied Diagrams

2.2.1 UML

UML 는 “Unified Modeling Language”의 약자로서, 그 목적은 표준 방식으로 시스템의 설계를 시각화하는 것이다. 이 프로젝트는 UML을 사용하여 그 시각화를 더욱 효율적으로 하고, 독자들이 시스템을 더 잘 이해하도록 돕는다.

2.2.2 Use case Diagram

Use case Diagrams 는 사용자와 사용자가 관련된 다양한 사용 사례 사이의 관계를 보여주는 시스템과의 사용자 상호 작용을 나타낸다. 이는 개발이 완료되지 않은 새로운 소프트웨어 프로그램에 대한 시스템/소프트웨어 요구 사항의 주요 형태이다. 이 문서는 시스템이 무엇을 해야 하는지에 대한 단순화된 지침의 역할을 하며, 단계가 수행되는 순서를 나타내지 않으며, 사용 사례, 행위자 및 시스템 간의 관계 중 일부만 요약한다.

2.2.3 Sequence Diagram

Sequence Diagrams 는 interaction 의 맥락에서 객체 간의 상호 작용을 수행하는 방법을 자세히 설명하는 상호 작용 다이어그램이다. 이들은 사용 사례 또는 작동을 시각화하는 하위 시스템, 시스템 또는 시스템과 사용자 간에 발생하는 높은 수준(high-level) 의 상호 작용을 묘사한다.

2.2.4 Class Diagram

Class Diagrams 는 시스템의 클래스, 속성, 운영 및 객체 간의 관계를 보여줌으로써 시스템의 구조를 설명하는 정적 구조 다이어그램의 일종이다. 이들은 시스템에서 작업할 때 개발자와 다른 팀 구성원을 돕는 클래스 집합과 클래스 간 관계 집합으로 구성된 다이어그램을 통해 시스템에서 분류자의 정적 구조를 제공하기 때문에 객체 지향 모델링의 기반이 된다.

2.2.5 Context Diagram

Context Diagrams 은 시스템과 외부 구성 요소 간의 정보 흐름을 보여주는 데 사용된다. 이는 데이터 흐름도 중 가장 높거나 가장 기본적인 수준이며, 그래서 상황 도표는 데이터 전송과 관련된 프로세스를 지정함으로써 시스템 전체에서 정보가 어떻게 흐르는지 문서화하는 도표이다. 따라서 Context diagrams 는 Level 0 Data Flow Diagrams라고도 한다.

2.2.6 Entity Relationship Diagram

Entity Relation Diagram 는 시스템 내의 다양한 엔티티가 서로 어떻게 관련되어 있는지를 시각적으로 표현하는 방법이다. 이 때의 엔티티는 사람, 역할, 이벤트, 장소, 제품 및 로그와 같이 시스템에 사용되는 모든 유형의 비즈니스 개체 등을 일컫는다.

2.3 Applied Tools

2.3.1 ERD Plus

ERD Plus 는 Entity Relationship Diagrams, Relational Schemas, Star Schemas, 그리고 SQL DDL 문장을 만드는 데이터베이스 모델링 도구이다. 또한, ERD Plus 는 Entity Relationship Diagrams (ERDs), Relational Schemas (Relational Diagrams), Star Schemas (Dimensional Models)를 빠르고 쉽게 만들 수 있는 웹 기반 데이터베이스 모델링 도구이다. ERD Plus 를 사용하면 표준 ERD 구성요소 (Entities, Attributes, Relationships)들을 그릴 수 있다.

2.3.2 Microsoft PowerPoint

Microsoft PowerPoint를 사용하면 이미 알고 있는 익숙한 슬라이드쇼 제작 도구에 액세스할 수 있으며, 프레젠테이션을 빠르고 쉽게 작성, 편집, 보기, 표시 또는 공유할 수 있다. PowerPoint는 모든 장치에서 쉽게 액세스할 수 있도록 최근 슬라이드 및 프레젠테이션을 빠르게 볼 수 있다. PowerPoint를 사용하면 강력하고 사용자 지정 가능한 슬라이드 및

프레젠테이션으로 지속적인 인상을 남길 수 있다. PowerPoint는 프레젠테이션에 사용할 템플릿과 자동 설계 아이디어를 제공한다. PowerPoint는 기업인과 교사, 학생 등 다양한 이용자들이 프레젠테이션 소프트웨어로 널리 활용되고 있으며, 이를 설득 스킬 형태로 활용하는 가장 적절한 방법 중 하나이다.

2.4 Project Scope

스마트 약 디스펜서의 목표는 노인과 시각장애인을 주 타겟층으로 하여 약 복용이 원활하게 이루어질 수 있고자 하는 프로젝트이다. 기본적으로 본 프로젝트는 사용자와 사용자 간의 실시간 교류에 목적을 두고 있으므로, 이를 위한 데이터 서버가 구비되어야 할 것이고, 또한 하드웨어와 앱 간의 교류를 위한 서버도 구비되어야 한다. 타겟층이 건강에 취약하므로 약을 거르는 일이 없도록 하기 위해 다양한 형태의 알림 기능, 알람 기능 등을 지원할 것이다. 이 외에도 가능한 한 사용자에게 친화적인 환경의 제공을 통해 사용자가 스스로 학습에 흥미를 갖고 임할 수 있도록 본 개발자들은 노력할 것이다.

2.5 References

이 문서는 다음의 자료들을 참고하였다.

- Team 7, 2021 Spring, Software Design Document, SKKU.

  https://github.com/skkuse/2021spring_41class_team7

- Team 4, 2021 Fall, Software Design Specification, SKKU.

  https://github.com/skkuse/2021fall_41class_team4

- Team 16, 2021 Fall, Software Design Specification, SKKU.

  https://github.com/skkuse/2021fall_41class_team16

3. System Architecture - Overall

3.1 Objectives

이 장에서는 시스템 아키텍처를 백엔드와 프론트엔드의 디자인의 구성을 통하여 설명한다.

3.2 System Organization

스마트 약 디스펜서는 사용자가 애플리케이션을 통하여 약 세부사항을 관리하고 그 데이터를 디스펜서 기기가 이용하여 사용자에게 약을 제공한다. 아래는 스마트 약 디스펜서의 Model View Controller 패턴이다. 프론트엔드에서는 사용자에게 회원가입, 로그인, 아이디/비밀번호 찾기, 비밀번호 재등록, 약 등록 화면, 사용자 초기 화면, 사용자 알림 화면 등을 뷰를 통해서 전달한다. 사용자 뷰에서는 컨트롤러에게 데이터를 전달하며, 상태를 모델로부터 받아 표시한다. 뷰에서 입력받은 데이터들은 백엔드에서는 사용자가 입력한 계정, 프로필, 약에 관한 정보들이 컨트롤러를 통해서 데이터베이스로 전달된다. 데이터베이스에서는 전달받은 데이터를 데이터베이스 내에 저장하여 추후에 컨트롤러에게 이 데이터를 다시 제공할 수 있다. 또한, 바뀐 상태에 대하여 뷰에게 변화 알림을 주어 사용자 뷰를 전환할 수 있다. 모델 내의 서버에서는 데이터베이스를 활용하여 디스펜서 기기에게 지시사항을 내리고, 디스펜서는 적절한 행동을 취한 이후 서버에게 상태를 전달한다. 이 MVC 패턴 과정에서 모델, 뷰, 컨트롤러가 데이터를 전달하는 과정은 HTTP protocol을 사용하며, JSON format으로 저장된다.

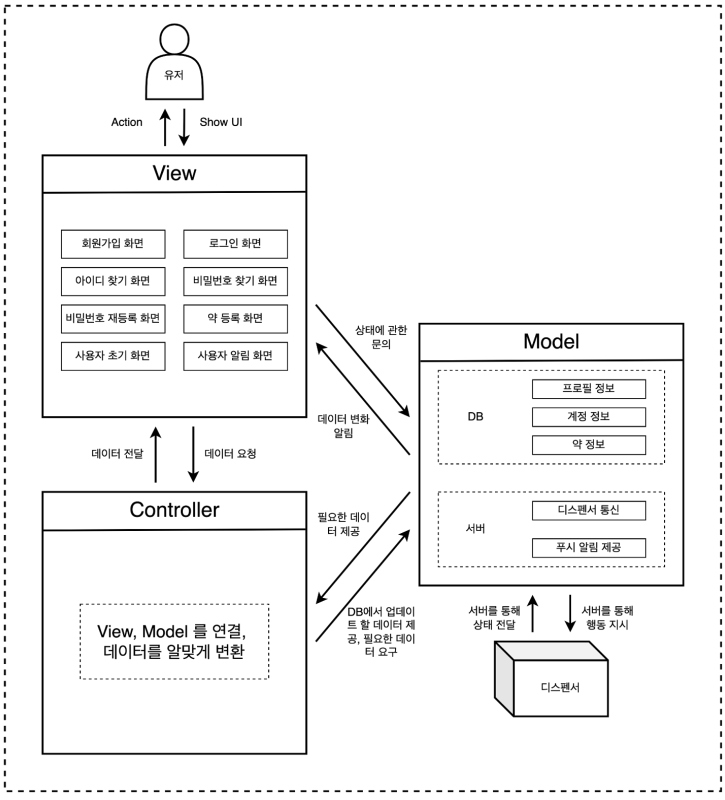

그림 1 Model-View-Controller

3.2.1 Context Diagram

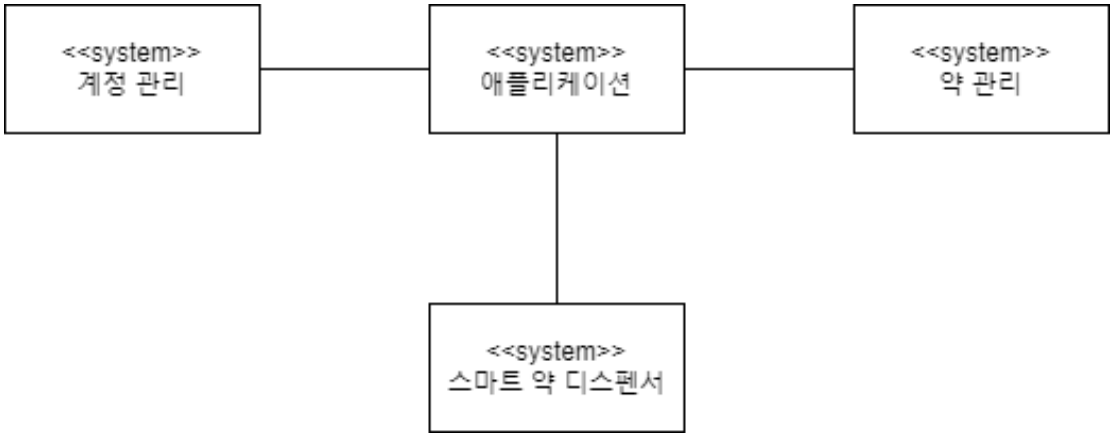

그림 2 Context Diagram

3.2.2 Sequence Diagram

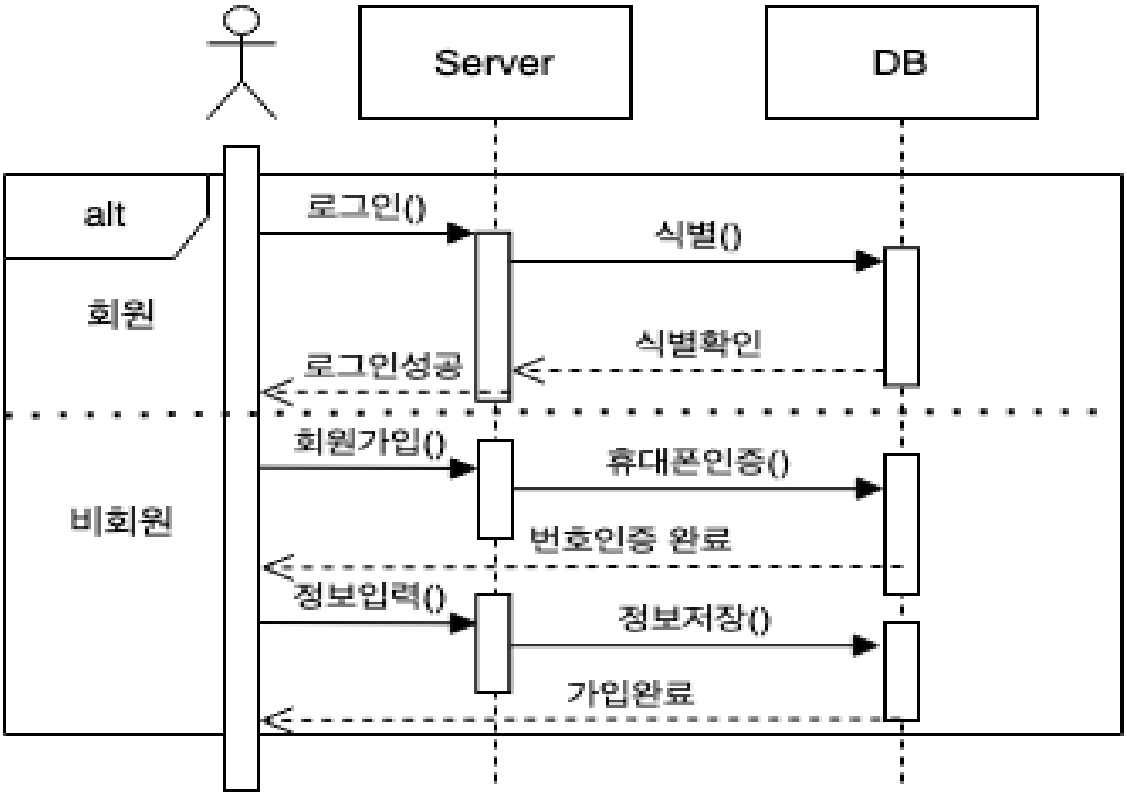

그림 3 Sequence Diagram - 로그인 / 회원가입 Sequence

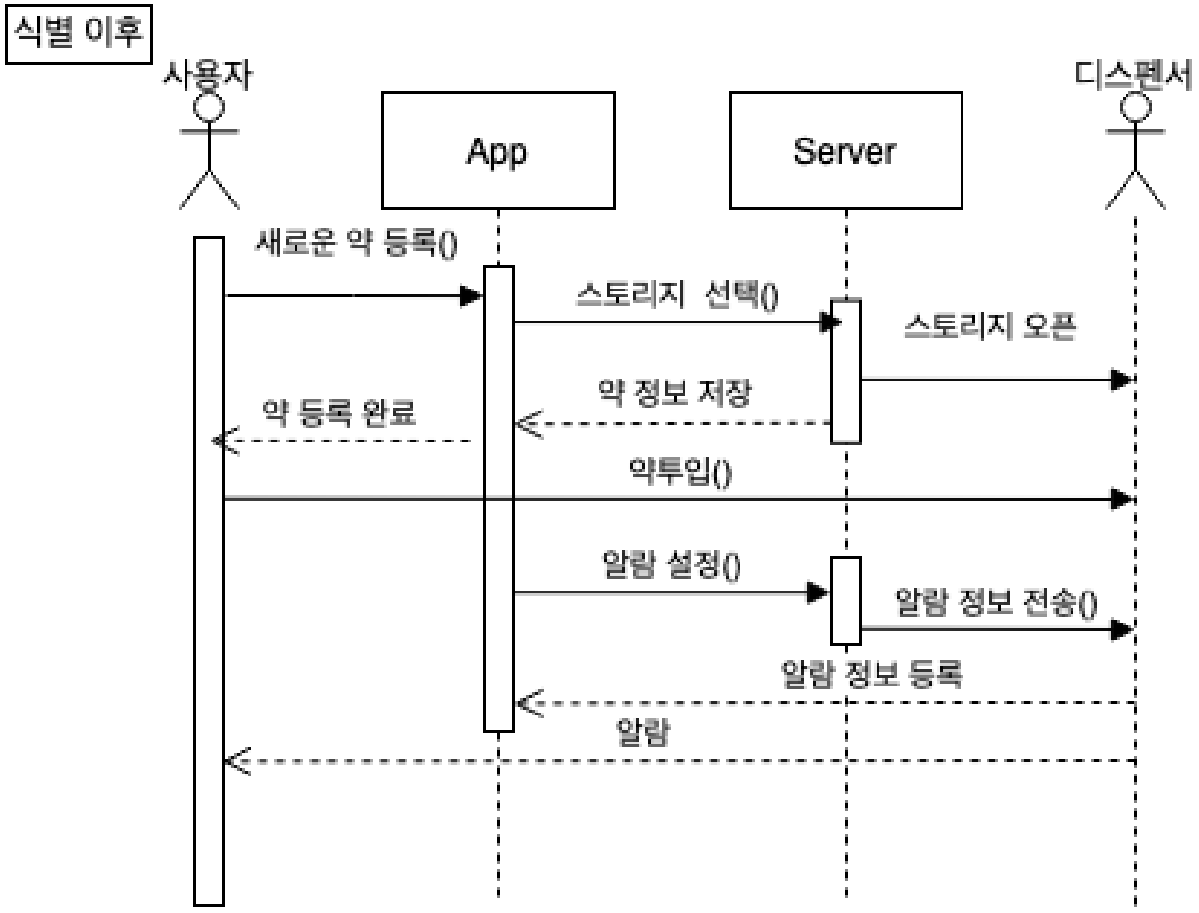

그림 4 Sequence Diagram - 약 등록 Sequence

3.2.3 Use Case Diagram

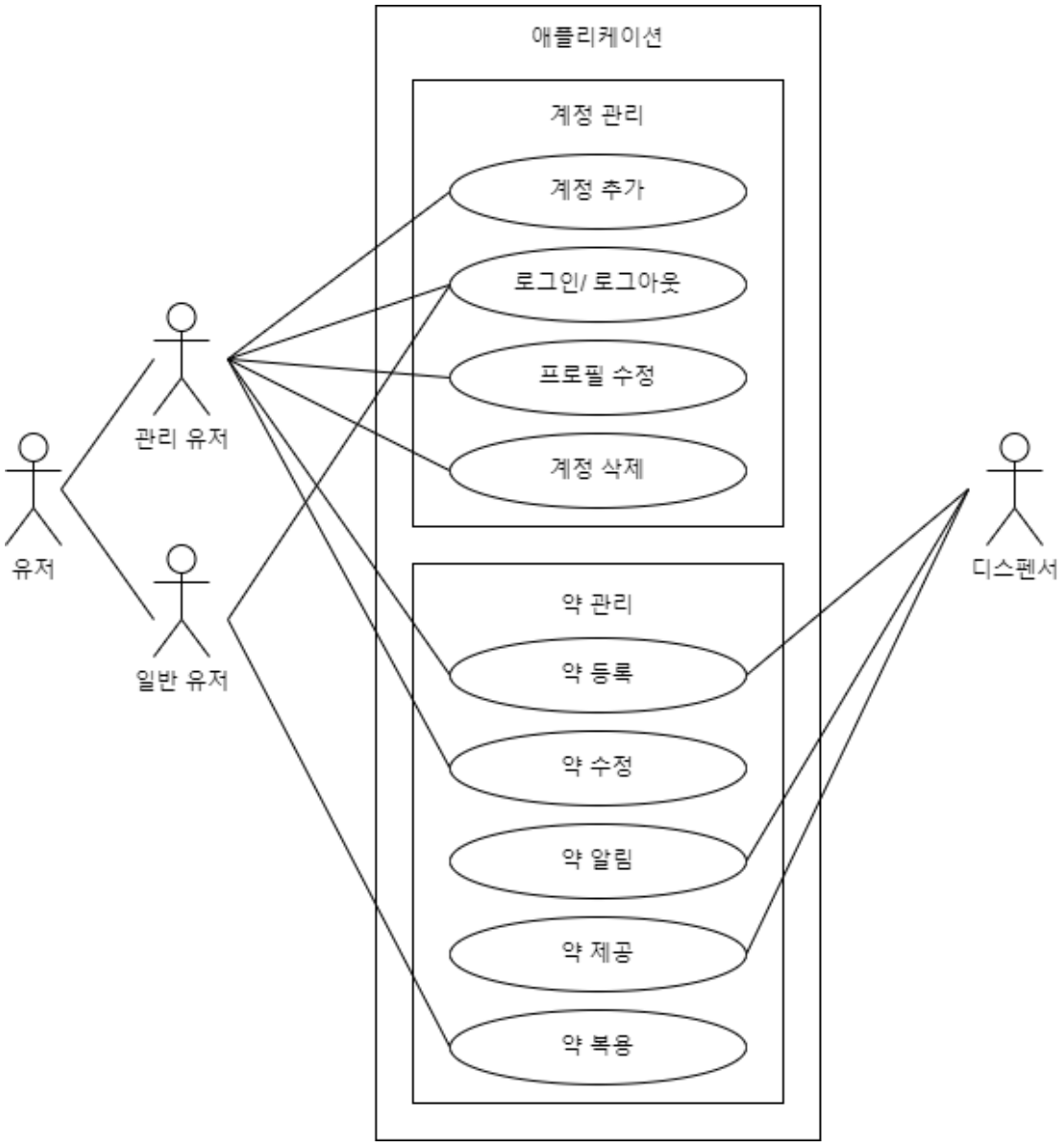

그림 5 Use Case Diagram

4. System Architecture - Frontend

4.1 Objectives

System Architecture에서는 component들 간의 관계를 class diagram과 sequence diagram으로 설명하고, 각 component들의 속성과 method를 명시한다.

4.2 Subcomponents

4.2.1 로그인 및 회원가입

4.2.1.1 Attributes

해당 object가 가진 attribute는 다음과 같다.

- user_id : 사용자의 id이다. id는 unique해야 한다.
- user_pass : 사용자의 비밀번호이다.
- name : 사용자의 이름이다.
- birth : 사용자의 생년월일 정보이다.
- phone : 사용자의 전화번호이다.

4.2.1.2 Methods

해당 object가 가진 methods는 다음과 같다.

- getAccount()
- getProfile()
- setAccount()

4.2.1.3 Class diagram

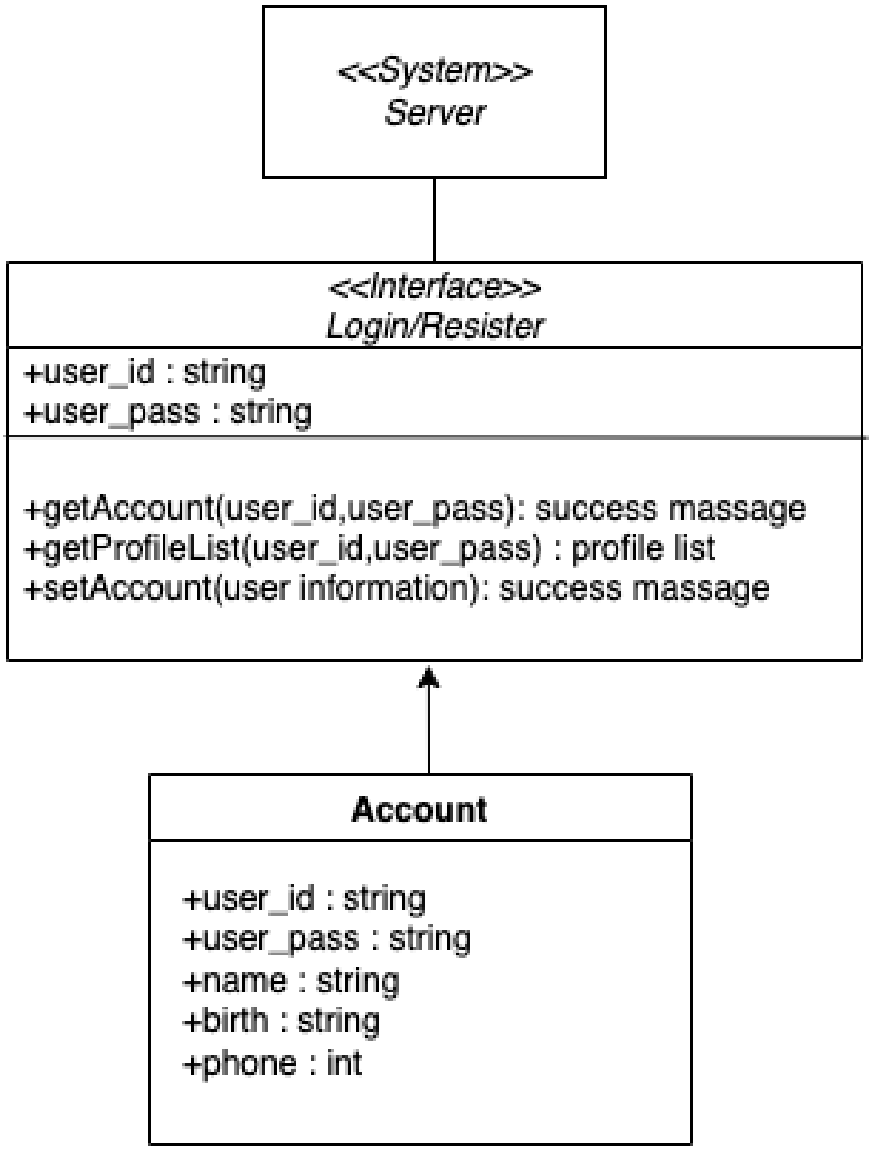

그림 6 로그인 및 회원가입 - Class Diagram

4.2.1.4 Sequence diagram

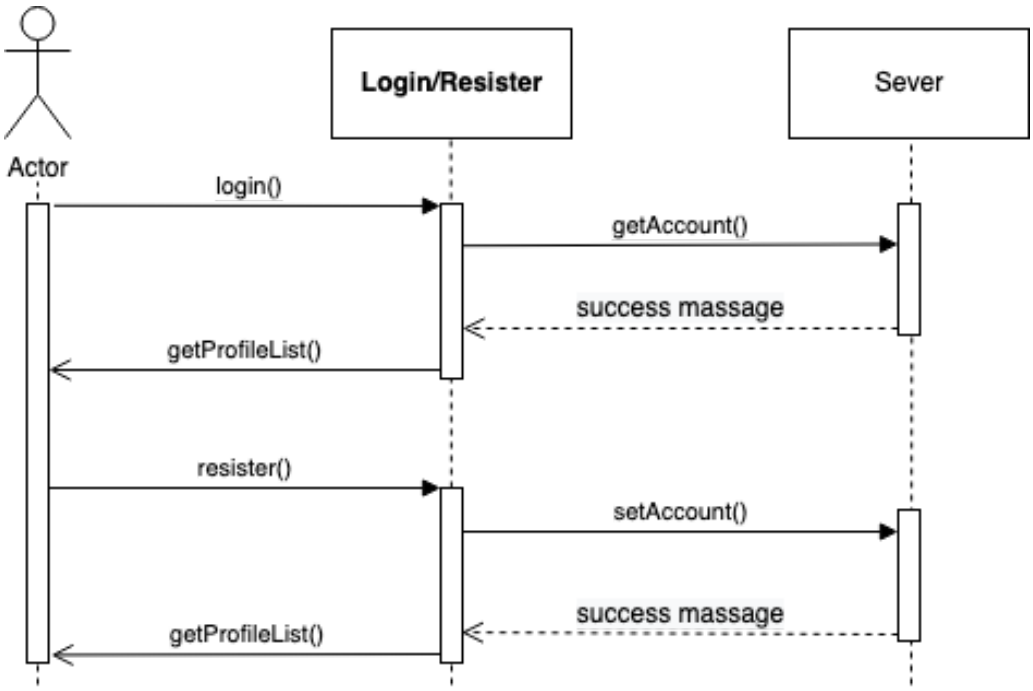

그림 7 로그인 및 회원가입 - Sequence Diagram

4.2.2 계정 프로필 관리

4.2.2.1 Attributes

해당 object가 가진 attributes는 다음과 같다.

- user_id : 현재 사용자의 id이다.
- profile_list : 현재 생성되어 있는 프로필 목록이다.
- profile_name : 각 프로필의 이름이다.

4.2.2.2 Methods

해당 object가 가진 methods는 다음과 같다.

- getProfile()
- getProfileList()
- editProfile()
- deleteProfile()
- createProfile()

4.2.2.3 Class diagram

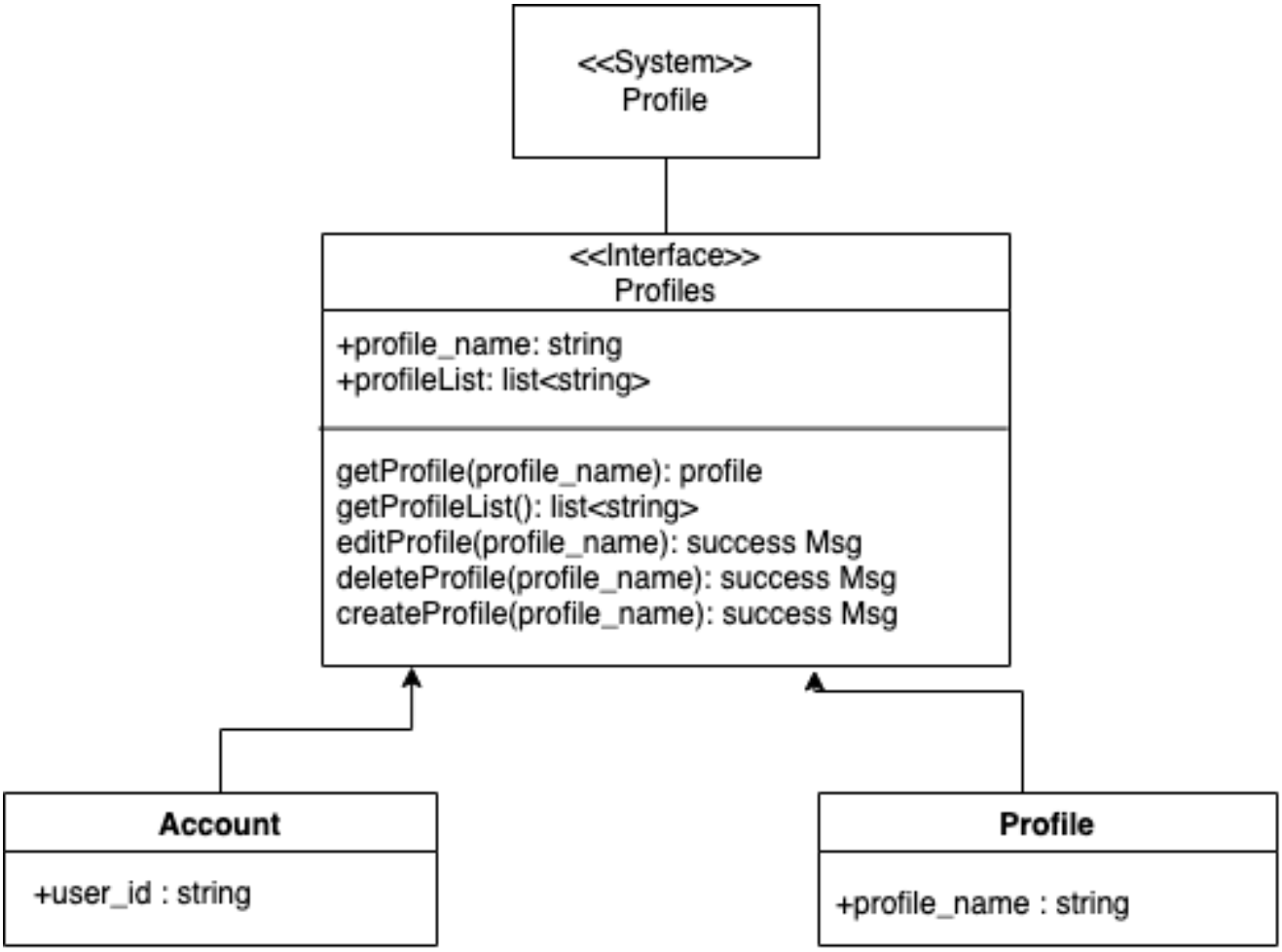

그림 8 계정 프로필 관리 - Use Case Diagram

4.2.2.4 Sequence diagram

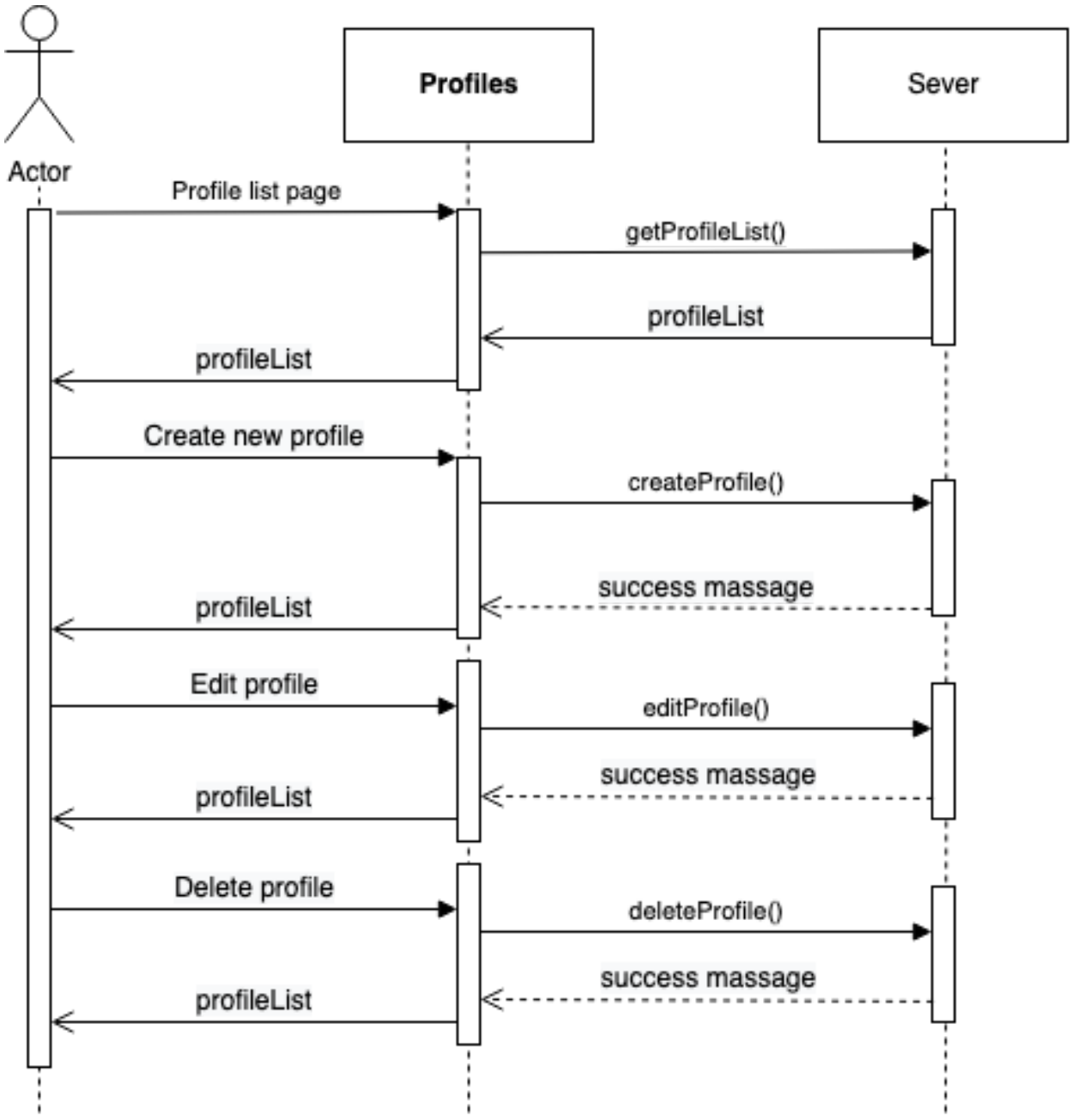

그림 9 계정 프로필 관리 - Sequence Diagram

4.2.3 약 복용 관리

4.2.3.1 Attributes

해당 object가 가진 attributes는 다음과 같다.

- user_id : 현재 사용자의 id이다.
- profile_name: 현재 접속한 profile의 이름이다.
- med_list : 현재 복용중인 약의 목록이다.
- med_name: 생성되어 있는 약의 이름이다.
- med_id: 생성되어 있는 약의 id이다.
- admin: 사용자가 관리자인지 아닌지를 나타내는 속성이다.
- med_time: 해당 약에 설정된 복용 시간을 나타낸다.
- med_storage: 해당 약이 저장된 스토리지 정보를 나타낸다.
- med_user: 해당 약에 접근 권한을 갖는 사용자 정보를 나타낸다.

4.2.3.2 Methods

해당 object가 가진 methods는 다음과 같다.

- getMed()
- newMed()
- deletMed()
- editMed()
- getPermission()
- setStorage()
- setTime()
- setUser()

4.2.3.3 Class diagram

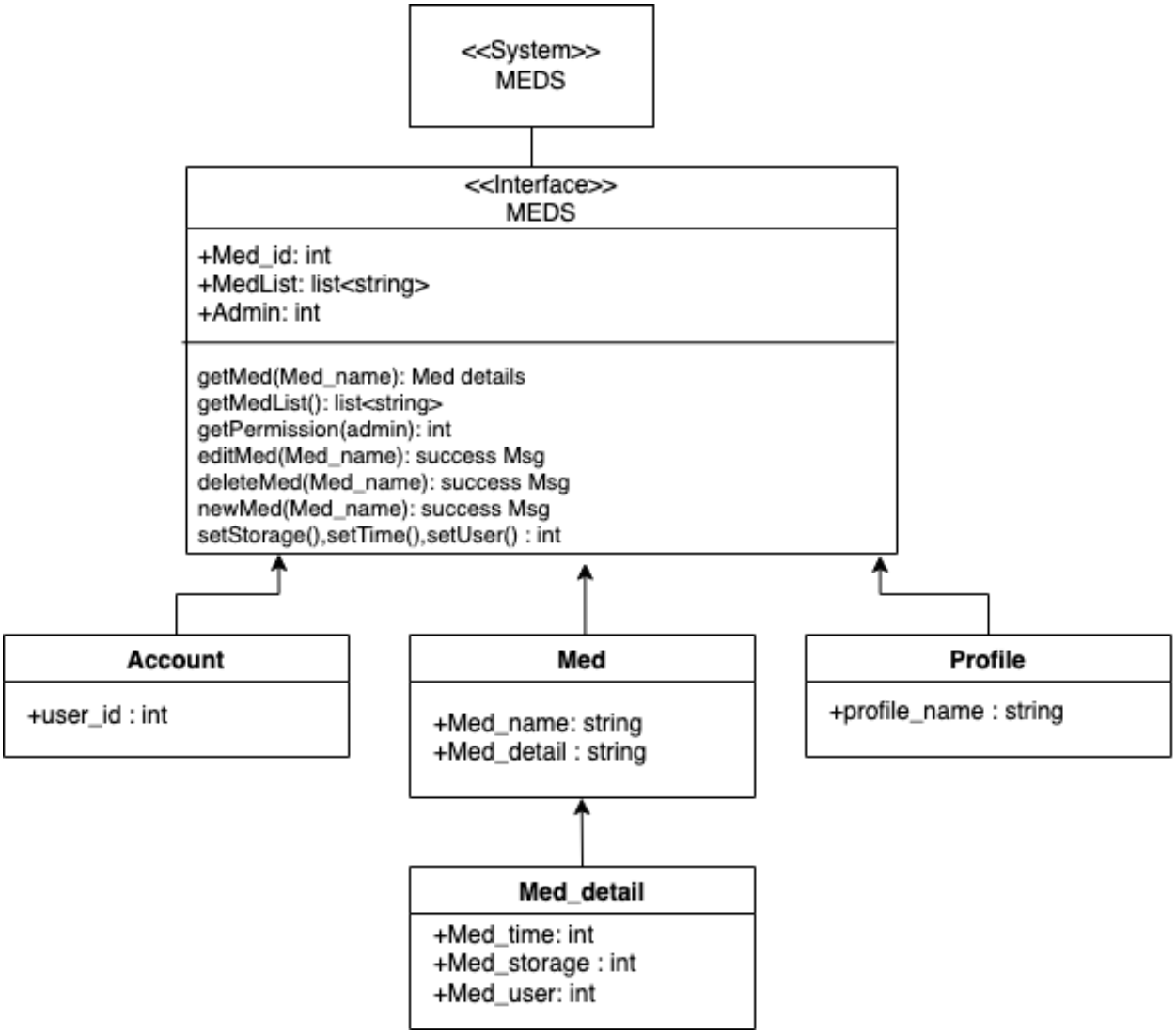

그림 10 약 복용 관리 - Class Diagram

4.2.3.4 Sequence diagram

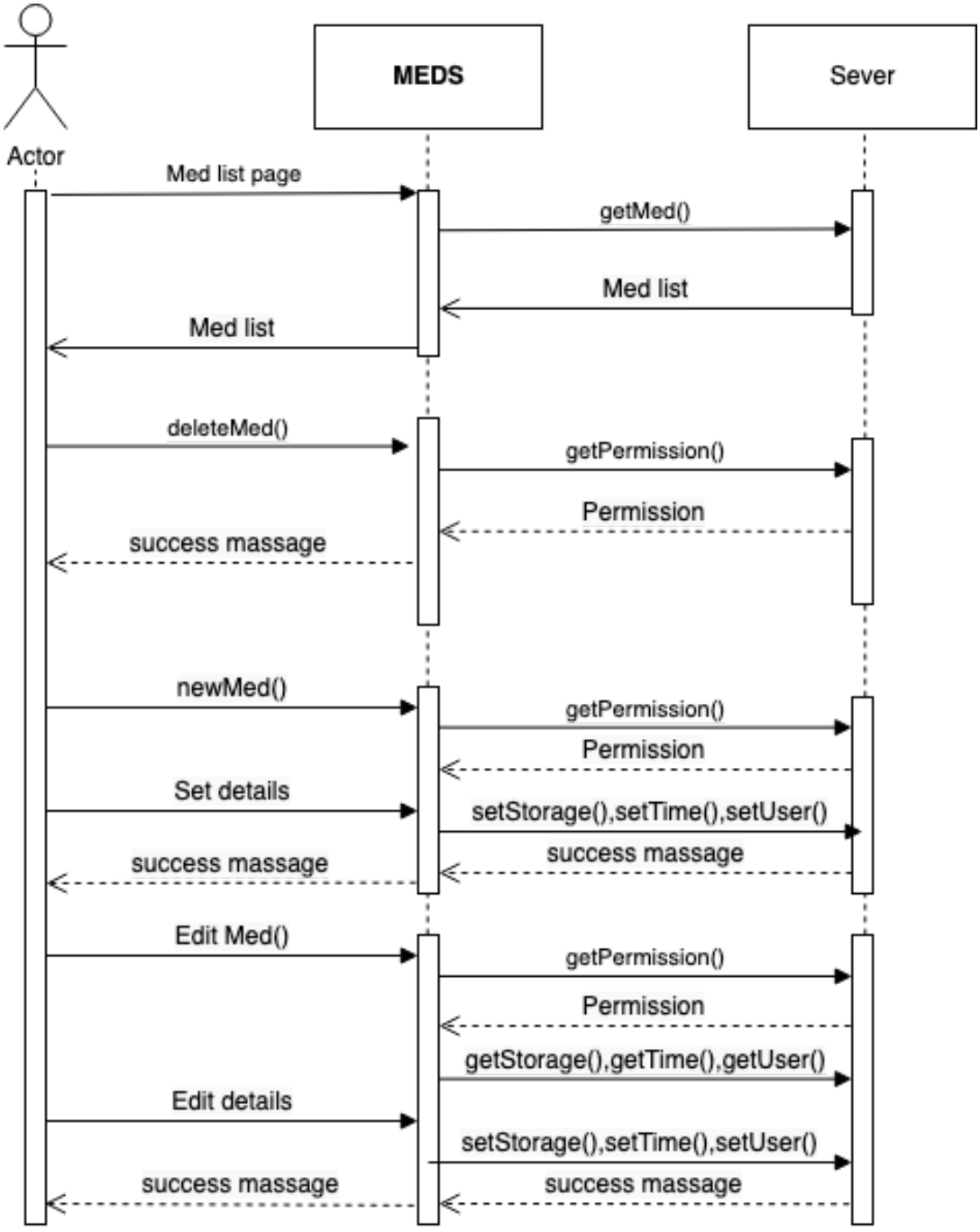

그림 11 약 복용 관리 - Sequence Diagram

4.2.4 알림

4.2.4.1 Attributes

해당 object가 가진 attribute는 다음과 같다.

- alert_id : 알림의 id이다.
- alert_list : 지금까지 온 알림들의 목록이다.
- alert_day : 알림의 날짜 정보이다.
- alert_info : 알림의 내용 정보이다.

4.2.4.2 Methods

해당 object가 가진 methods는 다음과 같다.

- getAlert()

4.2.4.3 Class diagram

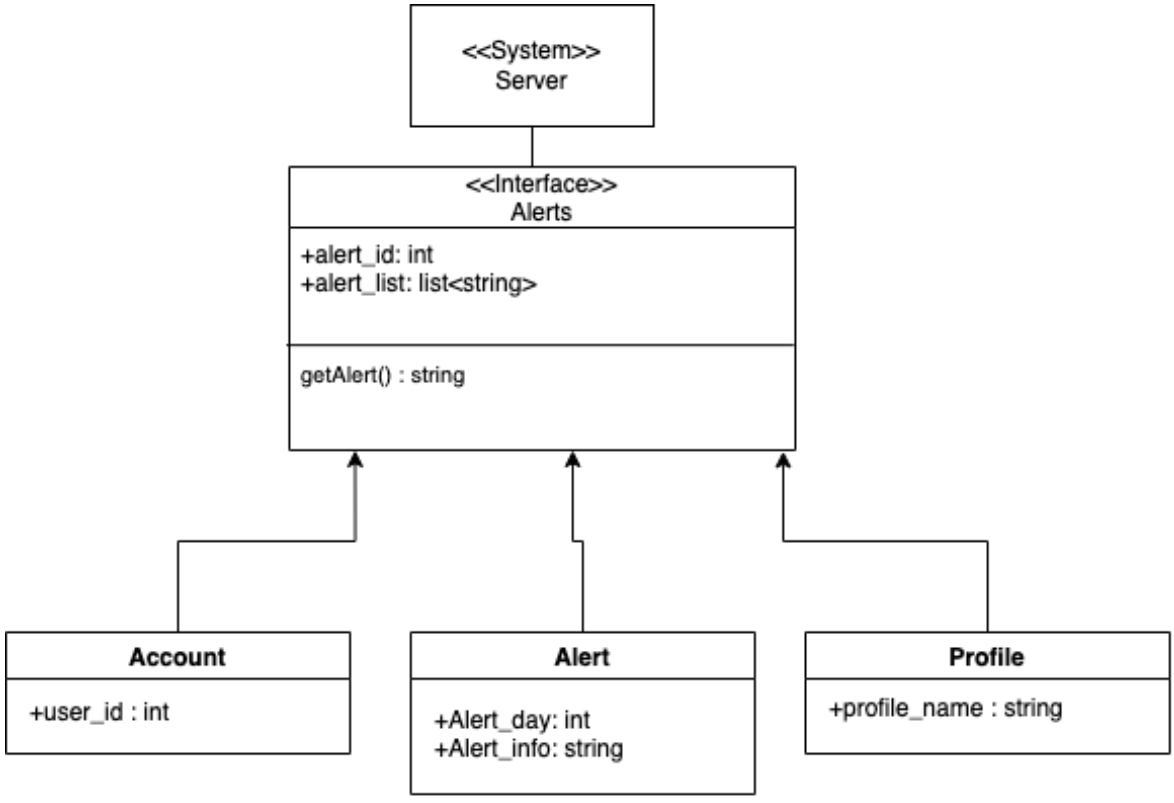

그림 12 알림 - Class Diagram

4.2.4.4 Sequence diagram

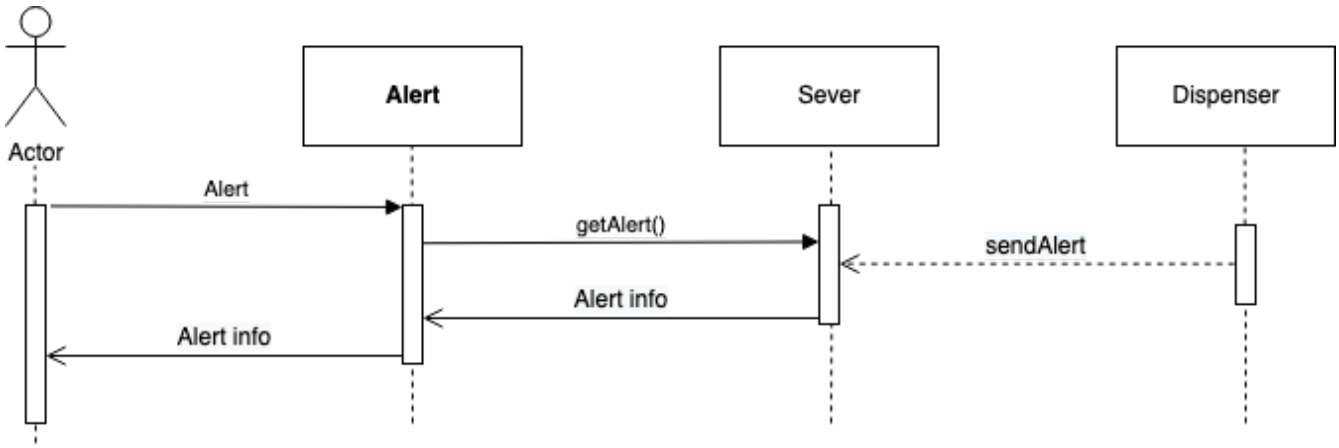

그림 13 알림 - Sequence Diagram

5. System Architecture - Backend

5.1 Objectives

5장에서는 Database, 디스펜서, 스마트폰 디바이스를 연결짓는 백엔드 시스템 아키텍처에 대해서 자세히 다룰 예정이다.

5.2 Overall Architecture

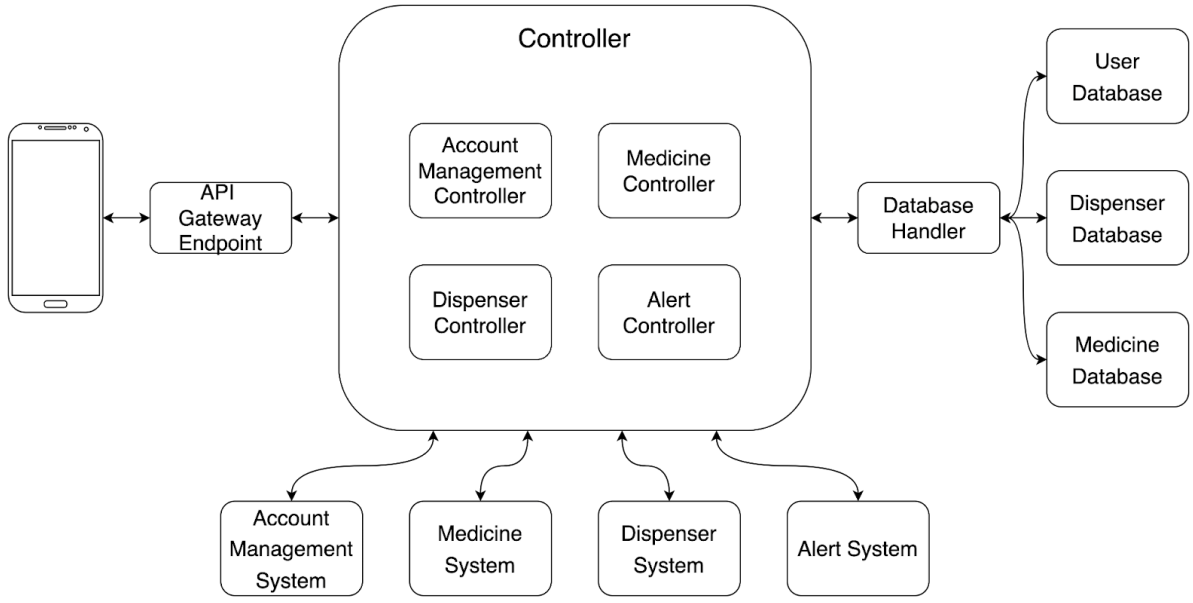

그림 14 Overall Architecture

5.2.1 System

스마트 약 디스펜서의 아키텍처에서 시스템은 4가지로 이루어져 있다. 계정 관리를 다루는 Account Management System, 약에 관련된 부분을 다루는 Medicine System, 디스펜서 기기와 관련된 부분을 다루는 Dispenser System, 그리고 알람에 관련된 내용을 다루는 Alert System이 그 전부이다.

5.2.2 Handler

Database Handler는 데이터베이스를 사용하는 것을 용이하게 하는 역할을 한다.

5.2.3 Controller

Controller는 API Gateway Endpoint, 각각의 시스템 요소들 그리고 데이터베이스 핸들러를 통해 구성 요소들이 연결될 수 있도록 도와주는 역할을 한다.

5.3 Subcomponents

5.3.1 Account Management System

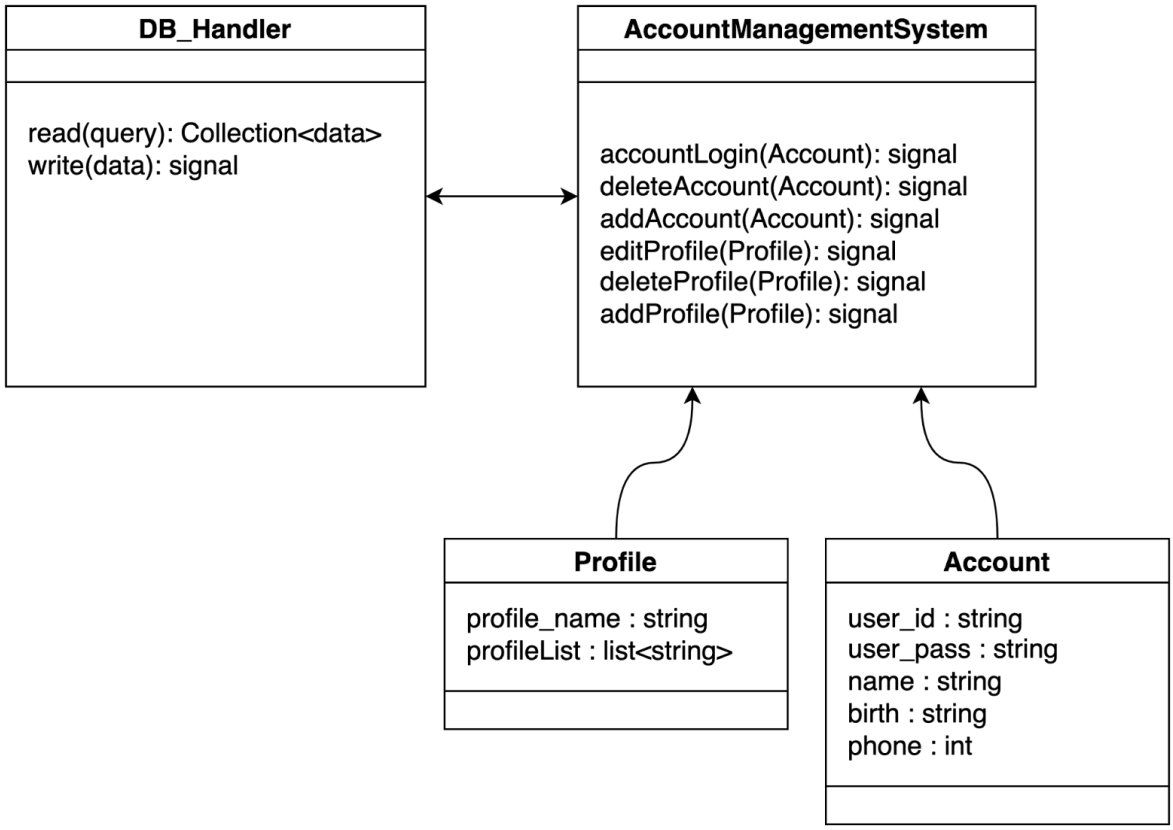

그림 15 Account Management System - Class Diagram

Account Management Class를 통해서 사용자는 계정 정보, accountLogin()함수를 이용해 계정 로그인을 하고 edltProfile()함수를 통해 프로필 정보를 바꿀 수 있다. 성공 여부에 대해서 그 결과를 유저에게 보여준다. 프로필을 삭제 또는 추가를 할 때 각각 deleteProfile(), addProfile()함수를 통해 데이터를 다루고, 다룬 내용을 DB_Handler를 통해 데이터베이스에 변경 사항을 전달한다.

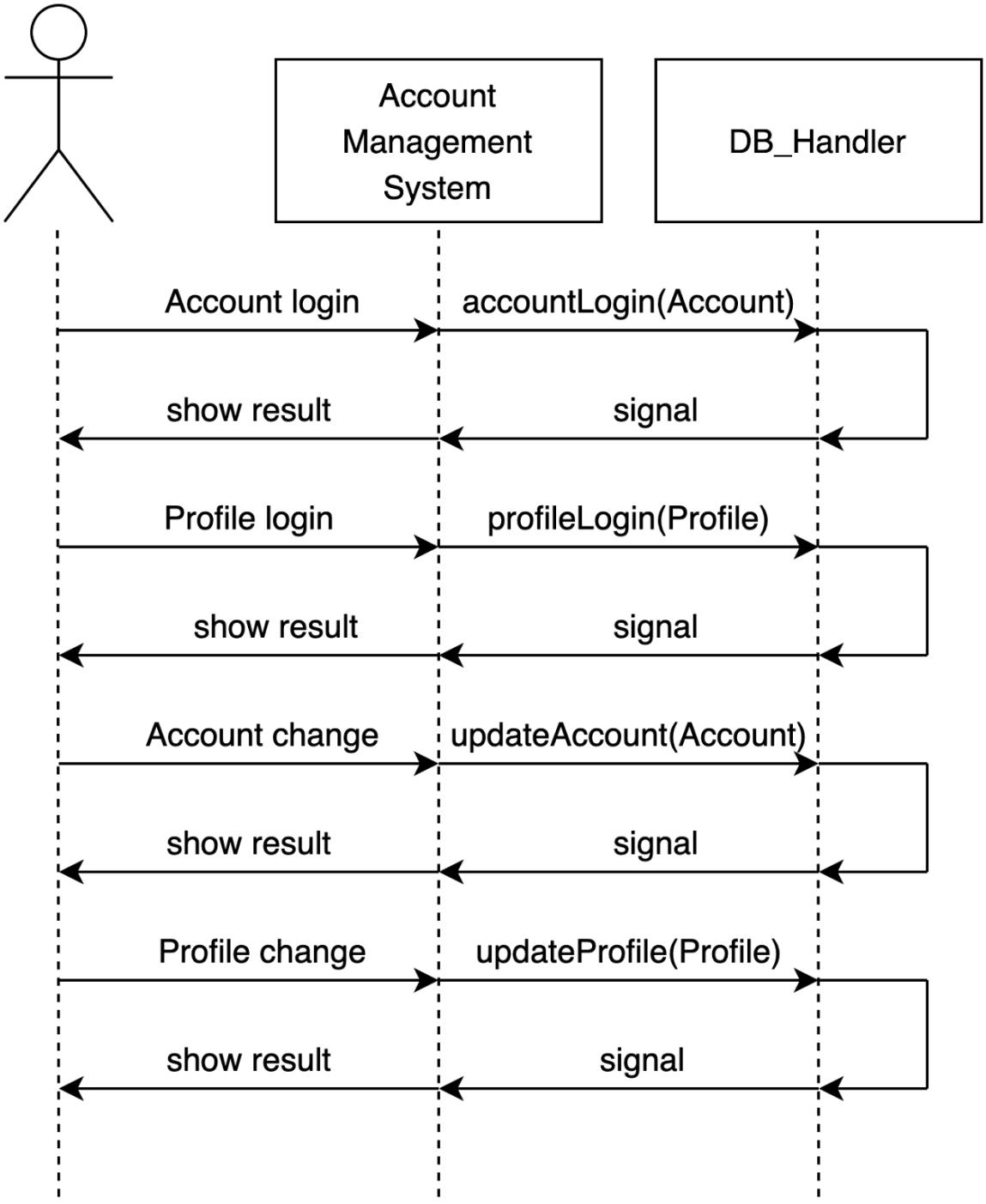

그림 16 Account Management System - Sequence Diagram

5.3.2 Medicine System

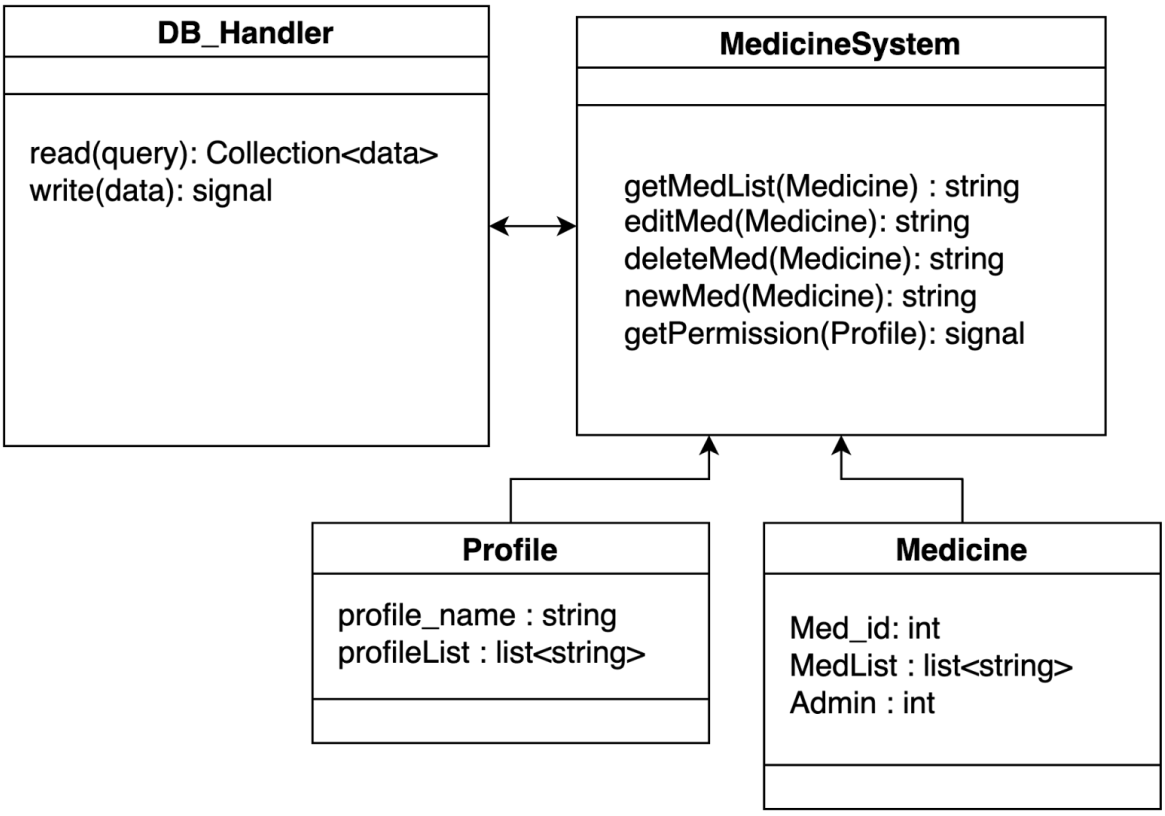

그림 17 Medicine System - Class Diagram

Medicine System은 사용자가 약의 정보에 대해서 생성, 조회, 수정, 삭제하는 것을 도와주는 클래스이다. 사용자가 조회를 원할 때에는 getMedList()함수와 getPermission()함수를 통해 권한을 가진 사용자에 한해 조회가 가능하게 한다. 특정 약을 삭제하고 싶은 경우에는 deleteMed()함수를 통해 삭제하며 삭제의 성공에 대한 시그널을 모바일 디바이스에 전달한다. 새로운 약을 추가하는 경우에는 newMed() 함수를 통해서 약을 추가하기 위해 유저로부터 전달받은 데이터를 데이터베이스에 저장한다. 그 성공 여부에 대한 시그널을 모바일 디바이스에 전달한다.

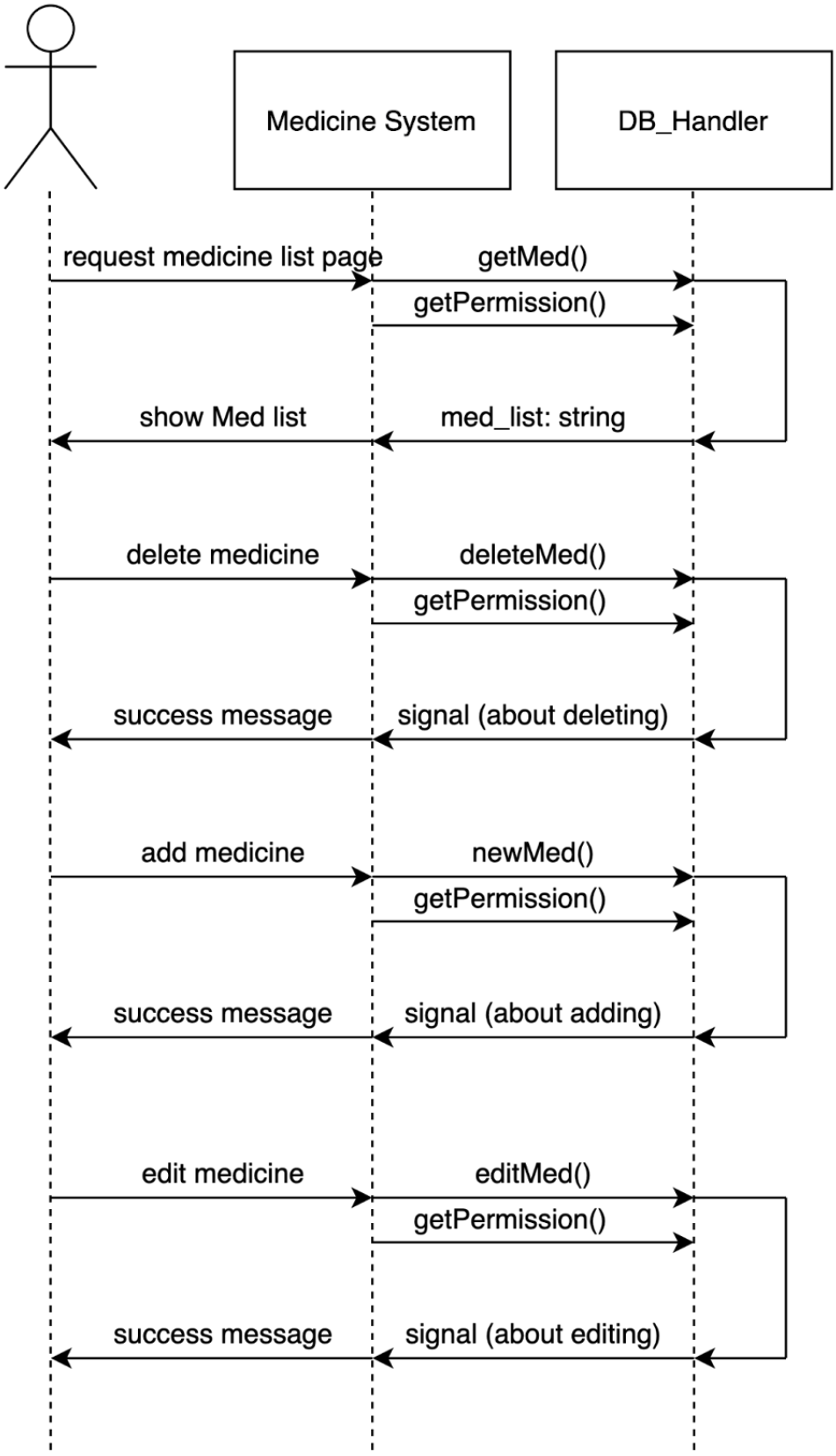

그림 18 Medicine System - Sequence Diagram

5.3.3 Dispenser System

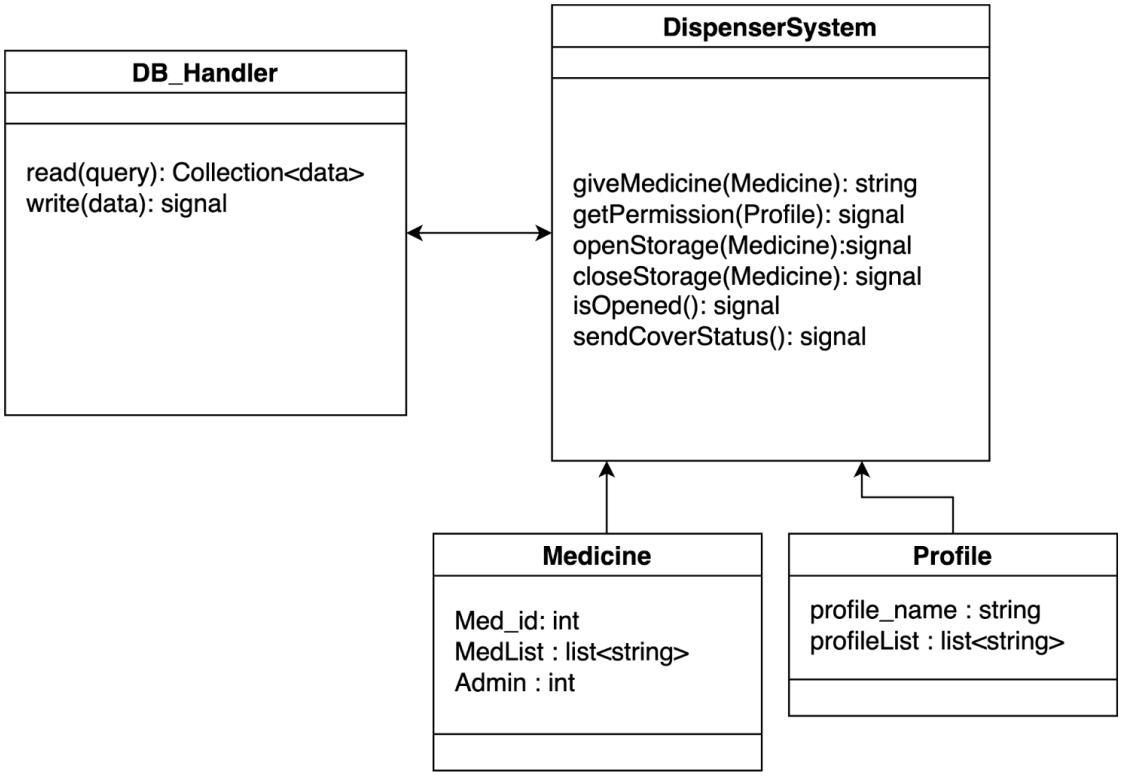

그림 19 Dispenser System - Class Diagram

Dispenser System 클래스에서는 사용자가 약에 대해 요구할 때 giveMedicine(), getPermission() 함수를 통해서 약을 받을 수 있는 사용자인지 확인하고 약을 요청한다. 약을 디스펜서에 채우는 상황에서 openStorage()함수를 통해 출력할 약이 있는지 확인하고 스토리지의 덮개를 열어달라는 요청을 전달하고 그 승낙 여부에 대해 다시 유저에게 전달한다. 사용자가 약을 채워넣은 이후에 덮개를 닫으면 closeStorage()함수를 통해서 그 내용을 DB_Handler로 전달한다. 그 과정에서 덮개가 잘 덮어졌는지의 여부에 대해서 isOpened()함수를 통해 확인하고 그 내용에 대해 Handler로 전달하며 덮개가 닫히는 과정에 이상이 없다는 사실을 저장한다. 마지막으로 덮개의 상태에 대해 sendCoverStatus() 함수를 이용해 DB_Handler로 전달한다.

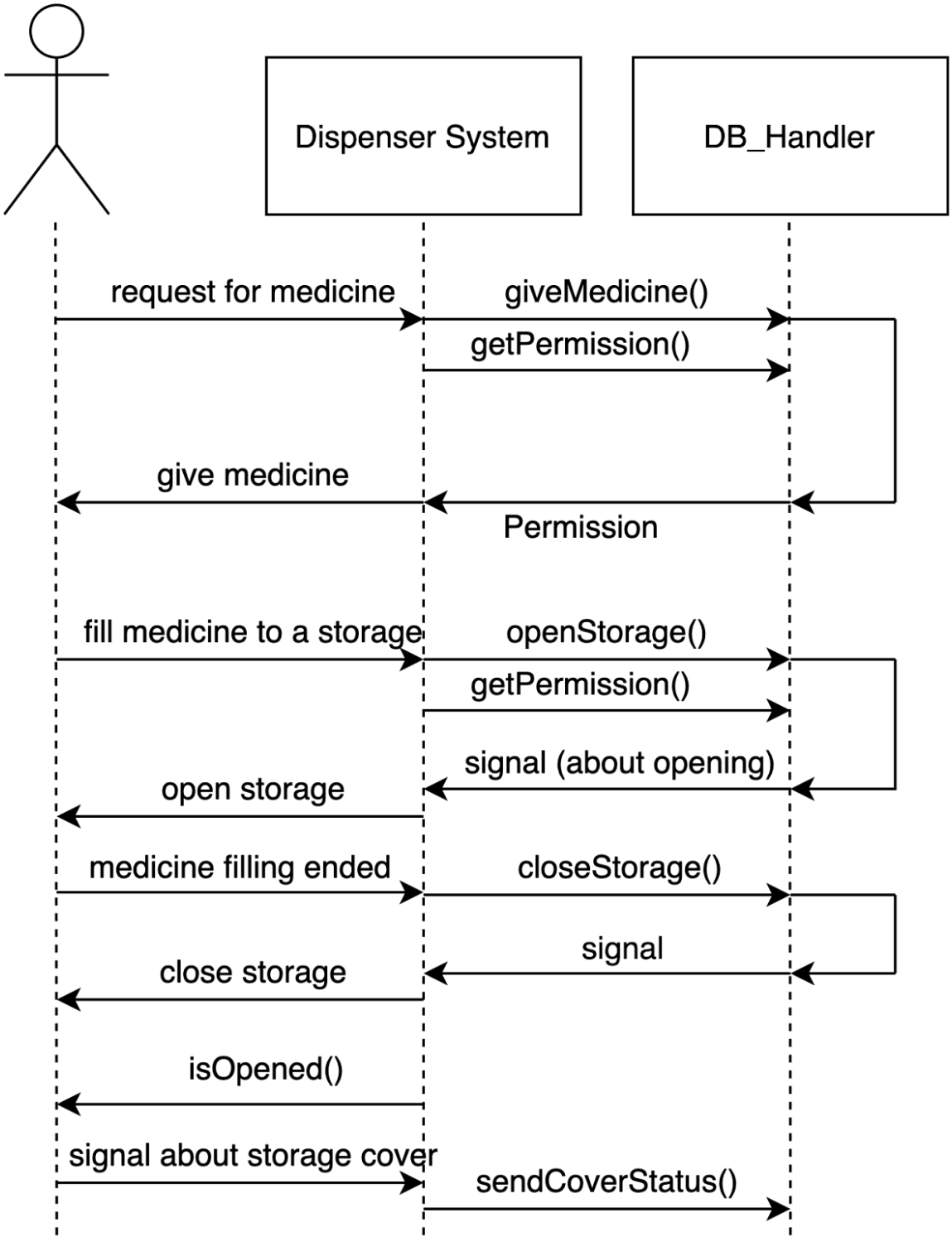

그림 20 Dispenser System - Sequence Diagram

5.3.4 Alert System

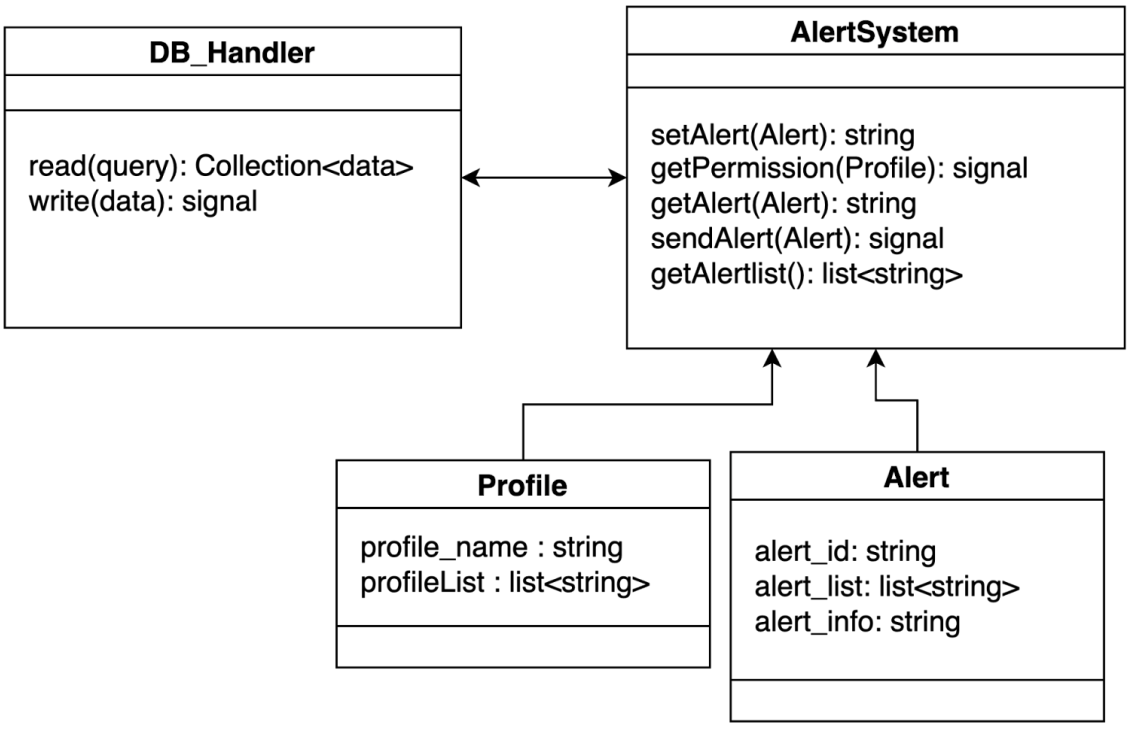

그림 21 Alert System - Class Diagram

Alert System Class에서는 디스펜서, 모바일 디바이스에서 알림을 내보내고 설정하는 모든 작업을 주관한다. User가 어플리케이션에서 알림을 설정하는 경우에 setAlert() 함수를 통해 DB_Handler에 설정된 알림을 전달한다. 이후 알림이 필요한 경우에 getAlert() 함수를 통해서 DB에 있는 알림 내용들을 읽어와서 필요한 경우 sendAlert() 함수를 통해 디스펜서나 모바일 디바이스에서 알림을 내보낼 수 있도록 한다. getAlertlist() 함수를 사용해서 설정된 모든 알림들을 확인할 수 있다.

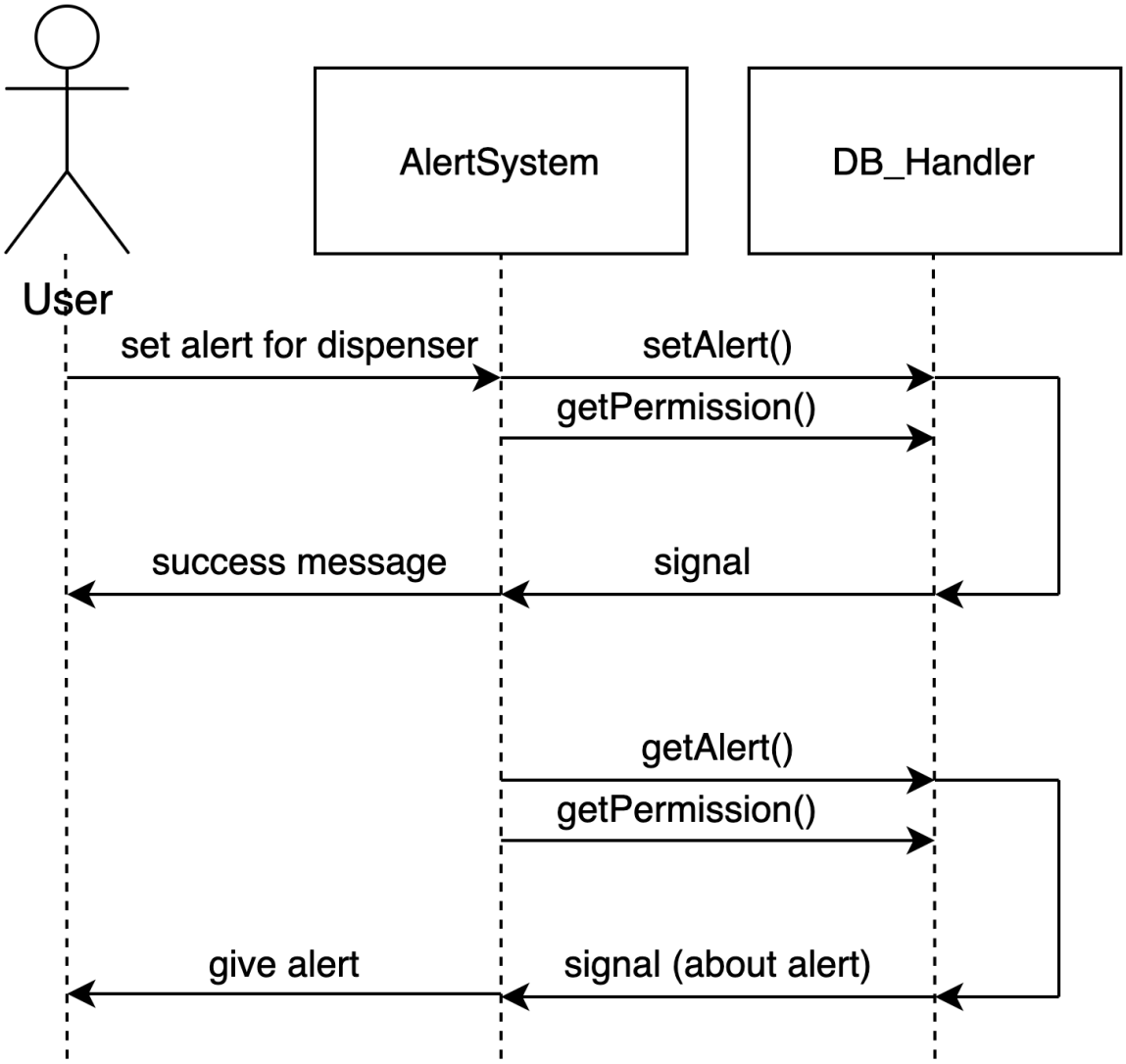

그림 22 Alert System - Sequence Diagram

6. Protocol Design

6.1 Objectives

Protocol design에서는 약 디스펜서 애플리케이션과 서버 간의 HTTP 통신에 있어서 어떤 방식으로 데이터를 주고받는지에 대해 설명한다.

6.2 전달 형식

6.2.1 HTTP

HTTP는 인터넷에서 정보를 주고 받을 때 서버, 클라이언트가 소통을 원활하게 할 수 있도록 도와주는 규칙 체계다. 클라이언트가 알맞은 형식에 맞게 요청(requests)를 서버에 보내면 서버는 클라이언트에게 알맞은 응답(responses)을 보낸다. 본 프로젝트에서 클라이언트는 스마트폰 디바이스이고 서버는 firebase다. 요청은 크게 start-line, header, body로 나뉘는데 start-line에서는 서버가 특정 동작을 하도록 하는 지시를 담는다. header에서는 요청에 대한 구체적인 정보를 담고 body는 주로 보낼 데이터를 담는 역할을 한다. 응답도 status-line, header, body로 나뉘며 비슷한 역할을 한다. 본 프로젝트에서는 HTTP를 다루며 JSON의 형태로 주로 정보를 주고 받는다.

6.2.2 JSON

JSON은 주로 문자열 기반의 데이터 포멧이다. JSON을 사용하면 데이터를 더 일관성 있고 쉽게 주고 받을 수 있다. 주로 JSON 객체의 형태로 많이 쓰이며 그 형식은 다음과 같다.

```
{
    "name" = '홍길동',
    "age" = 30,
    "ID" = 123456
}
```

이를 키-값 패턴이라고도 한다. 본 프로젝트에서는 주로 키-값의 구조를 가지는 JSON 객체의 형태로 데이터를 주고 받는다.

6.3 Authentication

6.3.1 로그인 및 회원가입

6.3.1.1 회원가입

HTTP Request

```
post {url}/user/registration
```

Request Body

Parameters	
name	이름 (string)
user_id	아이디 (string)
user_pass	비밀번호 (특수문자 포함 7자 이상 15 이하)
phone	휴대폰 번호 (string)

JSON Representation

```
{
  “registration”:
    {
        “name”: “홍길동”
        “user_id”: “qwerty12345”
        “user_pass”: “ghdrlfehd12!”
        “phone”: “01098761234”
    }
}
```

Response Body

Parameters	
Code	회원가입 성공 여부 (1000: 성공, 1001: 양식 불일치 1002: 이미 존재하는 아이디 1003: 휴대전화 미인증)
Authorization	회원 고유 번호

JSON Representation

```
{
    “Code”: “1000”
    “Authorization”: “id1abcde31”
}
```

6.3.1.2 로그인

HTTP Request

```
post {url}/user/login
```

Request Body

Parameters	
user_id	아이디 (string)
user_pass	비밀번호 (특수문자 포함 7자 이상 15 이하)

JSON Representation

```
{
  “login”:
    {
        “user_id”: “qwerty12345”
        “user_pass”: “ghdrlfehd12!”
    }
}
```

Response Body

Parameters	
Code	로그인 성공 여부 (1000: 성공, 1001: 비밀번호 불일치 1002: 존재하지 않는 아이디)
Authorization	회원 고유 번호

JSON Representation

```
{
    “Code”: “1000”
    “Authorization”: “id1abcde31”
}
```

6.3.2 계정 프로필 관리

HTTP Request

```
post {url}/user/profileselect
```

Request Body

Parameters	
profile_name	프로필 이름 (string)
Modify	수정/선택화면 옵션 (0: 선택화면, 1: 수정화면 컨텍스트 박스 생성)

JSON Representation

```
{
  “profileselect”:
    {
        “profile_name”: “홍길동”
        “Modify”: “0”
    }
}
```

Response Body

Parameters	
Code	프로필 접속 성공 여부 (1000: 접속 성공 1001: 접속 실패 - 인터넷 미연결)
Token	프로필 접속 성공 시

JSON Representation

```
{
    “Code”: “1000”
    “Token”: “profilename1”
}
```

6.4 약 복용 관리

6.4.1 약 등록 및 수정

6.4.1.1 기본 정보 입력

HTTP Request

```
post {url}/med/register/basicinfo
```

Request Body

Parameters	
med_name	약 이름 (string)
med_time	약 시간 (string)
med_amount	약 용량 (integer)

JSON Representation

```
{
   “basicinfo”:
   {
      “med_name” : “소화제”,
      “med_time” : “2022-06-03-20-30”,
      “med_amount” : “10”
   }
}
```

Response Body

Parameters	
Code	약 등록 성공 여부 (integer) (1000: 성공, 1001: 정보 미기입 1002: 용량 초과)

JSON Representation

```
{
   Code: “1000”
}
```

6.4.1.2 스토리지 지정

HTTP Request

```
post {url}/med/register/storage
```

Request Body

Parameters	
med_name	약 이름 (string)
med_storage	약 저장공간 (string)

JSON Representation

```
{
   “storage”:
   {
      “med_name” : “소화제”,
      “med_storage” : “A05”
   }
}
```

Response Body

Parameters	
Code	약 등록 성공 여부 (integer) (1000: 성공, 1001: 저장공간 미선택)

JSON Representation

```
{
   Code: “1000”
}
```

6.4.1.3 복용 시간 설정

HTTP Request

```
post {url}/med/register/time
```

Request Body

Parameters	
med_name	약 이름 (string)
med_time	약 시간 (string)

JSON Representation

```
{
   “time”:
   {

      “med_name” : “소화제”,
      “med_time”: [
         “2022-06-03-20-30”,
         “2022-06-04-20-30”,
         “2022-06-05-20-30”,
      ]
   }
}
```

Response Body

Parameters	
Code	복용시간설정 성공여부 (integer) (2001: 성공, 2002: med_name 값 오류, 2003: med_time 값 오류)

JSON Representation

```
{
   “Code” : “2001”
}
```

6.4.2 약 삭제

HTTP Request

```
del {url}/med/delmed
```

Request Body

Parameters	
med_id	약 아이디 (int)
med_name	약 이름(string)

JSON Representation

```
{
   “delmed”:
   {
      “med_name” : “소화제”,
      “med_id”: a021398
   }
}
```

Response Body

Parameters	
Code	성공여부 (integer) (1000 : 성공, 1001 : 실패, 권한없음)

JSON Representation

```
{

   “Code” : “1001”

}
```

6.5 약 알림 확인

HTTP Request

```
get {url}/alert/alertlist
```

Request Body

Parameters	
user_id	유저아이디

JSON Representation

```
{
   “alertlist”:
   {
      “user_id” : “홍길동”,
    }
}
```

Response Body

Parameters	
code	성공여부 (integer) (1000 : 성공, 1001 : 실패, 권한없음)
alert_day	알람이 온 날짜
alert_info	각 알람들의 내용
alert_list	현재 알람들의 목록

JSON Representation

```
{
   code : 1000
   alert_list : [
         alert_day :20220501
         alert_info : “홍길동님이 약을 복용하셨습니다"
      ]
}
```

---

7. Database Design

7.1 Objectives

이 장은 데이터베이스 구조와 표현을 기술한다. 각 데이터베이스 내의 엔티티들과 그 사이의 관계를 ER 다이어그램과 관계 스키마로 표현하고, 실제 데이터베이스 구현을 위한 SQL DDL을 작성한다.

7.2 ER Diagram

스마트 약 디스펜서 데이터는 3 개의 엔티티로 구성된다. 유저, 약, 디스펜서로 세 개의 데이터베이스가 존재하며 아래의 ER 다이어그램이 이를 표현한다.

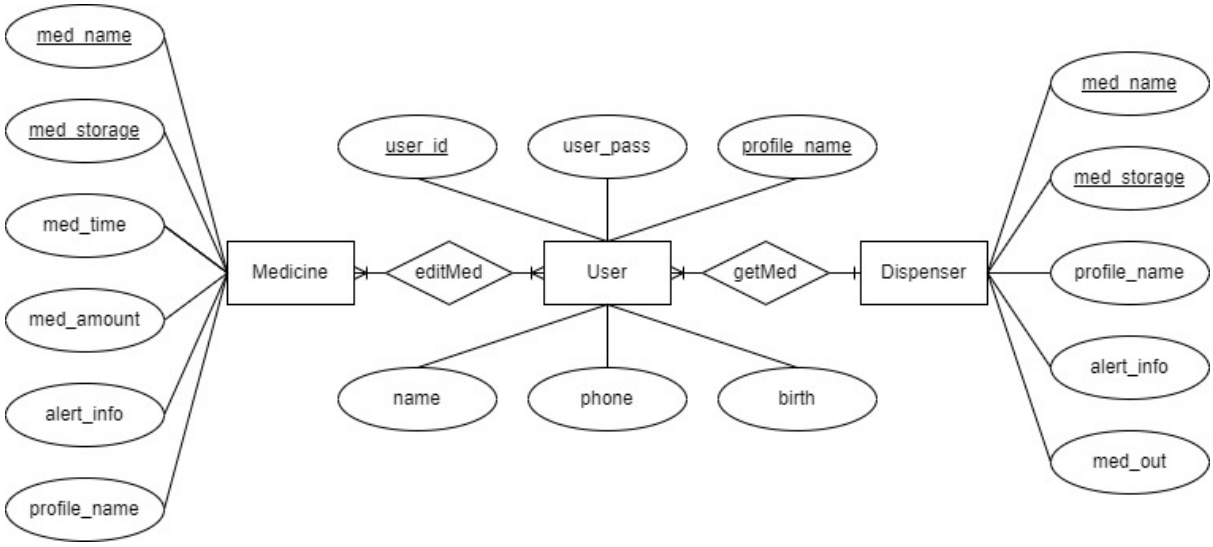

그림 23 ER Diagram

7.2.1 Entities

7.2.1.1 User Entity

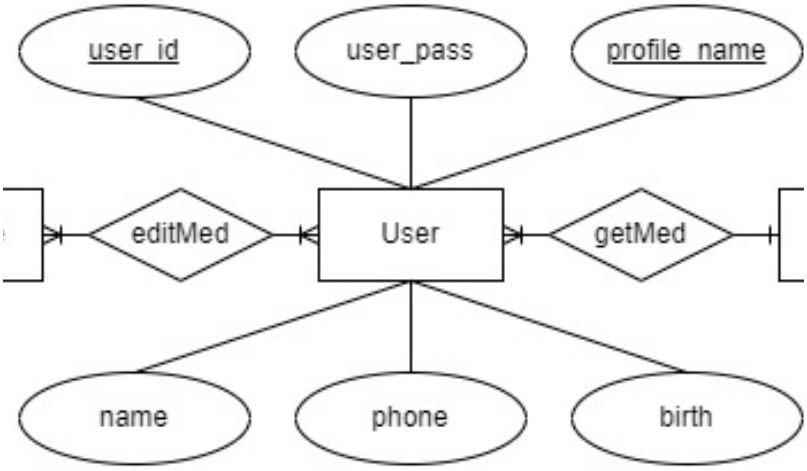

그림 24 User Entity

User entity는 유저에 관한 정보를 저장한다. 유저의 아이디, 비밀번호, 프로필 이름, 이름, 전화번호, 생년월일을 포함한다. user_id와 profile_name은 기본키이며 중복이 존재할 수 없다. 한 명 이상의 User는 한 개 이상의 Medicine 엔티티와 한 개의 Dispenser를 가질 수 있다.

7.2.1.2 Medicine Entity

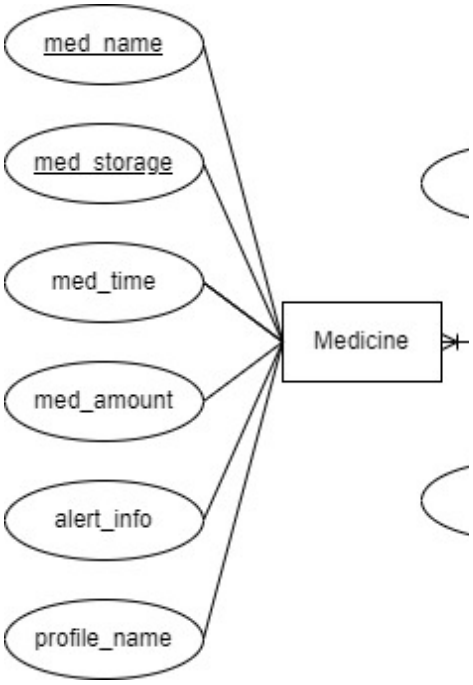

그림 25 Medicine Entity

Medicine entity는 약에 대한 정보를 저장한다. 약의 이름, 스토리지 저장 공간, 복용되는 빈도수, 남은 양, 알림 설정, 프로필 이름을 포함한다. med_name과 med_storage는 기본키이며 중복이 존재할 수 없다. profile_name을 user entity로부터 외래키로 받아온다. 한 개 이상의 Medicine은 한 명 이상의 User와 관계를 맺을 수 있다.

7.2.1.3 Dispenser Entity

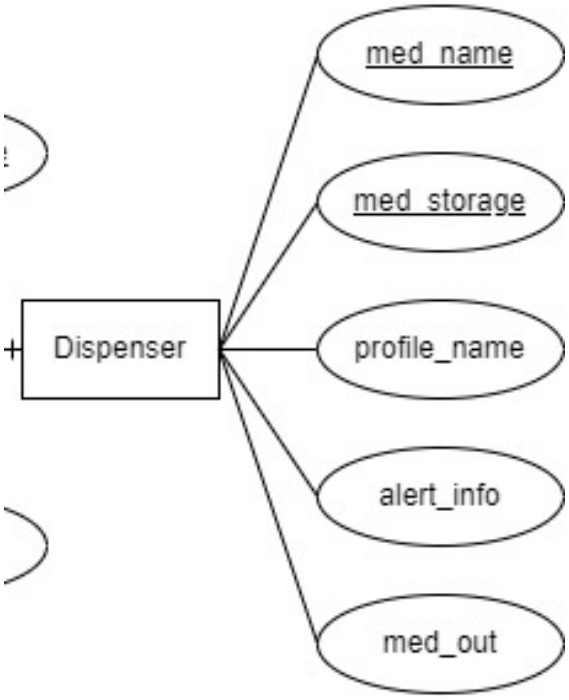

그림 26 Dispenser Entity

Dispenser entity는 디스펜서에 사용에 필요한 정보를 저장한다. 약의 이름, 스토리지 저장 공간, 프로필 이름, 알림 설정, 약 배출의 여부를 표현한다. med_name과 med_storage는 기본키이며 중복될 수 없다. 약 이름과 스토리지 저장 공간, 알림 설정을 medicine entity에서 외래키로 받아오며, 프로필 이름을 user entity에서 외래키로 받아온다. 하나의 디스펜서와 한 명 이상의 유저가 관계를 맺을 수 있다.

7.3 Relational Schema

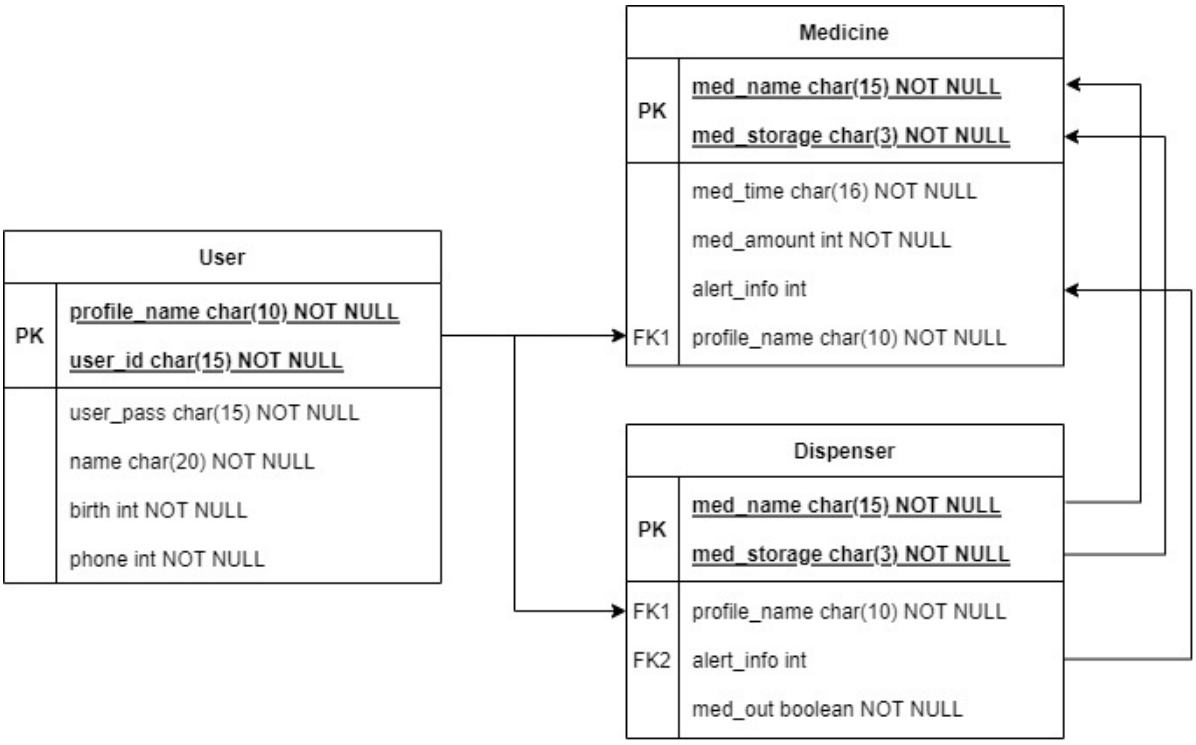

그림 **27** Relational Schema

7.4 SQL DDL

7.4.1 User

```
CREATE TABLE User
(
    profile_name CHAR(10) NOT NULL,
    user_id CHAR(15) NOT NULL,
    user_pass CHAR(15) NOT NULL,
    name CHAR(20) NOT NULL,
    birth INT NOT NULL,
    phone INT NOT NULL,

    PRIMARY KEY (profile_name),
    PRIMARY KEY (user_id)
);
```

7.4.2 Medicine

```
CREATE TABLE Medicine
(
    med_name CHAR(15) NOT NULL,
    med_storage CHAR(3) NOT NULL,
    med_time CHAR(16) NOT NULL,
    med_amount INT NOT NULL,
    alert_info INT,
    profile_name CHAR(10) NOT NULL,

    PRIMARY KEY (med_name),
    PRIMARY KEY (med_storage),
    FOREIGN KEY (profile_name) REFERENCES
User(profile_name)
);
```

7.4.3 Dispenser

```
CREATE TABLE Dispenser
(
    med_name CHAR(15) NOT NULL,
    med_storage CHAR(3) NOT NULL,
    profile_name CHAR(10) NOT NULL,
    alert_info INT,
    med_out boolean NOT NULL,

    PRIMARY KEY (med_name),
    PRIMARY KEY (med_storage),
    FOREIGN KEY (profile_name) REFERENCES
User(profile_name),
    FOREIGN KEY (med_name) REFERENCES
Medicine(Med_Name),
    FOREIGN KEY (med_storage) REFERENCES
Medicine(Med_Storage),
    FOREIGN KEY (alert_info) REFERENCES
Medicine(alert_info)
);
```

8. Testing Plan

8.1 Objectives

이 챕터에서는 개발 시험, 릴리스 시험 및 사용자 시험의 세 가지 주요 하위 그룹을 포함하는 시험 계획을 설명한다. 이러한 테스트는 제품의 잠재적 오류와 결함을 감지하고 결점이 없는 완전한 작동과, 신뢰할 수 있는 제품을 시장과 고객 출시를 보장한다는 점에서 매우 중요하다.

8.2 Testing Policy

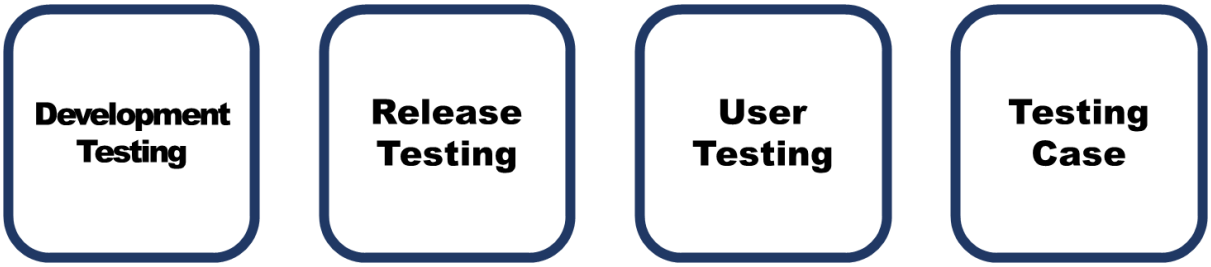

그림 28 Testing Policy의 요소

8.2.1 Development Testing

Development test 는 앞으로 생길 여러 잠재적인 위협들을 방지하고, 시간/금전적 비용을 경감하며, 넓은 범주에서 문제사항들을 예방하고 발견하는 전략으로써 진행된다. 이 단계의 스마트 약 디스펜서는 충분한 테스트를 거치지 못하였기 때문에 불안정할 수 있고, 구성 요소들간에 충돌이 발생할 수 있다. 따라서, 정적 코드 분석, 데이터 플로우 분석, 피어 코드 리뷰, 단위 테스트 등의 여러 테스트가 이루어져야 한다. 이런 과정들을 통해 성능, 신뢰도, 보안성 등의 수준을 충분히 끌어올린다.

8.2.1.1. Performance

스마트 약 디스펜서가 기존의 알약 디스펜서와 가장 큰 차이를 나타내는 부분은 음성인식 기능, 다양한 사용자의 등록 가능성, 상비약의 보관 가능성 등이므로, 알약 등록이나 알약 스토리지 오픈 등에 딜레이가 과해서는 안 될 것이다. 통신상태에 문제가 없다는 가정하에, 기본적으로 앱의 실행 이후 3 초 이내 로그인 화면에 진입할 수 있도록 앱 최초 실행 과정에서 최적화에 신경을 써야 한다. 또한 로그인 프로세스는 1.5초 이내에 진행되어야 하고, 각 기능의 선택에 따른 화면의 전환과 데이터의 표시는 1초 이내에 이루어질 수 있도록 최적화를 해야 한다. 또한 스마트 약 디스펜서에는 기존의 디스펜서들의 알약 저장 기능이 여전히 남아있고, 관련해서 업로드할 강의 영상의 인코딩은 1시간 강의 기준 1분 이내에 완료되어야 한다.

이런 요구 사항들은 애플리케이션 자체적인 최적화 및 개별 기기의 하드웨어적 스펙에 따라 좌우되는 부분이므로, 애플리케이션의 최소 기기 요구사양인 2GB RAM, 1.6GHz CPU, 32GB 저장공간, Android 6.0 또는 iOS 11.0 이상을 만족하는 기기를 이용하여 테스트가 진행되어야 할 것이며, 이 기기들을 기준으로 위의 제한 시간 내에 실행이 완료되도록 최적화되어야 할 것이다. 또한 모든 알약 관련 자료의 업로드는 최소 10Mbps 에서 최대 100Mbps 의 속도를 지원해야 하고, 다운로드는 10Mbps에서 200Mbps의 속도를 지원해야 한다. 이 테스트는 pdf, jpg, pptx, png, tar, zip 등의 다양한 확장자를 갖는 파일에 대해 진행될 예정이며, 인터넷 연결 상태에 따른 전송 속도의 차이를 분석하여 업/다운로드 가능한 자료의 크기 또한 제한되어야 할 것이다.

또한 약 디스펜서의 스토리지 오픈까지의 시간은 모바일 애플리케이션에서 기기에서 버튼 클릭 후 약 1초 이내의 딜레이로 서버에 적용되어 작용까지 이루어져야 한다. 동시에 한 스토리지에는 최대 5명의 유저가 편집기능을 실행한 채로 접근할 수 있어야 한다. 이와 함께 알림 기능은 0.1초 이내의 딜레이만이 허용된다. 이 부분에서는 서버의 시간을 실시간으로 입력 받아 미리 설정된 시간에 사용자에게 1초의 오차범위 내에서 전송되도록 한다. 이를 위해 실시간으로 서버에 반영, 다시 다른 유저들에게 적용되는 과정까지 시간을 측정하는 테스트가 있을 예정이다.

8.2.1.2. Reliability

스마트 약 디스펜서는 기존의 플랫폼에 더해 수많은 서브 컴포넌트로 구성되어 있다. 따라서 스마트 약 디스펜서가 오류 없이 작동하기 위해서는 각각의 서브 컴포넌트가 제대로 작동해야 하고, 이들이 아키텍처에 따라 정확히 결합되어야 할 것이다. 따라서, development test 는 unit developing stage 에서부터 진행되어야 하고, 매번 다른 컴포넌트가 추가될 때마다 반복적으로 진행되어야 한다.

8.2.1.3. Security

사용자의 개인 정보를 보호하는 것은 개발자 차원에서 당연히 안전장치를 마련해야 할 부분이다. 본인 또는 개인 정보 수집에 대해 약속되지 않은 사용자가 개인 정보에 무단으로 접근하는 것은 절대 있어서는 안된다. 시스템의 보안성을 위해서 우리는 모든 버전 애플리케이션의 개발 및 진화가 완료된 뒤, 수동으로 코드에 대한 리뷰를 진행하고, 이로부터 보안 이슈를 잡아내어 수정해야 한다. 또한 개발자 차원에서의 직접적 분석 이외에도 Ostorlab 등의 애플리케이션 자동 분석 서비스를 이용할 수 있다.

8.2.2 Release Testing

성공적인 소프트웨어 개발 프로젝트란, 그 설계, 구현 뿐만 아니라 배포까지 포함하는 개념이다. 아무리 무결점에 최적화가 잘 된 어플리케이션일지라도 배포하는 방법이 잘못되어있다면 그 어플리케이션은 구동과정에서 문제를 일으킬 가능성이 있다. 따라서, release test 는 소프트웨어의 배포가 문제없이 진행되어, 설계된 애플리케이션이 사용자에게 계획대로 사용될 수 있는지를 테스트하는 과정이다. 모순처럼 느껴질 수 있지만, 이러한

release test 는 반드시 소프트웨어의 정식 배포 전에 실행되어야 한다. 소프트웨어 배포 생명 주기(Software Release Life Cycle)에 따르면, release test 는 기본적인 구현이 끝난 시점부터 진행하며, 이를 '알파' 테스트라고 한다. Development test 는 이 알파 테스트 기간부터 계속해서 진행되며, 알파 테스트의 결과에 따라 '베타' 버전을 새로이 배포, '베타' 테스트에서는 실제 유저들과 함께 테스트를 진행하게 될 것이다. 베타 테스트가 끝난 뒤, 개발자 뿐만 아닌, 유저들로부터의 의견도 수렴하여 어플리케이션의 수정을 진행할 수 있을 것이다.

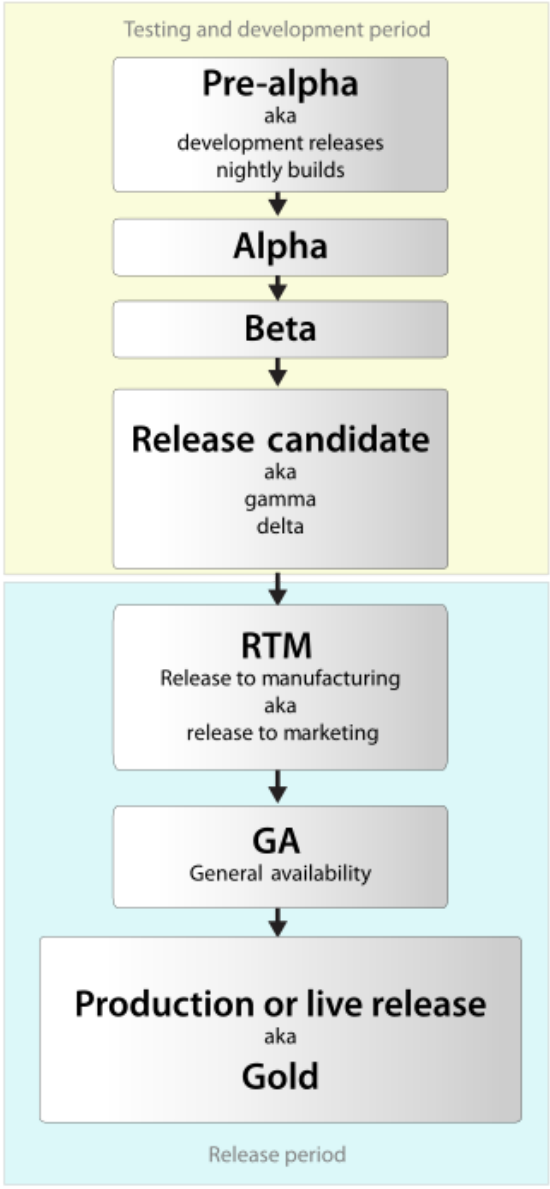

그림 29 소프트웨어 생명 주기(Software Release Life Cycle)

8.2.3 User Testing

위의 테스트에 덧붙여, 우리는 실제 디스펜서의 사용자가 처할 법한 상황을 상정할 필요가 있다. 이를테면 지역 문화센터의 휴대폰 강좌 수강생 및 기타 중장년층 집단에 대해서 스마트 약 디스펜서의 베타 버전을 배포하고, 이를 통한 수업의 진행으로 실제 상황에 대한 테스트를 진행할 수 있다. 스마트 약 디스펜서를 통한 간단한 테스트를 진행한 후, 각각의 사용자들로부터 피드백을 수렴하여 스마트 약 디스펜서 의 개선에 이용할 것이다.

8.2.4 Testing Case

우리의 목적은 스마트 약 디스펜서가 “본래 기획한 의도대로 사용될 수 있는가”, “실제 구동과정에 예기치 못한 오류가 생기는가”, “실제 사용자의 환경에서 충분히 제 성능을 낼 수 있는가”의 3 가지 과제에 대해 문제없이 기능을 수행할 수 있도록 하는 것이다. 각각의 과제는 “기획하지 않은 의도의 사용에 대한 테스트”, “시스템의 허점을 찌르는 경우에 대한 테스트”, “네트워크 환경 및 사용자의 실제 기기 상황에서 사용시에 생기는 문제점에 대한 테스트”를 진행하여 문제를 파악할 수 있을 것이며, 이를 기반으로 시스템 평가서를 작성할 것이다.

9. Development Plan

9.1 Objectives

이 챕터에는 어플리케이션 개발에 사용된 여러 외부 기술 및 프로그램이 기술되어있다.

9.2 Frontend Environment

9.2.1 Adobe Photoshop

그림 30 Adobe Photoshop 로고

Adobe Photoshop 은 그래픽 에디터이다. 이 프로젝트에서, Adobe Photoshop 은 디스펜서의 앱 UI를 디자인하기 위해 사용된다.

9.2.2 Adobe Xd

그림 31 Adobe Xd 로고

Adobe Xd 는 웹 디자인 또는 모바일 어플리케이션의 UI 및 UX 디자인을 위한 프로토타이핑 소프트웨어이다. Adobe 사에서 무료로 제공하는 프로그램으로, 이 프로그램의 주된 장점 중 하나는 디자인한 어플리케이션에 대해 즉각적으로 프로토타입을 만들어 시각화 할 수 있고, 만들어진 프로토타입을 개발팀원들과 즉각적으로 공유가 가능하다는 점이다. 따라서 디자인 과정에서의 빠른 피드백이 가능하므로 Adobe Xd 를 사용한다

9.2.3 Flutter

그림 32 Flutter 로고

Flutter 는 구글에서 제공하는 크로스 플랫폼 개발 환경이다. Flutter 를 통해 스마트 약 디스펜서의 구현된 소스코드는 Android OS, iOS 의 두 환경에서 구동 가능한 제품으로 컴파일 될 수 있다. 또한 Flutter의 사용은 웹 브라우저, Windows, Mac, Linux 등의 수많은 환경에서 가능하기 때문에 다수의 개발자가 통일된 환경에서 컴파일 하는 것이 가능하다는 장점이 있다.

9.3 Backend Environment

9.3.1 Github

그림 33 GitHub 로고

Github 는 소프트웨어 개발 버전을 관리하여 다수의 개발자들 간의 협업을 지원하는 코드 호스팅 플랫폼이다. 이를 통해 개발팀은 스마트 약 디스펜서 프로젝트를 동시다발적으로 개발할 수 있고, 개발된 요소들을 쉽게 취합할 수 있도록 한다.

9.3.2 Firebase

그림 34 Firebase 로고

Firebase 는 실시간 데이터베이스, 클라우드 저장소, 앱 테스트 등의 기능의 지원을 통해 어플리케이션의 개발을 돕는 통합 어플리케이션개발 플랫폼이다. 우리는 실시간 데이터베이스 관리 기능을 통해 사용자들의 개인정보, 사용자가 복용 중인 알약 관리 등을 수월하게 진행할 수 있다. Firebase 가 지원하는 실시간 데이터베이스 지원 기능 덕분에 스마트 약 디스펜서 의 사용자들이 업로드한 데이터는 일괄적으로 반영되어 공유가 가능하다.

9.3.3 AWS

그림 35 AWS 로고

AWS 는 Amazon Web Services 의 약자로, 아마존닷컴의 클라우드 컴퓨팅 사업부이다. AWS 는 다른 웹 사이트나 클라이언트측 응용 프로그램에 대해 온라인 서비스를 제공하고 있으며, 이러한 서비스의 상당수는 최종 사용자에 직접 공개되는 것이 아니고, 다른 개발자가 사용 가능한 기능을 제공하는 플랫폼을 제공하는 PaaS이다. 이 프로젝트에서는 약 디스펜서의 하드웨어와 애플리케이션 간의 서버, 그리고 사용자의 정보를 저장하는 서버를 구동하는 데에 사용된다.

9.3.4 KALDI

그림 36 Kaldi 로고

Kaldi 는 음성 인식 및 신호 처리를 위해 C++ 로 작성된 오픈 소스 음성 인식 툴킷 으로 Apache License v2.0 에 따라 무료로 사용할 수 있다. Kaldi는 유연하고 확장 가능한 소프트웨어를 제공하는 것을 목표로 하며 인식 시스템을 구축하기 위해 자동 음성 인식(ASR) 연구원이 사용하도록 고안되었다. 이 프로젝트에서는 디스펜서의 하드웨어 속 임베디드 시스템에 KALDI를 내장하여 사용자의 음성을 기계가 인식하여 자동으로 실행할 수 있도록 하였다.

9.4 Constraints

본 시스템은 이 문서에서 언급된 내용에 근거하여 설계되고 구현되었다. 세부적인 내용은 개발자의 선호에 따라 변동되어 구현될 수 있지만, 다음에 언급되는 제약 조건은 반드시 지켜져야 한다.

9.4.1 개발자들에 대한 제약 조건

a. 가능한 경우, 독점 software 대신 무료 Open-source software 를 사용할 것

b. 테스트를 통해 시스템을 최대한 신뢰 가능하고 안전하게 유지할 것

c. 조기 최적화를 최소화하면서 시스템을 최적화할 것

9.4.2 사용자들에 대한 제약 조건:

a. 100Mbps 이상의 인터넷 연결이 요구됨

b. 사용자는 애플리케이션 구동을 위한 다음의 휴대폰 최소 사양을 만족할 것

: Android 6.0 이상 또는 iOS 11.0 이상

9.5 Assumptions and Dependencies

이 시스템에 대한 Assumptions(가정) 과 dependencies(종속성) 은 다음과 같다.

9.5.1 Assumptions

a. 이 시스템은 사용자 인증을 위해 기존의 회원 데이터베이스를 사용할 수 있는 것으로 가정한다.
b. 각각의 사용자는 알약에 대한 사용 권한이 있는 것으로 간주된다.
c. 알약을 등록하는 과정에서, 같은 종류의 약은 최대한 인접한 스토리지 칸에 배정한다.
d. 프로필, 계정, 알약 등의 정보를 수정하는 화면은 이를 등록하는 화면과 유사한 것으로 한다.
e. 일반적으로 아침, 점심, 저녁에 해당하는 식사를 하는 시간이 아니더라도, 사용자가 설정한 시간이 존재한다면 해당 시간에 맞게 알람을 등록한다.
f. 알람 기능에 대해서는, 약 디스펜서 하드웨어의 시각적 및 청각적 알람과 애플리케이션 상의 알람이 동시에 울리도록 한다.
g. 약 등록 내의 '찾아보기' 기능에 대해서, 사용자가 알약의 정보에 접근할 수 있도록 '약학정보원' 사이트의 내용을 인용한다.
h. 처방전에 존재하는 QR 코드에 대한 정보는 모두 '약학정보원' 사이트에 존재하는 것으로 간주한다.
i. 알림을 전송하는 것에 대해서는, 어떠한 문맥적 또는 어법적 오류를 허용하지 않는다.
j. 연령대별 사용자의 선호도에 맞도록 하기 위해, 60세 미만인 사용자에게는 '글씨 크기'의 디폴트 값을 '보통'으로, 60세 이상인 사용자에게는 '크게'를 디폴트 값으로 설정한다. (이는 환경설정 화면에서 작게-보통-크게 중 하나로 수정 가능하다.)

9.5.2 Dependencies

a. 사용자는 '약 디스펜서 애플리케이션'을 실행하기 위한 최소 요구 사항을 충족하는 휴대폰에 액세스할 수 있어야 한다.
b. Kaldi 라는 프로그램이 음성인식 기능을 구현하기 위해 사용될 것이다.

10. Supporting Information

10.1 Software Design Specification

본 Software Design Specification은 IEEE Recommendation (IEEE Recommended Practice for Software Requirements Specifications, IEEE-Std_830)을 바탕으로 작성되었다.

10.2 Document History

날짜	버전	설명 (편집 파트)	참가자
2022/05/04	0.1	문서 작업 시작	전체
2022/05/04	1.1	1, 2	김민종
2022/05/04	1.2	3	원현선
2022/05/05	1.3	4	김하늘
2022/05/05	1.4	5	강창우
2022/05/06	1.5	7	원현선
2022/05/06	1.6	8, 9	김민종
2022/05/07	1.7	10	원현선
2022/05/07	1.8	6.1, 6.2, 6.4.1.3	강창우
2022/05/08	1.9	6.3.1, 6.3.2	김민종
2022/05/09	1.10	6.4.1.1, 6.4.1.2	원현선
2022/05/09	1.11	6.4.2, 6.5	김하늘
2022/05/10	2.0	목차 작성	강창우
2022/05/10	2.1	용어 통일 / 오타 수정 / 최종 서식 수정	전체