Design Specification

Software Engineering Team 9

2017314886 김준태

2016312917 김지연

2015313803 양진율

2016313513 이승윤

2015314213 이재원

Table of Contents

1. Preface

1.1. Readership

본 문서의 독자는 시스템을 직접적으로 개발하는 software engineer, 시스템을 설계하는 architecture, 시스템 개발에 참여하는 모든 stakeholder로 정의된다.

1.2. Document Structure

1.2.1. Preface

Preface에서는 본 문서의 독자를 정의하고, 전반적인 구조에 대해 소개한다. 구조를 소개할 때에는 각 목차의 목적을 서술한다.

1.2.2. Introduction

Introduction에서는 본 문서에서 system design을 설명하기 위해 사용된 모든 종류의 diagram과 tool에 대해 서술한다.

1.2.3. System Architecture

System Architecture에서는 본 프로젝트를 통해 개발하고자 하는 시스템의 전반적인 내용을 서술한다. System과 각 Subsystem의 관계를 대략적으로 서술하고, 이들이 실제 하드웨어에 어떻게 할당되는지를 서술한다.

1.2.4. Protocol Design

Protocol Design에서는 Subsystem들이 상호작용하기 위해 준수해야할 프로토콜에 대하여 서술한다. 통신하는 메시지의 형식과 용도, 의미를 설명한다.

1.2.5. Database Design

Database Design에서는 requirement specification에서 서술한 database requirement를 바탕으로 수정사항을 반영하여 다시 작성하였다. 각 data entity의 attribute와 relation을 ER diagram을 통해 표현하고, 이를 통하여 Relational Schema을 작성한다.

1.2.6. Testing Plan

Testing Plan에서는 시스템이 요구사항을 만족시키는지를 확인하고, 시스템의 내부적 결함을 찾기 위한 test를 서술한다. 각 test를 수행하기 위한 test policy를 서술하고, 각 상황에 기대되는 입력, 출력 동작을 test case를 통해 서술한다.

1.2.7. Development Plan

Development Plan에서는 시스템을 개발하기위해 사용되는 개발 도구와 프로그래밍 언어 등의 개발 환경에 대해 서술한다. 또한, 이후 진행될 개발 일정에 대해서도 서술한다.

2. Introduction

2.1. Objectives

Introduction에서는 본 문서에서 system design을 설명하기 위해 사용된 모든 diagram과 tool에 대해 서술하고, 본 시스템의 개발 범위에 대해 서술한다.

2.2. Applied Diagram

2.2.1. UML

UML(Unified Modeling Language)은 객체 관련 표준화 기구 OMG(Object Modeling Technique)에서 발표되었으며, 소프트웨어 공학에 사용되는 표준화된 범용 modeling language이다. UML은 객체 지향 소프트웨어 설계를 위해 사용되어왔던 여러 종류의 diagram을 통합한 modeling language로, 시스템 개발 과정에서 요구사항을 분석하고 시스템의 설계 및 구현하는 단계에서 시스템의 구조를 시각화 하여 개발자는 물론 시스템과 관련된 stakeholder들과의 의사소통도 원활하게 하기 위해 만들어졌다.

2.2.2. Class Diagram

Class Diagram은 객체 지향 설계에 사용되는 기본적인 Diagram으로, 시스템의 정적인 구조를 표현해 준다. Class Diagram에서는 시스템의 object들을 class로 표현하며, 각 class는 name, attribute, method의 내용들을 포함한다. Class Diagram을 통하여 각 class사이의 상속관계와 같은 relationship 또한 표현할 수 있다.

2.2.3. Sequence Diagram

Sequence Diagram은 시스템, 객체, 클래스 간의 상호작용을 시간의 흐름에 따라 표현한

UML diagram이다. 주로 시스템의 동적인 측면을 모델링하는 과정으로, use case를 실현하기 위한 각 객체 간에 데이터 교류, 메시지 시퀀스 등을 묘사하게 된다.

2.2.4. ER Diagram

ER(Entity Relationship) Diagram은 데이터베이스를 entity와 entity들 간의 relationship으로 표현하는 Conceptual Model로, 앞선 diagram들과는 달리 UML에 포함되는 diagram은 아니다. 이 때, 각 entity는 현실 세계에서의 객체로서 유형·무형의 정보를 대상으로 하며, 하나 이상의 attribute로 구별할 수 있는 것을 의미한다. relationship은 각 entity간의 상관관계를 나타내며, 실선으로 표현된다.

2.3. Applied Tool

2.3.1. Draw.io

Draw.io는 무료로 이용할 수 있는 웹 기반의 diagram software로, flowchart나 sequence diagram, class diagram 등 다양한 diagram을 쉽게 작성할 수 있게 해주는 도구이다. 구글 드라이브와 연동이 가능하며, 로컬파일로도 저장이 가능하기 때문에, 본문에 사용된 대부분의 diagram은 draw.io를 통하여 작성되었다.

2.4. Project Scope

MODU는 프로젝트를 진행하기 위하여, 프로젝트 제안자와 개발자가 자신에게 적절한 상대를 찾도록 도와주는 팀 매칭 시스템이다. 이를 위하여 MODU는 사용자에게 다양한 기능들을 제공한다. 우선, 사용자가 시스템에 가입하고, log in, log out하기 위한 Authentication 기능. 다른 사용자에게 노출될 자신의 정보들을 관리하기 위한 Portfolio Manage 기능. 시스템에 프로젝트를 등록하고 관리하기 위한 Project Manage 기능. 사용자가 자신에게 필요한 프로젝트와 개발자들을 찾기 위한 Search 기능. 알아보고자 하는 프로젝트와 개발자에 관한 상세한 정보를 사용자에게 보여주기 위한 Project Info, Developer Info 기능. 관심 있는 프로젝트에 참여 의사를 전달하기 위한 Participation Inquiry 기능. 관심 있는 개발자를 자신의 프로젝트에 참여시키고자 하는 의사를 전달하기 위한 Invite 기능. 관심 있는 개발자와 프로젝트에 대한 북마크 기능을 하는 Follow기능. 사용자 간의 의사소통과 시스템 알림을 전달하기 위한 User message, System Message 기능. 프로젝트가 완료된 후 프로젝트를 진행한 개발자 상호간의 평가를 기록하기 위한 Peer Assessment 기능. 본 프로젝트의 구현 범위는 이와 같다.

3. System Architecture

3.1. Objective

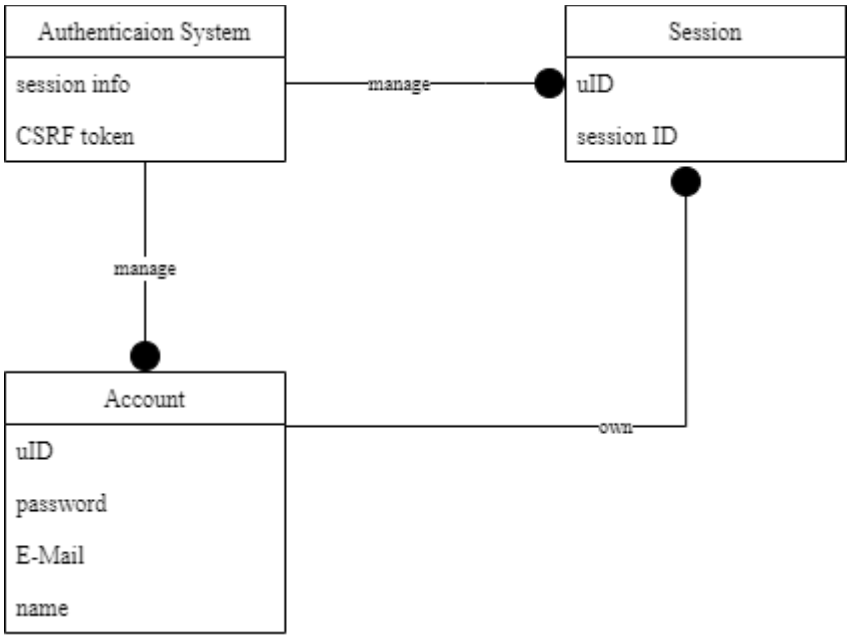
System architecture에서는 현재 개발하고자 하는 시스템의 전체적인 구조에 대해 서술한다. 시스템의 각 subcomponents의 구조와 관계는 sequence diagram과 class diagram을 통해 보다 상세히 서술한다.

3.2. System Organization

3.3. Subcomponents

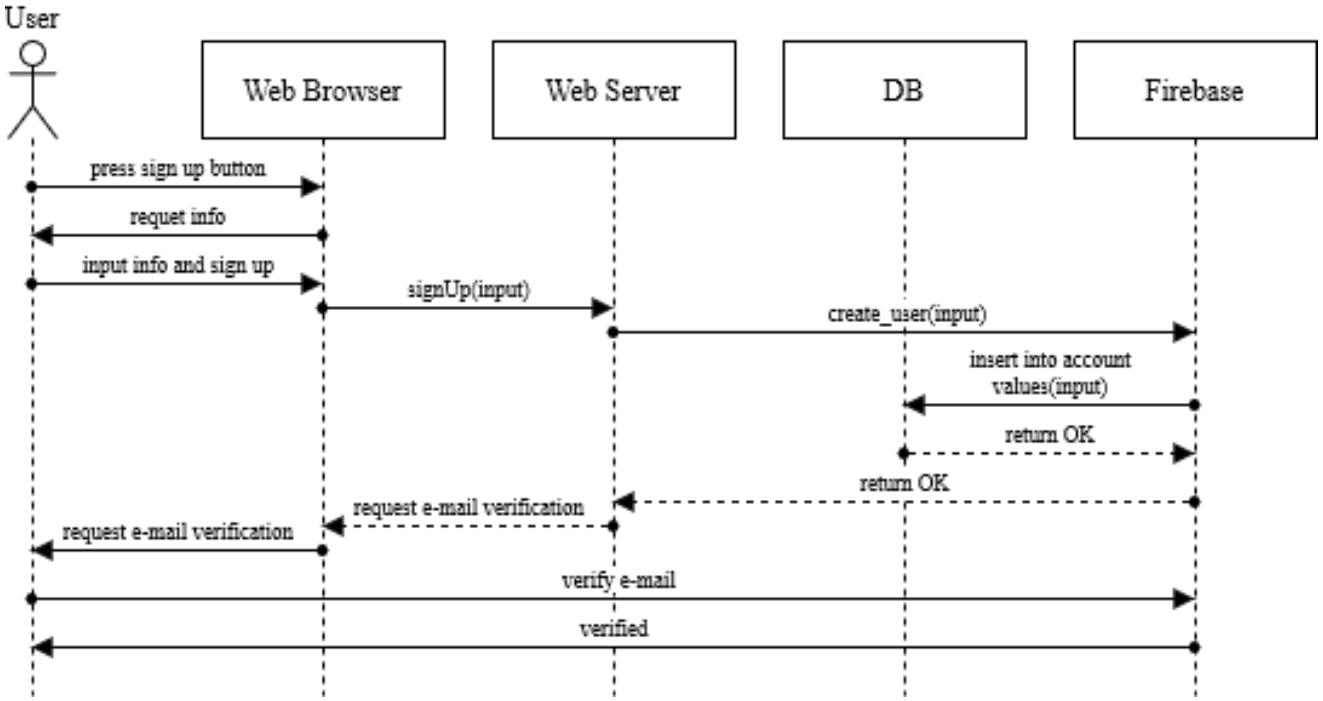
3.3.1. Authentication

-Authentication Class Diagram

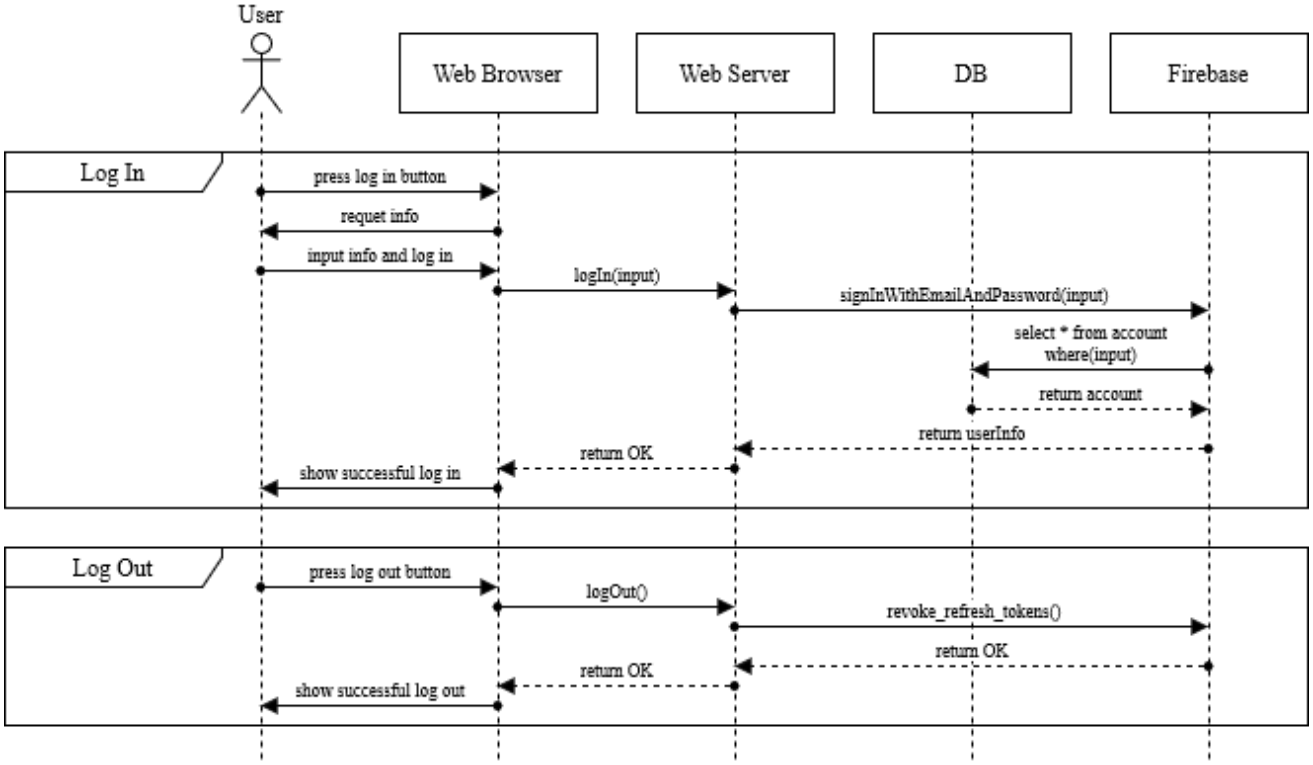


Authentication system은 세션 관리를 맡으며 계정과 세션이 매칭이 되도록 관리한다. 즉 account class는 최대 하나의 세션을 소유한다.

-Sign up Sequence Diagram



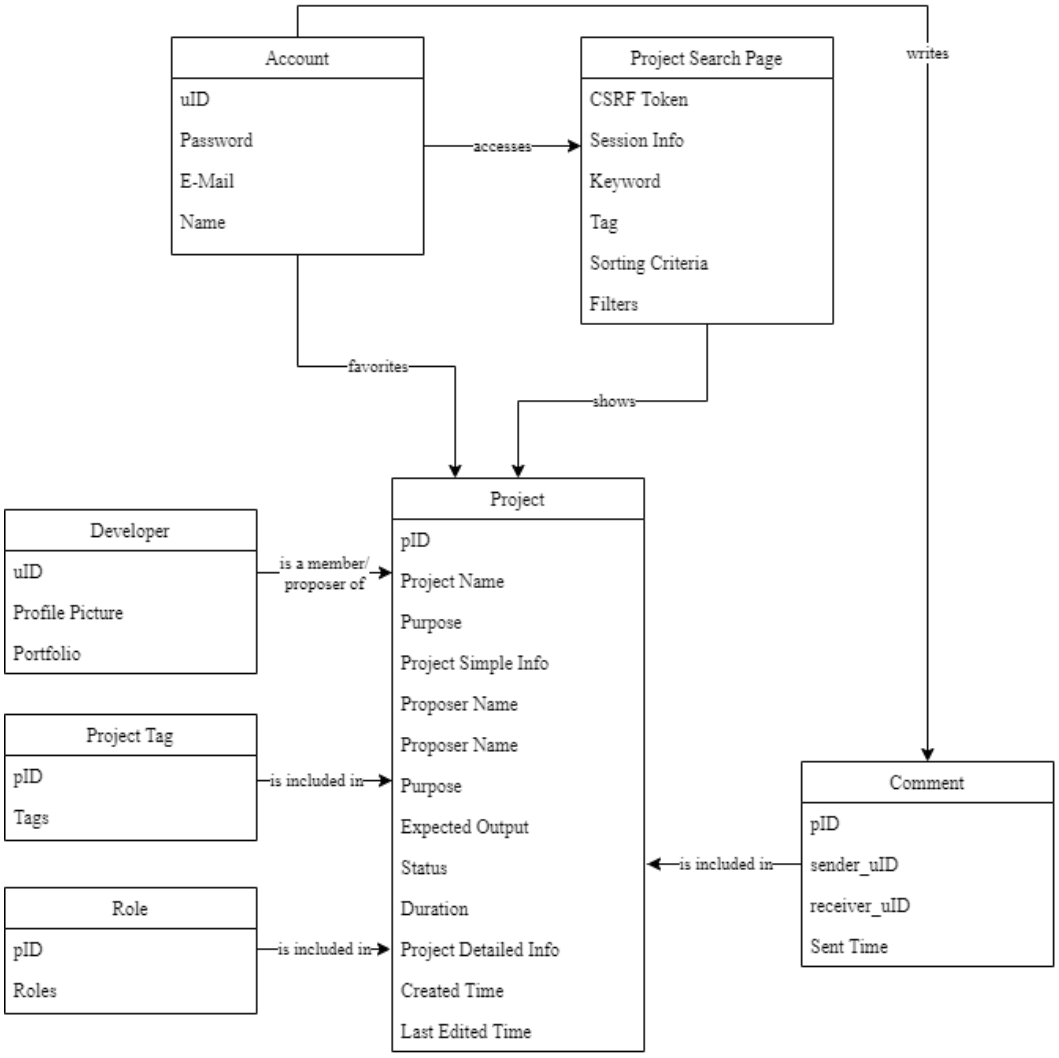
-Log in & Log Out Sequence Diagram



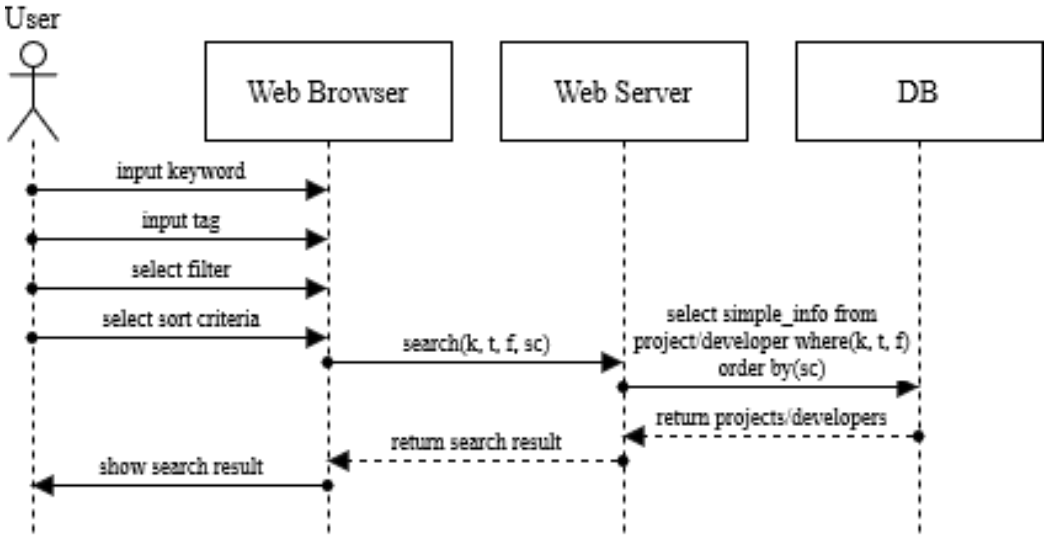
3.3.2. Project

- Search

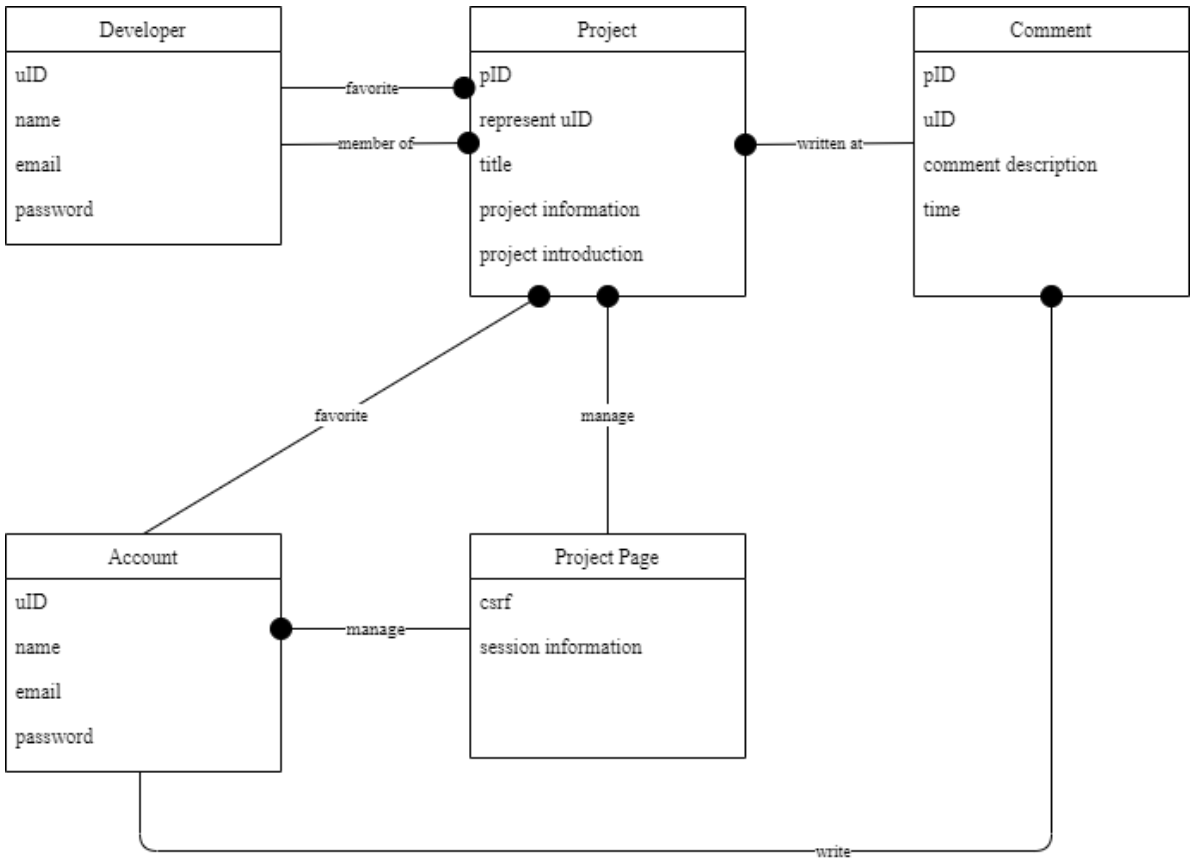
1. Class Diagram



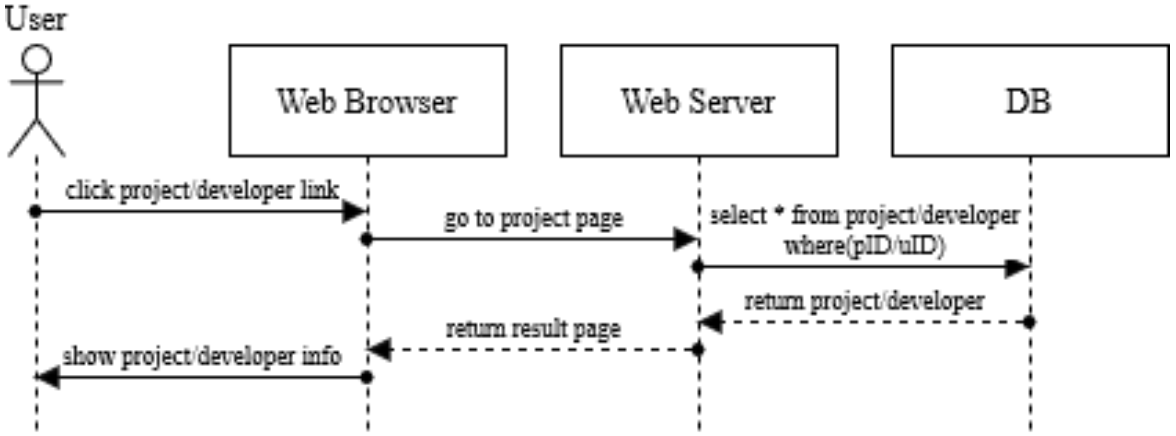
2. Sequence Diagram



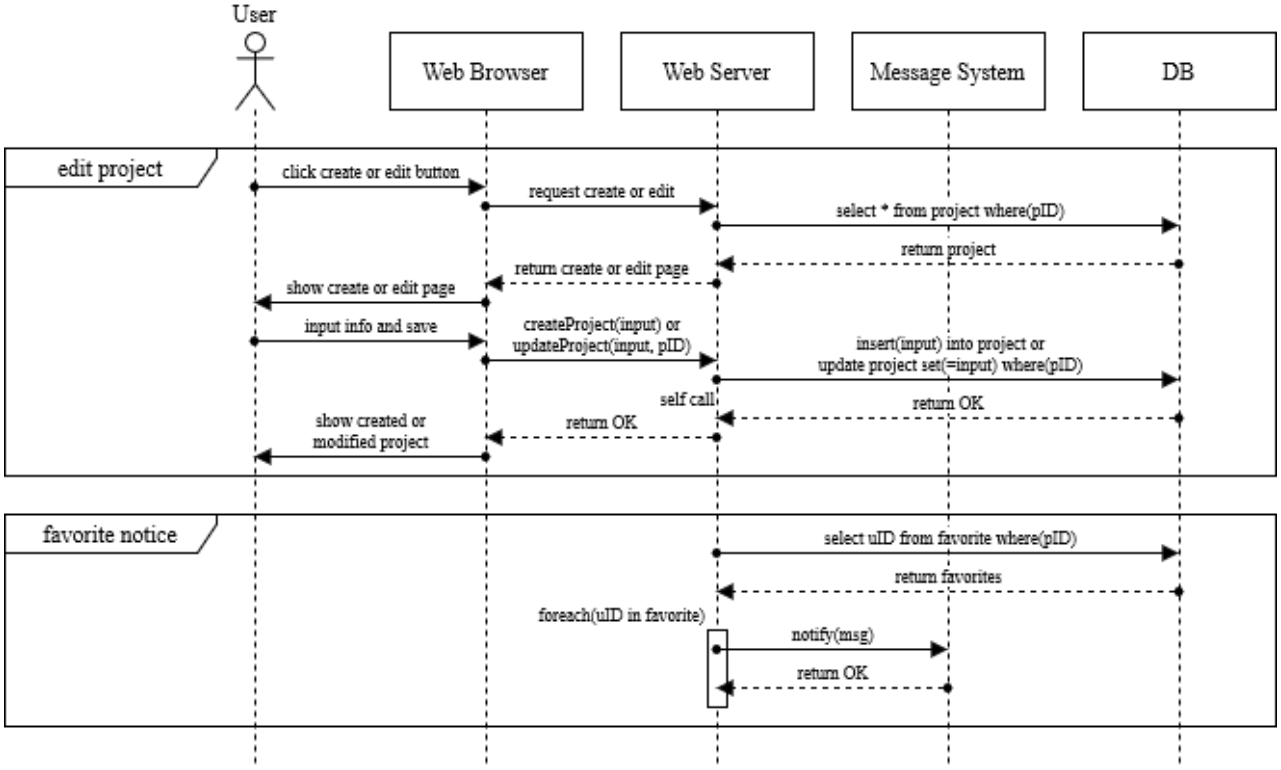
-Project Info & Manage Class Diagram



- Project Info Sequence Diagram



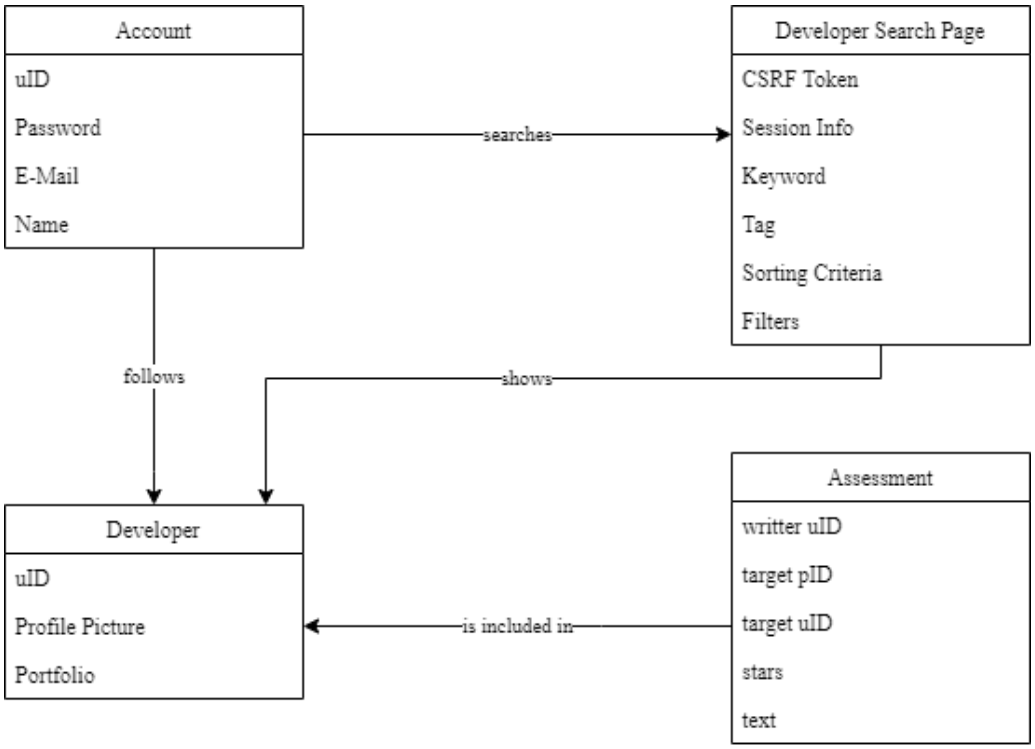
- Project Manage Sequence Diagram



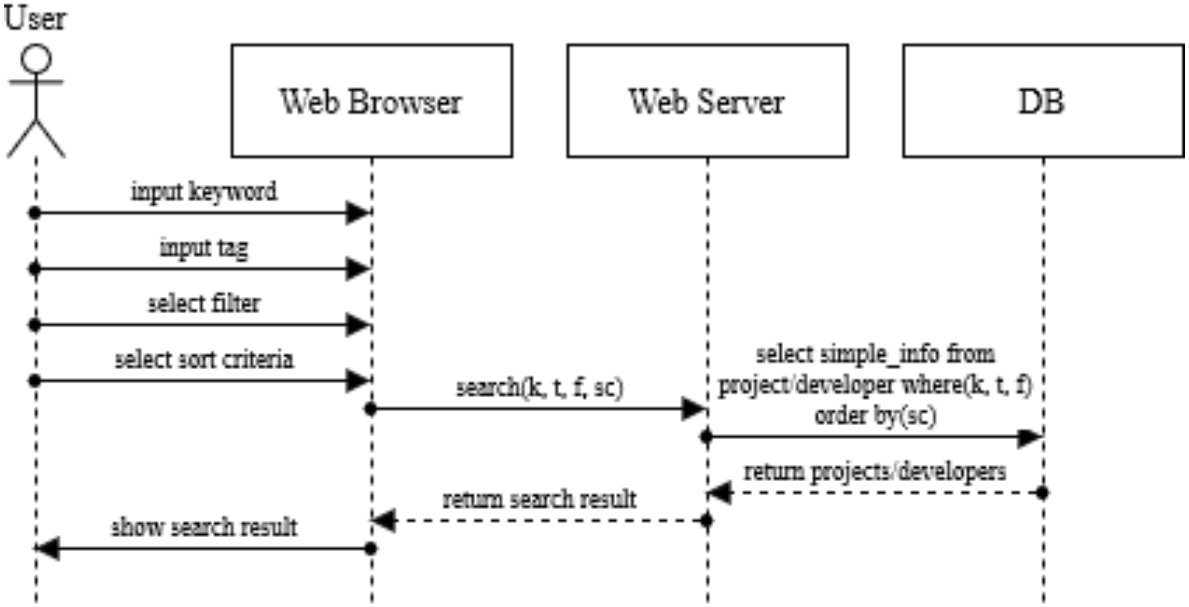
3.3.3. Portfolio

- Search

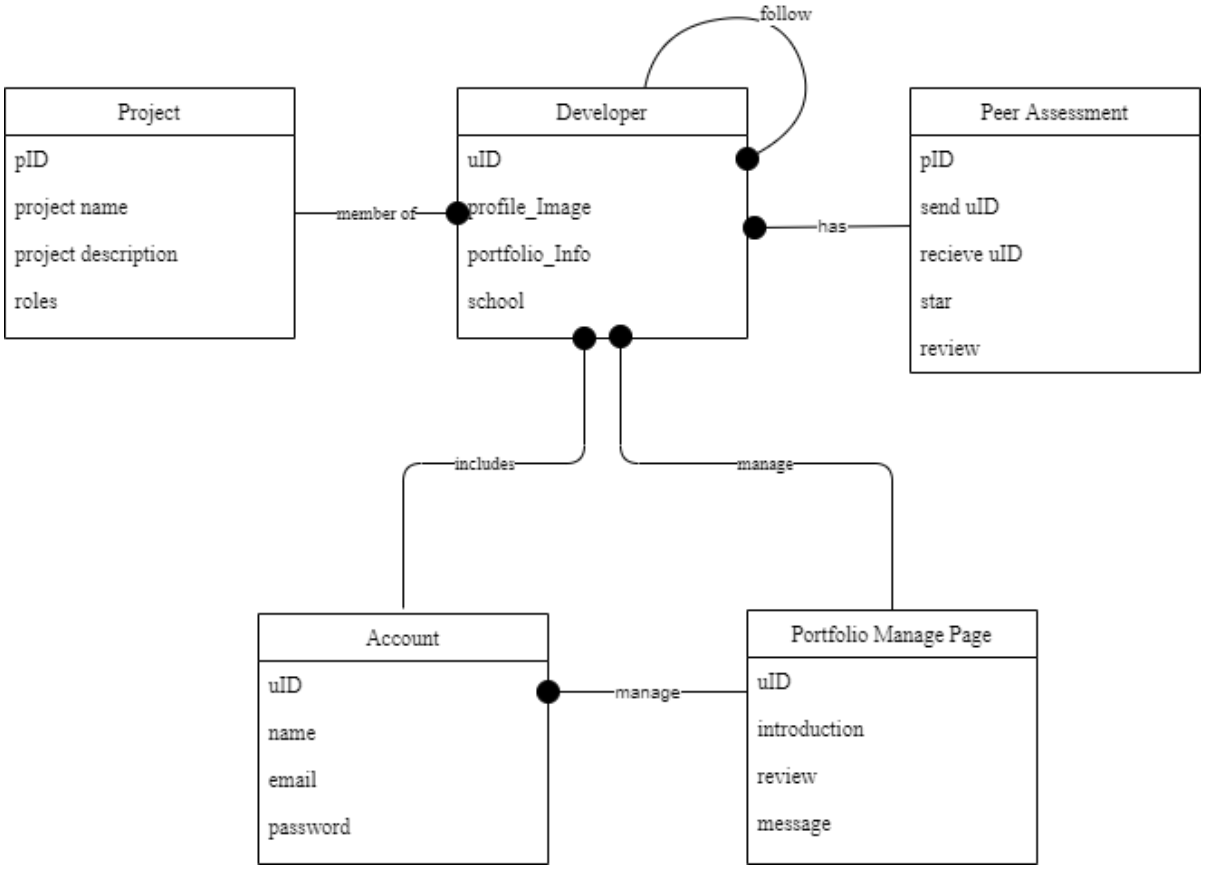
1. Class Diagram



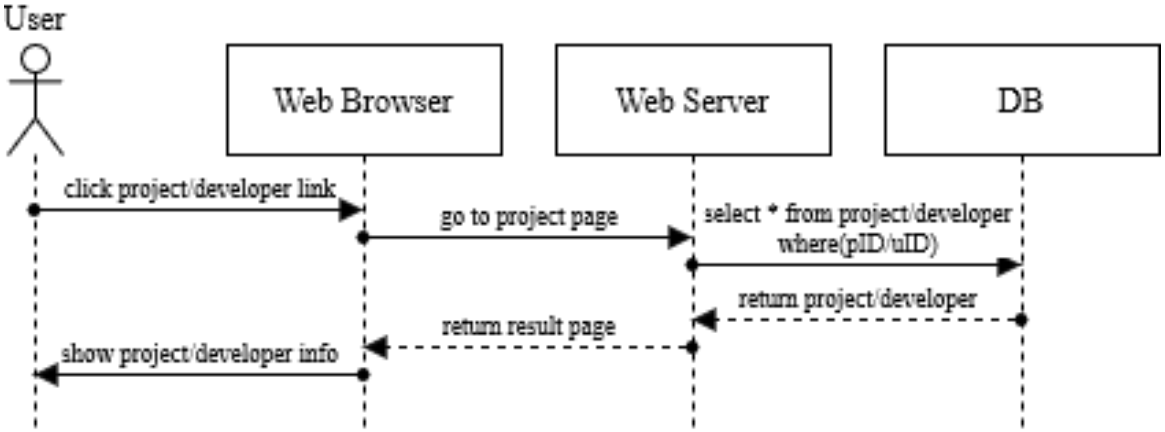
2. Sequence Diagram



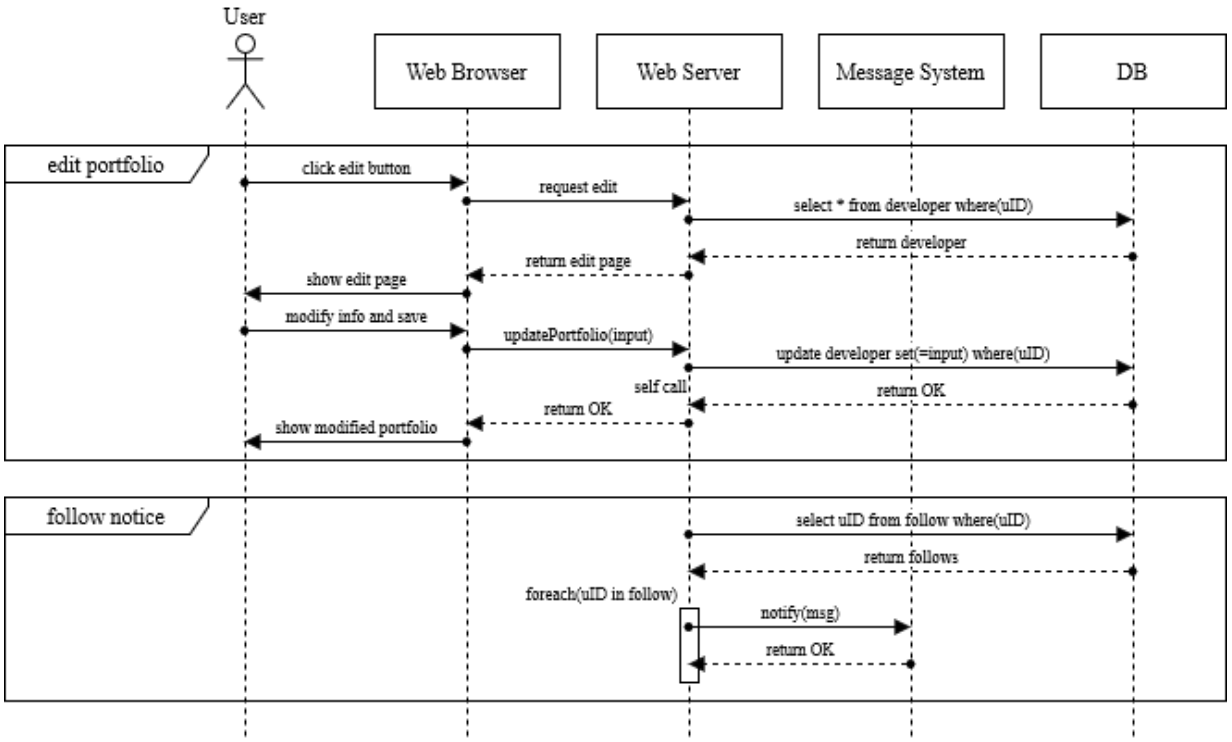
- Portfolio Info & Manage Class Diagram



-Project Info Sequence Diagram

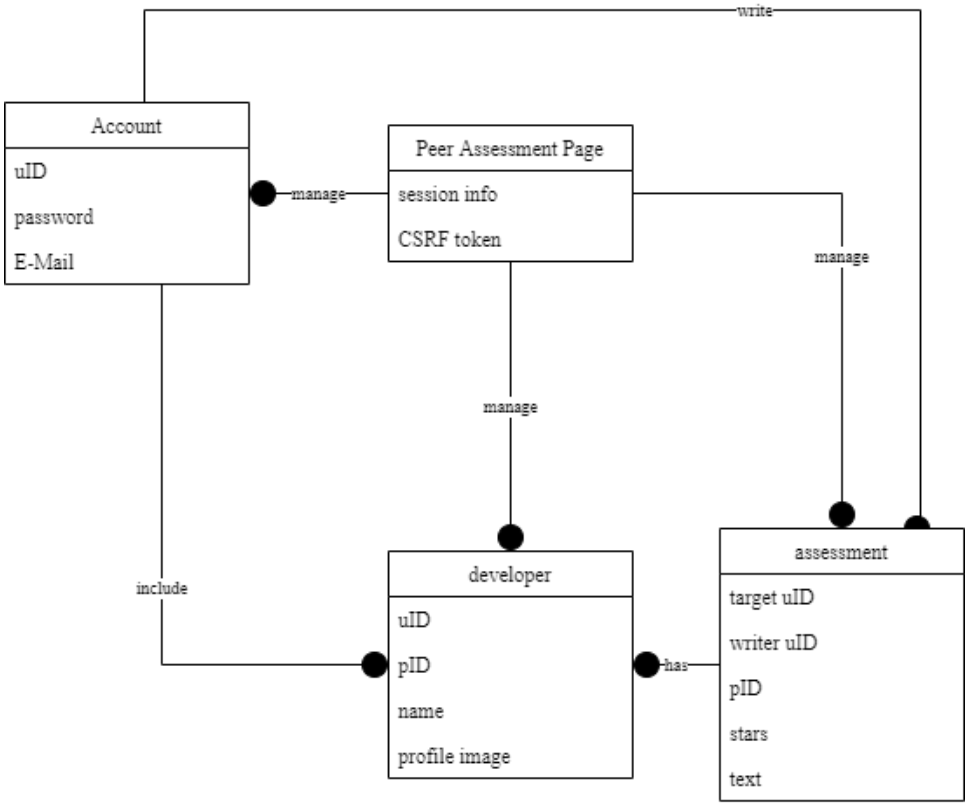


- Portfolio Manage Sequence Diagram

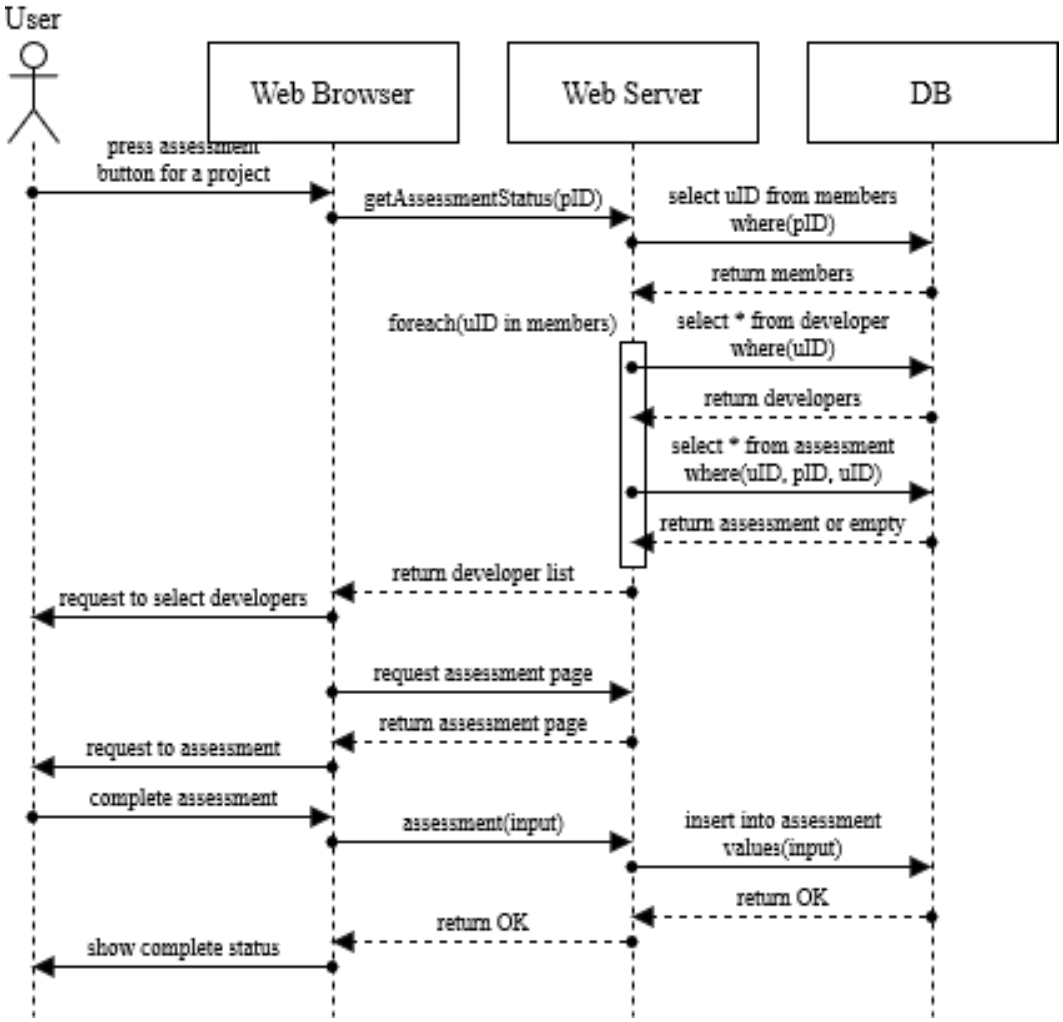


3.3.4. Peer Assessment

1. Class Diagram

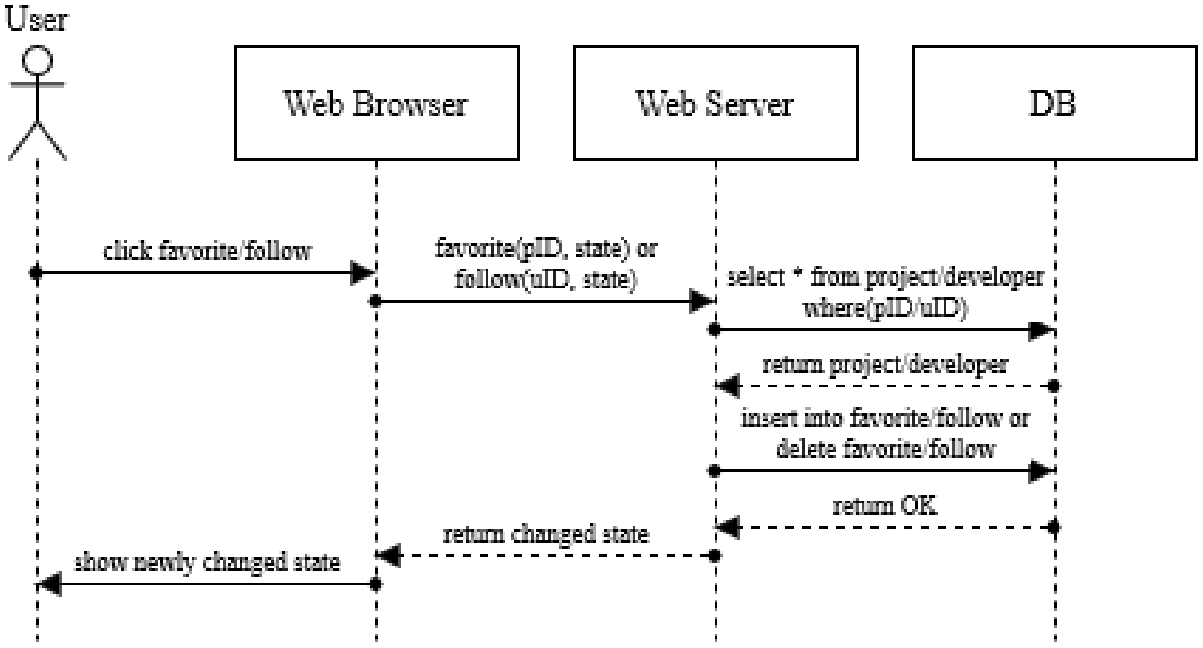


2. Sequence Diagram



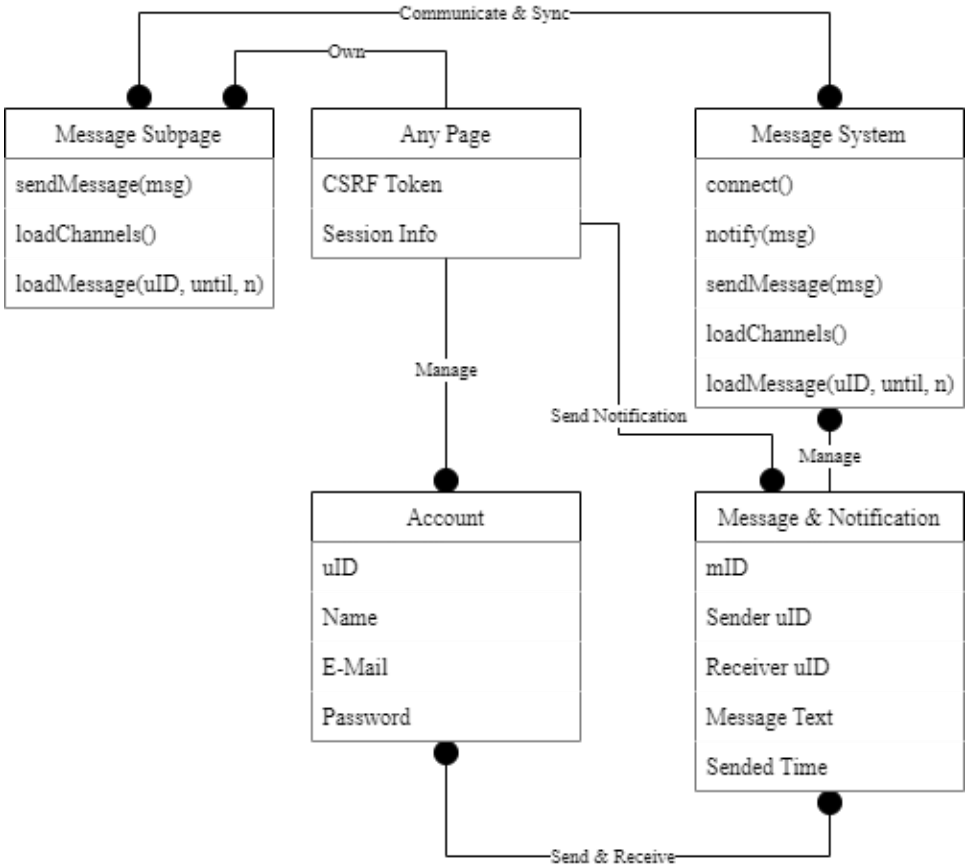
3.3.5. Favorite/Follow

- Sequence Diagram

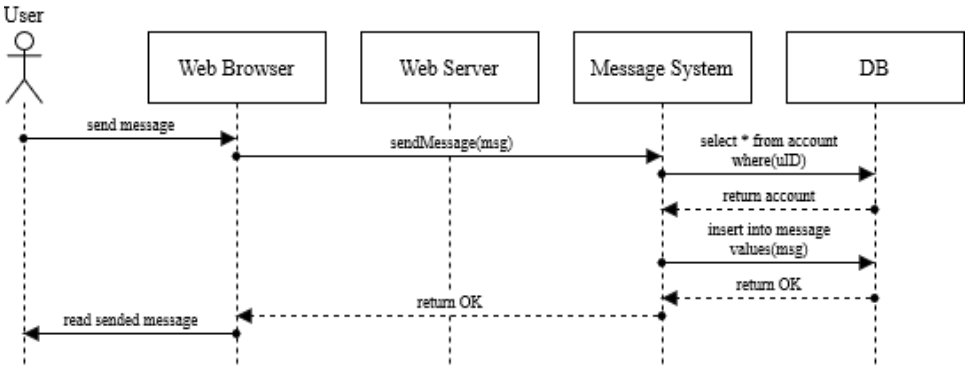


3.3.6. Message & Notification

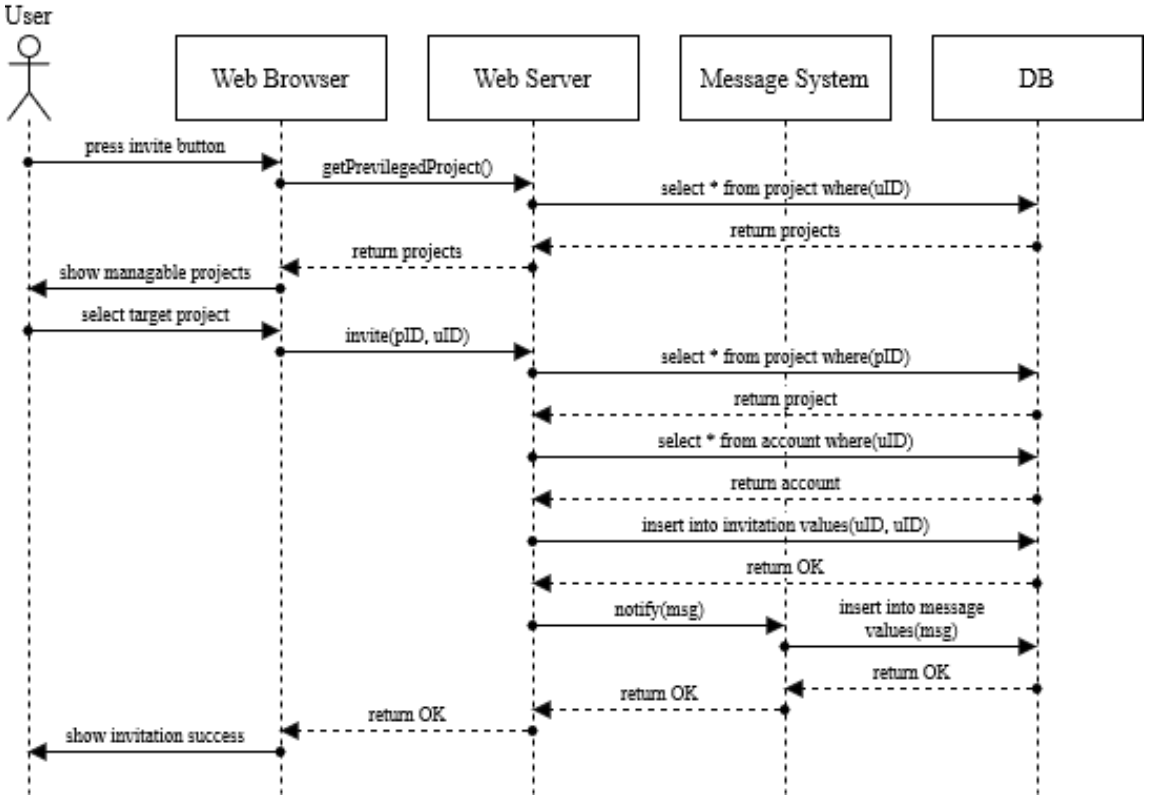
-Class Diagram



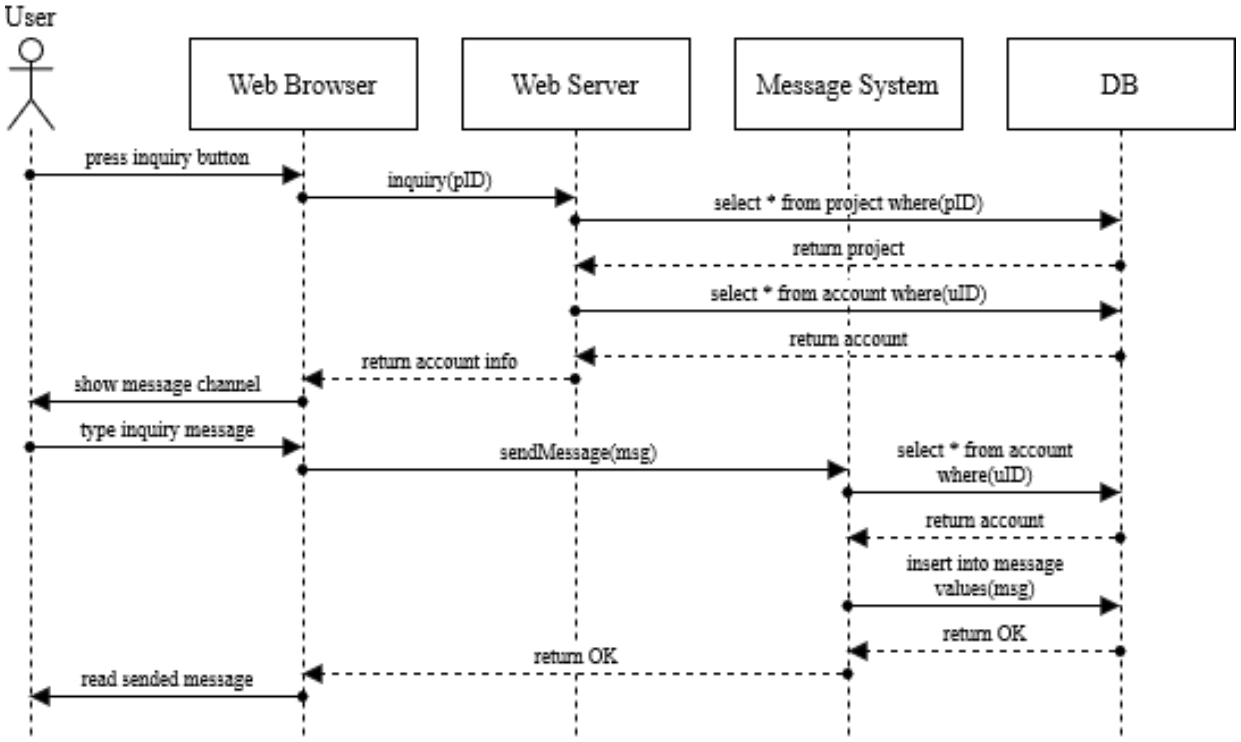
-User Message (Message Send) Sequence Diagram



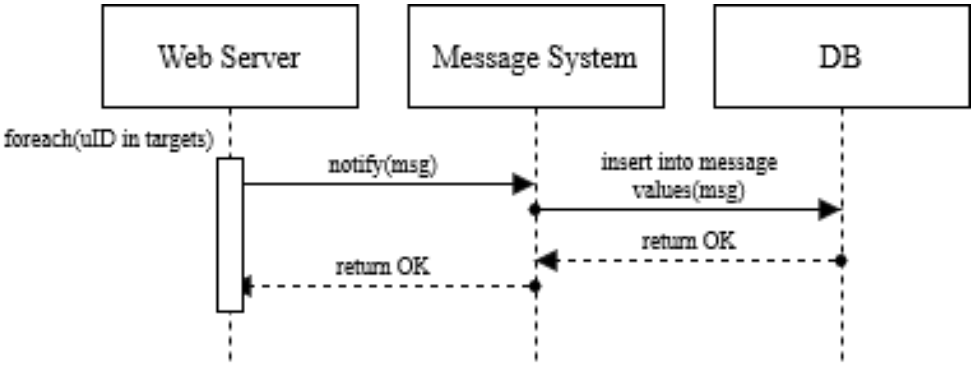
-Invite Sequence Diagram



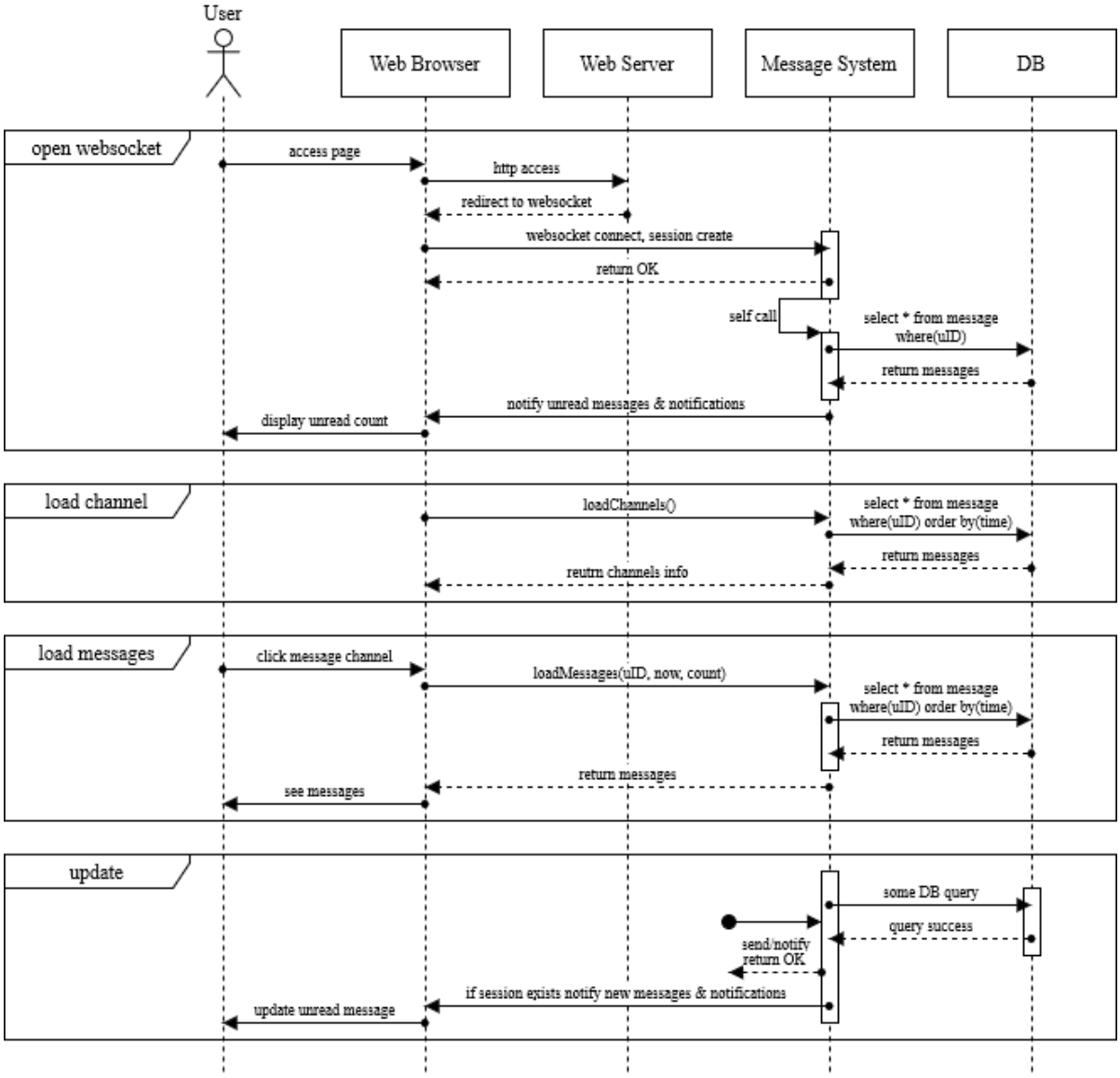
-Participation Inquiry Sequence Diagram



-System Message (Notification Send) Sequence Diagram

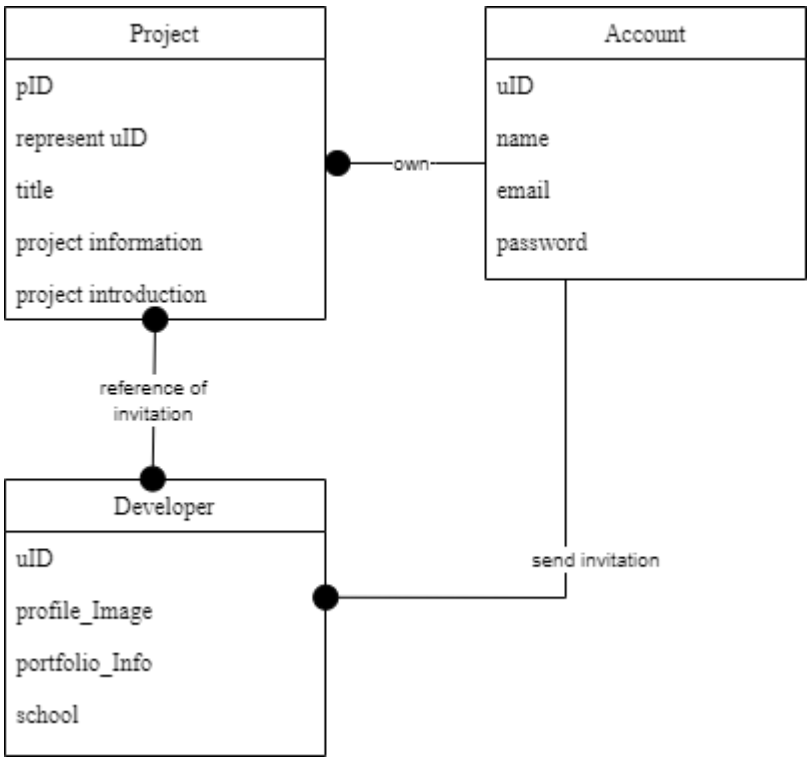


-Message & Notification Receive Sequence Diagram

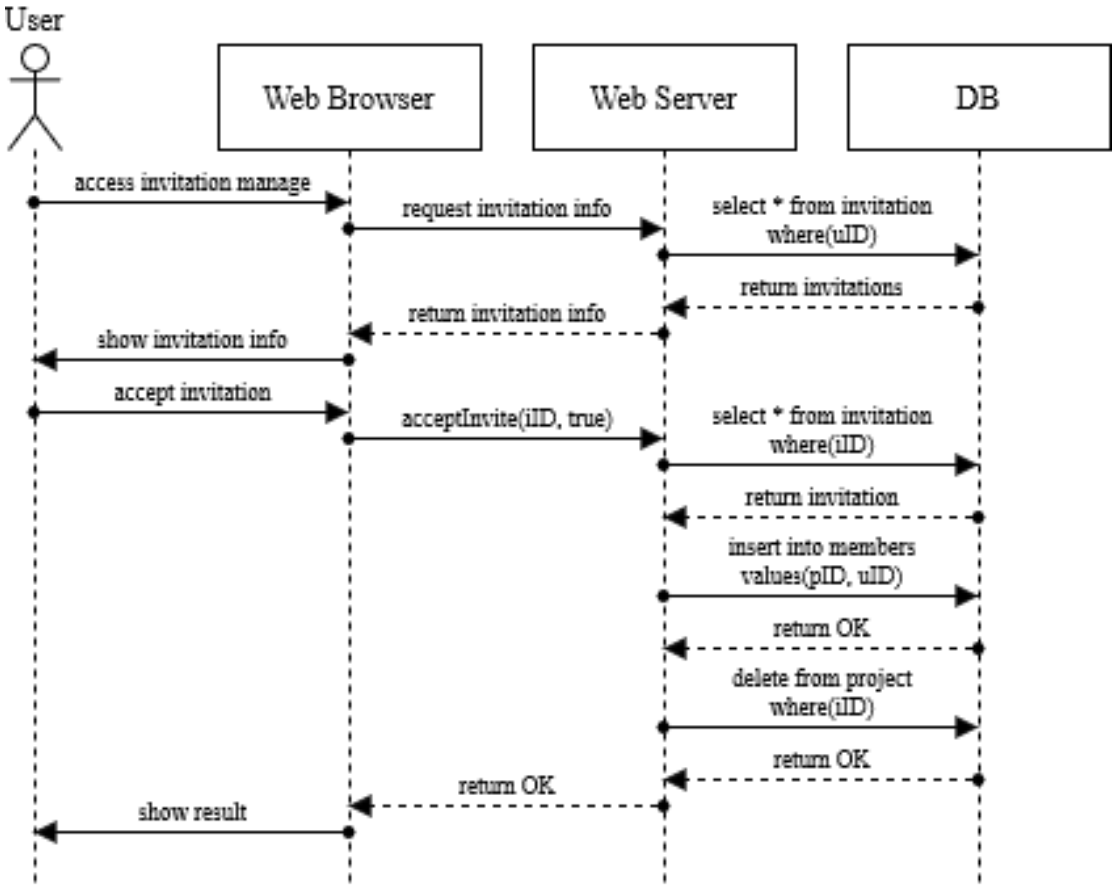


3.3.7. Invitation Accept

1. Class Diagram



2. Sequence Diagram

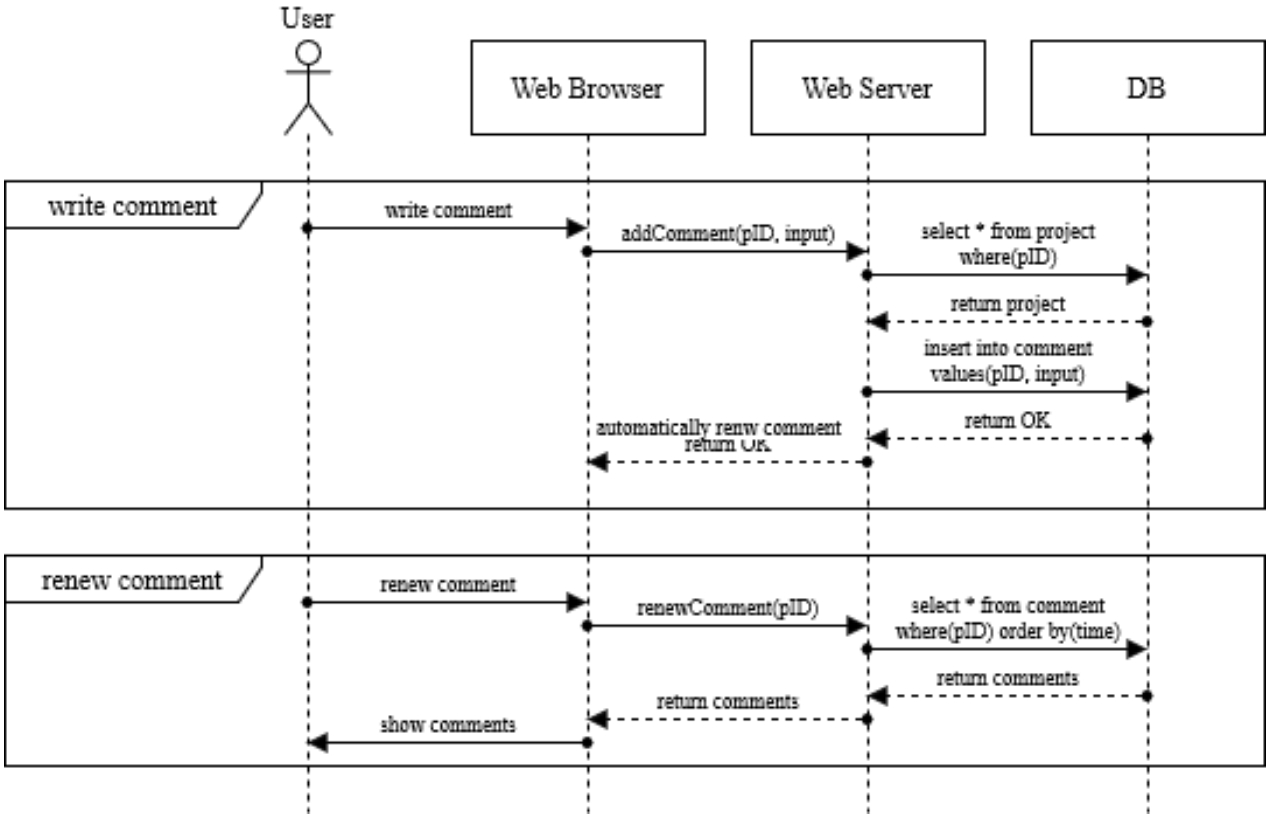


3.3.8. Comment

1. Class Diagram

생략

2. Sequence Diagram

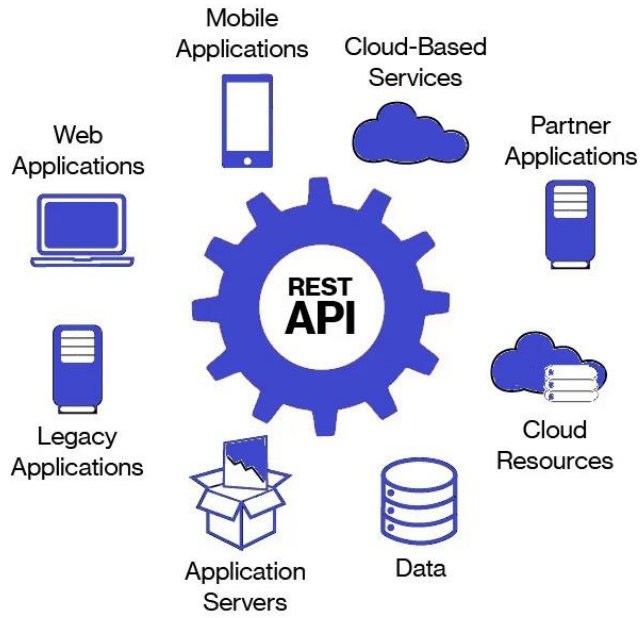


4. Protocol Design

4.1. Objective

본 챕터에서는 각 subsystem 간의 상호작용에 이용되는 프로토콜에 어떤 구조가 사용되었는지 설명하고, 각 인터페이스가 어떻게 정의되어 있는지 서술한다.

4.2. Rest API



본 시스템은 웹 인터페이스인 HTTP를 사용하여 프론트엔드와 백엔드 사이의 통신을 하며, 요청과 응답 형식은 REST API에 따른다. REST API는 서버에 저장되어 있는 각 자원을 이름으로 구분하여 해당 자원의 상태를 주고받는 API의 설계 형식을 의미한다. 즉, 자원 표현에 의한 상태 전달을 돕는다. REST API는 크게 자원, 행위, 표현으로 구성되어 있다.

1) 자원(Resource): URI
 - 각 자원은 고유한 URI를 가진다.

-Client는 URI를 이용해서 자원을 지정하고 해당 자원의 상태에 대한 조작을 서버에 요청한다.

2) 행위(Verb): HTTP Method
 - 서버의 자원에 접근해 상태를 조작하기 위한 요청 행위이다.

- HTTP Method를 통해 표현되며 POST, GET, PUT, DELETE 메서드를 사용한다.

3) 표현(Representation): JSON
 - Client가 자원의 상태에 대한 조작을 요청하면 서버는 이에 적절한 응답을 보낸다.

-JSON, XML, TEXT 등 여러 형태로 나타내어진다.

4.3. Details

4.3.1. Authentication

4.3.1.1. Sign up

1. Request

Method	POST	
URI	/signup	
Request Parameters	E-Mail	User E-Mail
	Password	User password
	Name	User name

2. Response

Success Code	201 Created	
Failure Code	400 Bad Request	
Success Response Parameters	Success	True
Failure Response Parameters	-	-

4.3.1.2. Log in

1. Request

Method	POST	
URI	/login	
Request Parameters	E-Mail	User E-Mail
	Password	User password

2. Response

Success Code	200 OK	
Failure Code	403 Forbidden	
Success Response Parameters	SID	사용자 인증 토큰
Failure Response Parameters	Success	False
	Message	Failure reason

4.3.1.3. Log out

1. Request

Method	POST	
URI	/logout	
Request Parameters	-	-

2. Response

Success Code	200 OK	
Failure Code	400 Bad Request	
Success Response Parameters	success	true
Failure Response Parameters	-	-

4.3.2. Project

4.3.2.1. Search

1.Request

Method	GET	
URI	/project	
Request Parameters	Keyword	Search by keyword
	Tag	Search by tags
	Sorting Criteria	Search by applying sorting criteria
	Filters	Search by applying filters
	SID	사용자 인증 토큰

2. Response

Success Code	200 OK	
Failure Code	404 Not Found (검색 조건에 해당하는 project 없음)	
Success Response Parameters	Projects	Project list
Failure Response Parameters	message	Fail reason 검색 조건에 해당하는 project 존재하지 않음

4.3.2.2. Re-search

1.Request

Method	PATCH	
URI	/project	
Request Parameters	Keyword	Search by keyword
	Tag	Search by tags
	Sorting Criteria	Search by applying sorting criteria
	Filters	Search by applying filters
	From	number
	To	number
	SID	사용자 인증 토큰

2. Response

Success Code	200 OK	
Failure Code	404 Not Found (검색 조건에 해당하는 project 없음)	
Success Response Parameters	Projects	Project list
Failure Response Parameters	message	Fail reason 검색 조건에 해당하는 project 존재하지 않음

4.3.2.3. Write

1. Request

Method	POST	
URI	/project/{project ID}/create	
Request Parameters	title	Project Title
	Project Info	Project Information
	Project Intro	Project Introduction
	SID	사용자 인증 토큰

2. Response

Success Code	303 See Other	
Failure Code	400 Bad Request	
Success Response Parameters	Location	/project/{project ID}
Failure Response Parameters	message	Failure reason Format에 제시된 부분을 모두 작성하지 않음

4.3.2.4. Edit

1. Request

Method	POST	
URI	/project/{project ID}/edit	
Request Parameters	title	Project Title
	Project Info	Project Information
	Project Intro	Project Introduction
	SID	사용자 인증 토큰

2. Response

Success Code	303 See Other	
Failure Code	400 Bad Request	
Success Response Parameters	Location	/project/{project ID}
Failure Response Parameters	message	Failure reason Format에 제시된 부분을 모두 작성하지 않음

4.3.3. Portfolio

4.3.3.1. Search

1.Request

Method	GET	
URI	/developer	
Request Parameters	Keyword	Search by keyword
	Tag	Search by tags
	Sorting Criteria	Search by applying sorting criteria
	Filters	Search by applying filters
	Authorization	사용자 인증 토큰

2. Response

Success Code	200 OK	
Failure Code	404 Not Found (검색 조건에 해당하는 developer 없음)	
Success Response Parameters	Developer	Developer list
Failure Response Parameters	message	Fail reason 검색 조건에 해당하는 developer 존재하지 않음

4.3.3.2. Re-search

1.Request

Method	PATCH	
URI	/developer	
Request Parameters	Keyword	Search by keyword
	Tag	Search by tags
	Sorting Criteria	Search by applying sorting criteria
	Filters	Search by applying filters
	From	number
	To	number
	SID	사용자 인증 토큰

2. Response

Success Code	200 OK	
Failure Code	404 Not Found (검색 조건에 해당하는 developer 없음)	
Success Response Parameters	Developer	Developer list
Failure Response Parameters	message	Fail reason 검색 조건에 해당하는 developer 존재하지 않음

4.3.3.3. Edit

1. Request

Method	POST	
URI	/developer/{user ID}/edit	
Request Parameters	Profile Image	User Profile Image
	Portfolio Info	User Portfolio Information
	School	User School Information
	SID	사용자 인증 토큰

2. Response

Success Code	303 See Other	
Failure Code	400 Bad Request	
Success Response Parameters	Location	/mypage
Failure Response Parameters	message	Failure reason Format에 제시된 부분을 모두 작성하지 않음

4.3.4. Peer Assessment

4.3.4.1. Developer List

1. Request

Method	GET	
URI	/project/{project ID}/developerlist	
Request Parameters	-	-

2. Response

Success Code	200 OK	
Failure Code	403 Forbidden	
Success Response Parameters	Project member info	개발자 목록 (body에 json으로)
Failure Response Parameters	-	-

4.3.4.2. Assessment

1. Request

Method	POST	
URI	/mypage/assessment	
Request Parameters	uID	대상 유저
	pID	대상 프로젝트
	Stars	별표 형태로 평가된 평가 항목들
	text	별표 평가에 추가된 짧은 내용의 텍스트 평가

2. Response

Success Code	303 See Other	
Failure Code	403 Forbidden	
Success Response Parameters	Location	/mypage/assessment
Failure Response Parameters	-	-

4.3.5. Comment

1. Request

Method	POST	
URI	/project/{project ID}/comment/	
Request Parameters	content	Comment content
	SID	사용자 인증 토큰

2. Response

Success Code	201 Created	
Failure Code	403 Forbidden	
Success Response Parameters	-	-
Failure Response Parameters	message	Failure reason

4.3.6. Favorite/Follow

4.3.6.1. Favorite/Unfavorite

1. Request

Method	POST	
URI	/project/{project ID}/favorite	
Request Parameters	favorite	0 or 1
	SID	사용자 인증 토큰

2. Response

Success Code	200 OK	
Failure Code	403 Forbidden	
Success Response Parameters	-	-
Failure Response Parameters	message	Failure reason

4.3.6.2. Follow/Unfollow

1. Request

Method	POST	
URI	/developer/{user ID}/follow	
Request Parameters	follow	0 or 1
	SID	사용자 인증 토큰

2. Response

Success Code	200 OK	
Failure Code	403 Forbidden	
Success Response Parameters	-	-
Failure Response Parameters	message	Failure reason

4.3.7. Message

4.3.7.1. Open websocket

1. Request

Method	GET	
URI	wss://{server URL}/websocket	
Parameters	Upgrade	websocket
	SID	사용자 인증 토큰

2. Response

Success Code	101 Switching Protocols	
Failure Code	400 Bad Request	
Success Response Parameters	Upgrade	websocket
	Connection	Upgrade
Failure Response Parameters	-	-

4.3.7.2. Invite

1. Request

Method	POST	
URI	/project/{project ID}/invite	
Parameters	uID	target user ID
	SID	사용자 인증 토큰

2. Response

Success Code	201 Created	
Failure Code	403 Forbidden	
Success Response Parameters	-	-
Failure Response Parameters	-	-

4.3.7.3. Participation Inquiry

1. Request

Method	POST	
URI	/project/{project ID}/inquiry	
Request Parameters	SID	사용자 인증 토큰

2. Response

Success Code	200 OK	
Failure Code	403 Forbidden	
Success Response Parameters	uID	target project maintainer user ID
Failure Response Parameters	message	Failure reason

4.3.7.4. Privileged Project

1. Request

Method	GET	
URI	/mypage/PreviledgedProjectList	
Parameters	-	-
	SID	사용자 인증 토큰

2. Response

Success Code	200 OK	
Failure Code	403 Forbidden	
Success Response Parameters	type	json
	body	Privileged projects
Failure Response Parameters	message	Failure reason Page Error

4.3.7.5. Invitation Accept/Delete

1. Request

Method	POST	
URI	/mypage/invitation	
Parameters	iID	invitation ID
	isAccepted	Boolean value of whether the request is accepted or not
	SID	사용자 인증 토큰

2. Response

Success Code	200 OK	
Failure Code	403 Forbidden	
Success Response Parameters	-	-
Failure Response Parameters	message	Failure reason Page Error

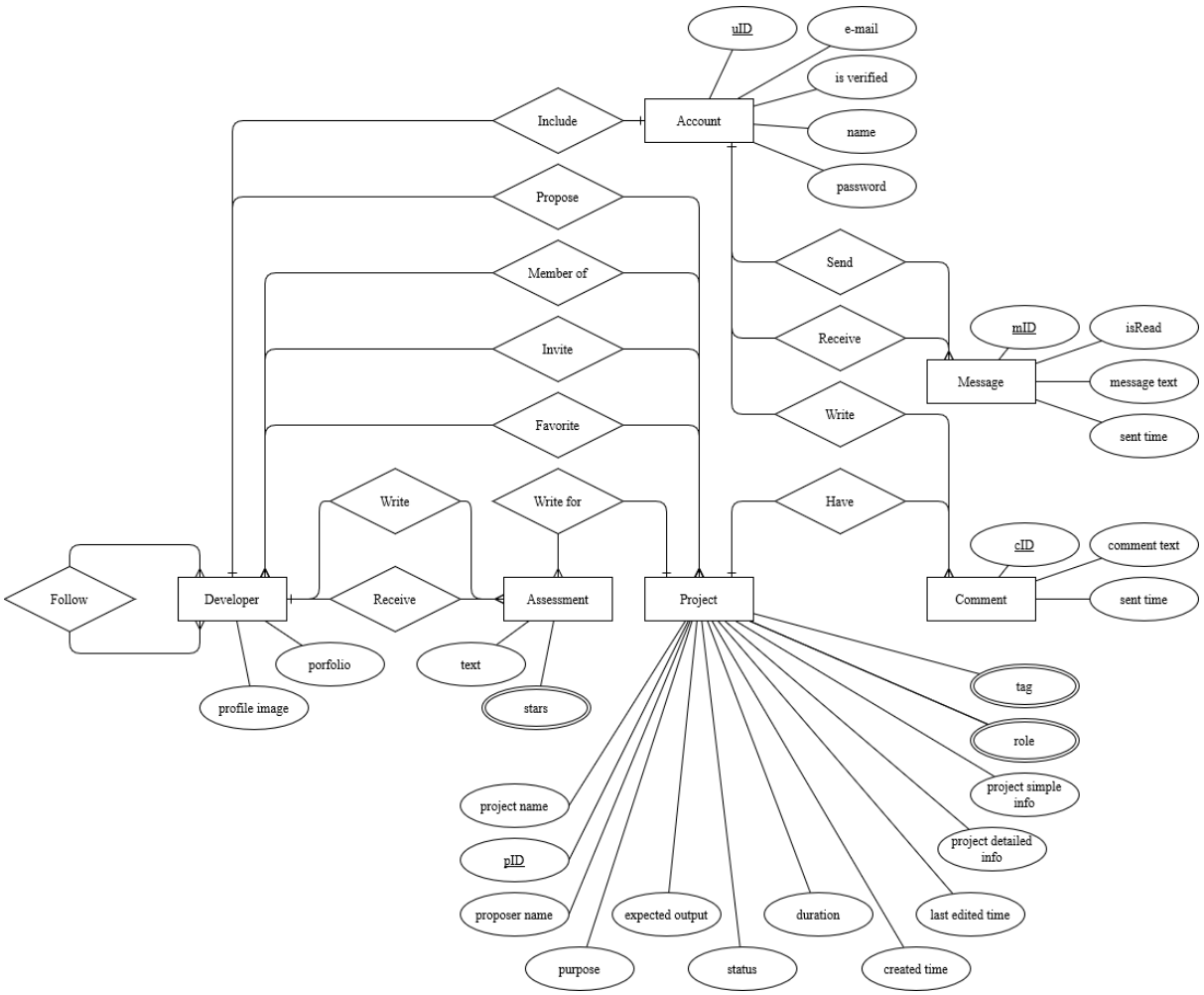
5. Database Design

5.1. Objective

Database Design에서는 Requirement Specification에서 기술한 데이터베이스 요구사항을 기반으로 수정 사항을 반영하여 작성하였다. ER Diagram을 통해 각 Entity와 Entity 사이의 relationship을 표현하고, 이를 활용하여 Relational Schema를 작성한다.

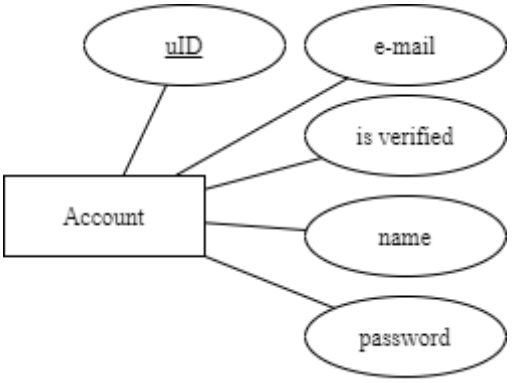
5.2. ER Diagram

MODU에서는 Account, Developer, Project, Assessment, Comment, Message로 총 6개의 Entity가 존재한다. 각각의 Entity는 직사각형 박스의 형태로 표현되고, Entity간의 관계는 마름모꼴로 표현된다. 특정 Entity가 없어도 되는 경우 Client Entity쪽에 작은 원을 표시한다. 각 Entity가 가지는 속성은 타원형으로 표현되고, Primary Key는 라벨에 밑줄을 그어 표시한다. 여러 Attribute를 허용하는 경우 테두리를 이중으로 표시한다.



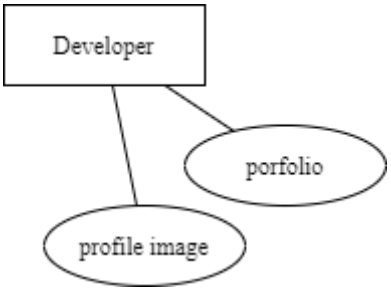
5.3. Entities

5.3.1. Account



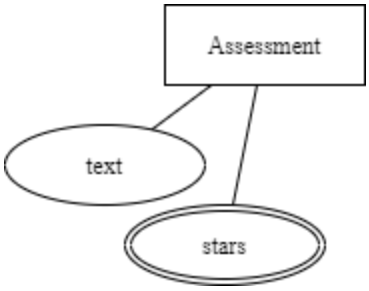
Account Entity는 사용자의 정보를 표현한다. uID 속성이 Primary Key이며, 이름, 이메일, 패스워드 정보를 가지고 있다.

5.3.2. Developer



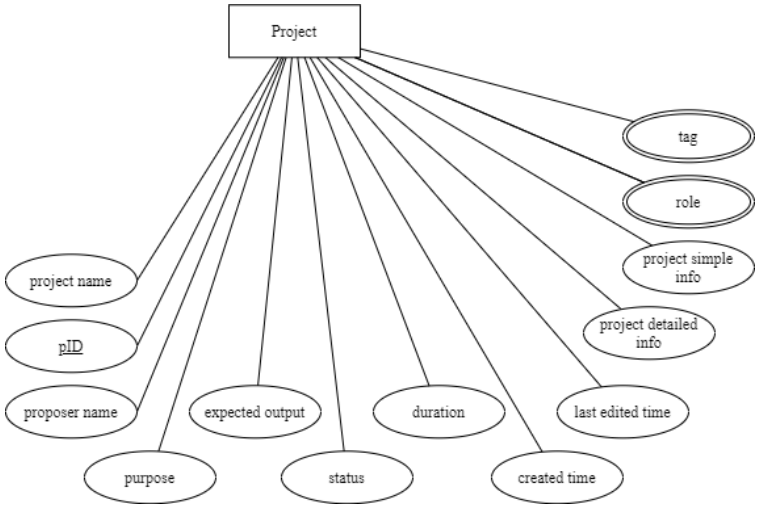
Developer Entity는 개발자의 정보를 표현한다. Primary Key는 존재하지 않으며, 기타 속성으로는 프로필 이미지, 포트폴리오가 있다.

5.3.3. Assessment



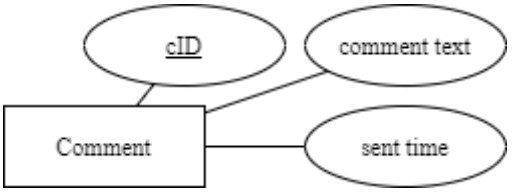
Assessment Entity는 개발자 상호평가 정보를 표현한다. Primary Key는 존재하지 않으며, 기타 속성으로는 stars, text가 있다. Stars는 여러 개의 값을 가질 수 있다.

5.3.4. Project



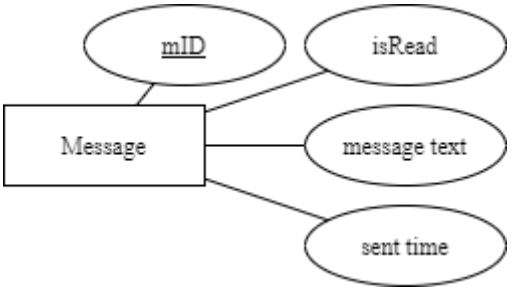
Project Entity는 프로젝트의 정보를 표현한다. pID 속성이 Primary Key이다. 이 외에도 프로젝트 이름, 프로젝트 간단 정보, 프로젝트 상세 정보, 프로젝트 진행상황, 제안자 이름, 프로젝트 목적, 프로젝트 예상 결과, 최초 생성 시간, 마지막 수정 시간, 태그, 모집 역할 정보를 가지고 있다. 모집 역할과 태그는 여러 개의 값을 가질 수 있다.

5.3.5. Comment



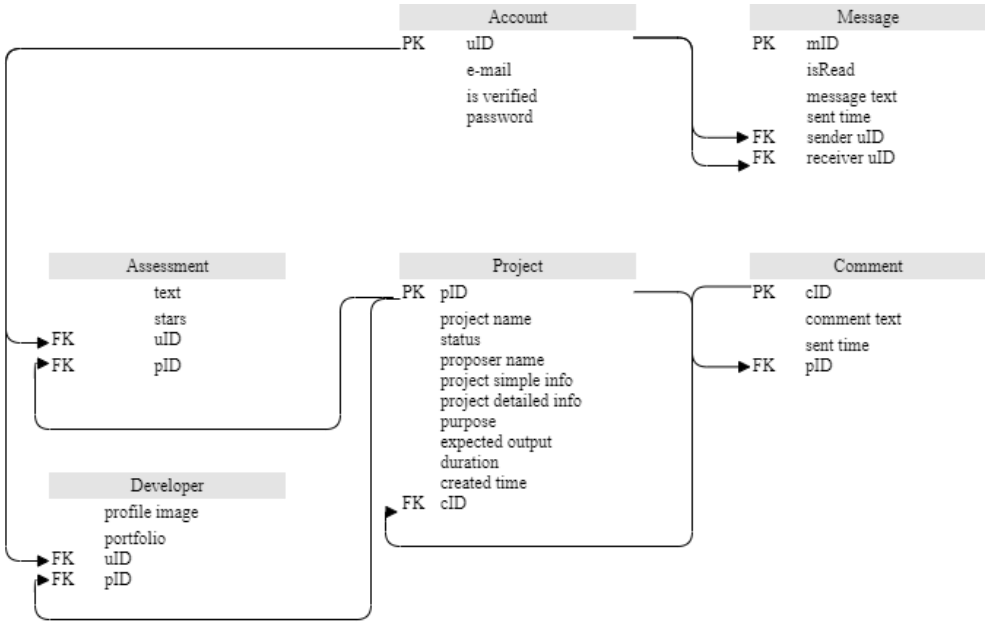
Comment Entity는 프로젝트의 코멘트 정보를 표현한다. cID 속성이 Primary Key이며, 코멘트 내용, 전송 시간 정보를 가지고 있다.

5.3.6. Message



Message Entity는 메시지 정보를 표현한다. mID 속성이 Primary Key이며, 메시지 내용, 전송 시간, 사용자가 메시지 수신 여부에 대한 정보를 가지고 있다.

5.4. Relational Schema



6. Testing Plan

6.1. Objectives

This section establishes a preliminary plan for system testing to determine if the system is operating as intended to identify and analyze the system for defects after completion of the system.

6.2. Testing Policy

During the development of the 'MODU' system, testing is divided into three stages. It is divided into Developing Testing, Release Testing, and User Testing.

6.2.1. Development Testing

Development testing is aimed for synchronizing application of broad spectrum of defect prevention and detection strategies in order to reduce software development risk.

6.2.1.1. Security

6.2.1.1.1. Cross-Site Scripting (XSS) protection

XSS is one of the most basic vulnerability attack methods on the web, and refers to a technique in which a malicious user puts a script in a site to be attacked. If the attack is successful, the user accessing the site executes the inserted code, and usually performs unintended actions or steals sensitive information such as cookies or session tokens.

Django's template converts the particularly dangerous characters <,>, ',' ', &, etc. into special characters (& lt;, & gt ;, etc ..) that only Django can recognize, making the script inoperative so that most XSS Prevents attacks. However, there are many other ways to attack this vulnerability, which requires a lot of attention and testing.

6.2.1.1.2. Cross site request forgery (CSRF) protection

CSRF refers to an attack that causes users to request an intended action (modification, deletion, registration, etc.) to a specific website regardless of their will. XSS is aimed at a user trusting a particular website, whereas CSRF aims at a state where a particular website is crediting the user's web browser.

When a victim enters a web page with a prepared attack code, the victim sends a request (GET, POST) prepared by the attacker to the server without his knowledge. That's why CSRF protection is done by examining all requests. Django has built-in protection for most types of CSRF attacks, so we need to test if the CRSF token is used where the request is possible.

6.2.1.1.3. SQL injection protection

SQL injection refers to an attacker executing arbitrary SQL code in the main database.

This attack could leak information from the Web or delete records from the database.

Django protects the resulting SQL from potential database attackers by using a query set. However, when using raw query or private sql, we should test the existence of parameters accessible to the user.

6.2.1.1.4. Clickjacking protection

Clickjacking protection causes a user to click on a link other than the one they wanted to click without notice. In other words, the user thinks that he or she clicked on a web page or button, but actually the user clicked the content of another page. Django has a clickjacking defense in the form of X-Frame-Options middleware. This prevents sites from being trapped in frames within supported browsers. However, there is a risk that it will not work in browsers other than the supported browsers (Internet Explorer 8+, Firefox 3.6.9+, Opera 10.5+, Safari 4+, Chrome 4.1+), so we should test it and look for alternatives

6.2.2. Release Testing

Release testing is a process to test the newer version /build of a software/ application to make sure that is flawless and doesn`t have any issues and works as intended. It has to be done prior to release.

6.2.3. User Testing

User testing can be explained by the concept of usability testing. We should set up scenario and realistic situation that can proceed necessary user tests. Since MODU requires free recruitment and interactive communication between members, it is necessary to minimize the intervention of administrator. However, it is necessary to intervene in matters that violate the internal policy, such as illegal activities or creation of false profiles. Because these two opposite goals have to be achieved, the dependability of the MODU must also be tested.

6.3. Testing Case

From basic member management to service use from the user's point of view, the overall test will be conducted on the provision and use of MODU services.

7. Development Plan

7.1. Objectives

This section describes the programming language and the IDE that is going to be used.

7.2. Frontend/Backend Environment

7.2.1. Django

Django is a full-stack Python web framework that enables rapid development and clan design. Django enables rapid development, reassures security, and is scalable and flexible. The core architecture of

Django is MVC (Model-View-Controller). Django comes with all the necessary essentials needed to solve common cases. Using Django and its plentiful open-source packages, our team can build a website quickly with high security with perfection.

7.2.2. Docker

Docker is an open platform designed to make it easier to develop, deploy, and run applications using containers. Containers allow developers to package up an application with all the parts needed such as libraries, and deploy it as one package. With the attribute of the container, the developer can be assured that the application will run on any other Linux machine regardless of any different settings that the machine compared to the machine used to build and test the code. Docker is lightweight but enables the program to run directly within the host machine's kernel. Using Docker, our team is able to use the NAS image to execute the NAS container for the server.

7.2.3. Firebase

Firebase Authentication enables authentication of users using passwords and popular federated identity providers such as Google, Facebook, and Twitter. This authentication system allows developers to save time on developing web service methods for authentication, and provides detailed analytics. Using this system, our team can safely authenticate users simply using their school email accounts while verifying that each user is a student.

7.3. Schedule

Phase	Due Date
Concept Decision	5/2
Proposal & UI Design	5/3
Requirements Specification	5/13
Design Specification	5/24
Code and Test	6/7
Presentation	6/14
Code Review	6/22-6/26