Software Design Specification

for

Codemy

2017314333 김영우
2018312707 김준형
2017310127 박충현
2018315379 송지은
2020311119 조성현
2017314593 조현준

소프트웨어공학개론 4조

2023. 05. 21

목차

1. Preface **6**

1.1. Objective 6

1.2. Readership 6

1.2.1. User Requirement Readership 6

1.2.2. System Requirement Readership 6

1.3. Document Structure 6

1.3.1. Preface 6

1.3.2. Introduction 6

1.3.3. Overall System Architecture 6

1.3.4. System Architecture - Frontend 6

1.3.5. System Architecture - Backend 7

1.3.6. Protocol Design 7

1.3.7. Database Design 7

2. Introduction **8**

2.1. Applied Diagram 8

2.1.1. Class Diagram 8

2.1.2. Sequence Diagram 8

2.1.3. State Diagram 8

2.1.4. ER Diagram 8

2.2. System Overview 8

3. System Architecture – Overall **10**

3.1. Objective 10

3.2. System Organization 10

3.3. Sub system 11

3.3.1. 사용자 시스템 11

3.3.2. 문제 추천 시스템 12

3.3.3. 문제 풀이 시스템 13

3.3.4. 코드 분석 시스템 14

3.3.5. 학생 관리 시스템 15

4. System Architecture – Frontend **16**

4.1. Objectives 16

4.2. Main page 16

4.2.1. Attributes 16
4.2.2. Methods 17
4.2.3. State Diagram 17
4.3. Problem List Page 19
4.3.1. Attributes 19
4.3.2. Methods 19
4.3.3. State Diagram 19
4.4. Problem Page 20
4.4.1. Attributes 20
4.4.2. Methods 21
4.4.3. State Diagram 21
4.5. View Feedback Page 22
4.5.1. Attributes 22
4.5.2. Methods 22
4.5.3. State Diagram 22
4.6. Submission List Page 23
4.6.1. Attributes 23
4.6.2. Methods 23
4.6.3. State Diagram 24
4.7. Create Feedback Page 25
4.7.1. Attributes 25
4.7.2. Methods 25
4.7.3. State Diagram 26
5. System Architecture – Backend **27**
5.1. Objectives 27
5.2. Overall Architecture 27
5.3. Subcomponents 28
5.3.1. User Management System 28
5.3.2. Problem Management System 30
5.3.3. Submission Management System 31
5.3.4. Feedback Management System 33
6. Protocol **35**
6.1. Objectives 35

6.2. HTTP 35
6.3. JSON 35
6.4. Protocol Description 36
6.4.1. 학생 회원가입 프로토콜 36
6.4.2. 강사 회원가입 프로토콜 37
6.4.3. 로그인 프로토콜 38
6.4.4. 문제 확인 프로토콜 39
6.4.5. 코드 제출 프로토콜 40
6.4.6. 피드백 확인 프로토콜 (학생) 41
6.4.7. 피드백 확인 프로토콜 (강사) 42
6.4.8. 코멘트 작성 프로토콜 (강사) 43
7. database Design **44**
7.1. Objectives 44
7.2. ER Diagram 44
7.3. Entity 45

그림 목차

Figure 1. System Architecture: 전체 시스템 구조 10

Figure 2. System Architecture: 사용자 시스템 11

Figure 3. System Architecture: 문제 추천 시스템 12

Figure 4. System Architecture: 문제 풀이 시스템 13

Figure 5. System Architecture: 코드 분석 시스템 14

Figure 6. System Architecture: 학생 관리 시스템 15

Figure 7. State Diagram of Main Page 18

Figure 8. State Diagram of Problem List Page 20

Figure 9. State Diagram of Problem Page 21

Figure 10. State Diagram of View Feedback Page 22

Figure 11. State Diagram of Submission List Page 24

Figure 12. State Diagram of Create Feedback Page 26

Figure 13. Overall Backend Architecture 27

Figure 14. User Management System Class Diagram 28

Figure 15. User Management System Sequence Diagram 29

Figure 16. Problem Management System Class Diagram 30

Figure 17. Problem Management System Sequence Diagram 30

Figure 18. Submission Management System Class Diagram 31

Figure 19. Grade Management System Sequence Diagram 32

Figure 20. Feedback Management System Class Diagram 33

Figure 21. Feedback Management System Sequence Diagram 34

Figure 22. 학생 회원가입 프로토콜 36

Figure 23. 강사 회원가입 프로토콜 37

Figure 24. 로그인 프로토콜 38

Figure 25. 문제 확인 프로토콜 39

Figure 26. 코드 제출 프로토콜 40

Figure 27. 피드백 확인 프로토콜 (학생) 41

Figure 28. 피드백 확인 프로토콜 (강사) 42

Figure 29. 코멘트 작성 프로토콜 (강사) 43

Figure 30. ER Diagram 44

1. Preface

1.1. Objective

이 장에서는 본 문서의 예상된 독자, 문서의 구조, 각 단원에서 전달하고자하는 사항에 대해 보다 세부적으로 설명한다.

1.2. Readership

1.2.1. User Requirement Readership

시스템이 제공하는 서비스와 운영 제약 조건에 대해 서술하며 사용자가 쉽게 이해하기 위해 자연어, 도표 등을 활용하여 작성한다. 주된 독자로는 Client manager과 contractor manager을 상정한다.

1.2.2. System Requirement Readership

시스템의 기능, 서비스 및 운영 제약 조건에 대한 자세한 설명을 서술한다. 계약의 일부가 될 수 있으므로 최대한 상세하게 기술해야 한다. 시스템 개발 과정에 참여하는 개발자, architects, system을 계약하는 client 등이 주된 독자가 된다.

1.3. Document Structure

1.3.1. Preface

본 문서의 예상된 독자, 문서의 구조, 각 단원에서 전달하고자하는 사항에 대해 보다 세부적으로 설명한다.

1.3.2. Introduction

이 장에서는 본 문서 시스템의 필요성과 시스템의 기능을 간략히 소개한다. 다른 시스템과 어떻게 작동하는지, 시스템이 조직의 비즈니스 또는 전략적 목표에 어떻게 적합하는지 설명한다.

1.3.3. Overall System Architecture

이 장에서는 본 시스템 구조 개요를 소개하고 다이어그램을 통해 시스템 간의 관계를 설명한다. 개발자가 시스템의 전체적인 구조를 이해하는 데에 도움을 준다.

1.3.4. System Architecture - Frontend

이 장에서는 프론트엔드 시스템의 구조, 속성 및 기능, 시스템을 구성하는 각 요소들의 관계를 기술한다. 사용자의 이해를 돕기 위해 자연어, 그림, 도표 등을 활용한다.

1.3.5. System Architecture - Backend

이 장에서는 백엔드 시스템의 구조를 시각화하여 제시하고, 시스템을 구성하는 각 요소들의 관계를 기술한다. 사용자의 이해를 돕기 위해 자연어, 그림, 도표 등을 활용한다.

1.3.6. Protocol Design

이 장에서는 서브 시스템 간의 상호작용에 필수적으로 준수해야 하는 프로토콜에 관해 설명하고, 통신 과정에서 전달되는 메시지의 형식과 용도, 의미를 설명한다.

1.3.7. Database Design

이 장에서는 database design의 전반적인 부분에 대해 설명한다. ER diagram을 통해 Entity와 Relationship을 설명하고, 각각의 Entity의 attribute에 대해 설명한다.

2. Introduction

이 장에서는 본 문서 시스템의 필요성과 시스템의 기능을 간략히 소개한다. 다른 시스템과 어떻게 작동하는지, 시스템이 조직의 비즈니스 또는 전략적 목표에 어떻게 적합하는지 설명한다.

2.1. Applied Diagram

2.1.1. Class Diagram

클래스 다이어그램은 시스템을 구성하는 클래스, 구성요소와 클래스 간의 관계를 보여주며 시스템의 구조를 설명하는 다이어그램이다. 이 다이어그램은 클래스를 상자로 나타내며 각 상자에는 클래스의 이름, 속성, 메소드가 포함되고 상자들은 화살표로 연결된다. 시스템의 구조를 시각화하여 설계 및 변경 사항을 추적하는 데 도움이 된다.

2.1.2. Sequence Diagram

시퀀스 다이어그램은 특정 use case 또는 use case instance 중 발생하는 상호작용을 순서대로 표현하는 다이어그램이다. 이 다이어그램은 언제 어떤 메시지가 전송되었는지를 시각적으로 보여주며 객체는 세로로 나열되고 상호작용은 화살표로 표시된다. 시퀀스 다이어그램은 시스템의 동작을 이해하고 디버깅하는 데 유용하다.

2.1.3. State Diagram

스테이트 다이어그램은 시스템이 외부 및 내부 이벤트에 대해 어떤 반응을 하는지 보여주는 다이어그램이다. 시스템이 가질 수 있는 상태나 조건들을 표현하며 시스템이 한 상태에서 다른 상태로 변화하는 이벤트도 표시된다. 스테이트 다이어그램은 복잡한 시스템을 모델링하고 시스템 동작을 시각화하는 데 유용하다.

2.1.4. ER Diagram

ER 다이어그램은 Entity-relationship Diagram의 약자로서 개체, 속성 및 개체 간의 관계를 나타내는 다이어그램이다. ER 다이어그램은 데이터 모델링을 시각적으로 이해하기 편하게 표현한다.

2.2. System Overview

Codemy는 교육 효율을 높이기 위해 개발된 학원 코딩 교육 서비스이다. 이 서비스를 사용하는 학원 강사들은 학생들이 코드를 제출한 후 ChatGPT에 입력하여 피드백을 받게 된다. 강사는 받은 피드백을 참고하여 코멘트를 남겨 학생에게 전달한다. 이를 통해 각 학생의 능력을 시각화하고 데이터 분석을 통해 학생들의 강점과 약점을 파악하여 보완할 수 있게 된다. 또한 ChatGPT는 학생들이 감당할 수 있는 수준의 문제를 추천하여 학생들이 문제를 풀고 학습할 수 있는 데이터 기반 서비스도

제공한다. Codemy에서 구현하고자 하는 사항은 다음과 같다.

a. Efficiency: 코드의 효율성. 코드가 돌아가는 속도를 빨라지게 하거나, 사용되는 메모리를 줄이는 등의 개선점을 보여준다. 더 나은 알고리즘을 사용하거나 반복되는 계산을 줄이거나 더 나은 자료구조를 사용하는 등의 방법으로 시간복잡도와 공간복잡도를 더 효율적으로 만들어준다.
b. Readability: 코드의 가독성. 다른 개발자들이 읽기 쉽게해서 유지보수 하기에 쉽게 만든다. 변수와 함수이름을 정해주거나 필요한 부분에 주석을 달거나 하는등의 개선점을 알려준다.
c. Correctness: 코드가 입력받은 원래 코드와 같은 결과를 출력하는지 확인한다. 특이 케이스들과 에러부분을 같이 수정하는지 확인한다.
d. Scalability: 입력값의 크기가 달라졌을때 코드가 마찬가지로 효율적인지 확인한다. 연산을 병렬화하고 작업을 더 잘게 쪼개는 등의 방법이 있다.
e. Modularity: 코드를 여러번 쓰일 수 있는 작은 부분들로 나눠서 중복되는 코드들을 줄이고 코드의 크기를 작게 만들 수 있다.
f. Security: 코드를 더 안전하게 만든다. 코드삽입 공격을 막기 위해 입력값이 적절한 형식인지 확인한다. 민감한 정보들이 제대로 암호화 되어 저장되는지 확인한다.

3. System Architecture – Overall

3.1. Objective

이 장에서는 본 시스템 구조 개요를 소개하고 다이어그램을 통해 시스템 간의 관계를 설명한다. 개발자가 시스템의 전체적인 구조를 이해하는 데에 도움을 준다.

3.2. System Organization

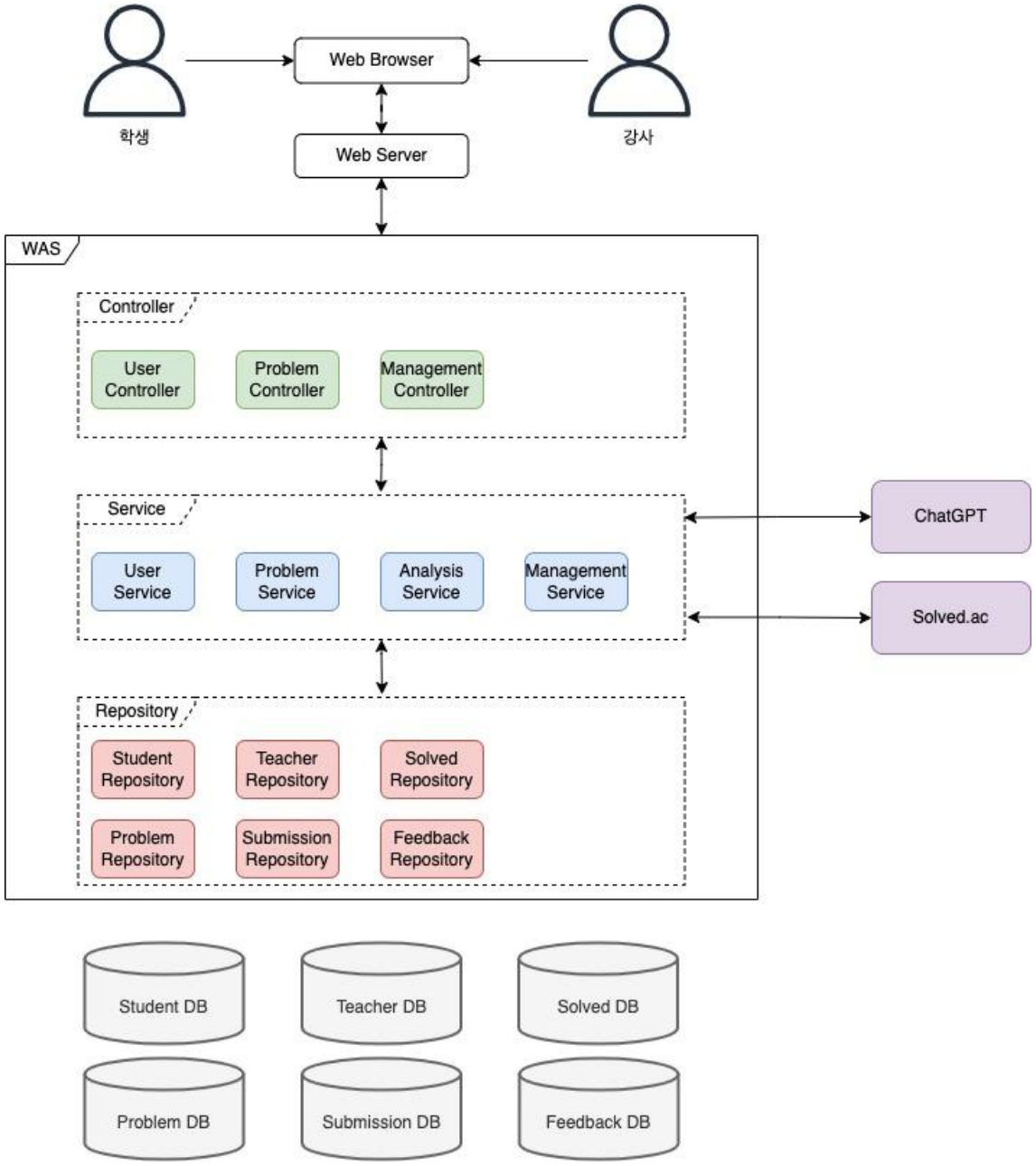

Figure 1. System Architecture: 전체 시스템 구조

전체 시스템 구조는 Web Server와 WAS 및 DB로 구성된 구조이다. 사용자는 Web Browser로 접속하여 시스템을 이용할 수 있다. WAS는 Spring Boot Application으로, 사용자가 요청한 HTTP 요청을 받아들이고, 이를 처리하여 응답을 반환하는 **Controller**, 비즈니스 로직을 구현하는 **Service**, 데이터베이스와의 연동을 처리하는 **Repository**로 구성되어 있다.

3.3. Sub system

3.3.1. 사용자 시스템

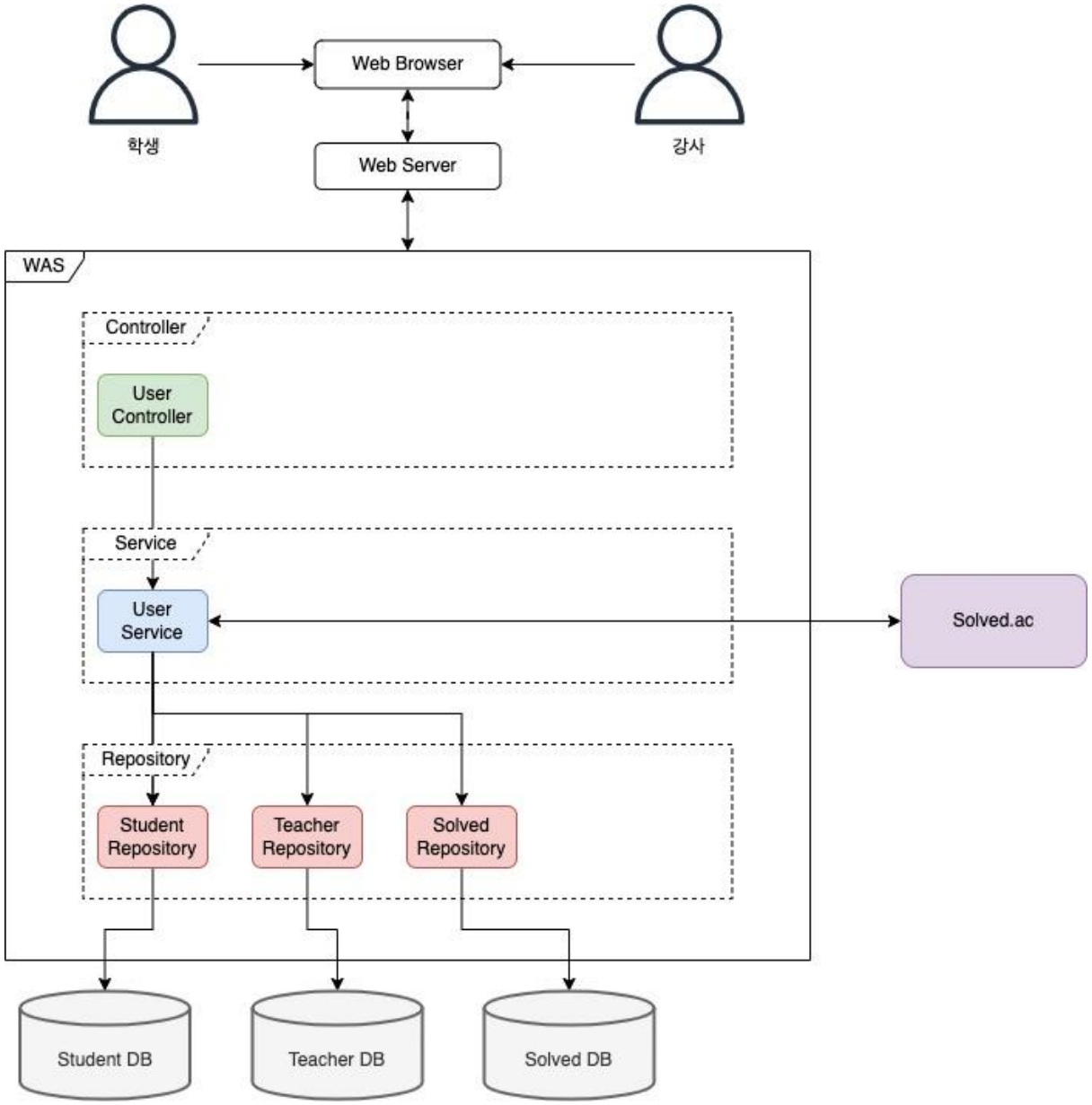

Figure 2. System Architecture: 사용자 시스템

사용자 시스템은 학생 및 강사의 회원가입 및 로그인과 같은 전반적인 사용자 관련

요구사항을 담당하는 시스템이다. 학생 회원가입 시, 백준 ID를 입력 받아 지금까지 풀었던 문제들을 조회하고 저장한다.

3.3.2. 문제 추천 시스템

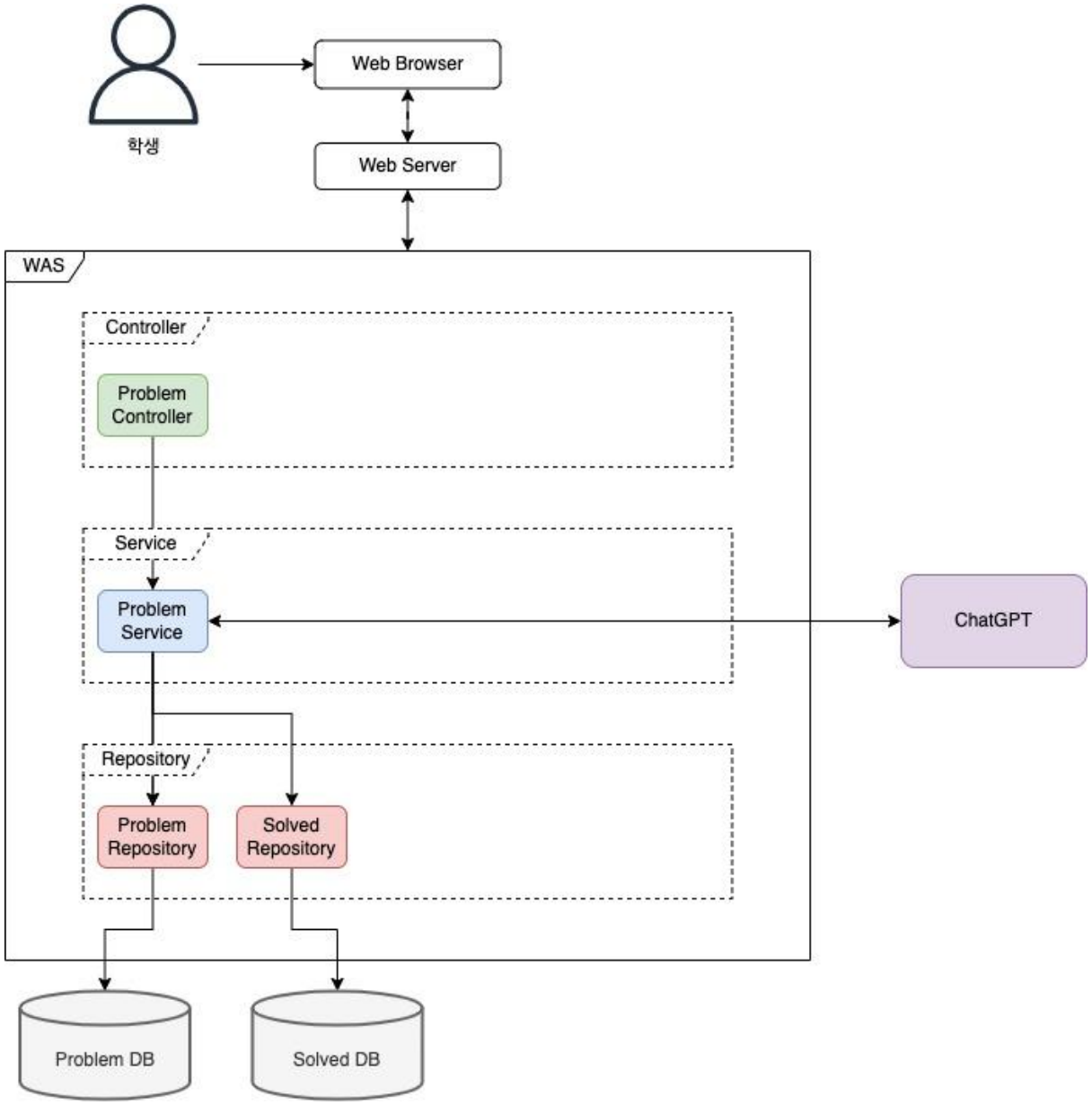

Figure 3. System Architecture: 문제 추천 시스템

문제 추천 시스템은 학생이 메인 페이지에 접속했을 때 지금까지 풀었던 문제들을 기반으로 ChatGPT를 이용하여 새로운 백준 문제들을 추천해주는 시스템이다.

3.3.3. 문제 풀이 시스템

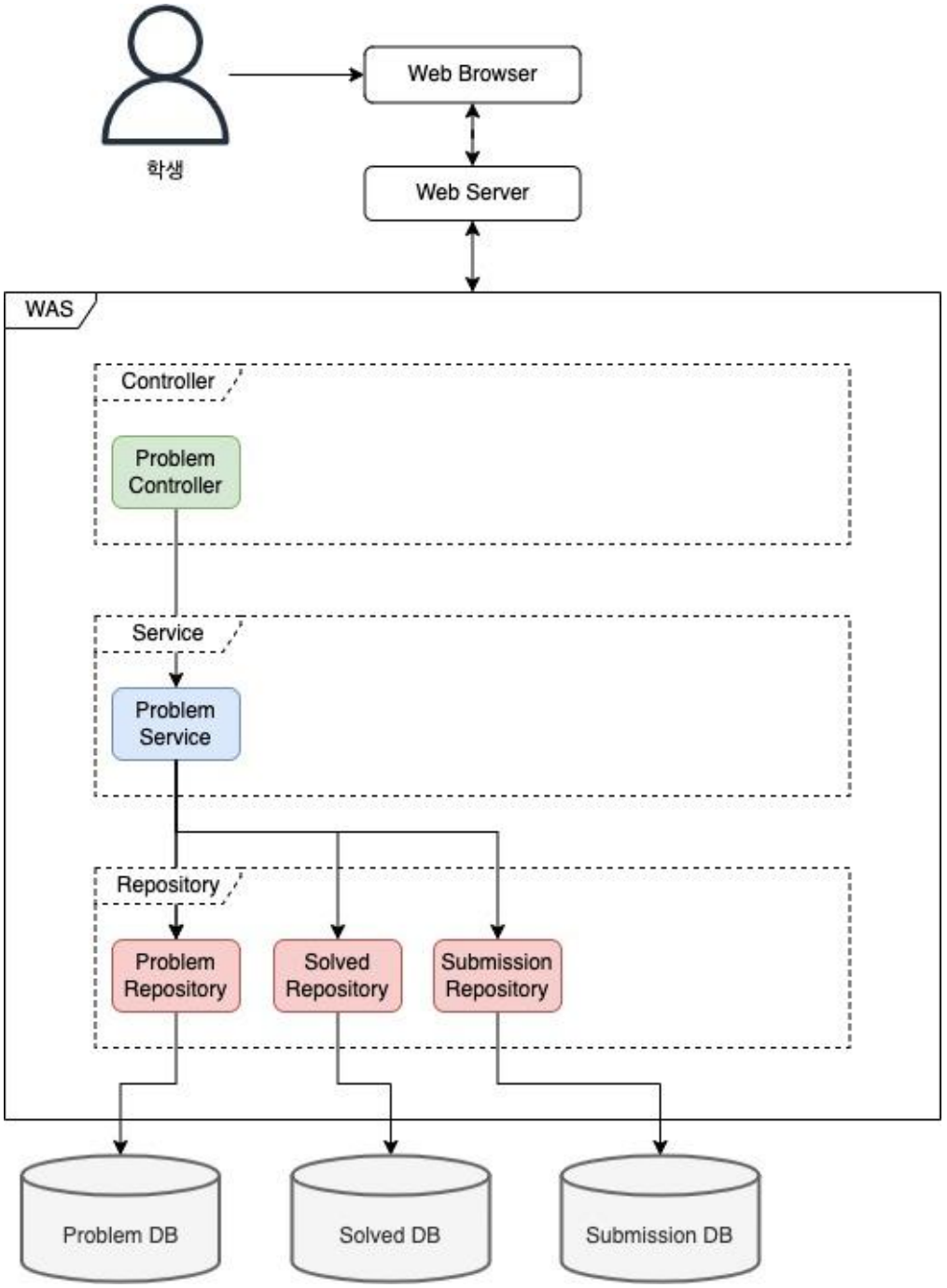

Figure 4. System Architecture: 문제 풀이 시스템

문제 풀이 시스템은 “문제에 대한 학생의 코드 제출” 요구사항을 처리하는 시스템이다. 코드 제출 이후 제출 코드는 Submission DB에 저장되고, 학생이 푼 문제는 Solved DB에 추가된다. 또한, 분석은 제출한 코드를 바탕으로 코드 분석 시스템을 통해 이루어진다.

3.3.4. 코드 분석 시스템

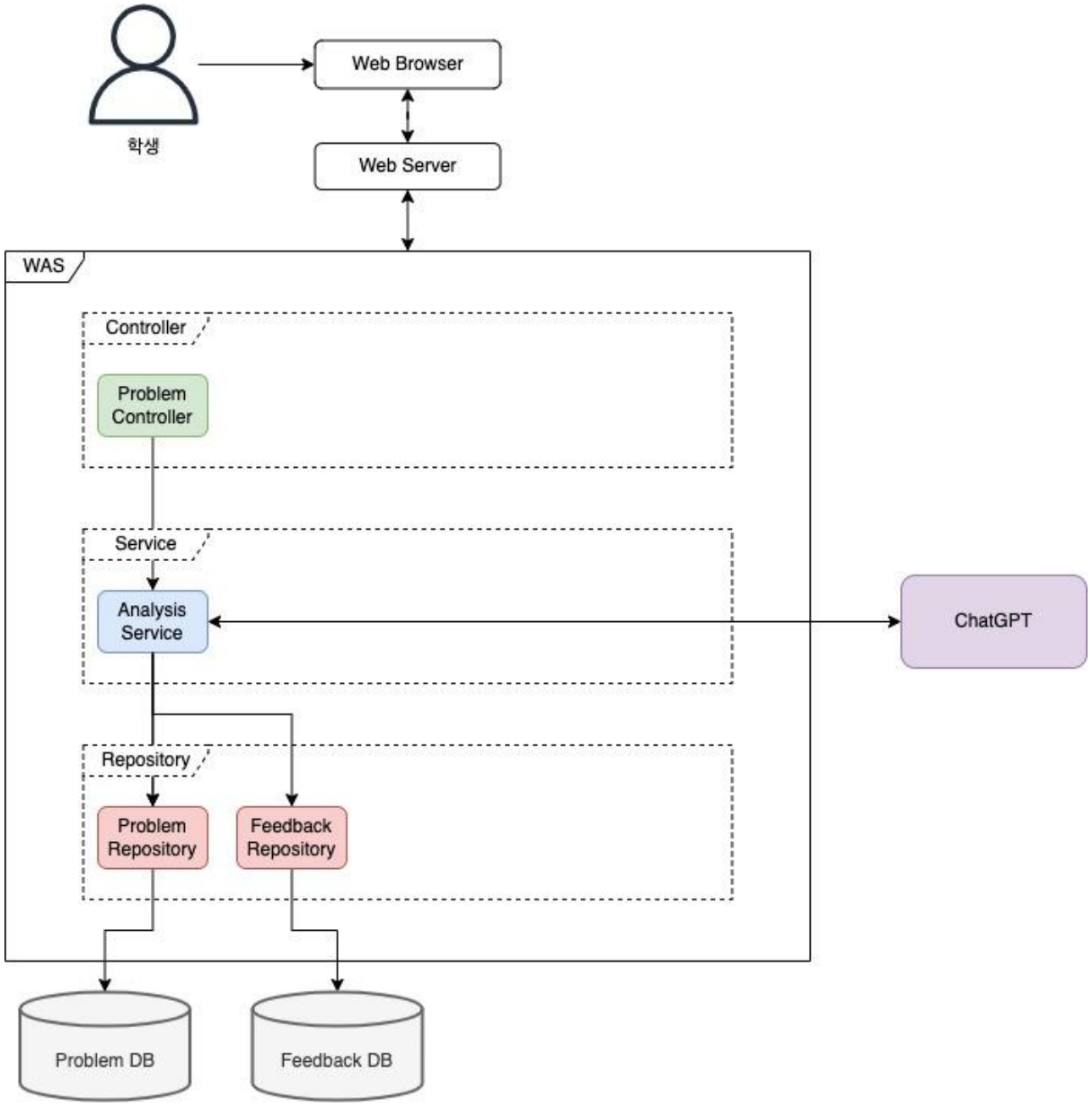

Figure 5. System Architecture: 코드 분석 시스템

코드 분석 시스템은 제출 코드에 대한 분석을 처리하는 시스템이다. ChatGPT를 통해 제출 코드에 대한 efficiency, readability, correctness, scalability, modularity, security에 대한 점수 및 코드에 대한 전체적인 피드백을 얻는다.

3.3.5. 학생 관리 시스템

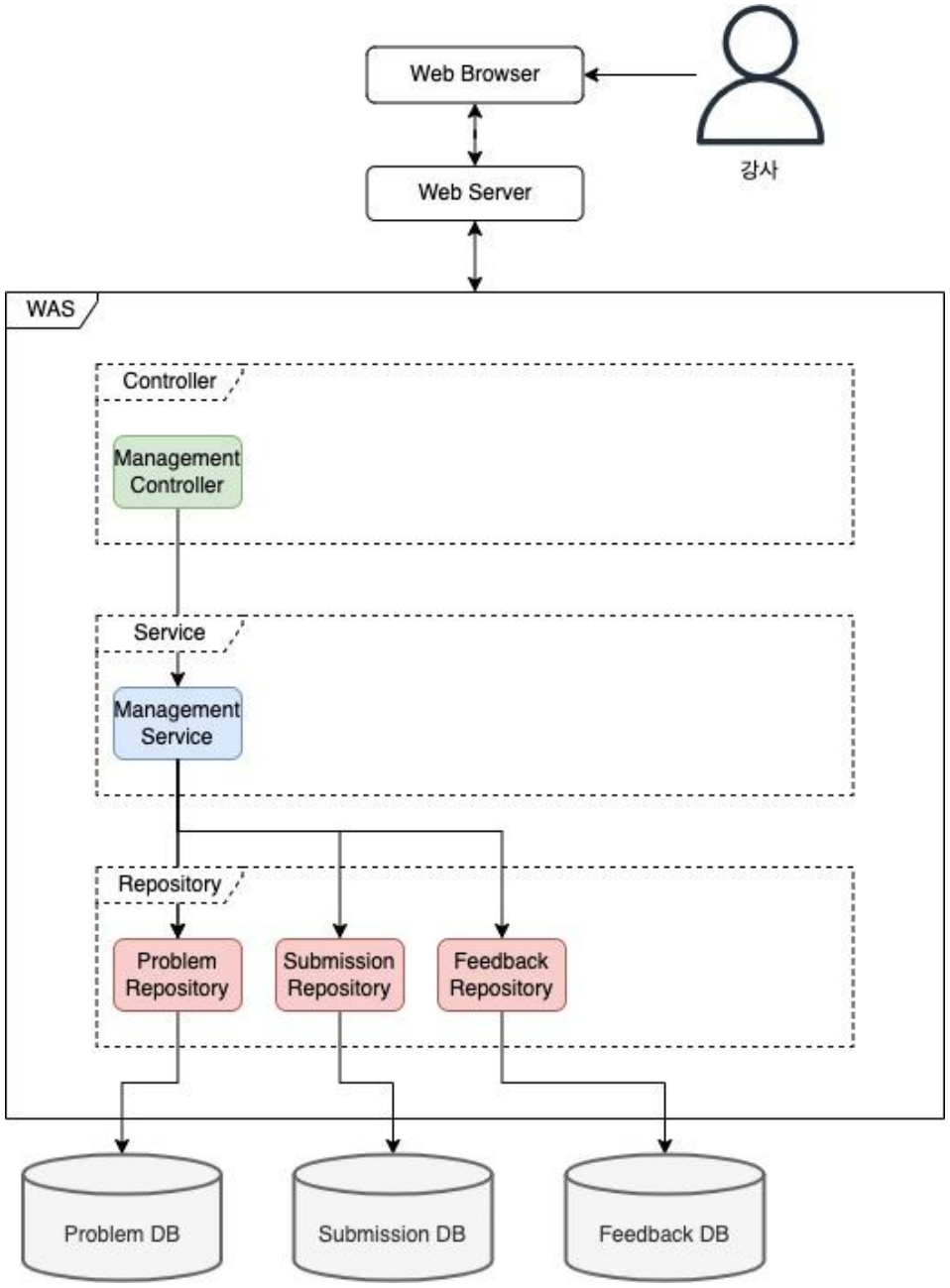

Figure 6. System Architecture: 학생 관리 시스템

학생 관리 시스템은 강사가 학생들을 관리하는 시스템이다. 이를 통해 강사는 학생들이 제출한 코드를 확인하고, ChatGPT의 피드백을 확인할 수 있으며, 이를 수정하여 학생에게 코멘트를 전달할 수 있다.

4. System Architecture – Frontend

4.1. Objectives

이 장에서는 프론트엔드 시스템의 구조, 속성 및 기능, 시스템을 구성하는 각 요소들의 관계를 기술한다. 사용자의 이해를 돕기 위해 자연어, 그림, 도표 등을 활용한다.

4.2. Main page

Codemy의 메인 페이지로, 처음 시스템에 접근하면 볼 수 있는 페이지이다. 페이지 중앙에는 Codemy가 제공하는 학원 코딩 교육 플랫폼에 대한 간략한 설명이 기재되어 있다. 또한 로그인 정보에 따라서 학생 사용자인 경우 문제와 코멘트 목록으로 이동할 수 있는 태그를 제공하고, 강사 사용자의 경우 학생이 제출한 답안을 확인할 수 있는 페이지로 이동하는 태그를 제공한다.

4.2.1. Attributes

메인 페이지는 크게 2가지 파트로 구성되어 있다.

- Head: 메인페이지의 head 부분으로, 사용자가 원하는 페이지로 이동할 수 있도록 하는 네비게이션 태그로 구성되어 있다.
- CodemyInfo : Codemy가 제공하는 서비스에 대한 간략한 설명을 제공하며, 플랫폼에 대한 더 자세한 설명을 담은 링크로 이동할 수 있는 버튼을 제공한다.

(1) Head는 아래의 속성을 가지고 있다.

Home : 홈 버튼이다.

Study : 문제 풀이 및 코멘트 시스템으로 이동할 수 있는 버튼이다. 학생은 문제 리스트로, 강사는 학생이 제출한 답안 리스트 페이지로 이동할 수 있다.

Login : Login 버튼이다.

(2) CodemyInfo는 아래의 속성을 가지고 있다.

ServiceInformation: Codemy 서비스에 대해 간략히 설명한다.

LearnMore: 자세한 서비스 내용과 결제 정보 등의 내용을 담은 페이지로 이동하는 버튼이다.

4.2.2. Methods

메인 페이지가 가지고 있는 method를 파트에 따라 나누면 다음과 같다.

(1) Head가 가지고 있는 method는 아래와 같다.

clickHome() : 사용자가 Home 버튼을 눌렀을 때 초기 상태의 메인 페이지로 돌아간다.

setLogin(): Login 정보를 입력하는 페이지로 넘어간다. 로그인이 완료되면 사용자의 id를 텍스트로 띄워준다.

clickStudy(): 학생 사용자는 문제 리스트와 피드백 현황을 볼 수 있는 페이지로 이동하고, 강사 사용자는 학생이 제출한 답안 목록을 볼 수 있는 페이지로 이동한다.

(2) CodemyInfo가 가지고 있는 method는 아래와 같다.

clickReadmore() : Codemy의 세부 서비스 정보를 보여주는 페이지로 이동한다.

4.2.3. State Diagram

아래는 사용자가 Codemy의 메인 페이지를 사용할 때의 상태를 그린 state diagram이다.

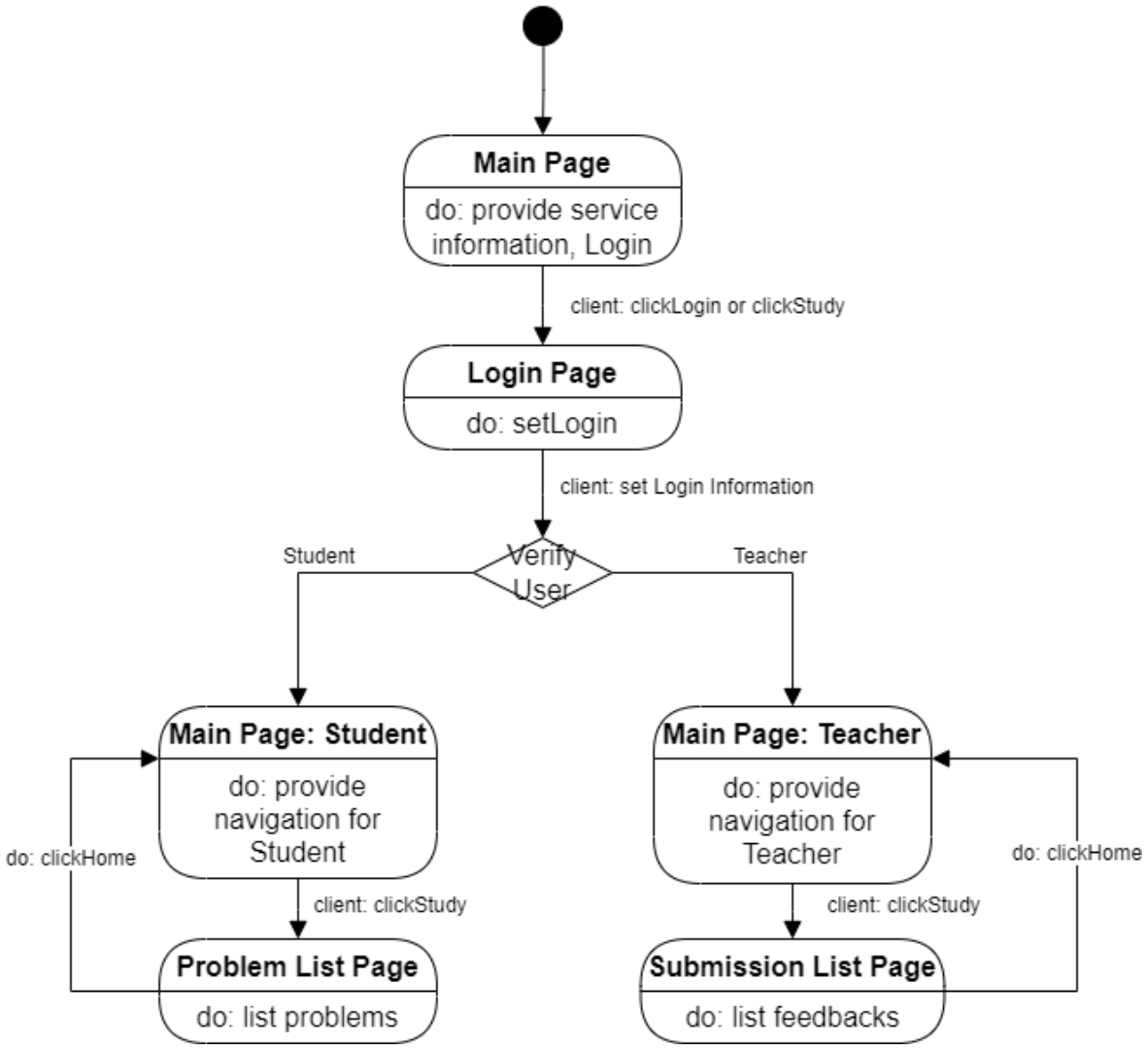

Figure 7. State Diagram of Main Page

사용자가 Codemy에 접속하면 Main Page는 페이즈 중앙에 서비스에 대한 간략한 설명을 제공한다. 학생 혹은 강사를 위한 서비스가 개별적으로 존재하기 위해서 사용자는 로그인을 수행해야 한다. Login 이후에 Study 태그를 클릭하면, 해당 사용자가 학생인지, 강사인지에 따라서 문제를 풀거나 푼 문제에 답변해줄 수 있는 페이지로 이동할 수 있다.

학생 사용자가 Study 태그를 클릭하는 경우, Problem List Page로 이동하게 된다. 이 페이지에는 학생이 추천받은 문제 목록들과 이미 푼 문제에 대한 코멘트 현황이 나열되어 있다. 이후 목록에 존재하는 문제를 클릭하면 문제를 풀고, 제출할 수 있는 페이지로 연결되며, 피드백 목록을 클릭하면 ChatGPT와 강사로부터 받은 피드백 정보를 열람하는 페이지로 이동하게 된다.

강사 사용자가 Study 태그를 클릭하는 경우, 현재까지 제출된 학생의 코드 답안 목록들을 확인할 수 있는 페이지로 이동하게 된다. 또한 학생이 제출한 코드를 클릭하면 해당 코드에 대한 ChatGPT의 피드백을 생성하거나 수동으로 코드에 대한 답변을 입력할 수 있는 페이지로 이동할 수 있다.

4.3. Problem List Page

학생 사용자가 Study 태그를 클릭했을 때 이동하는 페이지이다. 학생 자신이 추천받은 문제의 리스트가 표시되며, 문제를 클릭하면 해당 문제를 열람하고, 코드 답안을 제출할 수 있는 페이지로 이동하게 된다. 또한 이미 답안을 제출한 문제는 강사로부터 코멘트를 받게 되는데, 새로운 코멘트를 생성되면 학생은 문제 리스트 페이지에서 알림을 받을 수 있다. 코멘트를 확인하는 버튼을 누르면 코멘트 페이지로 이동하게 된다.

4.3.1. Attributes

ProblemList: 학생에게 배정된 문제들을 나열해주는 리스트이다.

CommentAlarm: 학생이 제출한 문제에 대한 코멘트가 생성되었음을 알려주는 알림이다.

4.3.2. Methods

showProblems(): 학생이 배정받은 문제들을 표 형식으로 보여주며, 문제를 클릭하면 문제를 푸는 페이지로 이동하도록 해준다.

alertComment(): 답안을 제출한 문제에 대한 새로운 코멘트가 등록되면, 문제 리스트에 파란색 점으로 표시하여 학생에게 알려준다.

4.3.3. State Diagram

아래는 Problem Page에서 제공하는 기능을 보여주며, 문제 혹은 코멘트를 클릭했을 때의 상태를 그린 state diagram이다.

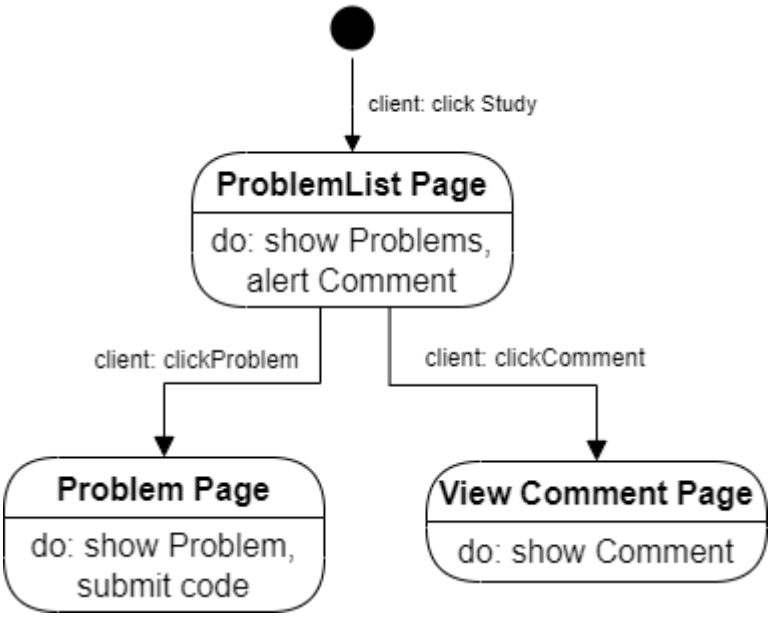

Figure 8. State Diagram of Problem List Page

Main Page에서 client가 Study 태그를 누르면 Problem List Page로 이동하게 된다. Problem List Page에서는 학생에게 배정된 문제를 리스트 형식으로 보여주며, 새로운 코멘트가 생성되면 문제에 표시를 해주어 학생에게 알려주는 기능을 제공한다.

학생이 문제 클릭하면, 해당 문제의 내용을 확인하고 코드를 제출할 수 있는 페이지인 Problem Page로 이동하게 된다. 또한 학생이 문제 리스트에서 코멘트를 클릭하는 경우, 강사가 등록해 놓은 코멘트를 열람할 수 있는 View Comment Page로 이동하게 된다.

4.4. Problem Page

학생 사용자가 Problem List Page에서 한 문제를 클릭하면 이동하는 페이지이다. 문제의 내용, 입력 및 출력 케이스 등 문제에 대한 상세한 설명을 보여준다. 또한 학생이 작성한 코드를 입력하고, 제출할 수 있는 섹션이 존재한다.

4.4.1. Attributes

Description: 문제 상황에 대한 설명이다.

TestInput: 작성한 코드를 테스트할 때 사용할 수 있는 테스트 입력값이다.

TestOutput: 작성한 코드를 테스트할 때 사용할 수 있는 테스트 출력값이다.

SubmitCode: 제출할 코드를 입력하는 섹션이다.

SubmitButton: 작성한 코드 답안을 제출하는 버튼이다.

4.4.2. Methods

showDescription(): 문제의 설명과 제한사항 등을 표시해준다.

showTestInput(): 코드의 테스트에 사용하는 입력값을 표시해준다.

showTestOutput(): 코드의 테스트에 사용하는 출력값을 표시해준다.

getAnswer(): 학생이 작성한 답안 코드를 입력받는다.

submit(): 학생이 작성한 답안 코드를 최종적으로 제출한다.

4.4.3. State Diagram

다음은 사용자가 코드를 작성하고 제출 버튼을 눌렀을 때의 Problem Page의 상태를 그린 state diagram이다

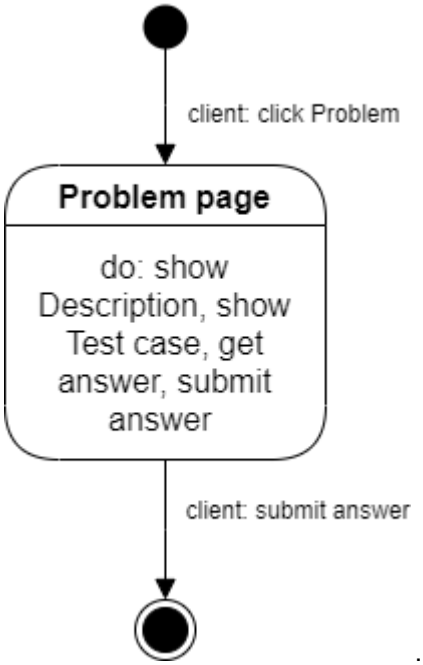

Figure 9. State Diagram of Problem Page

학생 사용자가 문제에 대한 답안을 작성하여 Problem Page에 입력한 후, 제출 버튼을 누르면 서버에 학생의 코드를 등록한 후 문제 풀이가 종료되게 된다.

4.5. View Comment Page

학생 사용자가 코드를 제출하면 강사가 이에 대한 최종적인 코멘트를 서버에 등록한다. 코멘트가 생성되면 Problem List Page에 알림이 띄워지며, 코멘트를 클릭하면 View Comment Page로 이동하게 된다. View Comment Page에서는 강사의 최종 코멘트를 확인할 수 있으며, 학생의 코딩 역량을 다각형 형태의 차트로 가시화하여 함께 보여준다.

4.5.1. Attributes

showSubmit: 학생이 제출한 코드를 보여준다.

showComment: 강사가 및 ChatGPT가 작성한 최종 코멘트를 보여준다.

showAchievement: 학생의 코딩 역량을 가시화하여 보여준다.

4.5.2. Methods

showSubmit(): 학생이 제출한 코드를 코드 블락으로 보여준다.

showComment(): 서버에 등록된 문제 피드백을 보여준다.

showAchievement(): 학생의 답안을 분석하여 도출한 코딩 역량을 가시화하여 보여준다.

4.5.3. State Diagram

다음은 학생 사용자가 특정 피드백을 클릭하여 View Comment Page에 방문했을 때의 상태를 그린 state diagram이다.

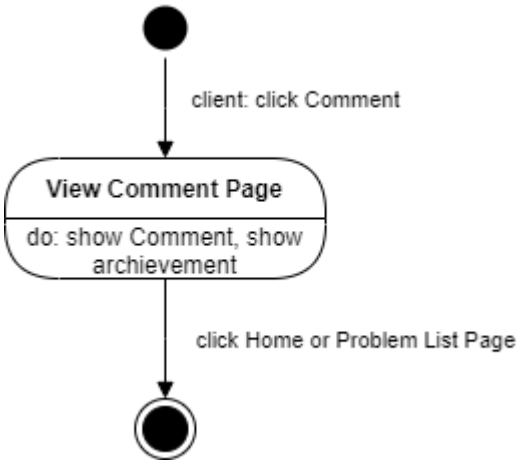

Figure 10. State Diagram of View Comment Page

학생 사용자가 피드백을 열람하는 페이지이다. 강사가 등록한 피드백을 확인할 수 있으며, Home 버튼 혹은 Problem List로 돌아가는 버튼을 클릭하여 페이지에서 나갈 수 있다.

4.6. Submission List Page

Codemy의 강사 페이지로, 최근 학생들이 제출한 코드를 순서대로 확인할 수 있는 페이지이다. 리스트 요소를 클릭하면 피드백을 이미 한 제출의 경우, 피드백을 수정할 수 있는 페이지로 이동하고, 피드백을 아직 하지 않은 제출인 경우 ChatGPT의 결과값을 받아와 피드백을 작성할 수 있는 페이지로 이동한다.

4.6.1. Attributes

- Home : 홈 버튼이다.
- Summary: 학생 전반에 대한 분석을 시각화하여 보여주는 페이지로 이동할 수 있는 버튼이다.
- StudentInfo: 학생 리스트를 확인할 수 있고 각 학생에 대한 조회가 가능한 페이지로 이동할 수 있는 버튼이다.
- SubmittedCodeList: 학생들이 제출한 코드들을 나열해주는 리스트이다.

4.6.2. Methods

clickHome() : 사용자가 Home 버튼을 눌렀을 때 초기 상태의 메인 페이지로 돌아간다.

clickSummary(): 학생 전반에 대한 분석을 시각화하여 보여주는 페이지로 이동한다.

clickStudentInfo(): 학생 리스트를 확인하는 페이지로 이동한다.

showSubmittedCode(): 학생들이 제출한 코드를 리스트 형식으로 보여주며, 코드를 클릭하면 피드백을 작성 또는 수정하는 페이지로 이동한다.

4.6.3. State Diagram

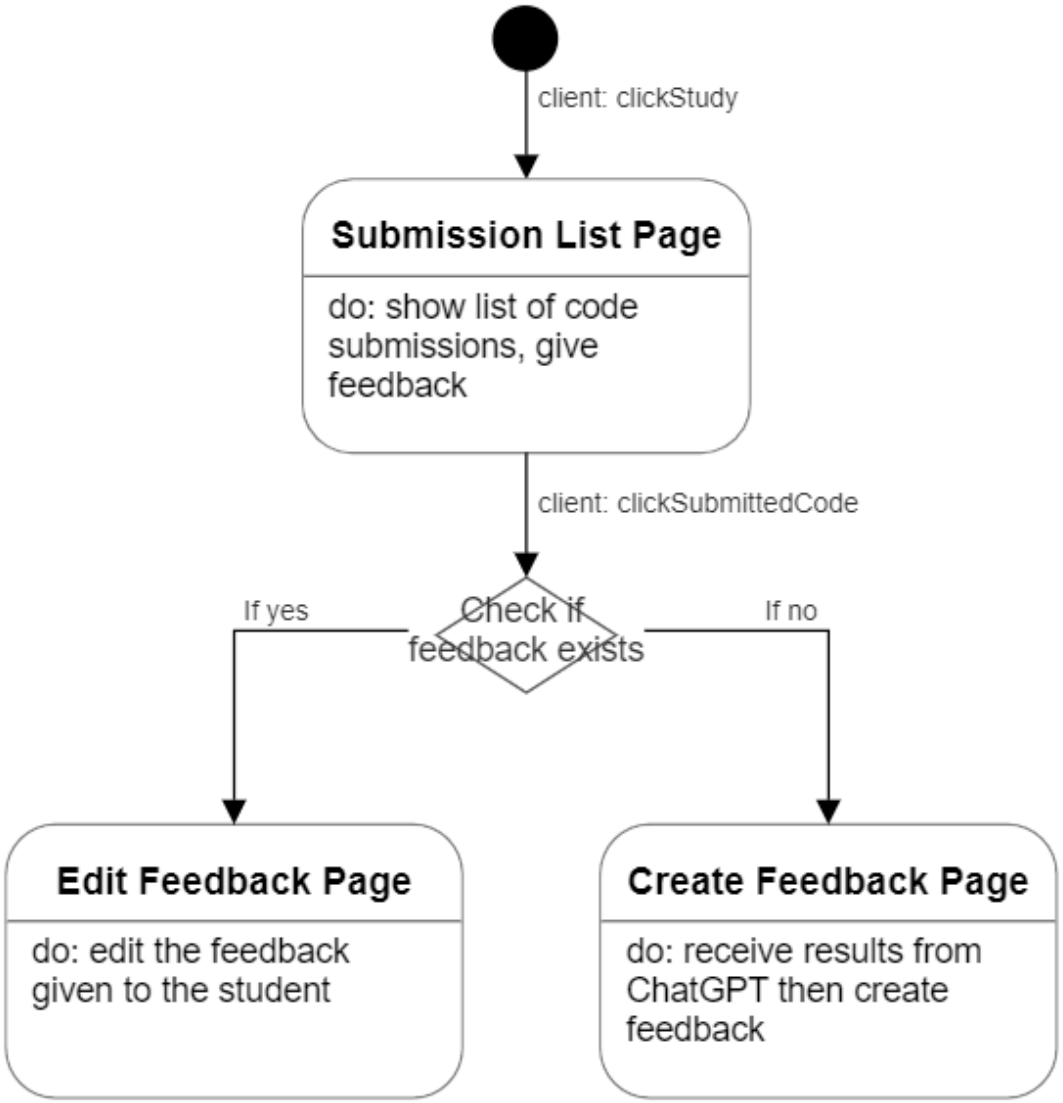

Figure 11. State Diagram of Submission List Page

Client가 강사일 경우, Main Page에서 Study 태그를 누르면 Submission List Page로 이동하게 된다. Submission List Page에서는 학생들이 제출한 코드를 리스트 형식으로 보여준다.

강사가 코드를 클릭하면, 해당 코드에 피드백이 존재하는지 여부를 확인하고 피드백이 존재하는 경우 Edit Feedback Page로 이동하게 된다. 또한, 피드백이 존재하지 않는 경우, ChatGPT의 결과값을 받아와 피드백을 작성할 수 있는 Create Feedback Page로 이동하게 된다.

4.7. Create Feedback Page

Codemy의 강사 페이지로, ChatGPT의 결과값을 받아와 피드백을 작성할 수 있는 페이지이다. 학생의 이름, 문제 번호, 제출된 코드의 메모리, 시간, 사용한 언어 그리고 학생이 제출한 코드가 표시된다. 또한, ChatGPT에서 출력한 피드백을 확인할 수 있으며 ChatGPT가 출력한 1차 피드백을 검토한 후 추가적인 피드백을 남길 수 있다. 이를 통해 학생의 코드를 시각화 한 차트도 확인할 수 있다.

4.7.1. Attributes

- Home : 홈 버튼이다.
- Summary: 학생 전반에 대한 분석을 시각화하여 보여주는 페이지로 이동할 수 있는 버튼이다.
- SubmissionList: 학생들이 제출한 코드를 리스트로 나열하여 순서대로 보여주는 페이지로 이동할 수 있는 버튼이다.
- StudentInfo: 학생 리스트를 확인할 수 있고 각 학생에 대한 조회가 가능한 페이지로 이동할 수 있는 버튼이다.
- StudentName: 코드를 제출한 학생의 이름이다.
- QuestionNumber: 학생이 제출하는 코드에 해당하는 문제 번호이다.
- CodeMemory: 제출된 코드의 메모리이다.
- CodeTime: 제출된 코드를 실행하는 데 걸린 시간이다.
- CodeLanguage: 학생이 제출한 코드에 사용된 언어이다.
- CodeSubmitted: 학생이 제출한 코드이다.
- ChatGptFeedback: ChatGPT가 학생의 코드를 분석한 후 남긴 피드백이다.
- ChatGptChart: 학생의 코드를 시각화 한 차트이다.
- Feedback: 피드백 작성란이다.

4.7.2. Methods

clickHome() : 사용자가 Home 버튼을 눌렀을 때 초기 상태의 메인 페이지로 돌아간다.

clickSummary(): 학생 전반에 대한 분석을 시각화하여 보여주는 페이지로 이동한다.

clickSubmissionList(): 학생들이 제출한 코드를 리스트로 나열하여 순서대로 보여주는 페이지로 이동한다.

clickStudentInfo(): 학생 리스트를 확인하는 페이지로 이동한다.

addFeedback(): 피드백을 작성한 후 학생에게 전달한다.

4.7.3. State Diagram

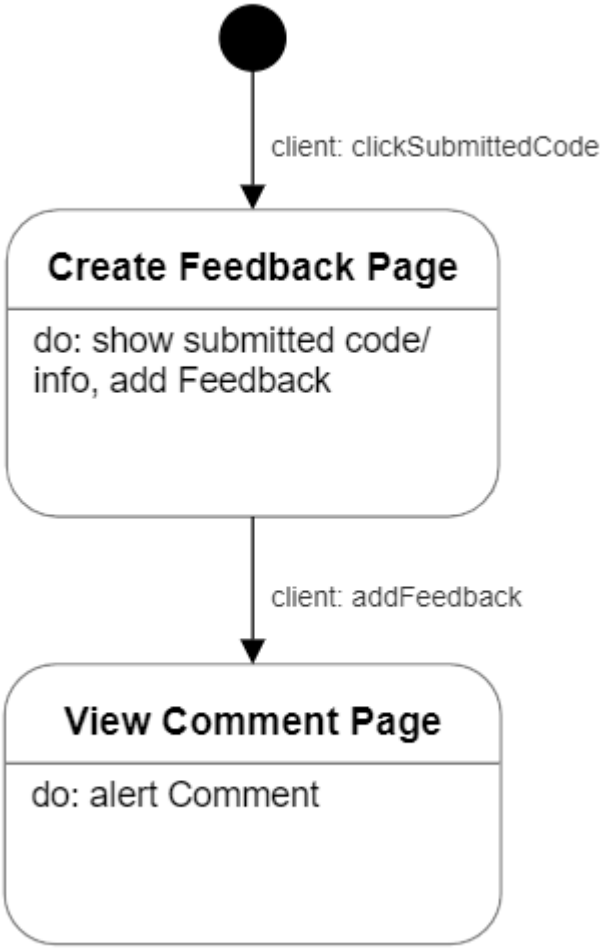

Figure 12. State Diagram of Create Feedback Page

강사가 제출된 코드를 클릭하면 해당 코드를 제출한 학생, 문제 번호, 제출된 코드의 메모리, 시간, 사용한 언어 그리고 학생이 제출한 코드가 표시된다. 또한 ChatGPT가 출력한 1차 피드백과 제출된 코드를 시각화 한 차트도 확인할 수 있다.

강사는 피드백 작성란에 피드백을 작성할 수 있고 완료하면 학생의 View Comment Page로 등록이 되며 학생은 코멘트를 열람할 수 있다.

5. System Architecture – Backend

5.1. Objectives

이 장에서는 백엔드 시스템의 구조를 시각화하여 제시하고, 시스템을 구성하는 각 요소들의 관계를 기술한다. 사용자의 이해를 돕기 위해 자연어, 그림, 도표 등을 활용한다.

5.2. Overall Architecture

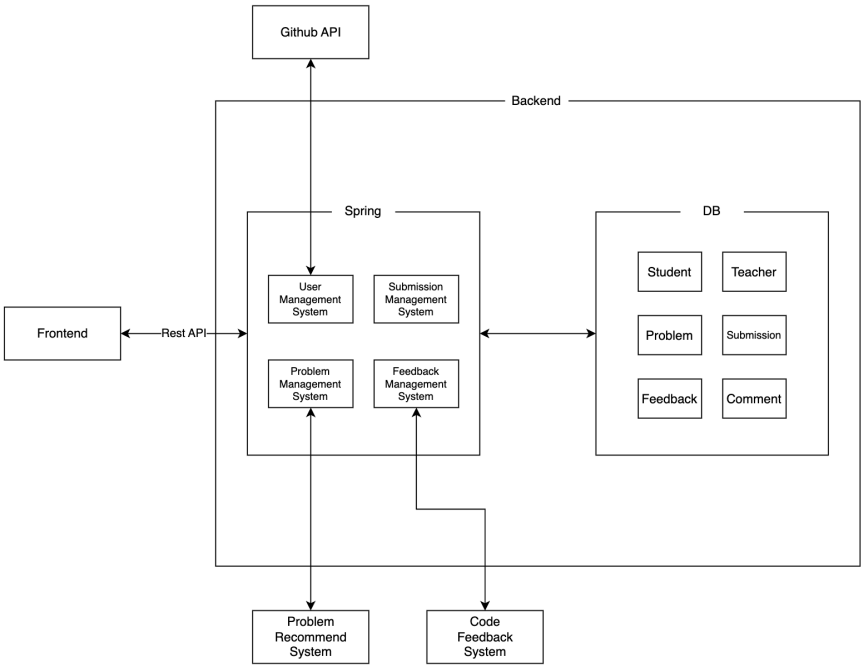

Figure 13. Overall Backend Architecture

Backend의 전체 구조는 다음과 같다. Frontend 서버에서 API를 호출하면 Spring framework를 이용해서 sub-system이 호출된다. Back-end 서버는 학생과 강사에 대한 정보와 회원가입 및 로그인 등의 기능을 담당하는 User Management System, 문제와 답안에 대한 채점을 담당하는 Problem Management System, 학생이 제출한 코드와 강사의 코멘트를 다루는 Submission Management System, 학생의 코드에 대한 ChatGPT의 피드백을 담당하는 Feedback Management System으로 총 4개의 sub-system으로 구성된다.

5.3. Subcomponents

5.3.1. User Management System

학생과 강사의 가입 및 로그인, 조회 등의 기능을 수행하는 시스템이다.

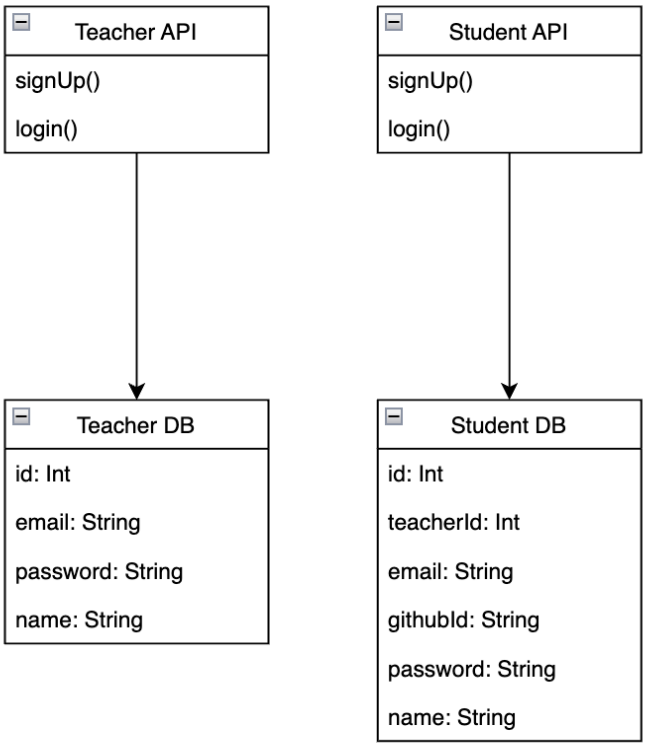

Figure 14. User Management System Class Diagram

- Teacher API Class

Frontend에서 signUp() API를 호출하면 함께 입력받은 email과 password, name을 이용해서 회원가입을 할 수 있다. 만약 해당 이메일이 이미 존재한다면 요청에 실패한다. 가입에 성공했다면 login() API를 통해서 인증 토큰을 발급받을 수 있다.

- Student API Class

Frontend에서 signUp() API를 호출하면서 함께 email, password, name, teacherId, githubId를 보내면 teacher와 연결시켜준다. 이메일이 이미 존재하거나 teacherId에 해당하는 강사가 존재하지 않는다면 가입에 실패한다. 가입에 성공했다면 login() API를 통해서 인증 토큰을 발급받을 수 있다.

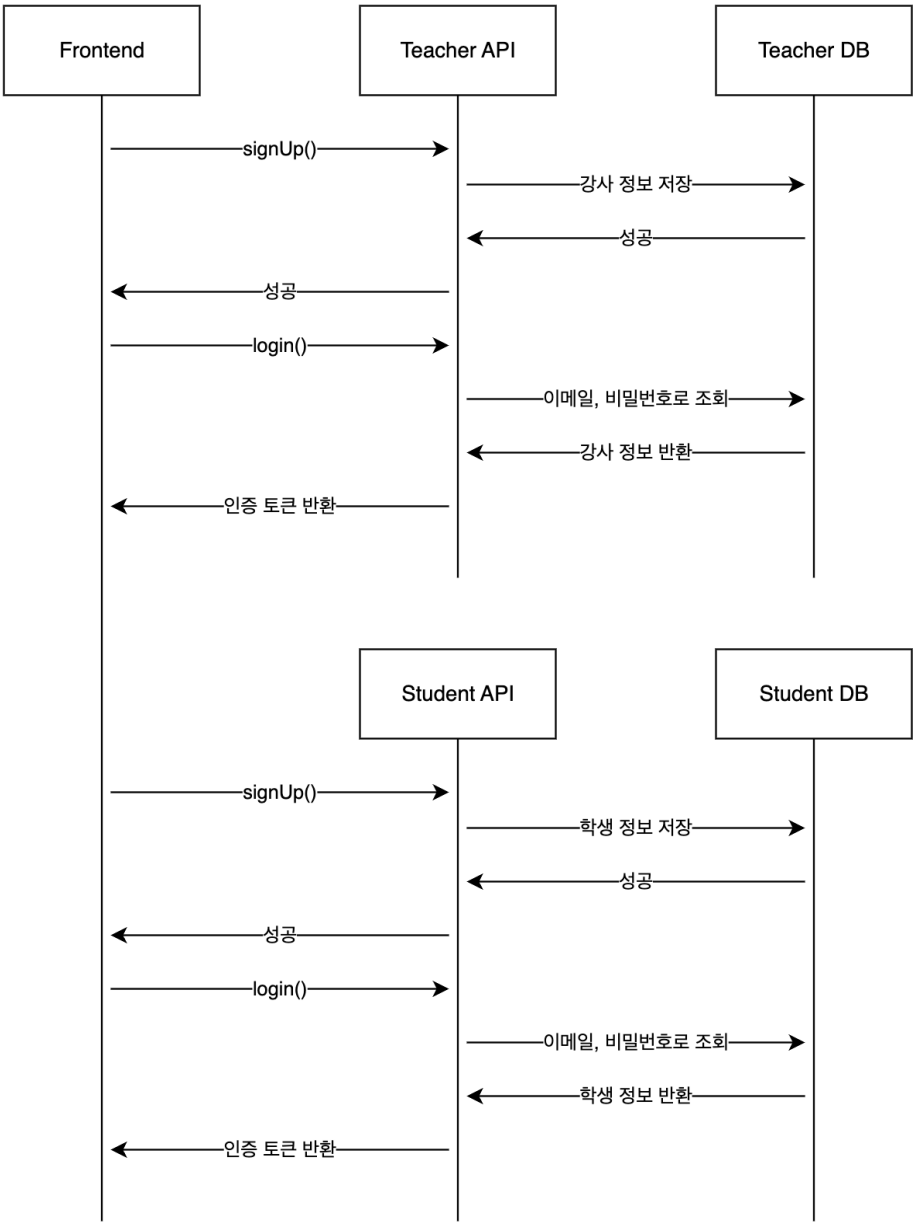

Figure 15. User Management System Sequence Diagram

5.3.2. Problem Management System

학생에게 맞춘 추천 문제를 제공해주는 시스템이다.

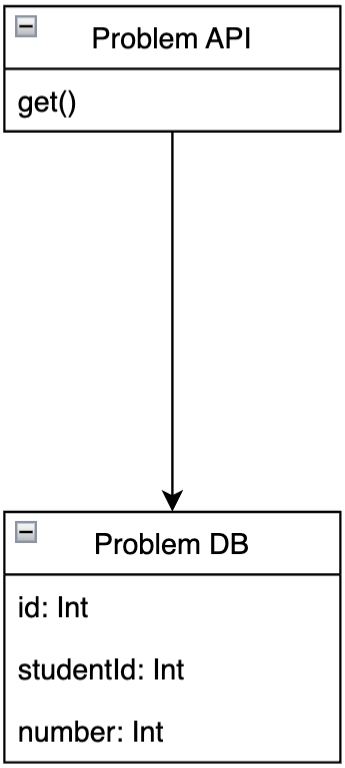

Figure 16. Problem Management System Class Diagram

- Problem API

문제의 id를 이용해서 문제에 대한 정보를 제공하는 get() API와 학생의 답안을 입력받아 submission API를 제공한다. get() API에서는 사용자에게 맞는 추천 문제를 제공하는데, 추천 문제가 없는 경우에는 새로운 추천 문제 5문제를 추가로 제공한다.

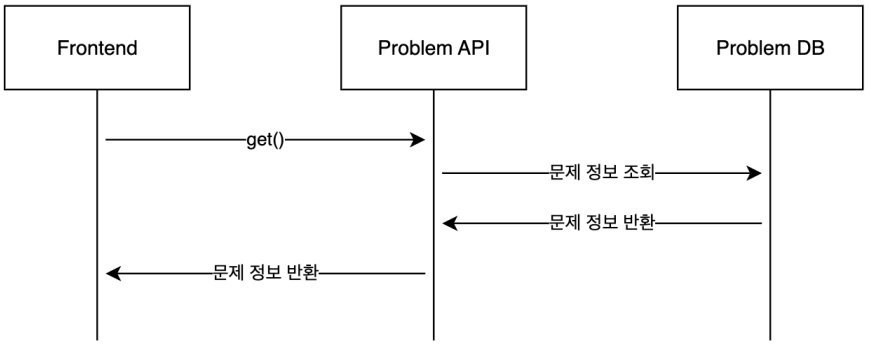

Figure 17. Problem Management System Sequence Diagram

5.3.3. Submission Management System

학생이 문제에 대한 답안 코드를 제출하고, 강사가 학생의 코드와 ChatGPT의 피드백을 이용해서 코멘트를 작성하는 과정을 담당하는 시스템이다.

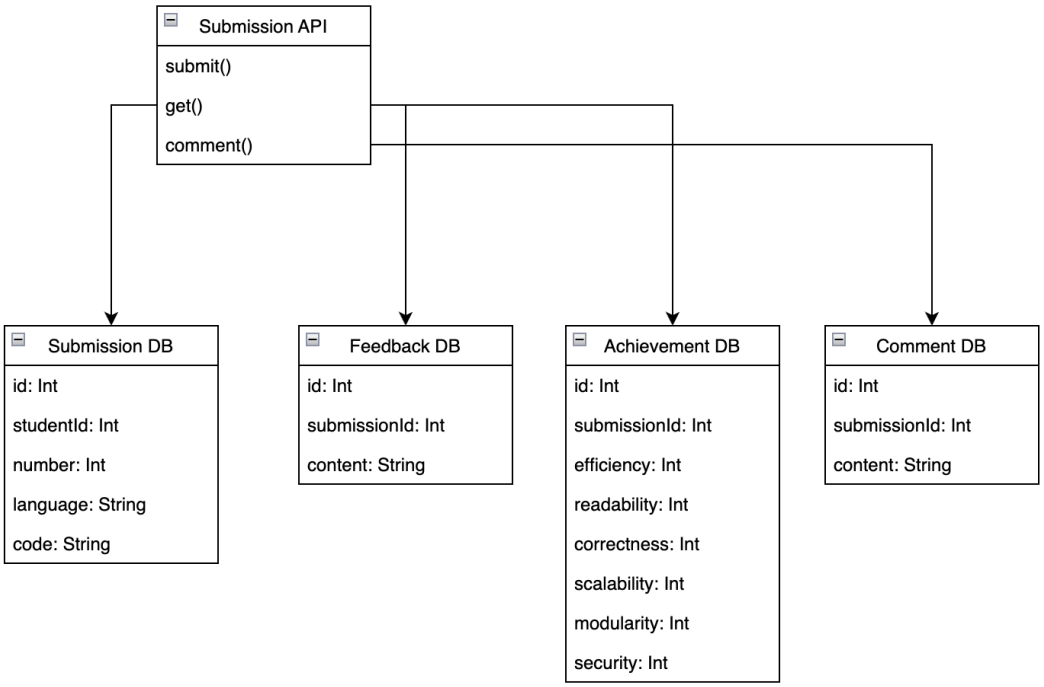

Figure 18. Submission Management System Class Diagram

- Submission API

submit() API는 frontend로부터 problem에 대한 정보와 제출 언어, 코드를 submission DB에 저장하는 API이다. get() 메서드는 학생이 제출한 submission과 feedback management system에서 만든 feedback을 반환해주는 메서드이다. comment() 메서드는 학생의 submission과 feedback을 바탕으로 강사가 작성한 comment를 제공하는 메서드이다. Comment가 추가되면 학생은 자신이 푼 문제에서 강사의 comment를 함께 확인할 수 있다.

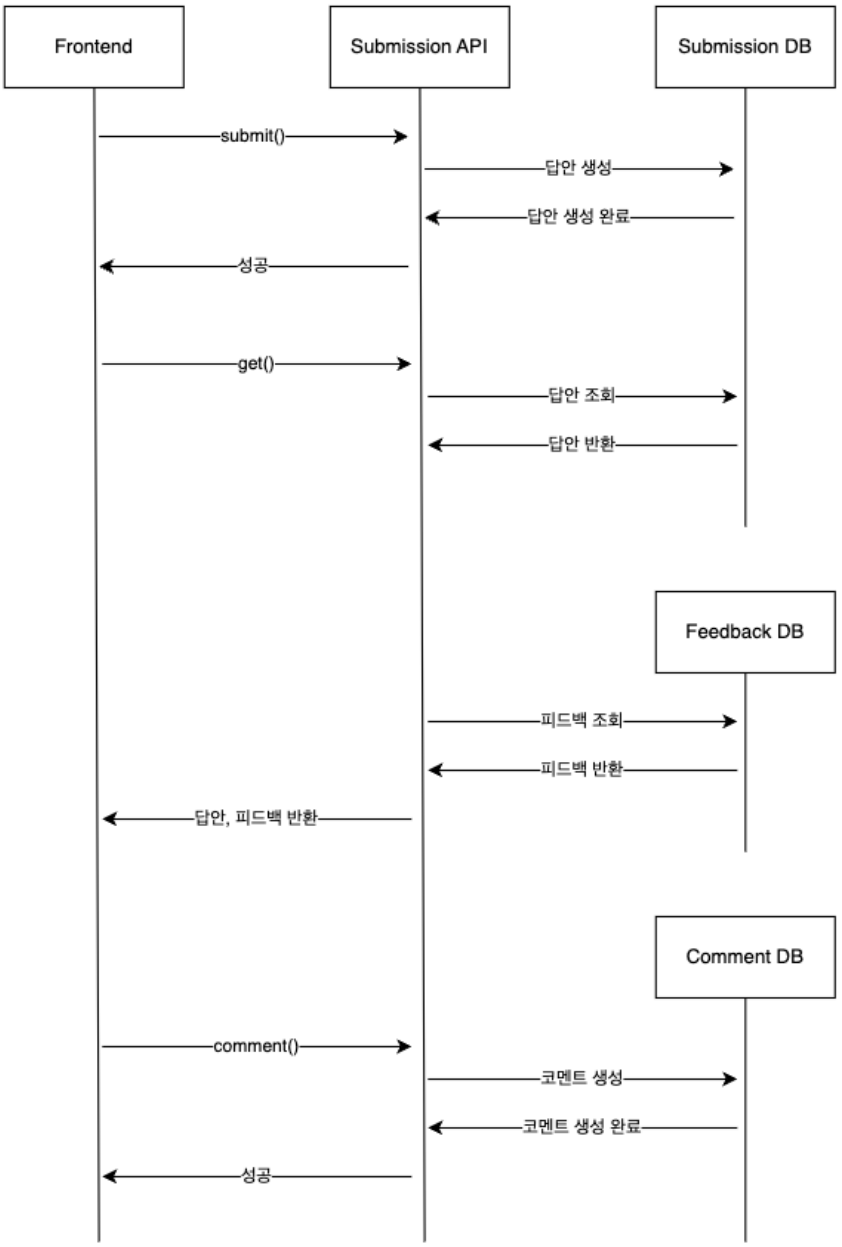

Figure 19. Grade Management System Sequence Diagram

5.3.4. Feedback Management System

학생이 제출한 답안을 이용해서 자동으로 피드백을 생성해주는 시스템이다.

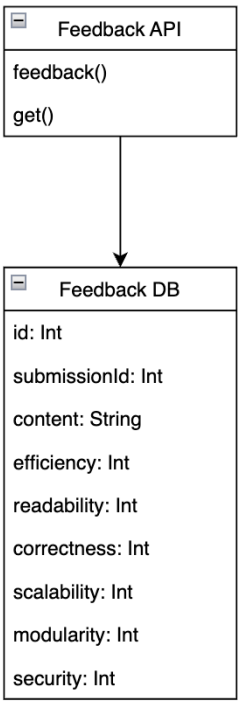

Figure 20. Feedback Management System Class Diagram

- Feedback API Class

get() 메서드는 학생의 답안에 대한 피드백을 반환하는 API이다. 피드백은 학생이 답안을 제출하는 시점에 만들어진다. 피드백에는 요약된 문장으로 제공되는 content와 efficiency, readability, correctness, scalability, modularity, security라는 6개의 관점에서의 점수로 이루어져 있다. 강사는 ChatGPT의 피드백을 참고해서 학생의 답안에 대해 comment를 작성한다.

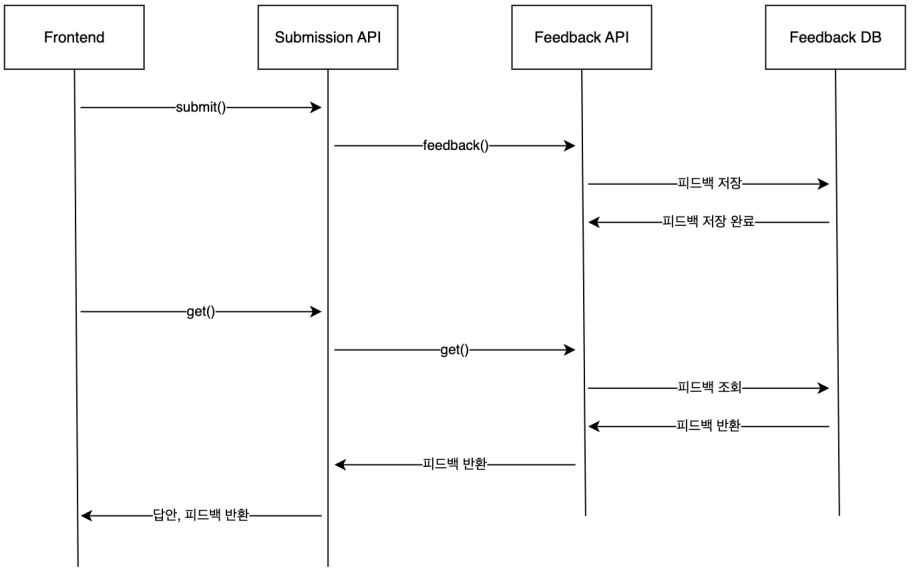

Figure 21. Feedback Management System Sequence Diagram

6. Protocol

6.1. Objectives

이 장에서는 서브 시스템 간의 상호작용에 필수적으로 준수해야 하는 프로토콜에 관해 설명하고, 통신 과정에서 전달되는 메시지의 형식과 용도, 의미를 설명한다.

6.2. HTTP

클라이언트-서버 프로토콜이며, 각각의 개별적인 요청은 서버로 보내지고, 서버는 요청을 처리한 후에 response라고 불리는 응답을 제공한다. 웹에서 이루어지는 모든 데이터 교환의 기초로, HTML 문서와 같은 리소스들을 가져올 수 있도록 하는 프로토콜이다.

6.3. JSON

객체 문법으로 구조화된 데이터를 표현하기 위한 문자 기반의 표준 포맷이다. 속성-값 쌍, 배열 자료형 등 시리얼화 가능한 값 또는 "키-값 쌍"으로 이루어진 데이터 오브젝트를 전달하기 위해 인간이 읽을 수 있는 텍스트를 사용한다. 특히, 인터넷에서 자료를 주고받을 때 그 자료를 표현하는 방법으로 알려져 있다.

6.4. Protocol Description

6.4.1. 학생 회원가입 프로토콜

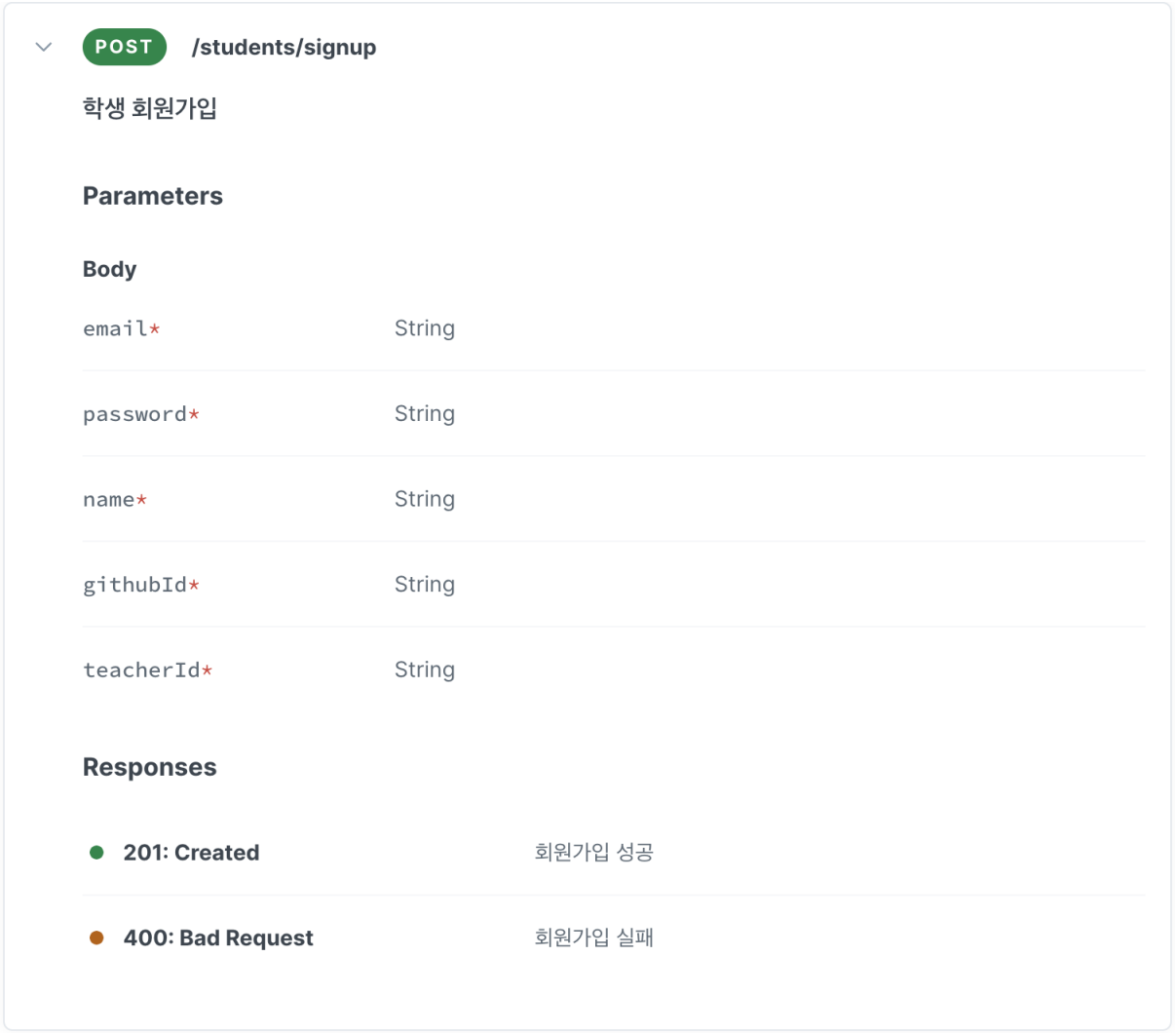

Figure 22. 학생 회원가입 프로토콜

6.4.2. 강사 회원가입 프로토콜

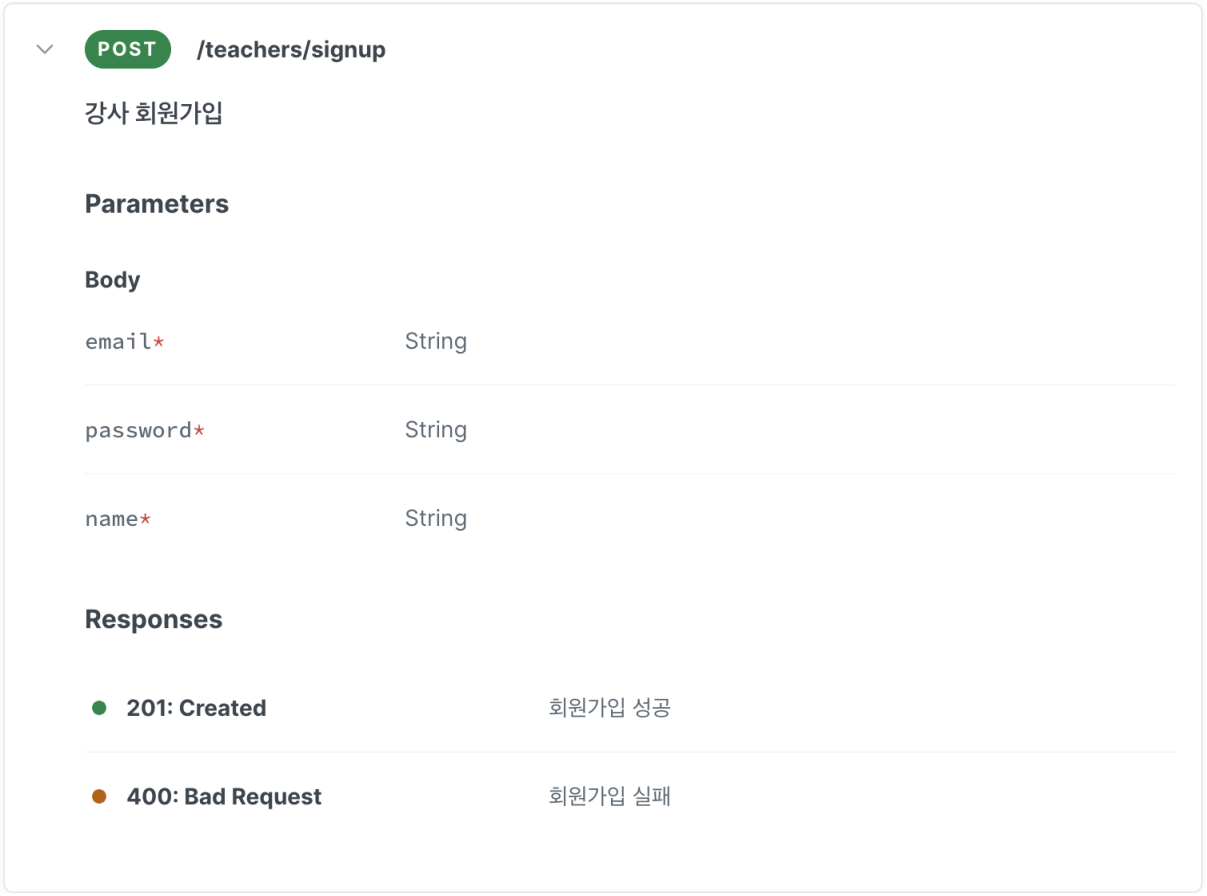

Figure 23. 강사 회원가입 프로토콜

6.4.3. 로그인 프로토콜

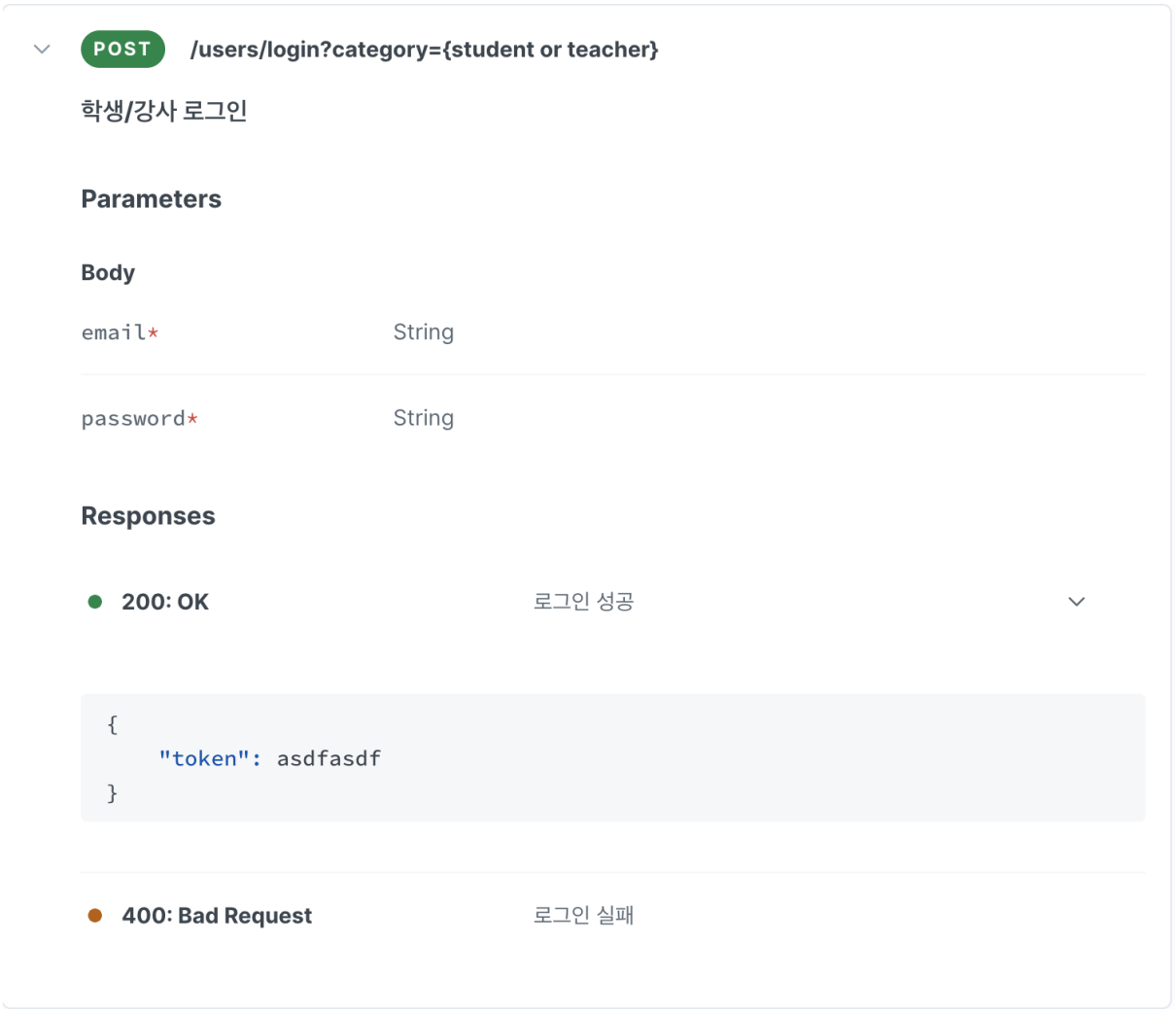

Figure 24. 로그인 프로토콜

6.4.4. 문제 확인 프로토콜

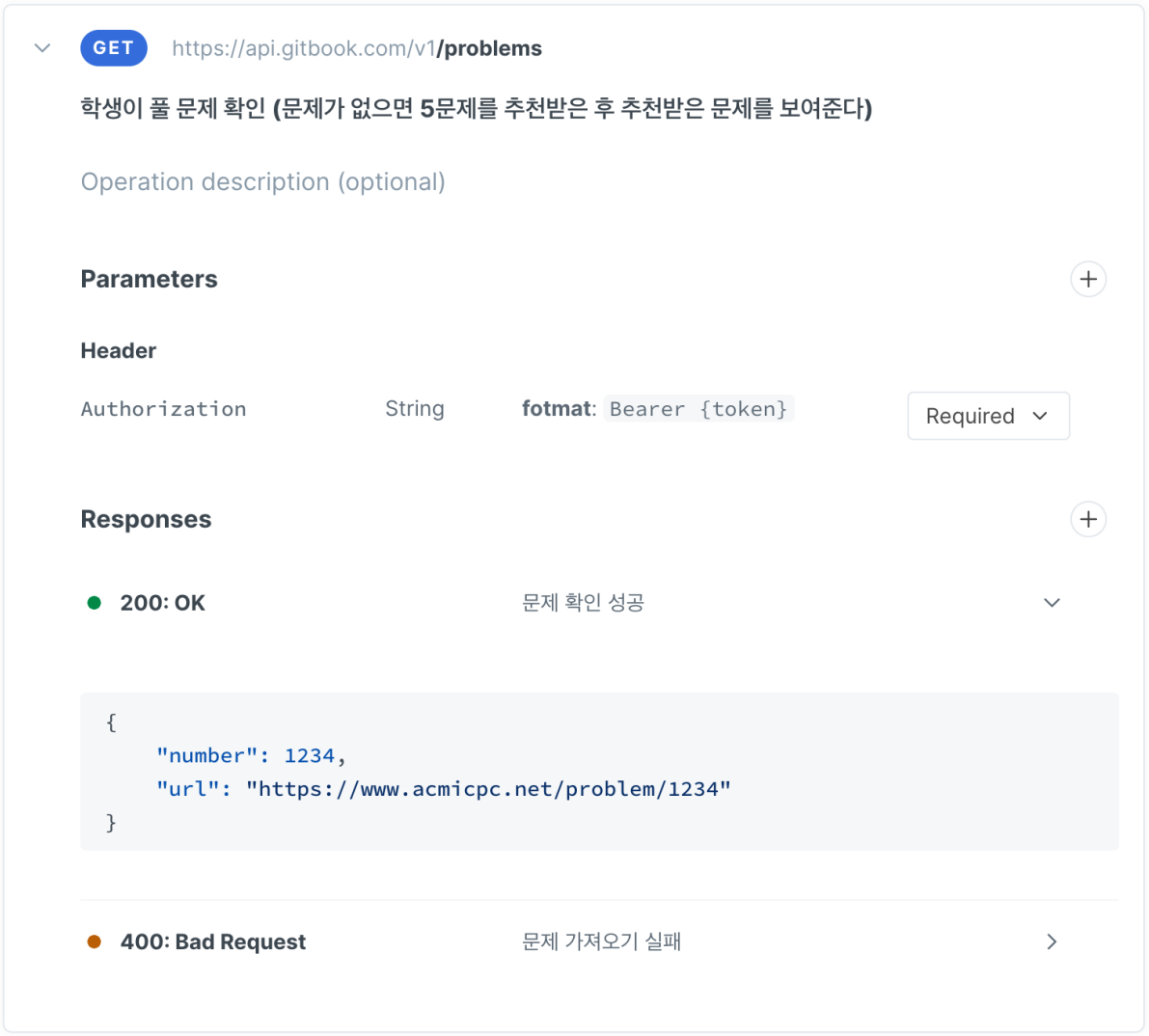

Figure 25. 문제 확인 프로토콜

6.4.5. 코드 제출 프로토콜

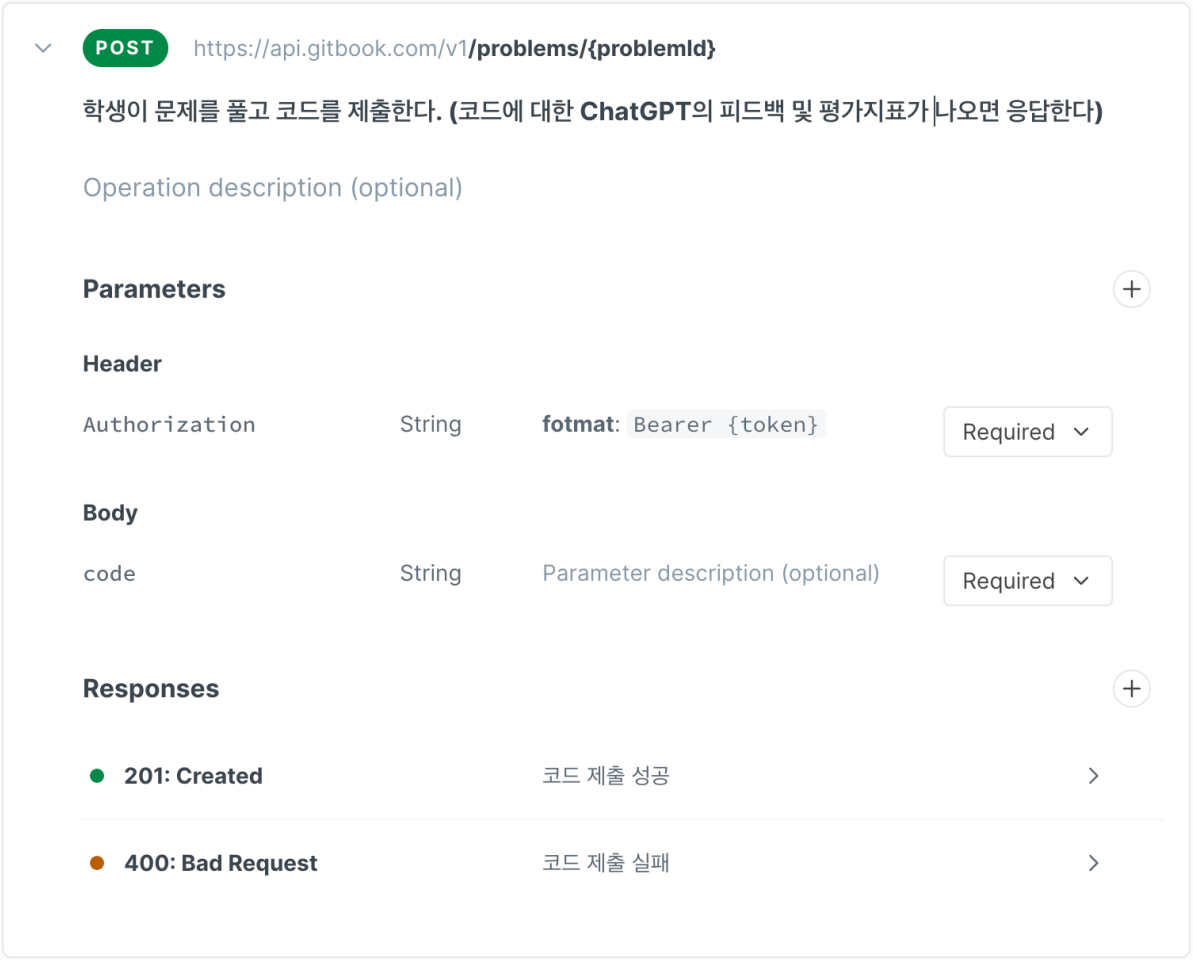

Figure 26. 코드 제출 프로토콜

6.4.6. 피드백 확인 프로토콜 (학생)

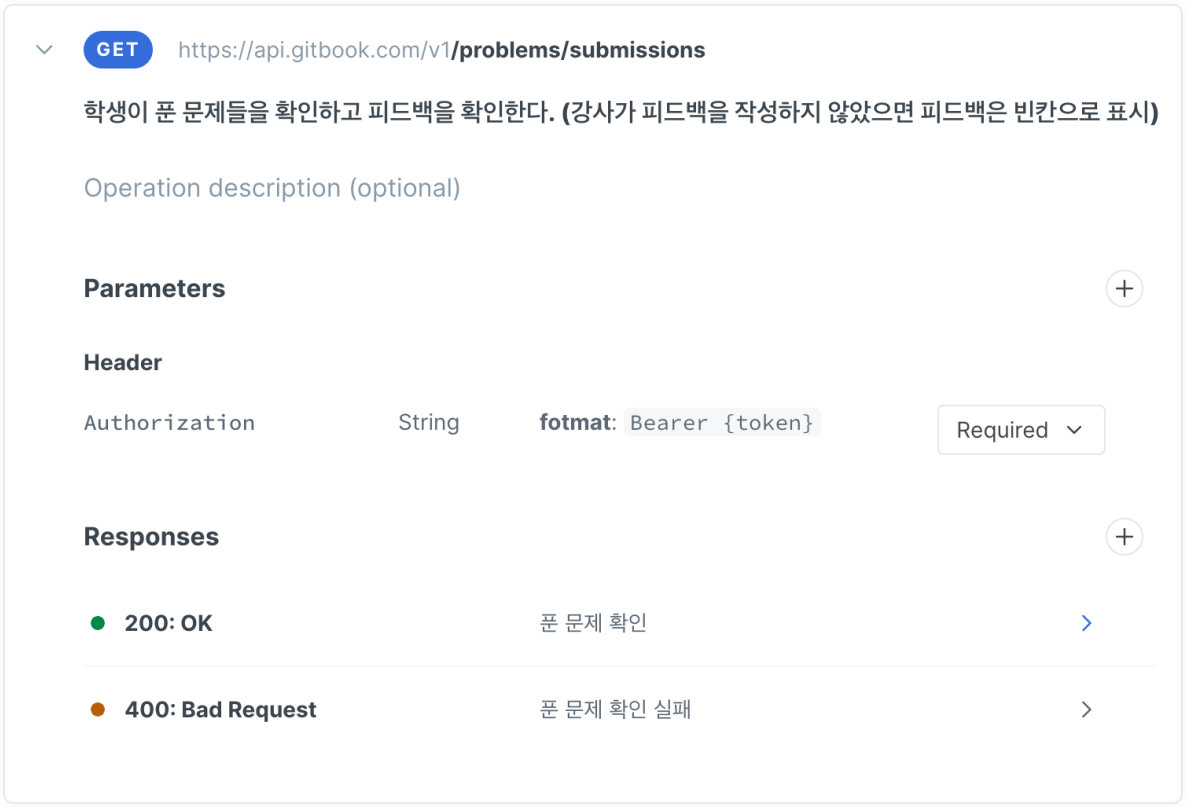

Figure 27. 피드백 확인 프로토콜 (학생)

6.4.7. 피드백 확인 프로토콜 (강사)

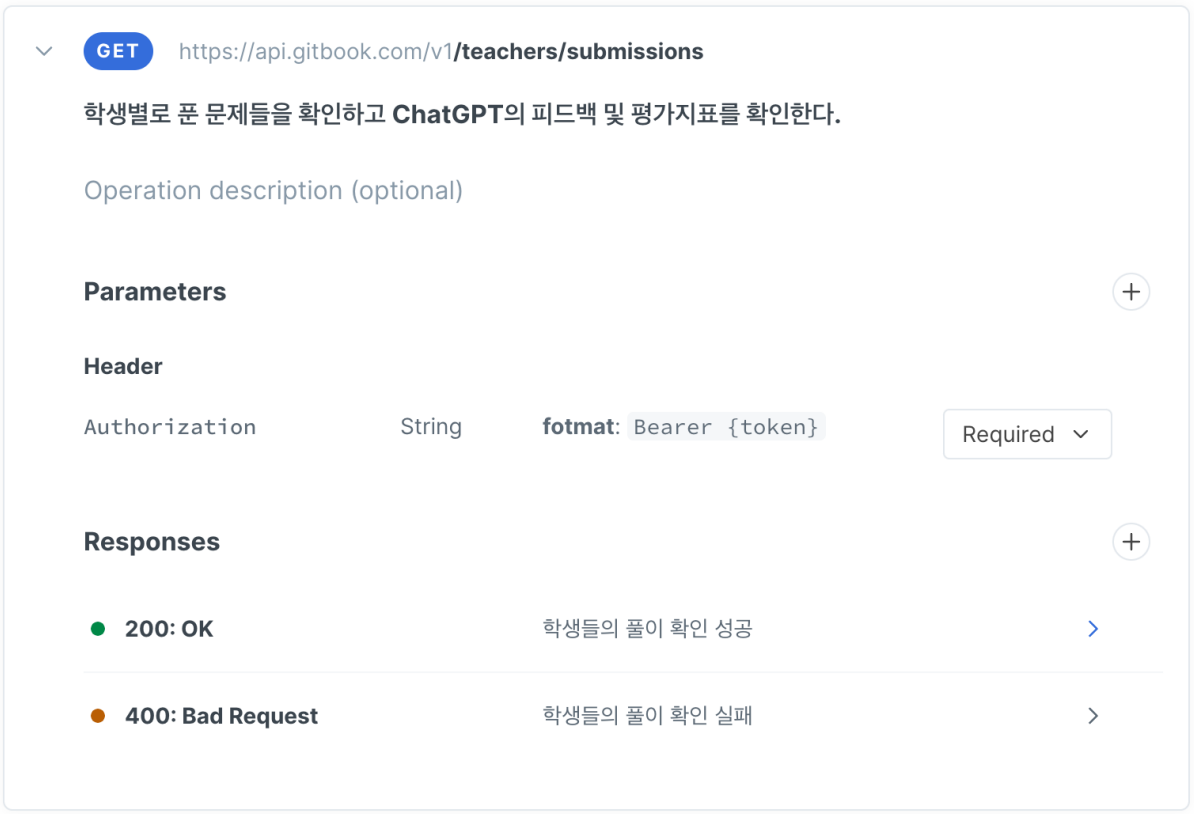

Figure 28. 피드백 확인 프로토콜 (강사)

6.4.8. 코멘트 작성 프로토콜 (강사)

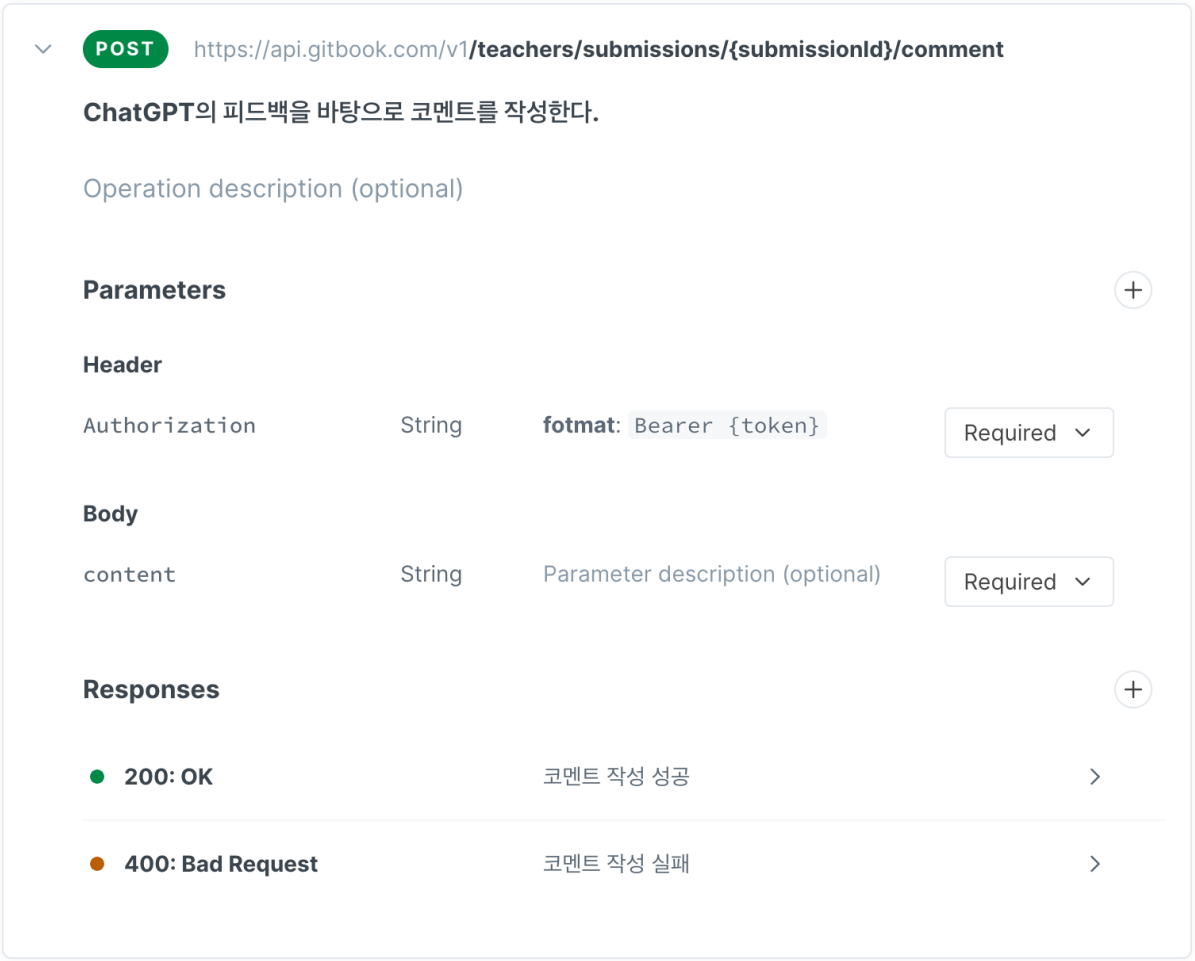

Figure 29. 코멘트 작성 프로토콜 (강사)

7. database Design

7.1. Objectives

Database design의 전반적인 부분에 대해 설명한다. ER diagram을 통해 Entity와 Relationship을 설명하고, 각각의 Entity의 attribute에 대해 설명한다.

7.2. ER Diagram

해당 시스템에는 Student, Teacher, Solved, Problem, Submission, Feedback, Analysis, Comment 총 8개의 Entity가 존재한다. ER diagram에서 각 Entity의 attribute를 표현하였다. 이중 파란색 attribute는 primary key, 검은색 attribute는 foreign key로 표현하였다.

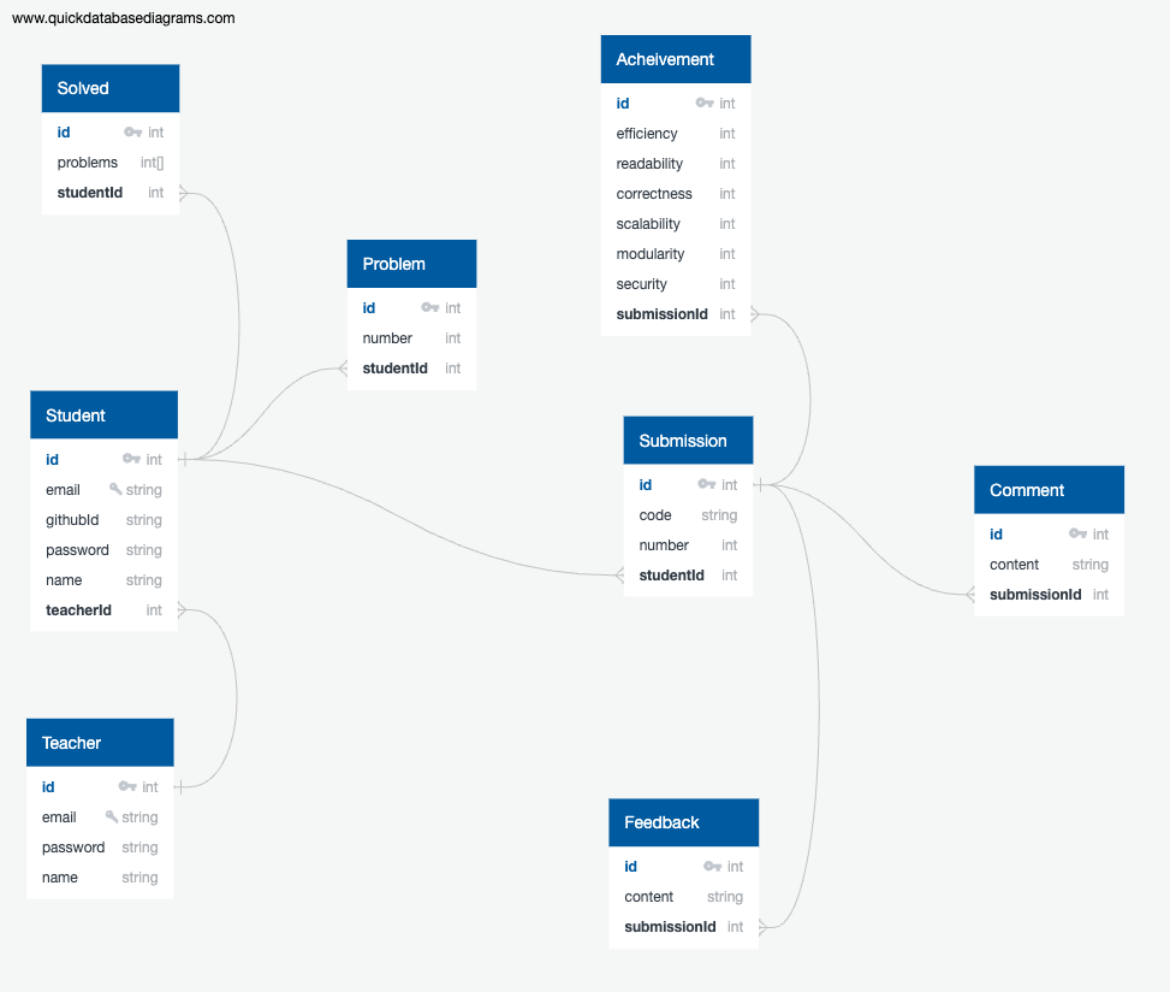

Figure 30. ER Diagram

7.3. Entity

Student Entity는 학생의 집합을 의미한다.

Teacher Entity는 강사의 집합을 의미한다.

Solved Entity는 학생이 지금까지 풀었던 문제의 집합을 의미한다. 처음 학생이 회원가입할 때 github id를 통해 어떤 문제를 풀었는지 확인하여 데이터를 추가한다. 또한, 추천받은 문제를 학생이 풀면 Solved Entity에 데이터가 추가된다.

Problem Entity는 학생이 추천받은 문제의 집합을 의미한다. 학생이 문제를 풀려고 할 때 추천받은 문제가 없으면 Problem Entity에 데이터가 추가된다.

Submission Entity는 학생이 제출한 문제의 집합을 의미한다. 학생이 문제를 풀고 코드를 작성하면 Submission Entity에 데이터가 추가된다.

Feedback Entity는 학생이 제출한 코드에 대한 ChatGPT의 피드백의 집합을 의미한다. 학생이 문제를 풀고 코드를 작성하면 코드에 대한 피드백 데이터가 Feedback Entity에 추가된다.

Achievement Entity는 학생이 제출한 문제에 대한 6가지 평가지표의 집합을 의미한다. 학생이 문제를 풀고 코드를 작성하면 코드에 대한 6가지 평가지표 데이터가 Analysis Entity에 추가된다.

Comment Entity는 학생이 제출한 문제에 대한 강사의 피드백 집합을 의미한다. 학생이 문제를 풀고 코드를 작성하면 해당 코드에 대해 ChatGPT의 피드백을 받을 수 있다. 이러한 피드백을 바탕으로 강사는 자신의 의견을 덧붙여 코드에 대한 피드백을 작성하면 강사의 피드백 데이터가 Comment Entity에 추가된다.