



온라인 저지 플랫폼

소프트웨어 요구사항 명세서

2022.10.25.

Introduction to Software Engineering 41

TEAM 10

팀장 최시열

팀원 김도환

김준석

박주현

윤민철

이수민

목차

1.	Introduction	6
1.1.	Purpose	6
1.2.	Scope	7
1.3.	Definitions, Acronyms, and Abbreviation	7
1.4.	References	8
1.5.	Overview	9
2.	Overall Description	9
2.1.	Product Perspective	9
2.1.1.	System Interfaces	9
2.1.2.	User Interfaces	10
2.1.3.	Hardware Interfaces	10
2.1.4.	Software Interfaces	11
2.1.5.	Communications Interfaces	11
2.1.6.	Memory Constraints	11
2.1.7.	Operations	12
2.1.7.1.	System administrator	12
2.1.7.2.	User	12
2.2.	Product Functions	13
2.2.1.	Login & Sign-up	13
2.2.2.	Admin Page Access	13
2.2.3.	Register Students	14
2.2.4.	Select Lecture Course	14
2.2.5.	Select Lecture Practice	14
2.2.6.	Save User Code	14
2.2.7.	Verifying User Code	14
2.2.8.	Submitting User Code	15
2.3.	User Characteristics	15
2.3.1.	System Administrator	15
2.3.2.	User	16

2.4. Constraints	16
2.5. Assumptions and Dependencies	16
3. Specific Requirements	17
3.1. External Interface Requirements	17
3.1.1. User Interfaces	17
3.1.2. Hardware Interfaces	31
3.1.3. Software Interfaces	32
3.1.4. Communication Interfaces	33
3.2. Functional Requirements	37
3.2.1. Use Case	37
3.2.2. Use Case Diagram	41
3.2.3. Data Dictionary	41
3.2.3. Data Flow Diagram	43
3.3. Performance Requirements	44
3.3.1. Static numerical requirement	44
3.3.2. Dynamic numerical requirement	44
3.4. Logical Database Requirements	45
3.5. Design Constraints	45
3.6. Standards Compliance	45
3.7. Software System Characteristics	45
3.7.1. Product Requirements	46
3.7.1.1. Usability Requirements	46
3.7.1.2. Performance Requirements	46
3.7.1.3. Security Requirements	46
3.7.2. Organizational Requirements	47
3.7.2.1. Environmental Requirements	47
3.7.2.2. Operational Requirement	47
3.7.3. External Requirements	47
3.7.3.1. Safety / Security Requirement	47
3.7.3.2. Regulatory Requirement	47
3.8. Organizing the Specific Requirements	48
3.8.1. Context Model	48

3.8.2.	Process Model	48
3.8.3.	Interaction Model	49
3.8.4.	Behavior Model	49
3.8.4.1.	Data Flow Diagram	49
3.8.4.2.	Sequence Diagram	49
3.9.	System Architecture	49
3.10.	System Evolution	50
3.10.1.	Limitation and Assumption	50
3.10.2.	Evolutions of Hardware and Change of User Requirements	50
4.	Supporting Information	51
4.1.	Software Requirement Specification	51
4.2.	Document History	51

그림 목차

[그림 1] Chrome 웹 브라우저 창 예시	17
[그림 2] 웹 브라우저 HTML 렌더링 예시	18
[그림 3] 로그인 및 등록 화면 예시	19
[그림 4] 로그인 오류 팝업 예시	20
[그림 5] 등록 화면 예시	21
[그림 6] 메인 화면 접근 예시	22
[그림 7] 헤드 섹션 예시	23
[그림 8] 문제 설명 섹션 예시	25
[그림 9] 테스트 케이스 섹션 예시	26
[그림 10] 코드 에디터 섹션 예시	27
[그림 11] 코드 에디터 섹션 예시	27
[그림 12] 결과 섹션 실행 결과 예시	28
[그림 13] 결과 섹션 채점 결과 예시	29
[그림 14] 결과 섹션 제출 결과 예시	29
[그림 15] 강의 교수자 화면 예시	30

[그림 16] 강의 교수자 과제 관리 화면 예시	31
[그림 17] Use Case Diagram	41
[그림 18] Data Flow Diagram	44
[그림 19] Context Model	48
[그림 20] Activity Diagram	49
[그림 21] Sequence Diagram	50
[그림 22] 시스템 구조	50

표 목차

[표 1] 약어 정리	8
[표 2] 전문용어 정의	9
[표 3] 웹 서비스에 대한 사용자의 GUI 기반 접근	17
[표 4] 사용자 GUI (1) 로그인 화면	18
[표 5] 사용자 GUI (2) 로그인 오류	19
[표 6] 사용자 GUI (3) 등록 화면	20
[표 7] 사용자 GUI (4) 등록 오류	21
[표 8] 사용자 GUI (5) 메인 화면	22
[표 9] 사용자 GUI (6) 메인 화면 – 헤드 섹션	23
[표 10] 사용자 GUI (7) 메인 화면 – 문제 설명 섹션	24
[표 11] 사용자 GUI (8) 메인 화면 – 테스트 케이스 섹션	25
[표 12] 사용자 GUI (9) 메인 화면 – 코드 에디터 섹션	26
[표 13] 사용자 GUI (10) 메인 화면 – 결과 섹션	27
[표 14] 사용자 GUI (11) 강의 교수자 화면	30
[표 15] 사용자 GUI (12) 강의 교수자 화면 오류	30
[표 16] 사용자 GUI (13) 강의 교수자 과제 관리 화면	31
[표 17] 브라우저 사용에 필요한 하드웨어 인터페이스	32
[표 18] 서비스 서버 소프트웨어 인터페이스	32
[표 19] 서비스 사용자 소프트웨어 인터페이스	33

[표 20] 로그인 HTTP 요청 형태	34
[표 21] 계정 등록 HTTP 요청 형태	34
[표 22] 강의/과제 목록 HTTP 요청 형태	35
[표 23] 과제 정보 HTTP 요청 형태	35
[표 24] 과제 채점 HTTP 요청 형태	36
[표 25] 과제 제출 HTTP 요청 형태	36
[표 26] 강의 등록 HTTP 요청 형태	36
[표 27] 학생 등록 HTTP 요청 형태	37
[표 28] 과제 등록 HTTP 요청 형태	37
[표 29] 사용례(1) 로그인	37
[표 30] 사용례(2) 계정 등록	38
[표 31] 사용례(3) 과제 출제	39
[표 32] 사용례(4) 과제 수행	39
[표 33] 사용례(5) 성적 확인(교수자)	40
[표 34] 사용례(6) 성적 확인(수강생)	40
[표 35] 사용례(7) 강의 등록	40
[표 36] 사용례(8) 수강생 추가	41
[표 37] 데이터베이스 Table 'User' 구조	42
[표 38] 데이터베이스 Table 'Class' 구조	42
[표 39] 데이터베이스 Table 'Enrollment' 구조	42
[표 40] 데이터베이스 Table 'Assignment' 구조	43
[표 41] 데이터베이스 Table 'Code' 구조	43
[표 42] 데이터베이스 Table 'Testcase' 구조	43
[표 43] 문서 이력	52

1. Introduction

1.1. Purpose

본 문서는 "Online Judge Platform" 서비스 (이하 '본 서비스') 제공을 위한 소프트웨어 요구 명세서이다. 본 서비스는 2022 년 2 학기 성균관대학교 소프트웨어공학개론 41 분반 10 팀(이하 '개발팀')에 의해 고안되고 개발된다. 본 요구 명세서에서 소프트웨어 시스템의 요구 사항을 요약 및 분석하였으며, 이에 근거하여 본 서비스의 시스템을 개발한다.

본 문서는 개발팀이 열람하는 것을 상정하여 작성되었다. 예외적으로, 교수, 조교를 포함한 성균관대학교 소프트웨어공학개론 수강자 또한 학습 및 교육의 용도로 본 문서를 열람할 수 있다. 본 문서를 재 배포 및 수정하는 것은 자유이나, 상업적 용도로 활용 시 반드시 개발팀의 허가를 얻어야 한다.

본 문서는 본 서비스의 개요와 요구 명세를 제시하기 위해 작성되었다. 본 서비스는 웹 어플리케이션 기반으로 구현되어, 온라인 코딩 테스트 플랫폼을 제작하기 위해 개발되었다. 서비스 내에서 사용자가 제출한 코드를 채점하는 것으로부터 시작하여, 최종적으로 사용자가 본인이 제출한 코드에 대한 평가와 분석까지 가능할 수 있도록 서비스를 제공한다. 이 과정에서 웹 서버, 웹 인터페이스 및 사용자의 코드를 채점하는 시스템이 개발되며 세부사항은 이하 문서에 명시한다.

1.2. Scope

본 서비스는 컴퓨터 과학을 공부하는 학생들이 Python 언어 기반 코딩 테스트를 실습할 수 있게 하기 위해 고안되었다. 작성한 코드를 채점하고 평가하여 시각화 하는 기능을 제공한다. 또한 웹 환경으로 배포하여 다수의 사용자가 실행 환경으로부터 자유롭도록 한다. 이를 제공하기 위해 React 기반 프론트 엔드와 Django 기반 서버를 구축한다.

본 서비스를 통해 사용자는 Python 기반 코딩 테스트 스킬을 익힐 수 있다. 본 서비스의 공급자는 확장성을 고려한 설계를 바탕으로 다른 언어에 대한 테스트 플랫폼으로 활용할 수 있다.

1.3. Definitions, Acronyms, and Abbreviation

이하의 표는 본 문서에서 활용된 약어의 정의를 나타낸다.

[표 1] 약어 정리

용어	설명
RAM	Random Access memory
GUI	Graphic User Interface
HTTP	HyperText Transfer Protocol
DNS	Domain Name Service
TCP	Transmission Control Protocol
SSH	Secure SHell
MD	MarkDown
HTML	HyperText Markup Language
CSS	Cascading Style Sheet
URL	Uniform Resource Locator

이하의 표는 본 문서에서 활용된 전문용어의 정의를 나타낸다.

[표 2] 전문용어 정의

용어	설명
Django	Micro web framework written in Python
SQLite	Light-weight Database Management System
React	JavaScript library for building user interfaces
npm	Package manager for the JavaScript
DirectX	Microsoft Based Multimedia Graphical API
OAuth	Industry-standard protocol for authorization

1.4. References

- IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications, In IEEEExplore Digital Library
<http://ieeexplore.ieee.org/Xplore/guesthome.jsp>
- IETF RFC 2616, HyperText Transfer Protocol 1.1.
<https://datatracker.ietf.org/doc/html/rfc2616>

- ECMA Script Standard, ECMA-262, Jun 2021.
<https://www.ecma-international.org/publications-and-standards/standards/ecma-262/>
- Flask Documentation
<https://flask.palletsprojects.com/>
- npm Documentation
<https://www.npmjs.com/>
- OAuth 2.0: Authorization and Authentication Protocol
<https://oauth.net/2/>

1.5. Overview

본 소프트웨어 요구 명세서의 이하 부분은 세 부분(Chapter 2.~4.)과 부록으로 구성되어 있다. Chapter 2.에서는 제품의 관점과 제품의 기능, 사용자 특징 및 제약조건에 대해 서술한다. 제품의 관점 항목은 하드웨어, 소프트웨어, 통신, 그리고 사용자의 인터페이스에 대해 서술한다. 제품의 기능 항목은 본 서비스가 제공하는 기능에 대해 서술한다. Chapter 3.에서는 세부적인 요구 명세가 기술된다. 각 인터페이스의 형태 및 변수들의 제약조건에 대해 서술한다. 또한 요구조건의 형태를 모델로서 나타낸다. Chapter 4.에서는 변경 이력 등 본 문서에 대한 추가적인 정보를 서술한다.

2. Overall Description

2.1. Product Perspective

본 제품은 Python 프로그래밍을 학습하는 강의의 과제를 수행하고자 하는 학생들을 위해 개발되었다. Python 을 사용할 환경이 없는 경우에도 온라인에서 해당 코드를 실행할 수 있는 환경을 구축하여 언제 어디서든 과제를 할 수 있다. 교육 대상이 되는 Python 인터프리터 버전은 3.10.X 버전이다. 사용자는 서비스를 통해 Python 코드를 작성하여 각 과제에 대응되는 문제들을 풀 수 있다. 이렇게 작성한 코드는 검증, 실행, 채점을 통해 작동을 확인할 수 있고 최종적으로 제출을 통해 코드를 제출할 수 있다. 본 제품이 가정하는 동시 사용자는 100 명 이내이며 그 이상의 접근에 대해서는 거부하거나 대기를 요청한다.

2.1.1. System Interfaces

사용자는 Chrome 웹 브라우저를 이용해 본 서비스를 사용한다. 이 때 Chrome 웹 브라우저의 버전은 100.X 이상을 대상으로 한다. 웹 브라우저는

사용자에게 로그인 기능을 제공함과 동시에, 로그인 하지 않은 계정의 서비스 이용을 제한한다. 이는 서버가 지나치게 많은 코드를 검사하거나 무차별적인 서비스 거부 공격(Denial of Service)을 당하는 상황을 방지하기 위함이다. 로그인에 사용되는 정보는 SQLite 데이터베이스를 통해 관리한다.

웹 서비스를 제공하기 위한 프론트엔드 환경은 NodeJS 프레임워크를 활용하여 구현한다. NodeJS 에 React 라이브러리를 설치하여 React 프레임워크를 기반으로 화면과 작동을 구성한다. 백엔드는 Python 기반의 프레임워크인 Django 를 사용한다. 이 Django 와 SQLite 를 연결하여 백엔드 서버와 데이터베이스를 연결한다. SQLite 는 Django 와의 연결이 용이하고, 또한 가벼운 구현으로 소규모 서비스를 제공하기 적합하다고 판단하여 선정했다.

코드를 테스트하기 위한 백엔드 환경은 Python 으로 작성된 채점 API 를 사용하여 APP 을 만들어 구축한다.

2.1.2. User Interfaces

사용자는 Chrome 웹 브라우저를 이용해 React 로 작성된 내용물을 읽고 상호작용할 수 있다. 웹 브라우저 기반의 구현이라 어떠한 환경이든 접근 가능하지만, 원활한 사용을 위해서는 데스크탑 PC 와 모니터 해상도 1920 * 1080 이상을 권장한다. 사용자가 수행할 수 있는 주요 기능은 로그인(회원가입), 문제 출제, 문제 열람, 코드 테스트의 네 가지이다. 또한 본 서비스는 이 기능들을 사용자에게 원활하게 제공하기 위한 웹 디자인적 요소들을 포함한다.

관리자는 강의 정보를 수정하기 위해서 직접 서비스가 제공되는 서버의 데이터에 접근해야 한다. 본 제품의 관리자는 프론트엔드 수준의 사후관리를 지원할 필요가 없으므로 별도로 구현하지 않는다. 서비스 서버는 HTTPS 프로토콜 기반의 IP 주소를 통해 접근할 수 있으며 관리자는 Django 에서 사용하는 데이터 및 채점 기능의 API 들의 접근을 수정하여 사용자에게 제공되는 웹 서비스를 수정할 수 있다.

2.1.3. Hardware Interfaces

사용자의 측면에서, 웹 브라우저 외의 하드웨어 의존성은 존재하지 않는다. 본 제품은 Chrome 웹 브라우저를 Windows 환경에서 사용한다고 가정하고 개발한다. Chrome 브라우저의 시스템 요구사항은 Windows 7 이상의 운영체제, 및 일정 수준 이상의 프로세서이다. Windows 7 의 최소사항과 조합하면 다음과 같다.

1. 1GHz 이상 32 비트(x86) 또는 64 비트(x64) 프로세서 (Core I5 이상)
2. 1GB RAM(32 비트) 또는 2GB RAM(64 비트)

3. 16GB 사용 가능한 하드 디스크 공간(32 비트) 또는 20GB(64 비트)

4. WDDM 1.0 이상 드라이버와 DirectX 9 그래픽 디바이스

또한 인터넷에 접근하기 위한 NIC 및 웹 브라우저의 GUI 와 소통하기 위한 마우스 등의 HID 를 포함한다.

관리자의 측면에서, 원활하게 Django 서버와 채점 API 제공하기 위한 하드웨어 요구조건은 다음과 같다. 원활한 서버 운영을 위해 서술된 것보다 더 고성능의 환경을 갖추기를 권장한다.

1. Ubuntu 20.04 운영체제 및 NIC 를 제공하는 메인보드

2. 4GB 이상의 RAM 환경

3. 25GB 이상의 하드 드라이브

2.1.4. Software Interfaces

상술했던 바와 같이 사용자는 Windows 7 이상의 운영체제에서 최신 안정화 Chrome 버전을 사용함을 기준으로 한다. Windows 10 기준 Chrome 100 버전 이상을 사용하는 환경을 가정한다. 또한 필수 사항은 아니지만 원활한 교육을 위해 웹 공간뿐 아닌 각 사용자의 기기에서도 Python 인터프리터 버전 3.10.X 을 사용하는 것을 권장한다.

관리자는 Ubuntu 20.04 이상의 운영체제에서 Django 를 사용하여 웹 서버를 구축한다. 웹 서버에서 사용자 정보를 다루기 위해 SQLite 데이터베이스를 활용한다. 사용자가 입력한 코드를 실행 및 검증하는 환경을 마련하기 위해 채점 API 를 활용한다.

2.1.5. Communications Interfaces

웹 페이지는 HTTP 프로토콜을 기반으로 사용자의 정보를 서버로 전달하며, 서버 또한 사용자에게 노출될 정보를 HTTP 프로토콜을 기반으로 전달한다. TLS 1.2 이상을 지원하기 위한 DNS 와 Root Certificate 는 본 제품에 포함되어 있지 아니하며 추후 관리자에 의해 추가될 수 있다.

2.1.6. Memory Constraints

서버의 메모리는 4GB RAM 을 최소사양으로 한다. 그러나 안정적인 서비스 제공을 위해 16GB 혹은 32GB 이상의 메모리 공간을 권장한다. 본 서버가 지원하는 최대 인원인 100 명에 대한 채점 API 를 통해 APP 을 생성할 수 있어야 한다.

사용자의 메모리는 1GB RAM 을 최소사양으로 한다. 그러나 안정적인 서비스 이용을 위해 Windows 10 이상의 운영체제 요구사항을 충족하고 4GB 이상의 메모리를 갖출 것을 권장한다. 이는 본 페이지가 제공하는 GUI 처리에 의한 성능 저하가 서비스 품질에 영향을 주지 않게 하기 위함이다.

2.1.7. Operations

2.1.7.1. System administrator

- 관리자 페이지
 - ✓ 관리자 계정일 경우 접근할 수 있다.
 - ✓ 관리자 간 공유할 내용 등을 기록할 때 활용한다.
 - ✓ 서버에 접근할 수 있는 권한을 DB 에 저장한다.
 - ✓ 강의 정보를 수정한다.

2.1.7.2. User

- 로그인
 - ✓ 사용자는 본인의 비밀번호 및 계정으로 서비스에 로그인할 수 있다.
- 등록
 - ✓ 사용자는 비밀번호 및 계정을 서비스에 등록할 수 있다.
 - ✓ 등록된 계정을 추후 로그인에 활용할 수 있다.
- 수강생 등록
 - ✓ 사용자(교수자)는 본인이 담당하는 강의의 수강생을 등록할 수 있다.
- 문제 선택
 - ✓ 사용자(수강생)은 본인이 풀 과제를 선택할 수 있다.
- 문제 풀이
 - ✓ 문제 페이지에서 코드를 작성할 수 있다.
 - ✓ 작성된 코드는 중간에 서버에 저장할 수 있다.
 - ✓ 파일에서 작성된 코드를 불러올 수 있다.
 - ✓ 코드를 파일로 저장할 수 있다

- 검사하기
 - ✓ 코드를 여러 번 실행할 수 있다.
 - ✓ 코드를 여러 번 채점할 수 있다.
 - ✓ 채점은 테스트케이스를 기준으로 성공여부를 제공한다.
 - ✓ 테스트케이스는 공개와 비공개가 있다.
- 제출하기
 - ✓ 과거 제출 코드는 여러 번 다시 풀기 가능하다.
 - ✓ Code Diff 를 통해 정답 코드와 비교한다.
 - ✓ 제출결과와 코드 설명, 관련 자료를 보여준다.
- 출제하기
 - ✓ 사용자(교수자)는 본인이 담당하는 강의의 과제를 출제할 수 있다.
- 점수 확인하기
 - ✓ 수강생이 제출한 코드와 채점된 점수를 확인할 수 있다.

2.2. Product Functions

2.2.1. Login & Sign-up

사용자가 웹 브라우저를 통해 서비스에 접근하면, 서비스는 사용자에게 로그인 혹은 계정 등록을 할 수 있는 인터페이스를 제공한다. 등록된 계정이 있을 경우 계정을 활용해 서비스를 이용할 수 있다. 등록된 계정이 없을 경우 신규 계정을 생성할 수 있다. 등록 과정에서 사용하는 계정명과 비밀번호는 함께 Hash 하여 서버의 데이터베이스에 저장한다. 로그인 시 해시값을 기반으로 성공 여부를 검증한다. 계정 정보가 잘못된 경우 재입력을 요청한다.

로그인하여 서비스에 접근할 경우 세션이 유지되며 세션의 수 및 세션의 유일성 여부를 이용해 서비스 품질을 향상할 수 있다. 본 제품이 지원하는 세션은 최대 100 개이며 동일 세션에서 여러 코드 검증을 요청할 수 없다. 세션의 수를 초과한 요청에 대해서는 서버 상태에 대해 공지한 뒤 로그인을 허락하지 않는다.

2.2.2. Admin Page Access

로그인에 성공한 계정은 사용자 계정 및 관리자 계정으로 구분된다. 관리자 계정의 경우 일반적인 계정 등록을 통해 얻을 수 없으며 다른 관리자에 의해 권한을 부여 받아야한다. 공급자는 본 제품을 사용할 때, 최초의 관리자 계정을 제공받는다. 이 계정을 활용하여 서버에 접근한 후 다른 관리자를 선정할 수 있다.

관리자 계정은 로그인 후 관리자 페이지로 접근할 수 있는 인터페이스를 제공받는다. 관리자 페이지로 접근할 때에 계정의 관리자 여부를 다시 한번 검증한다. 관리자임이 확인된 경우 관리자 간 정보를 공유하기 위한 페이지 및 서버에 접근하기 위한 권한을 DB 에 저장한다. 관리자는 서버에 직접 접근하여 다른 관리자를 선임하거나, 웹 서비스가 제공하는 내용들을 수정하거나, 서버 환경 자체를 수정할 수 있다.

2.2.3. Register Students

일반 사용자(교수자)가 로그인에 성공한 경우, 담당하는 강의에 강의를 수강하는 수강생을 추가할 수 있는 인터페이스가 제공된다. 사용자는 수강생의 학번을 입력하여 등록할 수 있고, 수강생들에게 과제를 부여하고 이들이 작성한 코드와 채점 결과를 확인할 수 있다.

2.2.4. Select Lecture Course

일반 사용자(수강생)이 로그인에 성공한 경우, 수강하는 강의 중 과제를 수행할 강의를 선택할 수 있는 인터페이스를 제공받는다.

2.2.5. Select Lecture Practice

수강하는 강의를 선택하면 과제를 선택할 수 있는 인터페이스가 제공된다. 이때 제출 여부를 그래픽적으로 나타내어 구분할 수 있다. 이때 제출한 경우 최종 제출의 점수를 같이 보여준다. 반면 제출이 안된 경우 미제출로 표기된다.

이때 선택된 과제는 별도의 창에서 문제 및 IDE 를 제공받는다. IDE 의 좌측 상단에는 수행할 과제가 작성되어 있고 하단에는 공개 테스트케이스의 리스트가 나열되어 있다. 사용자는 과제 내용을 만족하는 Python 코드를 IDE 에 작성할 수 있다. IDE 의 우측에는 실행, 채점, 제출의 결과는 보여주는 창이 있다. 이때 각 창의 크기는 조절이 가능하다.

2.2.6. Save User Code

과제 페이지에서 코드를 작성하고 중간에 저장할 수 있다. 중간 저장은 제출을 의미하지 않으며 최대 3 회까지 저장할 수 있다. 이때 추후에 이 저장된

버전을 선택할 경우 각 버전의 코드를 불러올 수 있어야 한다. 또한 외부의 파일에서 작성된 코드를 불러올 수 있다. 추가적으로 작성한 코드를 다운로드하는 것도 가능하다. 또한 코드를 최초의 skeleton 코드로 초기화할 수 있다.

2.2.7. Verifying User Code

검증, 실행, 채점을 여러 번 진행할 수 있다. ‘검증’버튼은 IDE 좌측창의 하단에 위치한 공개 테스트케이스 별로 위치하며 각 테스트케이스를 검증한다. 이때 결과값에 따라 pass/fail 의 결과를 제공한다. Fail 일 경우 코드의 결과값도 같이 제공한다.

‘실행’ 버튼을 눌러 제공하는 코드를 서버가 테스트하며 해당 결과를 좌측의 창의 Console 인터페이스를 통해 보여준다. 사용자가 제출한 코드가 정상적으로 동작하면 출력값을 제공한다. 반면 에러가 난 경우 에러 메시지에 따른 위치를 IDE 에서 하이라이트하고 해당 라인의 좌측 전구 아이콘을 누르면 해당 에러 메시지를 제공한다.

‘채점’ 버튼을 눌러 코드를 서버로 제출하면 서버에서 테스트케이스를 모두 테스트한다. 테스트를 진행하며 pass/fail 의 정보를 제공한다. 여기서 테스트케이스는 공개 테스트케이스와 비공개 테스트케이스로 나뉜다. 공개 테스트케이스가 실패한 경우 테스트케이스의 정보를 제공하지만 비공개의 경우 제공하지 않는다.

2.2.8. Submitting User Code

‘제출’ 버튼을 눌러 코드를 최종적으로 제출한다. 제출된 코드를 Code Diff 를 통해 교수자가 제공하는 정답 코드와 비교하여 바른 점을 표기한다. 이때 제출의 경우 최대 3 번 가능하다.

제출결과는 표절 검사, 기능 채점, 효율 채점, 가독성 채점을 한다. 표절 검사를 통과한 코드의 경우 나머지 점수를 IDE 의 우측페이지에서 최상단에 원형의 그래프로 보여준다. 해당 그래프 밑에서는 각각의 점수를 각 테스트케이스마다 보여준다.

그 밑에는 OpenAi Codex API 를 이용한 코드 설명이 있다. 이 이하는 관련 자료로 다른 문제, 영상, 학습 자료를 추천한다. 이때 각 추천사항은 특정 URL 로 연결되어 클릭 시 해당 페이지로 이동 가능하다.

2.3. User Characteristics

2.3.1. System Administrator

시스템 관리자는 본 서비스의 강의 데이터를 업데이트하고 서버 문제에 대응하는 사람을 뜻한다. 시스템 관리자는 시스템의 로그를 통해 문제를 확인한 뒤 이를 해결할 수 있어야 한다. 또한 SQLite의 쿼리문을 사용해 사용자가 문의하는 오류를 수정하거나 다른 관리자에 의해 변경된 사안을 확인할 수 있어야 하며 필요한 경우 데이터베이스에 데이터를 추가, 수정, 삭제할 수 있어야 한다. 이를 위해 HTML, CSS, JavaScript를 통해 React 기반의 웹 프레임워크와 SQLite, 서버 관리에 대한 지식을 갖추고 있어야 한다.

2.3.2. User

사용자는 본 서비스를 사용하여 과제를 출제하거나 제출하는 사람을 뜻한다. 본 서비스는 사용자를 프로그래밍 강의를 수강하는 성균관대학교 재학생 및 해당 강의의 교수자로 가정하였다. 사용자는 웹 브라우저를 사용할 수 있어야 하며 GUI 반응에 따른 새로운 상호작용이 가능해야 한다.

2.4. Constraints

본 시스템은 상기한 요구 조건과 후술할 세부 요소들을 바탕으로 고안되고 구현되어야 한다. 본 문서에 명시되지 않은 세부 조건들은 개발자의 재량에 의해 구현되고 설계될 수 있다. 그러나 그 과정에서 다음과 같은 원칙은 지켜져야 한다.

- 시스템의 사용자가 접근하기에 용이하고 사용이 편리하게 인터페이스를 구축할 것.
- 시스템의 성능이 저하되지 않도록 개발할 것
- 시스템의 유지 및 보수 비용을 최소화할 것.
- 외부 API를 사용하는 경우에는 보안과 성능이 보장된 API를 사용할 것
- 시스템 개발에 추가적인 비용이 필요하지 않도록 소프트웨어를 사용할 때 라이선스를 확인하고 가능한 오픈소스를 주로 사용할 것
- 시스템의 성능 향상을 위해 시스템 리소스를 낭비하지 않는 방향으로 개발할 것
- 시스템을 사용하는 인원이 늘어난다면 하드웨어의 개선만으로 늘어나는 인원을 수용할 수 있도록 확장 가능하게 설계할 것

2.5. Assumptions and Dependencies

본 서비스는 사용자가 문서에서 기술한 하드웨어와 소프트웨어의 요구조건을 만족하는 환경에서 접근하는 것을 가정한다. 따라서 웹 브라우저를 사용할 수 없는 환경에서 본 서비스에 접근할 경우 서비스의 정상적인 작동이 불가능하다.

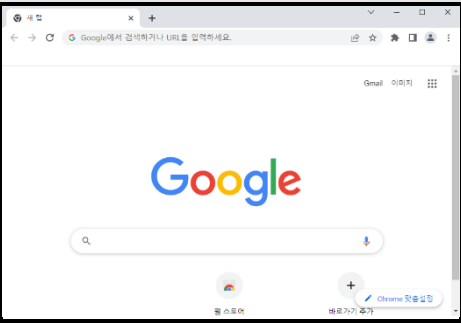
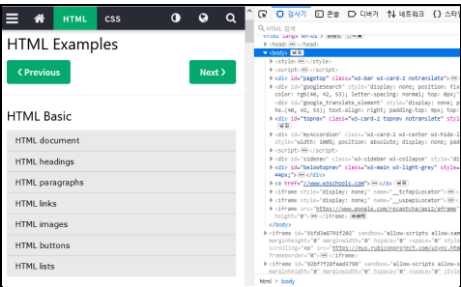
3. Specific Requirements

3.1. External Interface Requirements

3.1.1. User Interfaces


[표 3] 웹 서비스에 대한 사용자의 GUI 기반 접근

이름	웹 서비스에 대한 사용자의 GUI 기반 접근
목적 및 설명	사용자가 웹 서비스와 상호작용할 수 있는 가장 기본적인 인터페이스이다. 인터페이스는 브라우저의 기본 구현을 따르며 서버로부터 Html, CSS, JavaScript 를 받아 사용자의 화면에 전달한다. 사용자는 브라우저 창을 통해 제공된 정보를 바탕으로 서비스에 요청을 보내거나 서비스가 제공하는 자료를 열람할 수 있다.
입출력 형태	입력은 마우스 클릭, 호버링, 키보드 입력 등 HID 입력을 통해 이루어진다. 출력은 브라우저 JS 처리 및 HTML 링크 연산으로 이루어진다.
범위/정확도/오차범위	W3C 표준에 의한 CSS 처리 EMCAScript 표준에 의한 JavaScript 처리 WHATWG 표준에 의한 Html 처리 *표준을 지키지 않는 기술에 대해서는 브라우저 종류와 버전에 따라 동작이 다를 수 있다.
단위	Html DOM 컴포넌트와의 상호작용 JavaScript 이벤트와의 상호작용
시간/속도	서버 및 통신망 의존성: 페이지 로드 네트워크 지연시간 사용자 컴퓨터 성능 의존성: 페이지 렌더링 지연시간
다른 입출력 간 연관성	브라우저 확장 기능에 의한 페이지 상태 변화 브라우저 자체 기능에 의한 페이지 차단 및 Redirecting
화면의 형태	사용자의 브라우저의 기본 설정에 따름 크롬 브라우저 기준 아래 그림과 같음 DOM 및 JS 표준에 맞추어 렌더링
창의 형태	브라우저의 기본 설정에 따름 크롬 브라우저 기준 아래 그림과 같음 사용자 확장기능 및 테마에 맞추어 변화 [그림 1] Chrome 웹 브라우저 창 예시

이름	웹 서비스에 대한 사용자의 GUI 기반 접근
	
데이터 형태	<p>브라우저에서 인식하는 HTML 하이퍼링크 접근 브라우저에서 인식하는 JS 이벤트</p> <p>[그림 2] 웹 브라우저 HTML 렌더링 예시</p> 
종료 동작	브라우저 종료 혹은 페이지 이동

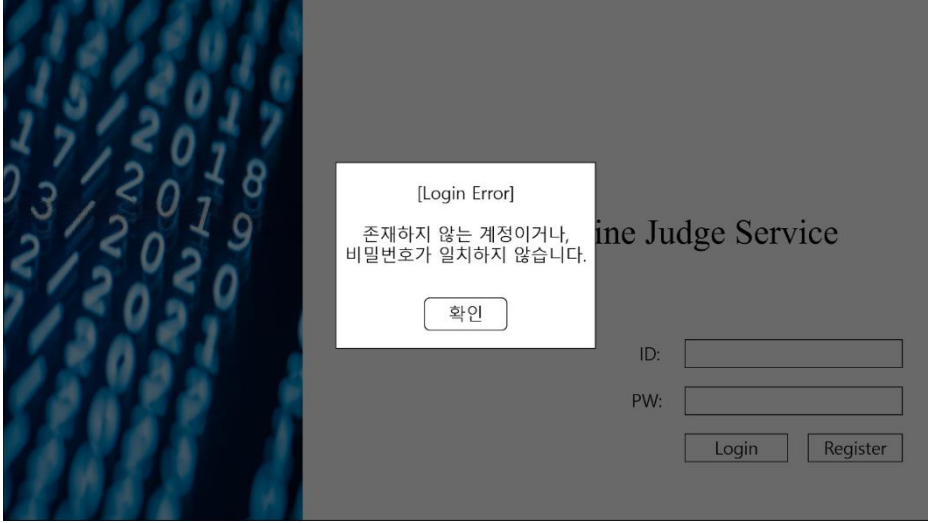
[표 4] 사용자 GUI(1) 로그인 화면

이름	사용자 GUI1. 로그인 화면
목적 및 설명	사용자가 IP 주소 혹은 도메인 이름을 이용하여 본 서비스에 접근할 때, 사용자에게 처음 제공되는 화면이다. 본 화면에서는 사용자가 로그인 또는 등록 페이지로 이동할 수 있는 환경을 제공한다.
입출력 형태	<p>본 인터페이스는 서버에 최초 접근할 때 수신한 Html, Css, JavaScript 를 입력으로 형성된다.</p> <p>‘Login’ 버튼을 눌러 사용자 계정, 사용자 비밀번호를 전송할 수 있다.</p> <p>‘Sign-up’ 버튼을 눌러 계정 등록 화면으로 전환할 수 있다.</p> <p>*이외의 동작은 상기 [표 3]의 웹 서비스 GUI에 상속됨</p>
범위/정확도/오차범위	표준을 지키지 않는 HTML 동작은 브라우저 버전에 따라 다를 수 있음 HTTP 패킷 전달은 TCP 기반 통신의 표준을 따른다.
단위	화면
시간/속도	인터페이스 표기: 상기 [표 3]의 웹 서비스 GUI에 상속됨 HTTP POST 요청 처리: 서버 처리 시간에 상속됨
다른 입출력 간 연관성	*상기 [표 3]의 웹 서비스 GUI에 상속됨
화면의 형태	<p>아래의 화면과 같은 형태를 띤다.</p> <ol style="list-style-type: none"> 상단에 위치한 사이트 이름

이름	사용자 GUI 1. 로그인 화면
	<p>2. 사용자의 계정과 비밀번호를 입력하는 공간</p> <p>3. 입력된 사용자의 정보를 전송할 Login 버튼</p> <p>4. 계정 등록 화면으로 전환할 Sign-up 버튼</p> <p>*세부적인 사항은 상기 [표 3]의 웹 서비스 GUI에 상속됨</p> <p>[그림 3] 로그인 및 등록 화면 예시</p> 
데이터 형태	계정, 비밀번호의 값은 문자열 붙이기 연산 후 해시하여 전송 전송 형태는 HTTP POST 표준을 따름.
종료 동작	서버 처리 결과에 의한 에러 팝업 서버 처리 결과에 의한 다음 동작 페이지

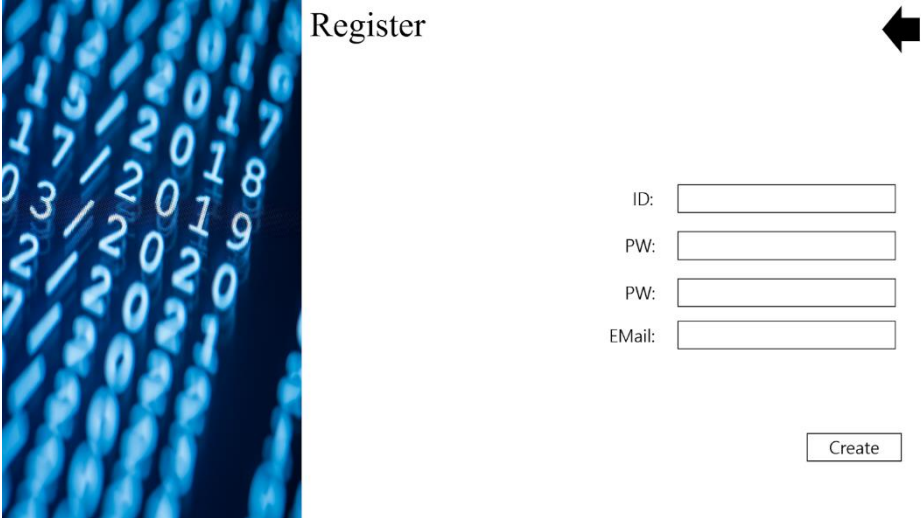
[표 5] 사용자 GUI (2) 로그인 오류

이름	사용자 GUI 2. 로그인 오류
목적 및 설명	상기 사용자 GUI 1. 로그인 화면에서 사용자가 서버에 요청한 로그인 HTTP POST의 처리 결과가 에러일 경우 발생한다.
입출력 형태	<p>서버로부터 HTTP Response를 수신하고, 해당 내용이 오류에 해당하는 경우 GUI 1의 JavaScript에 의해 팝업창이 발생한다. 팝업창에 표시되는 버튼을 클릭하여 창을 닫고 로그인 및 등록 화면 GUI로 돌아갈 수 있다.</p> <p>발생 가능한 오류의 종류는 다음과 같다.</p> <ol style="list-style-type: none"> 비밀번호가 일치하지 않음 존재하지 않는 계정임 <p>*상기 두 에러는 서버 측에서는 서로 다른 접근이지만 보안성을 위해 동일한 에러로 표기함</p> <p>*이외의 동작은 상기 [표 3]의 웹 서비스 GUI에 상속됨</p>
범위/정확도/오차범위	*상기 [표 4]의 웹 서비스 GUI 로그인 및 등록 화면에 상속됨
단위	화면
시간/속도	인터페이스 표기: 상기 [표 3]의 웹 서비스 GUI에 상속됨 HTTP POST 요청 처리: 서버 처리 시간에 상속됨

이름	사용자 GUI 2. 로그인 오류
다른 입출력 간 연관성	*상기 [표 3]의 웹 서비스 GUI 에 상속됨
화면의 형태	<p>아래의 화면과 같은 형태를 띤다.</p> <ol style="list-style-type: none"> 1. 에러 원인 표시 2. 에러 설명 표시 3. 이전 화면으로 돌아갈 수 있는 버튼 <p>*세부적인 사항은 상기 [표 3]의 웹 서비스 GUI 에 상속됨</p> <p>[그림 4] 로그인 오류 팝업 예시</p> 
데이터 형태	x
종료 동작	버튼 클릭으로 인한 로그인 및 등록 화면 GUI 로의 회귀

[표 6] 사용자 GUI (3) 등록 화면

이름	사용자 GUI 3. 등록 화면
목적 및 설명	상기 사용자 GUI 1. 로그인 화면에서 사용자가 'Sign-up' 버튼을 클릭한 경우 발생한다.
입출력 형태	<p>본 인터페이스는 GUI 1 의 로그인 화면에서 'Sign-up'버튼을 클릭한 후 수신한 Html, Css, JavaScript 를 입력으로 형성된다.</p> <p>사용자 계정, 비밀번호, 비밀번호 재확인, 이메일을 입력받은 후 "Create" 버튼을 통해 서버로 전달한다.</p> <p>*이외의 동작은 상기 [표 3]의 웹 서비스 GUI 에 상속됨</p>
범위/정확도/오차범위	*상기 [표 4]의 웹 서비스 GUI 로그인 화면과 동일함
단위	화면
시간/ 속도	<p>인터페이스 표기: 상기 [표 3]의 웹 서비스 GUI 에 상속됨</p> <p>HTTP POST 요청 처리: 서버 처리 시간에 상속됨</p>

이름	사용자 GUI 3. 등록 화면
다른 입출력 간 연관성	*상기 [표 3]의 웹 서비스 GUI 에 상속됨
화면의 형태	<p>아래의 화면과 같은 형태를 띤다.</p> <ol style="list-style-type: none"> 1. 사용자 계정, 비밀번호, 비밀번호 재확인, 이메일을 입력받는 공간 2. 입력된 정보를 서버로 전달할 “Create”버튼 3. 로그인 화면으로 돌아갈 수 있는 뒤로가기 버튼 <p>*세부적인 사항은 상기 [표 3]의 웹 서비스 GUI 에 상속됨</p> <p>[그림 5] 등록 화면 예시</p> 
데이터 형태	계정, 비밀번호, 이메일의 값은 문자열 붙이기 연산 후 해시하여 전송 전송 형태는 HTTP POST Response 표준을 따름
종료 동작	버튼 클릭으로 인한 등록 화면 GUI 로의 회귀

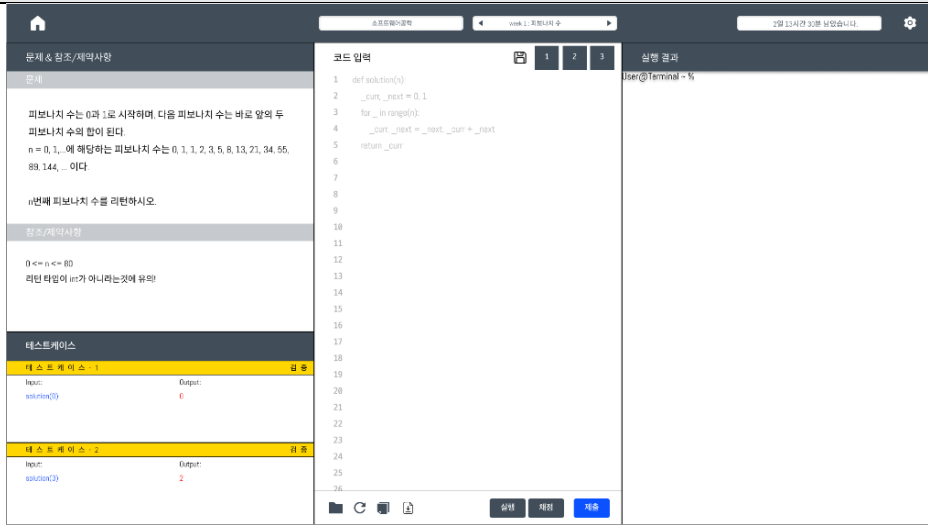
[표 7] 사용자 GUI (4) 등록 오류

이름	사용자 GUI 4. 등록 오류
목적 및 설명	상기 사용자 GUI 6. 등록 화면에서 ‘Create’ 버튼을 누른 경우, 올바르지 않은 정보가 서버로 전달되었을 때 발생한다.
입출력 형태	<p>서버로부터 HTTP Response 를 수신하고, 해당 내용이 오류에 해당하는 경우 GUI 3 의 JavaScript 에 의해 팝업창이 발생하며, 팝업창의 내용은 수신한 POST Response 의 Content 에 의해 결정된다. 팝업창에 같이 표시되는 버튼을 클릭하여 창을 닫고 등록 화면 GUI 로 돌아갈 수 있다.</p> <p>발생 가능한 오류의 종류는 다음과 같다.</p> <ol style="list-style-type: none"> 1. 이미 존재하는 계정명 2. 이미 등록된 이메일 3. 비밀번호 재확인 실패 <p>*이외의 동작은 상기 [표 3]의 웹 서비스 GUI 에 상속됨</p>
범위/정확도/오차범위	*상기 [표 4]의 웹 서비스 GUI 로그인 화면과 동일함
단위	화면

이름	사용자 GUI 4. 등록 오류
시간/ 속도	인터페이스 표기: 상기 [표 3]의 웹 서비스 GUI에 상속됨 HTTP POST 요청 처리: 서버 처리 시간에 상속됨
다른 입출력 간 연관성	*상기 [표 3]의 웹 서비스 GUI에 상속됨
화면의 형태	아래의 화면과 같은 형태를 띈다. *세부적인 사항은 상기 [표 3]의 웹 서비스 GUI에 상속됨 *에러 화면은 GUI 2. 로그인 오류와 동일한 팝업 형태임
데이터 형태	x
종료 동작	버튼 클릭으로 인한 등록 화면 GUI로의 회귀

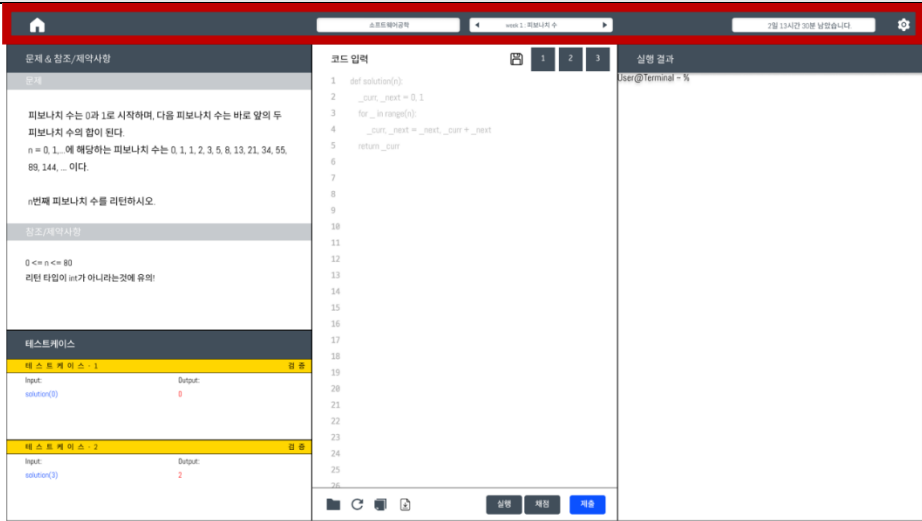
[표 8] 사용자 GUI(8) 메인 화면

이름	사용자 GUI 5. 메인화면
목적 및 설명	상기 사용자 GUI 1. 로그인 화면에서 사용자가 서버에 요청한 등록 HTTP POST의 처리 결과가 정상일 경우 나타난다. 이 화면에서 사용자는 과제를 선택하고 편의 기능을 제공하는 헤드 섹션, 문제와 참조, 제약사항을 확인할 수 있는 문제 설명 섹션, 테스트케이스를 확인할 수 있는 테스트 케이스 섹션, 코드를 작성할 수 있는 코드 에디터 섹션, 코드 에디터와 관련된 기능을 제공하는 기능 버튼 섹션, 코드 에디터에 작성된 코드를 실행한 결과가 출력되는 결과 섹션을 확인할 수 있다.
입출력 형태	서버로부터 수신한 Html, Css, JavaScript에 근거하여 페이지를 렌더링한다. 헤드 섹션, 문제 설명 섹션, 테스트 케이스 섹션, 코드 에디터 섹션, 기능 버튼 섹션, 결과 섹션이 포함된 Html이 표시된다. *이외의 동작은 상기 [표 3]의 웹 서비스 GUI에 상속됨
범위/정확도/ 오차범위	*상기 [표 3]의 웹 서비스 GUI에 상속됨
단위	화면
시간/ 속도	인터페이스 표기: 상기 [표 3]의 웹 서비스 GUI에 상속됨 HTTP GET 요청 처리: 서버 처리 시간에 상속됨
다른 입출력 간 연관성	*상기 [표 3]의 웹 서비스 GUI에 상속됨
화면의 형태	아래의 화면과 같은 형태를 띈다. *세부적인 사항은 상기 [표 3]의 웹 서비스 GUI에 상속됨 [그림 6] 메인화면 접근 예시

이름	사용자 GUI 5. 메인화면
	
데이터 형태	x
종료 동작	*상기 [표 3]의 웹 서비스 GUI에 상속됨

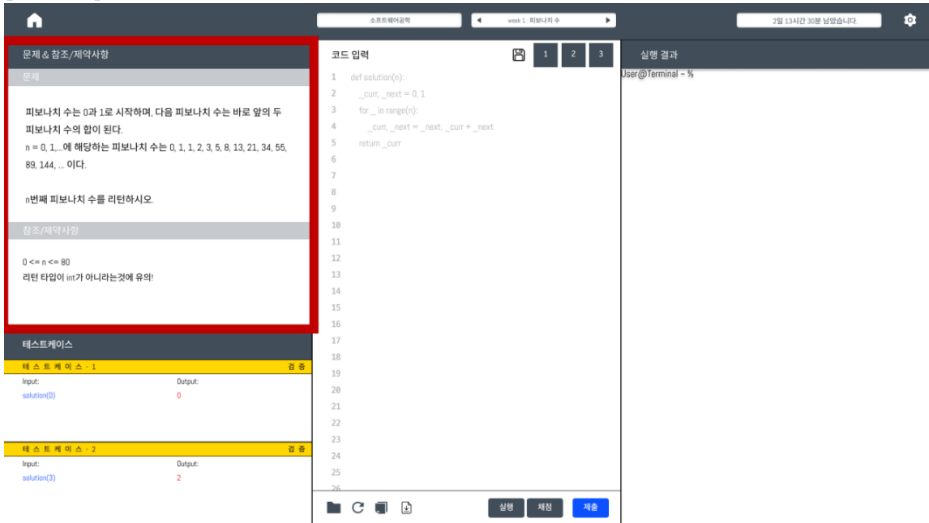
[표 9] 사용자 GUI (6) 메인 화면 - 헤드 섹션

이름	사용자 GUI 6. 메인 화면 - 헤드 섹션
목적 및 설명	상기 사용자 GUI 5의 일부로서 강의와 과제를 선택할 수 있고 편의 기능을 제공하기 위해 존재한다.
입출력 형태	서버로부터 수신한 Html, Css, JavaScript에 근거하여 섹션을 렌더링한다. 사용자가 수강중인 강의 목록과 강의의 과제 목록, 과제의 마감기한을 출력한다. *이외의 동작은 상기 [표 3]의 웹 서비스 GUI에 상속됨
범위/정확도/오차범위	학기중에 수강중인 강의 목록, 강의의 과제 목록 *상기 [표 3]의 웹 서비스 GUI에 상속됨
단위	섹션
시간/속도	인터페이스 표기: 상기 [표 3]의 웹 서비스 GUI에 상속됨 HTTP GET 요청 처리: 서버 처리 시간에 상속됨
다른 입출력 간 연관성	*상기 [표 3]의 웹 서비스 GUI에 상속됨
화면의 형태	아래의 화면과 같은 형태를 띈다. <ol style="list-style-type: none"> 1. 메인 화면으로 돌아올 수 있는 홈버튼 2. 현재 선택된 강의 명을 나타내는 수강중인 강의 목록 3. 현재 선택된 과제 명을 나타내는 과제 목록 4. 현재 선택된 과제의 마감 기한 5. 배경색이나 코드에디터를 변경할 수 있는 설정 버튼 *세부적인 사항은 상기 [표 3]의 웹 서비스 GUI에 상속됨 [그림 7] 헤드 섹션 예시

이름	사용자 GUI 6. 메인 화면 - 헤드 섹션
	 <p>The screenshot shows the '사용자 GUI 6. 메인 화면 - 헤드 섹션' (User GUI 6. Main Screen - Head Section). It features a sidebar on the left with sections: '문제 & 참조/제약사항' (Problem & Reference/Constraints), '입출/제약사항' (Input/Output/Constraints), and '테스트케이스' (Test Cases). The main area is divided into three panes: '문제 설명' (Problem Description) on the left, '코드 입력' (Code Input) in the center, and '실행 결과' (Execution Results) on the right. The '문제 설명' pane contains text about Fibonacci numbers and a sample input/output. The '코드 입력' pane shows a JavaScript function 'def solution(n)'. The '실행 결과' pane displays 'User@Terminal ~ %' and a list of test cases with their inputs and outputs.</p>
데이터 형태	JSON 형태로 강의 목록, 과제 목록 전달
종료 동작	*상기 [표 3]의 웹 서비스 GUI에 상속됨

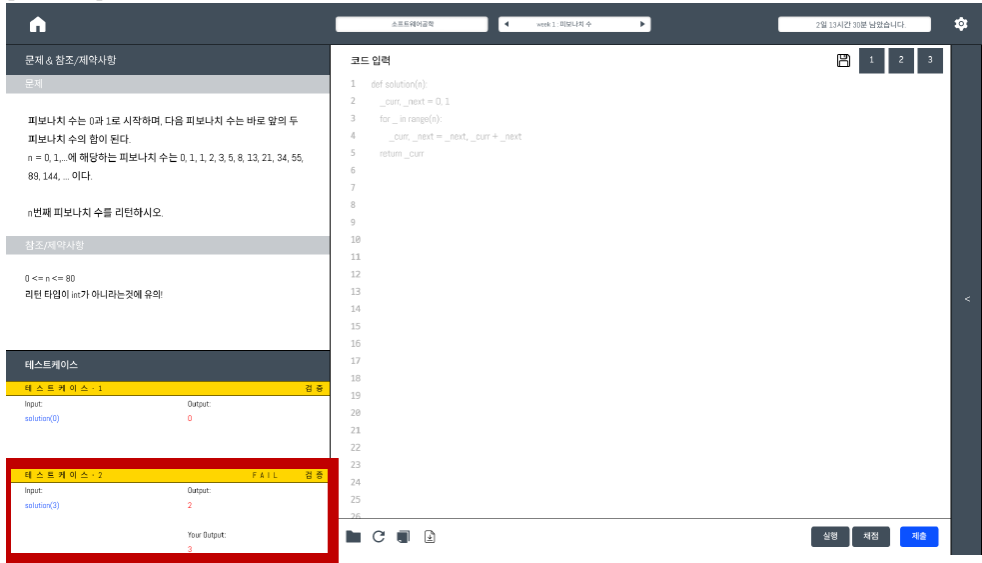
[표 10] 사용자 GUI (7) 메인 화면 - 문제 설명 섹션

이름	사용자 GUI 7. 메인 화면 - 문제 설명 섹션
목적 및 설명	상기 사용자 GUI 5의 일부분으로, 과제의 문제와 참조 및 제약 사항을 나타낸다.
입출력 형태	서버로부터 수신한 Html, Css, JavaScript에 근거하여 섹션을 렌더링한다. 사용자가 선택한 과제의 문제와 참조, 제약 사항을 출력한다. *이외의 동작은 상기 [표 3]의 웹 서비스 GUI에 상속됨
범위/정확도/오차범위	*상기 [표 3]의 웹 서비스 GUI에 상속됨
단위	섹션
시간/속도	인터페이스 표기: 상기 [표 3]의 웹 서비스 GUI에 상속됨 HTTP GET 요청 처리: 서버 처리 시간에 상속됨
다른 입출력 간 연관성	GUI 6. 헤드 섹션에서 선택된 과제에 따라 해당 섹션에 출력될 정보가 달라짐. *상기 [표 3]의 웹 서비스 GUI에 상속됨
화면의 형태	아래의 화면과 같은 형태를 띈다. 1. 문제 내용 2. 참조/제약사항 *세부적인 사항은 상기 [표 3]의 웹 서비스 GUI에 상속됨

이름	사용자 GUI 7. 메인 화면 - 문제 설명 섹션
	<p>[그림 8] 문제 설명 섹션 예시</p>  <p>The screenshot displays the '문제 설명 섹션' (Problem Description Section) of the Online Judge Platform. It includes a problem statement about Fibonacci numbers, a code editor with a solution in Python, and test cases with input/output pairs.</p>
데이터 형태	JSON 형태로 강의 목록, 과제 목록 전달
종료 동작	*상기 [표 3]의 웹 서비스 GUI에 상속됨

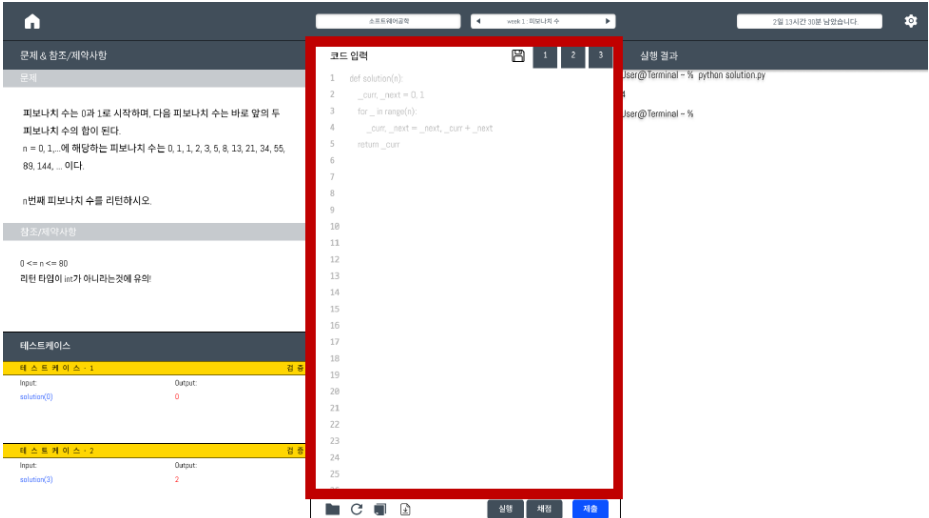
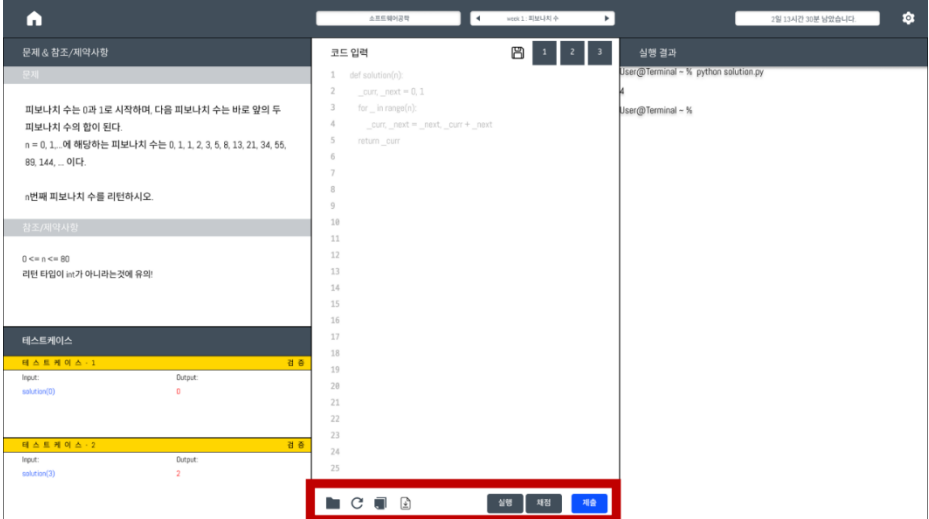
[표 11] 사용자 GUI (8) 메인 화면 - 테스트 케이스 섹션

이름	사용자 GUI 8. 메인 화면 - 테스트 케이스 섹션
목적 및 설명	상기 사용자 GUI 5의 일부로서, 문제의 테스트 케이스들을 나타낸다.
입출력 형태	<p>서버로부터 수신한 Html, Css, JavaScript에 근거하여 섹션을 렌더링한다.</p> <p>사용자가 선택한 과제의 문제의 테스트 케이스를 출력한다.</p> <p>테스트 케이스는 입력과 출력으로 이루어진다.</p> <p>*이외의 동작은 상기 [표 3]의 웹 서비스 GUI에 상속됨</p>
범위/정확도/오차범위	*상기 [표 3]의 웹 서비스 GUI에 상속됨
단위	섹션
시간/속도	<p>인터페이스 표기: 상기 [표 3]의 웹 서비스 GUI에 상속됨</p> <p>HTTP GET 요청 처리: 서버 처리 시간에 상속됨</p>
다른 입출력 간 연관성	<p>GUI 6. 헤드 섹션에서 선택된 과제에 따라 해당 섹션에 출력될 정보가 달라짐.</p> <p>GUI 9. 코드 에디터 섹션에서 작성된 코드에 따라 해당 섹션에 출력될 결과가 달라짐.</p> <p>*상기 [표 3]의 웹 서비스 GUI에 상속됨</p>
화면의 형태	<p>아래의 화면과 같은 형태를 띈다.</p> <ol style="list-style-type: none"> 1. 테스트 케이스의 입력 출력 2. 검증 버튼을 통해 사용자가 작성한 코드를 해당 테스트 케이스에 대해 검증 가능 3. 만약 성공한 경우, PASS 출력 4. 만약 실패한 경우, Fail 출력 및 사용자의 코드를 실행한 결과 출력 <p>*세부적인 사항은 상기 [표 3]의 웹 서비스 GUI에 상속됨</p>

이름	사용자 GUI8. 메인 화면 - 테스트 케이스 섹션
	<p>[그림 9] 테스트 케이스 섹션 예시</p> 
데이터 형태	JSON 형태로 강의 목록, 과제 목록 전달
종료 동작	*상기 [표 3]의 웹 서비스 GUI에 상속됨

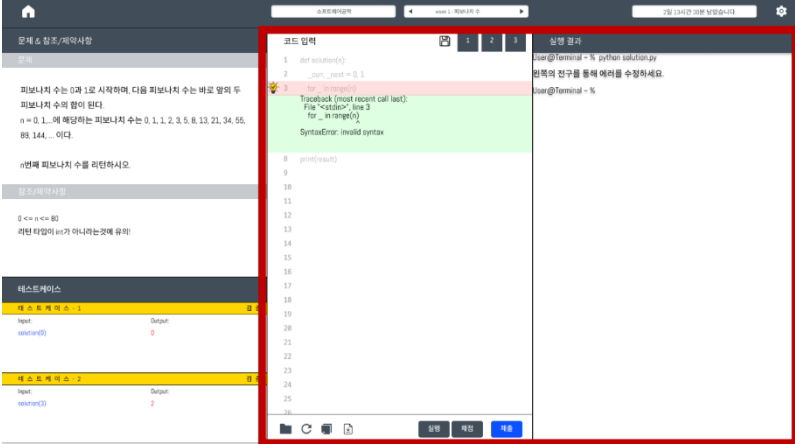
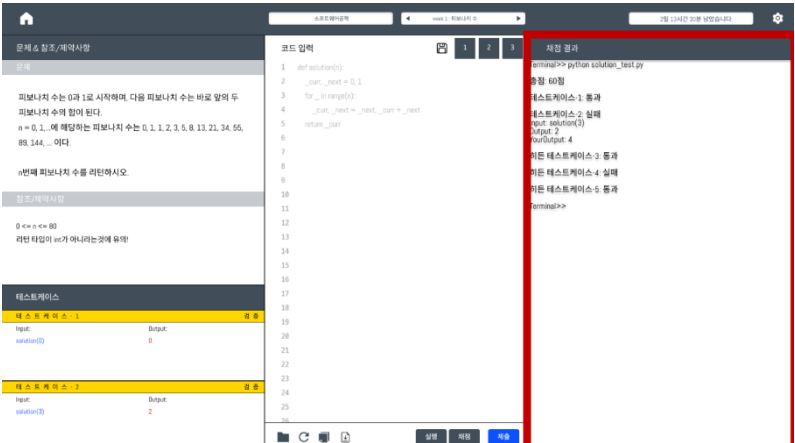
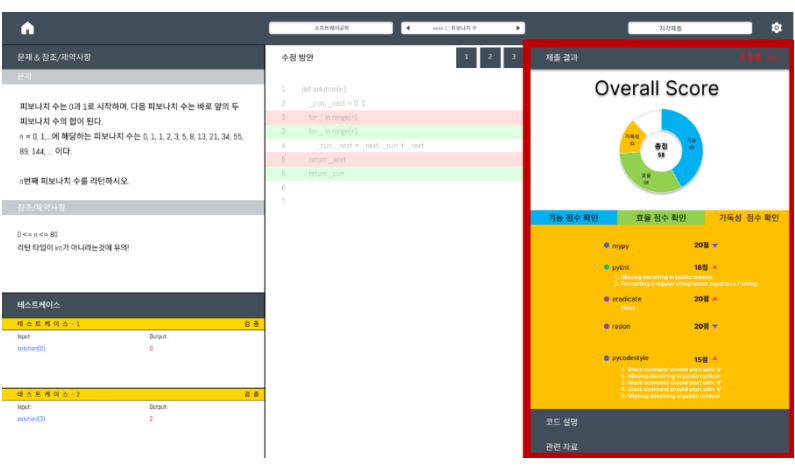
[표 12] 사용자 GUI (9) 메인 화면 - 코드 에디터 섹션

이름	사용자 GUI9. 메인 화면 - 코드 에디터 섹션
목적 및 설명	상기 사용자 GUI 5 의 일부로서, 사용자가 코드를 편집할 수 있는 공간과 코드 편집 관련 편의 기능을 나타낸다.
입출력 형태	<p>서버로부터 수신한 Html, Css, JavaScript 에 근거하여 섹션을 렌더링한다.</p> <p>사용자는 텍스트 형태로 Python 코드를 작성한 뒤, 기능 버튼을 통해 서버로 전달한다.</p> <p>이후 결과를 출력하기 위해 각각 실행, 채점, 제출 결과 화면을 결과 섹션에 나타낸다.</p> <p>*이외의 동작은 상기 [표 3]의 웹 서비스 GUI에 상속됨</p>
범위/정확도/오차범위	*상기 [표 3]의 웹 서비스 GUI에 상속됨
단위	섹션
시간/속도	<p>인터페이스 표기: 상기 [표 3]의 웹 서비스 GUI에 상속됨</p> <p>HTTP POST 요청 처리: 서버 처리 시간에 상속됨</p>
다른 입출력 간 연관성	<p>GUI 6. 헤드 섹션에서 선택된 과제에 따라 해당 섹션에 미리 출력되어 있을 스켈레톤 코드가 달라짐</p> <p>*상기 [표 3]의 웹 서비스 GUI에 상속됨</p>
화면의 형태	<p>아래의 화면과 같은 형태를 띤다.</p> <ol style="list-style-type: none"> 1. 스켈레톤 코드가 출력되어있는 코드 입력 공간 2. 코드를 중간 저장할 수 있는 버튼 3. 중간 저장한 코드를 불러올 수 있는 숫자 버튼 4. 파일로 작성된 코드를 읽어올 수 있는 버튼 5. 코드를 초기 상태로 초기화 할 수 있는 버튼

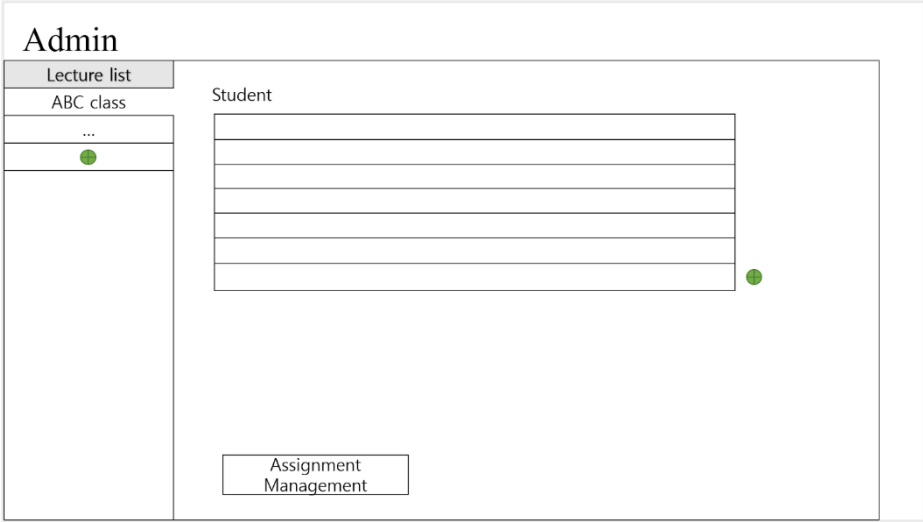
이름	사용자 GUI9. 메인 화면 - 코드 에디터 섹션
	<p>6. 작성 중인 코드를 복사할 수 있는 버튼</p> <p>7. 코드를 파일로 다운로드 할 수 있는 버튼</p> <p>8. 터미널을 통해 코드를 실행할 수 있는 버튼</p> <p>9. 코드를 채점할 수 있는 버튼</p> <p>10. 과제를 제출하기 위한 버튼</p> <p>*세부적인 사항은 상기 [표 3]의 웹 서비스 GUI에 상속됨</p> <p>[그림 10] 코드 에디터 섹션 예시</p>  <p>[그림 11] 코드 에디터 섹션 예시</p> 
데이터 형태	문자열의 형태로 작성한 코드를 서버로 전달
종료 동작	*상기 [표 3]의 웹 서비스 GUI에 상속됨

[표 13] 사용자 GUI (10) 메인 화면 - 결과 섹션

이름	사용자 GUI 10. 메인 화면 - 결과 섹션
목적 및 설명	상기 사용자 GUI 5 의 일부로서, 사용자가 코드를 작성한 뒤 터미널에서 실행, 채점, 제출한 결과를 나타낸다.
입출력 형태	서버로부터 수신한 Html, Css, JavaScript 에 근거하여 섹션을 렌더링 한다. 사용자는 텍스트 형태로 Python 코드를 서버로 전달한다. 서버는 코드를 입력받아 여러 성능을 측정한 후 결과를 전달한다. *이외의 동작은 상기 [표 3]의 웹 서비스 GUI 에 상속됨
범위/정확도/ 오차범위	*상기 [표 3]의 웹 서비스 GUI 에 상속됨
단위	섹션
시간/ 속도	인터페이스 표기: 상기 [표 3]의 웹 서비스 GUI 에 상속됨 HTTP GET 요청 처리: 서버 처리 시간에 상속됨 HTTP POST 요청 처리: 서버 처리 시간에 상속됨
다른 입출력 간 연관성	기존에 입력된 제출 코드에 따라 제출 횟수 및 저장 공간 사용 *상기 [표 3]의 웹 서비스 GUI 에 상속됨
화면의 형태	<p>아래의 화면과 같은 형태를 띤다.</p> <p>실행 결과</p> <ol style="list-style-type: none"> 1. 실행 결과가 정상적인 경우, 출력값 제공 2. 실행 결과가 에러일 경우, 에러 위치 하이라이트 및 에러 메시지 제공 <p>채점 결과</p> <ol style="list-style-type: none"> 1. 총점 제공 2. 테스트 케이스 별 Pass/Fail 여부 출력 3. 오픈 테스트 케이스의 경우, Fail 시 테스트 케이스 정보 출력 4. 히든 테스트 케이스의 경우, Pass/Fail 여부만 출력 <p>제출 결과</p> <ol style="list-style-type: none"> 1. 기존 제출 코드 불러오기 2. 정답 코드와 비교 제공 3. 표절을 출력 4. 점수 분야별 탭으로 확인가능 5. 기능 점수 확인 가능 6. 효율 점수 확인 가능 7. 가독성 점수 확인 가능 8. 전체 점수의 총합을 그래프로 상단에 표시 9. 하단에 코드 설명 제시 10. 관련 학습 자료 추천 <p>*세부적인 사항은 상기 [표 3]의 웹 서비스 GUI 에 상속됨</p> <p>[그림 12] 결과 섹션 실행 결과 예시</p>

이름	사용자 GUI 10. 메인 화면 - 결과 섹션
	 <p>[그림 13] 결과 섹션 채점 결과 예시</p>  <p>[그림 14] 결과 섹션 제출 결과 예시</p> 
데이터 형태	문자열의 형태로 작성한 코드를 서버로 전달 여러 성능 및 채점 결과를 JSON 형태로 전달
종료 동작	*상기 [표 3]의 웹 서비스 GUI에 상속됨

[표 14] 사용자 GUI(11) 강의 교수자 화면

이름	사용자 GUI 11. 강의 교수자 화면
목적 및 설명	강의 교수자 계정으로 로그인 성공한 경우, 나타나는 화면 강의를 수강하는 학생 관리를 가능하게 한다.
입출력 형태	서버로부터 수신한 Html, Css, JavaScript 에 근거하여 섹션을 렌더링 한다. 강의를 수강하는 학생의 정보를 입력한 뒤 서버로 전달한다. *이외의 동작은 상기 [표 3]의 웹 서비스 GUI 에 상속됨
범위/정확도/ 오차범위	*상기 [표 3]의 웹 서비스 GUI 에 상속됨
단위	화면
시간/ 속도	인터페이스 표기: 상기 [표 3]의 웹 서비스 GUI 에 상속됨 HTTP POST 요청 처리: 서버 처리 시간에 상속됨
다른 입출력 간 연관성	*상기 [표 3]의 웹 서비스 GUI 에 상속됨
화면의 형태	<p>아래의 화면과 같은 형태를 띈다.</p> <ol style="list-style-type: none"> 1. 현재 계정이 관리하는 강의 목록 출력 2. 선택한 강의를 수강하는 학생 목록 출력 3. 학생을 추가할 수 있는 버튼 4. 강의를 추가할 수 있는 버튼 5. 과제 관리 창으로 진입할 수 있는 버튼 <p>*세부적인 사항은 상기 [표 3]의 웹 서비스 GUI 에 상속됨</p> <p>[그림 15] 강의 교수자 화면 예시</p> 
데이터 형태	강의 정보와 학생 정보를 JSON 형식으로 전달
종료 동작	*상기 [표 3]의 웹 서비스 GUI 에 상속됨

[표 15] 사용자 GUI(12) 강의 교수자 화면 오류

이름	사용자 GUI 12. 강의 교수자 화면 오류
목적 및 설명	상기 사용자 GUI 11. 강의 교수자 화면에서 학생 정보를 등록하거나 강의 정보를 등록할 때 올바르지 않은 정보가 서버로 전달되었다면 발생한다.
입출력 형태	서버로부터 HTTP Response 를 수신하고, 해당 내용이 오류에 해당하는 경우 GUI 11 의 JavaScript 에 의해 팝업창이 발생하며, 팝업창의 내용은 수신한 POST Response 의 Content 에 의해 결정된다. 팝업창에 같이 표시되는 버튼을 클릭하여 창을 닫고 등록 화면 GUI 로 돌아갈 수 있다. 발생할 수 있는 오류는 다음과 같다. 1. 학생의 ID 가 중복됨 2. 강의의 ID 가 중복됨 *이외의 동작은 상기 [표 3]의 웹 서비스 GUI 에 상속됨
범위/정확도/오차범위	*상기 [표 4]의 웹 서비스 GUI 로그인 화면과 동일함
단위	화면
시간/ 속도	인터페이스 표기: 상기 [표 3]의 웹 서비스 GUI 에 상속됨 HTTP POST 요청 처리: 서버 처리 시간에 상속됨

[표 16] 사용자 GUI(13) 강의 교수자 과제 관리 화면

이름	사용자 GUI 13. 강의 교수자 과제 관리 화면
목적 및 설명	강의 교수자 화면에서 과제 관리 버튼을 클릭할 시 접근 가능
입출력 형태	서버로부터 수신한 Html, Css, JavaScript 에 근거하여 섹션을 렌더링한다. 해당 강의의 과제 정보를 작성한 뒤 서버로 전달 *이외의 동작은 상기 [표 3]의 웹 서비스 GUI 에 상속됨
범위/정확도/오차범위	*상기 [표 3]의 웹 서비스 GUI 에 상속됨
단위	화면
시간/ 속도	인터페이스 표기: 상기 [표 3]의 웹 서비스 GUI 에 상속됨 HTTP POST 요청 처리: 서버 처리 시간에 상속됨
다른 입출력 간 연관성	*상기 [표 3]의 웹 서비스 GUI 에 상속됨
화면의 형태	아래의 화면과 같은 형태를 띤다. 1. 해당 강의의 과제 목록 2. 문제 정보를 입력받기 위한 입력창 3. 제약 사항을 입력받기 위한 입력창 4. 정답 코드가 작성될 입력 창 5. 마감 기한을 설정할 수 있는 공간 6. 테스트 케이스를 추가할 수 있는 공간 입력에 따른 출력과 배점, 공개 여부를 설정해야함 *세부적인 사항은 상기 [표 3]의 웹 서비스 GUI 에 상속됨 [그림 16] 강의 교수자 과제 관리 화면 예시

이름	사용자 GUI 13. 강의 교수자 과제 관리 화면
데이터 형태	과제 정보를 JSON 형식으로 전달
종료 동작	*상기 [표 3]의 웹 서비스 GUI에 상속됨

3.1.2. Hardware Interfaces

[표 17] 브라우저 사용에 필요한 하드웨어 인터페이스

이름	Chrome 브라우저에 사용 가능한 하드웨어 인터페이스
Graphic Driver	Graphic Processing Unit Specific Driver
HIDs Driver	Mouse / Keyboard Specific Driver
NIC Driver	Network Interface Chip Driver

* 본 서비스 이용을 위해 의존성이 있는 프로그램 외 별도의 요구사항은 없다.

3.1.3. Software Interfaces

[표 18] 서비스 서버 소프트웨어 인터페이스

이름	서버 구축에 필요한 소프트웨어 인터페이스	
목적 및 설명	사용자의 로그인 및 등록을 처리하며 문제 목록 관리 및 문제 선택, 코드 작성, 채점, 분석을 제공하기 위해 서버가 요청하는 소프트웨어 패키지 인터페이스이다.	
NodeJS	설명	서비스 제공을 위한 웹 어플리케이션 엔진

이름	서버 구축에 필요한 소프트웨어 인터페이스	
18.12.0~	입력	서버 동작 구현을 위한 JavaScript 파일 및 Express 리소스
	출력	외부 접근 가능한 Html, Css, JavaScript 등 렌더링 자원
Django 3.2~	설명	서비스 제공을 위한 서버 엔진
	입력	서버 동작 구현을 위한 Python 파일 및 사용자 요청
	출력	사용자의 요청을 처리한 결과 값
SQLite3 3.39.4~	설명	사용자 데이터 보관 및 처리를 위한 데이터베이스 엔진
	입력	사용자 데이터 처리를 위한 질의문
	출력	결과 반영 혹은 질의 응답
unittest	설명	사용자의 입력한 코드로 테스트 케이스를 수행하기 위해 사용
	입력	사용자가 입력한 코드
	출력	기대 출력, 실제 출력
pip: copydetect 0.4.2	설명	사용자가 입력한 코드의 표절률을 측정하기 위해 사용
	입력	사용자가 입력한 코드
	출력	표절률
pip: multimetrix 1.3.0	설명	사용자가 입력한 코드의 효율을 측정하기 위해 사용
	입력	사용자가 입력한 코드
	출력	LOC, Halstead, Data flow 복잡도, Control flow 복잡도
pip: pylama 8.4.1	설명	사용자가 입력한 코드의 가독성을 측정하기 위해 사용
	입력	사용자가 입력한 코드
	출력	가독성 점수
pip: openai 0.24.0	설명	사용자가 입력한 코드에 대한 설명을 출력하기 위해 사용
	입력	사용자가 입력한 코드
	출력	입력된 코드에 대한 설명

[표 19] 서비스 사용자 소프트웨어 인터페이스

이름	서비스 이용을 위한 사용자 소프트웨어 인터페이스	
목적 및 설명	사용자가 본 서비스를 이용하기 위한 필수적, 선택적 소프트웨어 인터페이스이다.	
브라우저 Chrome 100~	설명	사용자가 본 제품과 상호작용하기 위한 기본 환경
	입력	HID 기반 사용자 입력
	출력	Html, Css, JavaScript 렌더링 페이지
※Python 3.10.x~	설명	사용자가 학습 내용을 실습해보기 위한 환경
	입력	사용자 입력 코드
	출력	실행 결과 및 오류 보고
SSH Client Latest Ver.	설명	서비스 관리자가 서버 관리를 위해 사용하는 서버 접속 통신
	입력	서비스 서버 내 TTY Shell 로그인 및 입력
	출력	서비스 서버 내 TTY Shell 출력
※Windows Terminal Latest Ver.	설명	사용자가 Python, SSH 를 사용하기 위한 Windows CLI
	입력	사용자 운영체제 내 TTY
	출력	명령 및 프로그램 실행 결과

* ※표기된 소프트웨어 인터페이스는 필수 요소가 아니다.

3.1.4. Communication Interfaces

[표 20] 로그인 HTTP 요청 형태

이름	로그인 HTTP 요청
용도	사용자가 서비스 이용을 위해 로그인 할 때 서버로 전송하는 요청
Method	POST
URL	main.com/login
Content Type	text/application
내용	사용자 계정 및 비밀번호 *비밀번호를 직접 전송하지 않고 계정명과 합친 해시값을 보낸다
서버 동작	Sqlite 를 활용하여 로그인 가능 여부 판단 로그인 된 경우 관리자 여부를 판단하여 페이지 응답
인코딩	Base64 인코딩 및 전송

[표 21] 계정 등록 HTTP 요청 형태

이름	계정 등록 HTTP 요청
용도	사용자가 서비스 이용을 위해 계정 등록할 때 서버로 전송하는 요청
Method	POST
URL	main.com/signup
Content Type	text/application
내용	사용자 계정 및 비밀번호, 이메일 *비밀번호 2 차 확인 과정은 사용자 JavaScript 수준에서 이루어진다.
서버 동작	Sqlite 를 활용하여 가입 가능 여부 판단
인코딩	Base64 인코딩 및 전송

[표 22] 강의/과제 목록 HTTP 요청 형태

이름	실습 코드 실행 HTTP 요청
용도	강의 목록과 강의 별 과제 목록을 가져올 때 서버로 전송하는 요청
Method	GET
URL	main.com/problem/list
Content Type	text/application
내용	사용자의 고유 ID 사용자가 수강 중인 강의명 강의별 과제 목록
서버 동작	Sqlite 를 활용하여 데이터 수집 및 응답
인코딩	Base64 인코딩 및 전송

[표 23] 과제 정보 HTTP 요청 형태

이름	과제 정보 HTTP 요청 형태
용도	과제의 상세 정보를 가져오기 위해 서버로 전송하는 요청
Method	GET
URL	main.com/problem/detail/pid
Content Type	JSON
내용	과제의 문제, 참조/제약 사항, 마감 일자, 테스트 케이스, 스켈레톤 코드, 정답 코드
서버 동작	사용자가 요청한 정보를 전달

이름	과제 정보 HTTP 요청 형태
인코딩	별도 인코딩 없음

[표 24] 과제 채점 HTTP 요청 형태

이름	과제 채점 HTTP 요청 형태
용도	해당 과제에 대해 사용자가 작성한 코드 채점 요청
Method	POST
URL	main.com/problem/grad/pid
Content Type	JSON
내용	사용자가 작성한 코드 총점, 각 테스트 케이스의 통과 여부, 공개 테스트 케이스의 정답
서버 동작	unittest 를 통한 코드 채점 후 결과 반환
인코딩	별도 인코딩 없음

[표 25] 과제 제출 HTTP 요청 형태

이름	과제 제출 HTTP 요청 형태
용도	해당 과제에 대해 사용자가 작성한 코드 제출 요청
Method	POST
URL	main.com/problem/submit/pid
Content Type	JSON
내용	사용자가 작성한 코드 작성된 코드에 대한 표절률 점수, 코드 설명, 추천 학습 자료
서버 동작	copydetect, multimetric, pylama 를 통한 코드 채점 후 결과 반환 openAI codex 를 통한 코드 설명 반환
인코딩	별도 인코딩 없음

[표 26] 강의 등록 HTTP 요청 형태

이름	강의 등록 HTTP 요청 형태
용도	관리자가 새로운 강의를 등록하고자 할 때 요청
Method	POST
URL	admin.com/lecture/create/lectureid
Content Type	JSON
내용	강의명, 강의번호

이름	강의 등록 HTTP 요청 형태
서버 동작	Sqlite 를 활용하여 강의 등록 가능 여부 판단
인코딩	별도 인코딩 없음

[표 27] 학생 등록 HTTP 요청 형태

이름	학생 등록 HTTP 요청 형태
용도	강의를 수강할 학생을 등록하고자 할때 요청
Method	POST
URL	admin.com/lecture/lectureid/student
Content Type	JSON
내용	강의명, 강의번호, 학생 ID
서버 동작	Sqlite 를 활용하여 학생 등록 가능 여부 판단
인코딩	별도 인코딩 없음

[표 28] 과제 등록 HTTP 요청 형태

이름	과제 등록 HTTP 요청 형태
용도	해당 강의에 과제를 등록하고자 할 때 요청
Method	POST
URL	admin.com/lectureid/problem/create/pid
Content Type	JSON
내용	과제 번호, 문제, 참조/제약 사항, 마감일, 스켈레톤 코드, 정답 코드, 테스트케이스
서버 동작	Sqlite 를 활용하여 과제 등록
인코딩	별도 인코딩 없음

3.2. Functional Requirements

3.2.1. Use Case

[표 29] 사용례 (1) 로그인

이름	로그인
행위자(Actor)	사용자
설명	사용자는 사전에 등록한 자신의 계정으로 서비스를 이용할 수 있다. 자신의 계정으로 서비스를 활성화하기 위해선 웹 로그인 창에 관련 정보를 입력하고 POST 메시지를 전송해야 한다. 이후 서버는 해당 값을 검증하고 로그인 여부를 판단한다.
정상 흐름	①사용자가 이미 등록된 계정과 비밀번호를 입력한다. ②계정명과 비밀번호 해시값을 HTTP POST 메시지로 전송한다. ③서버는 수신한 데이터를 Sqlite3 데이터베이스 질의로 검증한다. ④-A.사용자의 계정에 문제가 없을 경우 홈 화면을 띄운다. ④-B.사용자의 계정이나 서버에 문제가 있을 경우 팝업창을 통해 재시도를 요구한다.
전제 조건	서버에 100 개 이하의 세션이 유지되어 충분한 공간이 있다. 사용자는 사전에 계정과 비밀번호를 등록했다.
사후 조건	로그인 성공 시 계정과 세션을 결부하기 위해 브라우저 Cookie 를 수정한다.
가정	없음

[표 30] 사용례 (2) 계정 등록

이름	계정 등록
행위자(Actor)	사용자
설명	사용자는 서비스를 이용하기 위해 반드시 계정을 생성해야 한다. 각 계정은 이메일, 계정명, 비밀번호 3 개의 정보로 생성 가능하며 비밀번호는 사용자 브라우저의 javascript 를 이용하여 비밀번호 재입력 값과 비교한다. 실질적으로 계정을 생성하는 과정에서는 서버에 HTTP POST 요청을 전송하며 그 결과 혹은 에러 여부를 사용자에게 전달한다.
정상 흐름	①사용자가 생성을 원하는 계정명과 비밀번호, 이메일을 입력한다. ②비밀번호 재입력을 요구한다. 1 번의 입력과 다를 경우 재시도를 요구한다. ③계정명, 비밀번호의 해시값, 이메일 세 정보를 HTTP POST 요청으로 전송한다. ④서버는 수신한 데이터를 Sqlite3 데이터베이스 질의로 중복 검증한다. ⑤-A.사용자의 계정에 문제가 없을 경우 새로운 사용자를 추가한다.

이름	계정 등록
	⑤-B.사용자의 계정명이나 이메일이 중복되었을 경우 오류 메시지를 출력하고 재시도를 요구한다.
전제 조건	사용자는 본 서비스에 처음 접근한다. 동일한 이메일로 다른 계정이 존재하지 않는다.
사후 조건	계정 생성 시 User 테이블에 사용자를 추가한다.
가정	없음

[표 31] 사용례 (3) 과제 출제

이름	학습 자료 선택 및 수강
행위자(Actor)	사용자(교수자)
설명	교수자는 자신이 담당하는 수업에 프로그래밍 과제를 출제할 수 있다.
정상 흐름	①사용자는 과제를 출제할 강의를 선택한다. ②과제 제출 페이지로 이동하여 과제명, 본문, 마감일, 스켈레톤 코드, 관련자료, 테스트케이스를 입력한다. ③과제 정보를 HTTP POST 메시지로 전송한다. ④서버는 전송받은 과제 정보를 추가한다.
전제 조건	Enrollment 테이블에 교수자에 대한 정보가 있다.
사후 조건	Assignment 테이블에 과제 정보를 추가한다.
가정	없음

[표 32] 사용례 (4) 과제 수행

이름	관리자 페이지 접근
행위자(Actor)	사용자 (수강생)
설명	수강생은 자신이 수강하는 강의의 과제를 확인하고 웹 브라우저에서 코드를 작성하여 테스트케이스를 통해 점수를 확인하고 코드를 제출할 수 있다.
정상 흐름	①사용자는 과제를 수행할 강의를 선택한다. ②코드를 작성하기 위한 메인 화면으로 이동하여 헤더 섹션에서 과제를 선택한다. ③과제명을 통해 과제의 정보를 HTTP GET 으로 요청한다.

이름	관리자 페이지 접근
	④ 해당 과제의 본문, 마감일, 스켈레톤 코드, 관련자료, 테스트케이스를 메인 화면의 각 섹션에 띄운다. ⑤ 코드 에디터 섹션에서 코드를 작성한다. ⑥ 작성한 코드는 중간저장이 가능하며 테스트케이스를 통해 검증할 수 있다. 작성이 완료된 코드는 제출할 수 있다. ⑦ 제출된 코드는 HTTP POST 메시지로 전송된다.
전제 조건	사용자가 수강하는 강의의 수강생으로 저장되어 있고, 마감일이 지나지 않은 과제가 존재한다.
사후 조건	사용자가 작성한 코드가 Code 테이블에 추가된다.
가정	없음

[표 33] 사용례 (5) 성적 확인(교수자)

이름	학습 자료 선택 및 수강
행위자(Actor)	사용자(교수자)
설명	사용자는 자신이 출제한 과제에 대해 수강생이 제출한 코드와 점수를 확인할 수 있다.
정상 흐름	① 사용자는 성적을 확인할 강의를 선택한다. ② 관리 페이지로 이동하여 성적을 확인할 과제를 선택한다. ③ 과제명을 HTTP GET 메시지로 전송한다. ④ 서버로부터 받은 성적 정보를 화면에 띄운다.
전제 조건	사용자(수강생)의 점수와 코드를 데이터베이스에서 확인할 수 있다.
사후 조건	없음
가정	없음

[표 34] 사용례 (6) 성적 확인(수강생)

이름	학습 자료 선택 및 수강
행위자(Actor)	사용자(수강생)
설명	사용자는 자신이 제출한 과제의 점수 기록을 확인할 수 있다.
정상 흐름	① 사용자는 성적을 확인할 강의를 선택한다. ② 메인화면으로 이동하여 성적을 확인할 과제를 선택한다.

이름	학습 자료 선택 및 수강
	③사용자의 과제 정보를 HTTP GET 메시지로 요청한다. ④과제 설명, 테스트케이스, 사용자가 제출한 코드, 채점된 점수를 띄운다.
전제 조건	과제 정보와 사용자의 점수, 코드를 데이터베이스에서 확인할 수 있다.
사후 조건	없음
가정	없음

[표 35] 사용례 (7) 강의 등록

이름	학습 자료 선택 및 수강
행위자(Actor)	관리자
설명	시스템 관리자는 해당 학기에 성균관대학교에서 진행되는 강의 정보를 추가하고 지난 학기 강의 정보를 수정할 수 있다.
정상 흐름	①관리자는 관리자 페이지에서 강의 수정을 선택한다. ②강의 수정 페이지에서 해당 학기에 진행되는 강의 정보를 입력한다. ③강의 정보를 HTTP POST 메시지로 전송한다. ④서버는 받은 정보를 저장한다.
전제 조건	해당 학기가 지나면 Class 테이블이 초기화된다.
사후 조건	Class 테이블에 관리자가 입력한 정보가 추가된다.
가정	없음

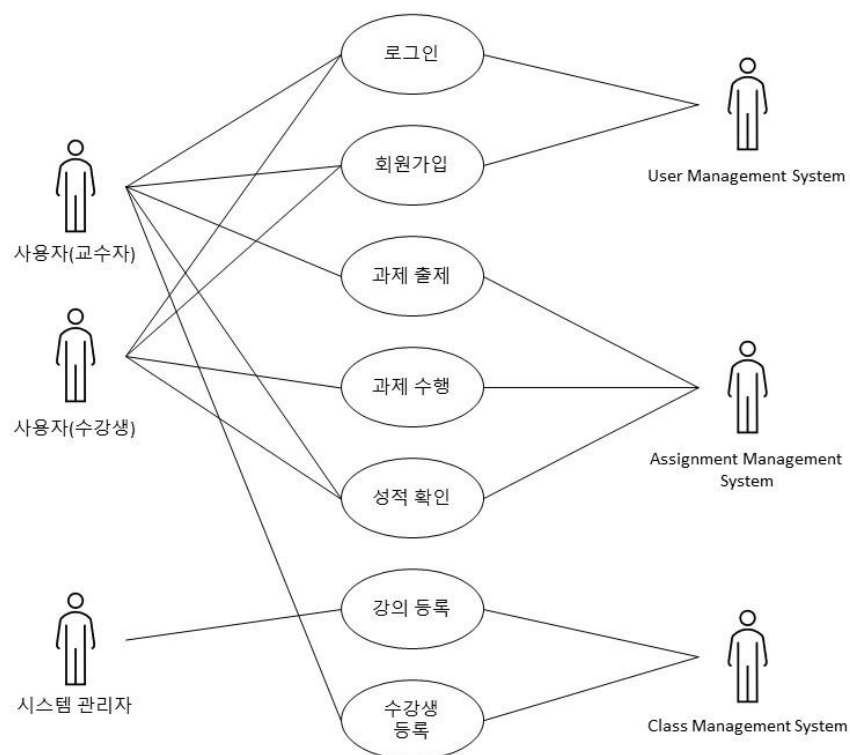
[표 36] 사용례 (8) 수강생 추가

이름	학습 자료 선택 및 수강
행위자(Actor)	사용자(교수자)
설명	교수자는 자신이 담당하는 수업에 수강생을 추가할 수 있다.
정상 흐름	①사용자는 수강생을 추가할 강의를 선택한다. ②수강생 관리 페이지로 이동하여 추가할 수강생의 정보를 입력한다. ③수강생 정보가 HTTP POST 메시지로 전달된다. ④서버에 전달된 수강생 정보가 데이터베이스에 추가된다.
전제 조건	사용자가 수강생을 추가하려는 강의의 교수자로 등록되어 있다.

이름	학습 자료 선택 및 수강
사후 조건	Enrollment 테이블에 사용자가 입력한 정보가 추가된다.
가정	없음

3.2.2. Use Case Diagram

[그림 17] Use Case Diagram



3.2.3. Data Dictionary

[표 37] 데이터베이스 Table ‘User’ 구조

Field	Key	Constraint	Description
user_id	PK	Not Null	학번
password_hash		Not Null	비밀번호
email		Not Null	이메일 정보
Admin		Boolean	초기값 False

* 새로운 사용자가 등록될 때 본 테이블에 값을 추가해야 한다.

[표 38] 데이터베이스 Table ‘Class’ 구조

Field	Key	Constraint	Description
class_id	PK	Not Null	강의의 번호
class_name		Not Null	강의명

* 시스템 관리자가 수업을 추가할 때 본 테이블에 값을 추가해야 한다.

[표 39] 데이터베이스 Table ‘Enrollment’ 구조

Field	Key	Constraint	Description
user_id	PK	Not Null	사용자 식별자
class_id	PK	Not Null	강의 식별자
is_student		Boolean	초기값 True 학생과 교수자 식별

* 사용자가 관련된 수업을 선택했을 때 테이블에 값을 추가해야 한다.

[표 40] 데이터베이스 Table ‘Assignment’ 구조

Field	Key	Constraint	Description
assign_id	PK	Not Null	과제 식별자
class_id		Not Null	강의 식별자
name		Not Null	과제명
content		Not Null	과제 설명명
due_date		Not Null	과제 마감일
skeleton code			스켈레톤 코드
related			관련 자료

[표 41] 데이터베이스 Table ‘Code’ 구조

Field	Key	Constraint	Description
user_id	PK	Not Null	사용자 식별자
assign_id	PK	Not Null	과제 식별자
code1			중간저장 코드 1
code2			중간저장 코드 2
code3			중간저장 코드 3
final_code			제출 코드

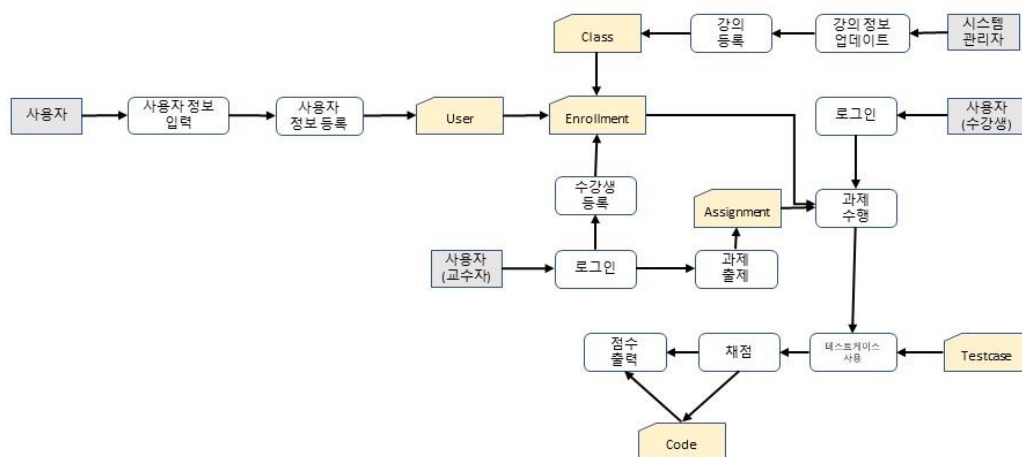
Field	Key	Constraint	Description
score		Not Null	초기값 0

[표 42] 데이터베이스 Table ‘Testcase’ 구조

Field	Key	Constraint	Description
assign_id	PK	Not Null	과제 식별자
testcase1			테스트케이스 1
testcase2			테스트케이스 2
testcase3			테스트케이스 3
testcase4			테스트케이스 4
testcase5			테스트케이스 5
hidden		boolean	초기값 False

3.2.4. Data Flow Diagram

[그림 18] Data Flow Diagram



3.3. Performance Requirements

다음 항목들은 본 서비스가 제공해야 할 성능의 요구조건을 나타낸다.

3.3.1. Static numerical requirement

- 본 시스템은 사용자가 복수의 코드 검증(실습 실행)을 수행하는 것을 제공하지 않는다. 각각의 사용자는 세션 및 쿠키를 통해 구분된다.
- 사용자의 코드 검증은 특별한 실습 조건이 아닌 한 5 초의 실행 제한시간을 가지며 그 시간을 벗어나면 시간 초과 실행으로 전달한다.
- 본 시스템은 하드웨어 요구조건을 만족하는 한, 브라우저 및 응용 프로그램에서 연결 종료라고 판단하는 TTL 이전에 정보를 전달해야 한다.

3.3.2. Dynamic numerical requirement

- 본 시스템은 복수의 관리자가 관리할 수 있으며, 루트 권한의 계정으로 접속하여 동시 작업이 가능하다.
- 본 시스템은 최대 100 명의 동시 사용자를 가정하고 있으며, 그 이상의 인원에 대해서는 최적의 품질을 제공하기 어렵다.
- 상기 실행시간 5 초에 근거하여, 각 실습의 검사 시간은 테스트 케이스의 수 X 5 초 안에 종료되어야 한다.
- 페이지 이동 등 데이터베이스에서 자료를 찾지 않고, 데이터베이스를 수정하지 않는 동작은 5 초 안에 수행되어야 한다.
- 데이터베이스에서 자료를 탐색하는 과정을 동반하는 각 기능은 10 초 이내에 수행되어야 한다.

3.4. Logical Database Requirements

본 시스템은 SQLite 데이터베이스를 사용한다. 서버와의 연동을 위해 NodeJS 의 SQLite3 플러그인을 사용한다. 사용자 정보, 강의 정보를 데이터베이스에 저장한다. 사용자가 추가될 때마다 사용자 테이블에 새로운 레이블이 생성된다. 코드랑 관련 테스트 케이스가 생성될 때마다 코드 테이블에 새로운 레이블이 생성이 된다. 코드 학습 여부 기록 테이블에서는 임의의 사용자가 임의의 코드를 마칠 때마다 그 순서쌍을 레이블의 형태로 저장을 하며, 코드가 미완성인 경우에는 임시저장을 하는데, 사용자의 코드 임시저장과 관련한 부분은 브라우저에 저장하는 것으로 한다. SQLite 데이터베이스는 본 시스템을 위해 필요한 모든 기능을 큰 성능 부하 없이 제공한다.

3.5. Design Constraints

본 시스템은 Apache2 라이선스를 따른다. NodeJS 기반의 웹 서비스를 제공한다. 서버는 사용자에게 교육 자료를 제공한다. 사용자가 입력한 코드를 서버가 수행하고 결과를 반환할 수 있다. 본 시스템은 Django, React, NodeJS, SQLite, openAI codex 를 기반으로 설계되었다.

3.6. Standards compliance

시스템 내의 모든 JavaScript 는 ECMAScript 표준을 따른다. NodeJS 구현 및 사용자에게 전달되는 JavaScript 양 측에 해당되는 내용이다. CSS 는 W3C 의 표준을 따른다. 사용자에게 전달되는 HTML 의 경우 NodeJS 의 템플릿 엔진 Express 를 활용한다. Express 표준과 HTML5 표준이 상충할 경우 Chrome 브라우저에서 해석하는 것을 기준으로 한다. 동일한 경우 HTML5 의 내용을 따른다.

서비스 가동 중 발생하는 모든 네트워크 통신은 TCP, HTTP 프로토콜 표준을 따르며 각 항목은 RFC 793, RFC 2616 에 명시된 바를 따른다.

3.7. Software System Characteristics

소프트웨어 시스템 특성은 비 기능적 요구조건으로부터 발생한다. 이 항목에서는 본 시스템의 비 기능적 요구조건을 소개한다. 비 기능적 요구조건은 제품 요구조건(Product Requirements), 조직 요구조건(Organization Requirements), 외부 요구조건(External Requirements) 3 개로 구분한다.

3.7.1. Product Requirements

제품 요구조건 항목은 본 시스템의 동작 중에 지켜져야 하는 요구조건들을 뜻한다. 본 시스템은 운영 중 다음과 같은 요구조건을 충족해야 한다.

3.7.1.1. Usability Requirements

본 시스템은 사용자에게 사용성 좋은 시스템을 제공해야 한다. 접근성이 좋은 브라우저 기반의 웹 어플리케이션으로 개발되었기 때문에, 사용자가 본 시스템을 사용할 때 별도의 설치 과정 및 요구 조건을 가지지 않는 것은 본 시스템의 주요 요구조건이다. 사용자가 본 시스템을 이용하기 위해 브라우저 이외의 도구를 사용하게 되는 것을 지양한다. 또한 본 시스템에서 제공하는 모든 교육자료는 관련 전문지식이 충분하지 않은 사람일지라도 이해할 수 있도록 설계되어 있다. 만일 충분한 배경지식이 있어야만 이해할 수 있는 내용이라면, 사용자(교수자)는 배경 지식에 해당하는 교육 자료 또한 적절히 구성하고 사용자(수강생)이 외부 서비스의 도움 없이 제공되는 내용을 이해할 수 있도록

설계해야 한다. 코드 작성을 위한 설명은 openAI codex 를 기반으로 제공이 된다. 마지막으로 제출이 완료되었으면, 문제와 관련된 학습 자료를 추천해준다.

3.7.1.2. Performance Requirements

본 시스템은 세션과 계정 기반으로 사용자를 구분할 수 있다. 사용자가 서버에 코드 검증을 요청하는 형태임을 고려하여 본 서비스는 서로 다른 100 명의 사용자 이상의 접근을 지양한다. 만일 더 큰 규모로 서비스를 확장할 경우 서버는 하드웨어, 소프트웨어 측면에서 개선이 필요할 수 있다. 또한 상대적으로 과부하의 대상이 되기 쉬운 웹 서비스 특성 상, 본 시스템은 한 사람이 동시에 여러 코드 검증을 시도하는 것을 허락하지 않는다.

상기한 요구 조건이 충족된다면, 본 시스템은 제공되는 모든 서비스를 브라우저 환경에서 무리 없이 사용할 수 있는 수준까지의 속도를 보장한다. 각각의 페이지 이동마다 5 초, 자료 요청 및 서버 자료 수정 등에 대해서는 10 초 이상의 지연이 발생하지 않도록 설계한다.

3.7.1.3. Security Requirements

본 서비스 이용 도중, 사용자는 서버에 임의의 데이터를 보낼 수 있다. 그 과정은 TCP 프로토콜을 기반으로 충분히 안전하지만, MITM 중간자 공격을 고려하여 사용자 데이터는 비식별화하여 전송한다. 사용자의 정보 보호 이외에도, 웹 서비스는 각 사용자에게 의해 데이터 유출 공격의 대상이 될 수 있다. 이러한 상황을 방지하기 위해 서버는 사용자가 전송하는 어떠한 HTTP 요청의 값도 신뢰하지 않고 검증한다. 또한 관리자 권한이 필요한 모든 동작에 대하여 세션 및 쿠키 기반으로 올바른 접근을 수행하고 있는지 확인한다.

3.7.2. Organizational Requirements

조직 요구조건은 사용자 집단 및 개발자 집단의 규제 및 절차에 의해 생성된 요구 조건을 뜻한다.

3.7.2.1. Environmental Requirements

- 본 시스템의 최종 라이선스인 Apache2 라이선스에 위배되거나 충돌하지 않는 한, 개발자는 어떠한 오픈 소스 API 든 활용할 수 있다. 단, 이 경우 본 요구사항 명세서를 적절히 수정하여야 한다.
- 개발자는 시스템 관리자가 데이터베이스, Python, Django, NodeJS 및 Express 등 개발에 쓰인 엔진에 대한 기본적인 지식을 가지고 있다고 가정할 수 있다. 본 프로그램을 작동하는 서버에서 루트 권한의 계정을 시스템 관리자가 접속할 수 있다면, 별도의 시스템 관리 어플리케이션 없이 서버를 수정할 수 있다.

3.7.2.2. Operational Requirement

- 본 서비스는 본 문서에서 정해진 범주 내에서 서비스 이용에 장애가 발생하지 않아야 한다.
- 사용자는 본 서비스를 통해 두 가지 목적을 이룰 수 있다. 첫째, 교수자는 예시 문제를 학생에게 제공할 수 있으며, 학생들이 작성한 코드를 평가할 수 있다. 둘째, 학생들은 교수자가 제공한 문제에 대한 코드를 작성한 후 평가받을 수 있다.

3.7.3. External Requirements

본 항목에서는 시스템의 운영과 개발에 있어 외부적으로 발생할 수 있는 요구조건을 서술한다.

3.7.3.1. Safety / Security Requirement

- 이 시스템은 각 개인정보가 충분히 비식별화되어, 서버에게 전송될 때 중간자 공격으로부터 원천적으로 자유롭게 설계되었다.
- 비식별화 과정에서 사용자의 비밀번호와 계정을 같이 해시했기 때문에, 레인보우 공격으로부터 안전하다.
- 이 시스템은 세션 및 브라우저 쿠키를 바탕으로 사용자를 구분한다. 브라우저 쿠키는 외부로 유출되지 않아야 한다.
- 사용자가 서버로 보내는 모든 코드는 실행할 때 완전히 격리된 환경에서 수행되어야 한다.

3.7.3.2. Regulatory Requirement

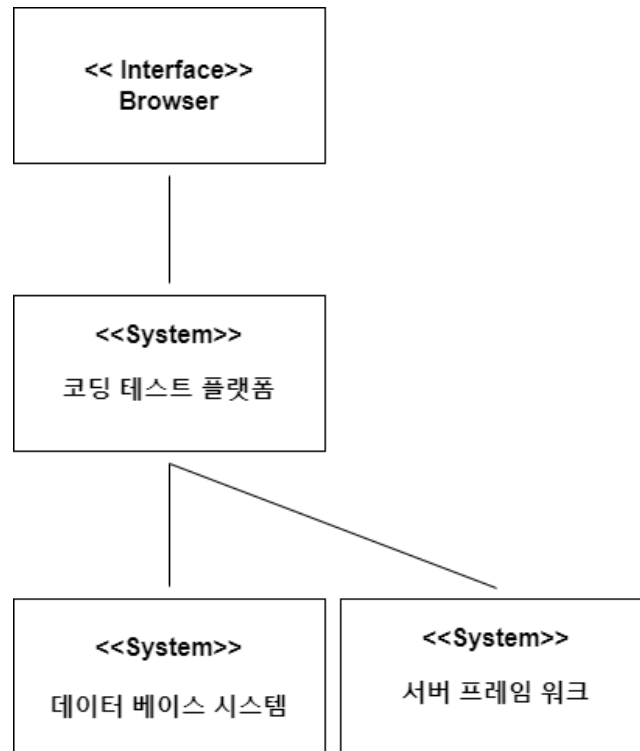
- 본 시스템에서 이용하는 React 및 python library 등 외부 API의 라이선스를 철저히 지켜야 한다. 만일 라이선스가 충돌할 경우 대체를 찾아야 한다.

3.8. Organizing the Specific Requirements

본 항목에서, 우리는 시스템 모델을 Unified Modeling Language(UML)에 기반한 가시적인 형태와 표를 기반으로 나타내고자 한다. 이러한 모델은 시스템, 서브 시스템, 시스템 내의 각 요소와 운영 환경 등 세부적인 요구사항의 형태를 명확하게 나타낼 수 있다.

3.8.1. Context Model

[그림 19] Context Model



3.8.2. Process Model

[그림 20] Activity Diagram



3.8.3. Interaction Model

3.2.2 항목의 User Case Diagram 형태로 표현되었다.

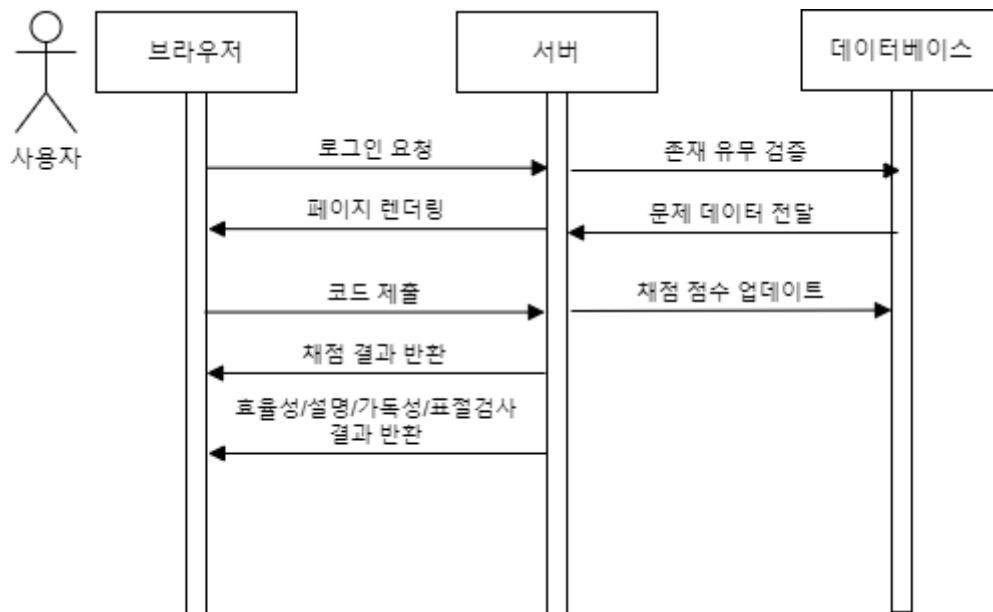
3.8.4. Behavior Model

3.8.4.1. Data Flow Diagram

3.2.4 항목에 Data Flow Diagram 형태로 표현되었다.

3.8.4.2. Sequence Diagram

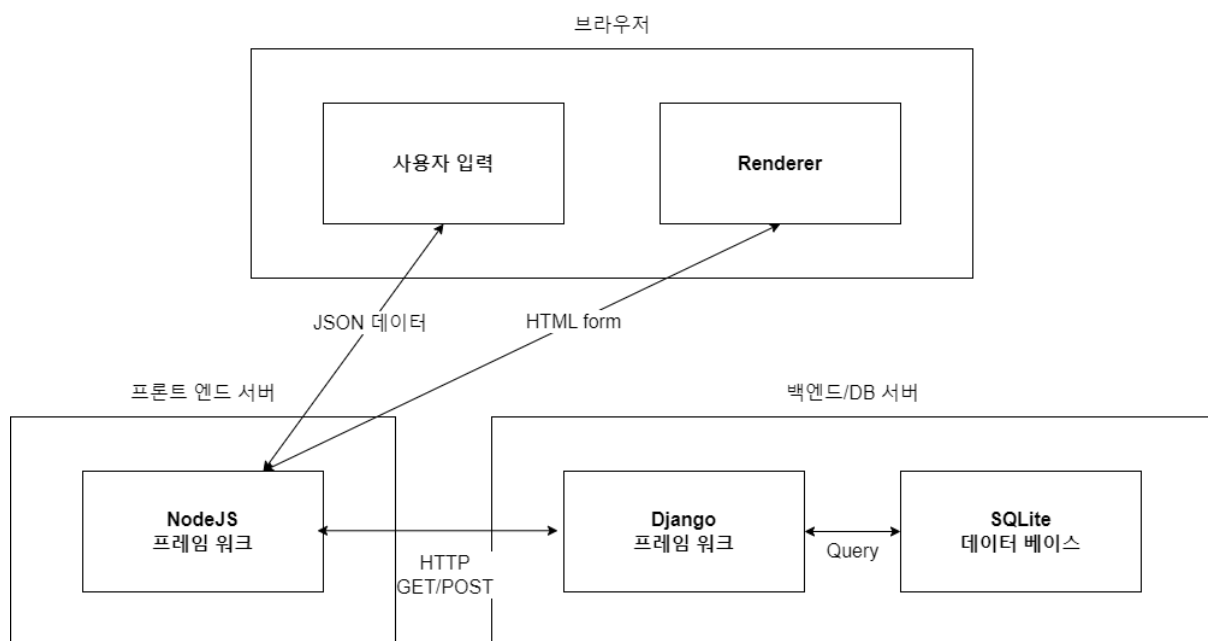
[그림 21] Sequence Diagram



3.9. System Architecture

본 항목은 시스템 구조를 고수준의 추상화된 도식 형태로 나타낸다. 시스템을 구성하는 각 요소 (사용자 브라우저, 서버)간 존재하는 기능과 통신에 대해 가장 근본적인 형태가 나타난다.

[그림 22] 시스템 구조



3.10. System Evolution

본 항목에서 우리는 시스템이 추후 하드웨어 개선 및 소프트웨어 개선, 혹은 새로운 요구사항이 발생할 때 대처하기 위한 기본적인 입장을 서술한다. 이 항목은 추후 본 소프트웨어의 방향성을 서술하는 것이기 때문에 시스템 디자이너는 이를 고려하여 시스템을 설계하여야 한다.

3.10.1. Limitation and Assumption

본 서비스는 브라우저 기반으로 사용자가 Python 프로그래밍을 연습할 수 있는 환경을 제공한다. 본 제품을 출시하는 시점에는 “Python 코딩 채점” 및 “Python 코드 분석과 피드백” 2 가지의 대분류밖에 없지만, 추후 시스템 관리자와의 협의를 통해 다양한 언어 지원이 가능해질 예정이다. 본 서비스는 다수의 사용자가 부하가 큰 여러 프로세스를 진행하는 극한의 상황을 가정하여 미들웨어와 리버스 프록시 엔진을 활용한 설계를 하였으며 다수의 사용자가 복잡한 문제를 풀 수 있도록 설계되었다.

3.10.2. Evolutions of Hardware and Change of User Requirements

본 서비스는 동시 접속자의 수를 100 명으로 제한하고 한 사람당 수행할 수 있는 코드 검증의 수를 제한하였다. 추후 하드웨어 및 소프트웨어가 개선되더라도 멀티 쓰레딩에 최적화 되어있지 않은 Python 의 언어적 특성상 멀티 쓰레딩이 어려울 수 있다. 이를 해결하기 위해 미들웨어 uwsgi 와 리버스 프록시 엔진 nginx 를 활용해 다수의 요청을 효과적으로 분산할 수 있도록 설계하였다. 관리자 및 설계자는 파이썬 백 엔드가 갖는 한계점이 오는 시점을 적절히 파악하여 비용 및 서비스 품질을 개선해야 한다.

사용자 요청 사항이 발생할 경우, 만일 새로운 기능을 개발하는 것이 아니라 단순히 내용물의 추가하는 것이라면 시스템 관리자 선에서 해당 내용을 추가할 수 있다. 새로운 기능을 개발해야 하는 경우, 본 요구사항 명세서에 있는 내용들을 기반으로 새로운 API 를 추가하거나 새로운 웹 서비스 페이지를 설계할 수 있다. Django 의 기능의 경우 Apps 와 Modules 내부의 Class 들에서 기능들이 관리되므로 쉬운 추가와 발전이 가능하다. 단 각 설계 단계에 있어 기존 요구사항 명세와 통일된 양식을 가져야 하며, 작성 후 본 문서 또한 수정되어야 한다.

4. Supporting Information

4.1. Software Requirement Specification

본 소프트웨어 요구 명세서는 IEEE 권장사항에 맞추어 작성되었다. 본 서비스에 적합한 요구 조건을 작성하기 위해 본래의 양식에서 일부 추가되거나 제외된 바 있다. (IEEE Recommended Practice for Software Requirements Specifications, IEEE-Std-830)

4.2. Document History

[표 43] 문서 이력

Date	Version	Description	Writer
2022.10.28.	1.0	팀원 회의 후 최초 버전 발행	Siyeol Choi, et al.