성균관대학교
SUNG KYUN KWAN UNIVERSITY

CodingCAT: 온라인 코딩 테스트 플랫폼

소프트웨어 요구사항 명세서

2022.10.30

소프트웨어공학개론
41분반 Team 2

Team Leader 박승호
Team Member 김동우
신주영
어영준
장병우
한태욱

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. Introduction

1.1. Objective

본 문서는 성균관대학교 학생을 위한 코딩 테스트 플랫폼, CodingCAT 서비스를 제공하기 위한 요구 사항 명세서이다. 이 서비스는 성균관대학교 이은석 교수님의 2022년 2학기 소프트웨어공학개론 강의 41분반의 2조에 의해 고안되고 개발된다.

본 문서는 서비스 개발팀 6명과 강의 조교님, 강의 교수님이 본 서비스의 주요 독자이며, 예외적으로 동일 강의 수강자 또는 학습 및 교육의 용도로 열람할 수 있다. 본 문서를 재배포 및 수정하는 것에 제약은 없으나, 상업적 용도로 활용 시 반드시 개발팀의 허가를 얻어야 한다.

본 문서는 명세에 의거하여 학생을 위한 코딩 테스트 서비스를 제공하기 위해 작성되었다. 소프트웨어 엔지니어로서 문제 해결력은 가장 기본이 되며 가장 근본이 되는 요구 역량이다. 이에 따라 여러 대학과 사교육 업체에서 소프트웨어 교육을 실시하고 있다. 현 실태에 맞춰 성균관대학교 학생들의 역량과 창의성을 기르기 위해 자체적인 코딩 테스트 플랫폼을 제공하고자 한다. 제공되는 서비스를 통해 한 단계 심화된 학습과 실습을 하고, 보다 발전된 역량을 기르는 것을 목표로 한다.

본 문서는 목표로 하는 플랫폼 CodingCAT을 개발하기 위하여 클라이언트와 서버를 디자인하는 과정에서 도출한 요구 사항들을 분석 및 정리한 것이다. 이 내용을 토대로 서비스가 구현되고 제공되며 세부 사항은 이하 문서에 명시한다.

1.2. Scope

본 서비스는 성균관대학교 학생들에게 Python 언어 코딩 테스트 환경을 제공하기 위해 고안되었다. 코드 작성 역량을 키우기 위한 플랫폼 및 작성한 코드의 동작, 가독성, 효율성을 평가하는 기능을 제공한다. 또한 사용자가 쉽게 접근할 수 있는 웹 환경으로

제공하여 사용 가능성을 높인다. 이를 제공하기 위해 Javascript 기반의 프론트엔드 서버와 Python 기반의 백엔드 서버를 구축하며, 서버 배포를 위해 AWS 서비스를 활용한다.

본 서비스를 통해 사용자는 문제 해결 능력을 기를 뿐만 아니라, 교강사와 학생이 소통하는 온라인 강의 플랫폼으로 이용될 수 있다. 시스템 엔지니어는 확장성을 고려한 설계를 바탕으로, 추후 본 서비스는 부가 기능을 제공하는 애플리케이션 혹은 다양한 언어를 지원하는 교육 플랫폼으로 확장될 수 있다.

1.3. Definitions, Acronyms, and Abbreviations

Table 1. List of Terms and Definitions

Term	Definition
Django	Back-end 서버를 구축하기 위한 Python 기반의 라이브러리이다.
Frontend	소프트웨어 프로그램에서 이는 사용자와 상호작용하는 모든 요소들의 총집합을 칭한다.
Git	무료 오픈 소스 distributed version control system이다. 전세계적으로 수많은 개발 프로젝트들의 관리를 위해 널리 사용되고 있다.
Keyboard	컴퓨터의 입력 장치로서 전통적인 Typewriter Keyboard를 계승하는 장치이다. 키보드를 구성하는 버튼은 기계식 장치 또는 멤브레인 방식을 따르는 전자식 장치로 구성되어 있다.
Mouse	컴퓨터에 사용되는입력장치 중 하나이며, 2차원의 움직임을 컴퓨터에 입력할 수 있다. 이 입력을 사용해 사용자는 컴퓨터에서 다양한 동작을 수행할 수 있다.
Node.js	주로 네트워크 애플리케이션을 만들기 위한 JavaScript runtime 환경이다.
OAuth	웹, 모바일, 및 데스크탑 애플리케이션에서 표준이되는 보안 인증

Term	Definition
	프로토콜이다.
React	user interface를 만들기 위한 JavaScript 기반의 라이브러리이다.
WEB Browser	Website에 접근하기 위해 필요한 소프트웨어이다. 대표적인 예시로 Google 사의 Chrome, Mozilla사의 Firefox 등이 있다.

Table 2. List of Acronyms and Abbreviations

Acronym, Abbreviation	Description
API	Application Programming Interface
CPU	Central Processing Unit
CSS	Cascading Style Sheets
DB	Database
HTML	HyperText Markup Language
JS	Javascript
JWT	Json Web Token
MITM	Man in the Middle Attack
PC	Personal Computer
RAM	Random-access Memory
SQL	Structured Query Language
URL	Uniform Resource Locator
VCS	Version Control System

1.4. References

- IEEE, “Recommended Practice for Software Requirements Specifications”. https://ieeexplore.ieee.org/document/720574
- Team 3, “Requirements Specification Team3”. https://github.com/skkuse/2022spring_41class_team3
- 국가법령정보센터, “저작권법". https://www.law.go.kr/%EB%B2%95%EB%A0%B9/%EC%A0%80%EC%9E%91%EA%B6%8C%EB%B2%95
- 국가법령정보센터, “개인정보 보호법". https://www.law.go.kr/%EB%B2%95%EB%A0%B9/%EA%B0%9C%EC%9D%B8%EC%A0%95%EB%B3%B4%EB%B3%B4%ED%98%B8%EB%B2%95
- Facebook, “React”. https://reactjs.org/
- Django Software Foundation, “Django Project”. https://www.djangoproject.com/
- IETF, “The OAuth 2.0 Authorization Framework”. https://oauth.net/
- SQLite Team, “SQLite”. https://www.sqlite.org/

1.5. Overview

본 소프트웨어 요구사항 명세서는 열 부분 (Chapter 2~11)과 부록으로 구성되어 있다. Chapter 2에서는 제품의 관점과 기능, 예상 사용자 특징, 구동 환경 및 제약 조건에 대해 서술한다. Chapter 3에서는 서비스 기능을 위한 인터페이스에 대한 요구사항을 서술한다. Chapter 4에서는 세부적인 기능 요구 사항을 표와 도표를 이용해 서술한다.

Chapter 5~7에서는 기능 요구사항 이외의 요구 사항을 상세히 서술한다. 여기에는 제품 요구사항, 조직 요구사항, 사회적 요구사항 등을 포함된다. Chapter 8~9에서는 시스템의 동작을 추상화하여 도표를 이용해 서술한다. Chapter 10에서는 시스템의 변경 가능성 및 유지보수를 위한 사항들을 서술한다. Chapter 11에서는 변경 이록 등 본 문서에 대한 부가적인 정보를 서술한다.

2. Overall Description

2.1. Product Perspective

CodingCAT은 독립형 웹 서비스 시스템이다. 본 섹션에서는 이 제품의 추상화된 작업 흐름을 설명하여 본 서비스 사용자가 얻을 수 있는 이점에 대해 간략하게 설명한다. 이 시스템의 주요 사용자는 학생과 교수자이다. 교수자는 과제를 생성하고 해당 수업의 학생들이 문제를 해결할 수 있도록 해야 한다. 학생은 웹사이트의 코드 편집기에서 직접 코드를 생성하고 주어진 테스트 케이스를 이용하여 솔루션을 테스트할 수 있다. 코드를 제출한 후 점수가 매겨진 결과를 보거나 출력 섹션에서 발생한 오류에 대한 알림을 받을 수 있다. 이 서비스를 통해 교수가 얻을 수 있는 이점은 시간이 많이 소요되는 과제 생성, 학생 제출물 접수, 과제 채점 과정 등을 최소화의 노력으로 진행할 수 있다는 점이다. 학생들은 실시간으로 자신의 점수를 확인하고 자신의 답변을 모범 답변과 비교할 수 있다. CodingCAT은 교수와 학생 모두가 더 쉽게 과제를 할당하고 완료할 수 있는 모든 Python 관련 코딩 과제를 위한 간소화된 프로세스와 단일 플랫폼을 제공한다.

2.2. Product Functions

2.2.1. User Authentication

본 서비스는 사용자의 Github 계정을 통해 로그인할 수 있는 시스템으로 구성되어있다. 이 시스템은 사용자의 Github 계정을 사용하여 CodingCat 계정에 액세스한다. 학생 사용자 계정과 교수자 사용자 계정 간에는 차이가 있다. 교수자 계정은

학생 계정보다 더 높은 권한을 가지므로 과제 및 코스를 만들고 편집할 수 있는 권한이 부여된다.

2.2.2. Assignments

본 서비스는 교수자 사용자만이 과제를 생성 또는 삭제할 수 있도록 설정 되어있다. 학생 사용자는 문제 설명과 과제 테스트 케이스를 읽을 수 있다. 학생 사용자는 과제 정답 코드에 대한 수정 권한이 없고, 과제를 제출한 경우에만 정답 코드를 확인할 수 있다. 학생 사용자는 채점을 위해 소스 코드를 제출할 수 있다. 성공적인 제출시 학생 사용자는 전체 점수를 볼 수 있다. 전체 점수는 기능 점수, 효율성 점수 및 가독성 점수의 세 가지 점수를 합산한 것이다. 결과는 또한 백분율 형식으로 표절 점수를 표시한다.

2.2.3. Course Enrollment

학생 사용자는 강의 목록을 볼 수 있고 새로운 수업에 수강 신청을 할 수도 있다. 또한 강의의 학생 사용자는 해당 강의에 대한 더 자세한 정보를 볼 수 있다.

2.2.4. Saving/Loading User Code

학생 사용자는 저장 아이콘을 클릭하여 최대 3개의 개별 소스 코드를 저장할 수 있다. 코드 편집기에 작성된 내용은 해당 저장 번호를 기준으로 저장된다. 그리고 가져오기 아이콘을 클릭하여 저장된 항목을 가지고 올 수 있다.

2.2.5. IDE

본 서비스는 코드 편집기와 상호 작용하는 모든 사용자를 위한 온라인 IDE를 제공한다. 코드 편집기를 활용하기 위해 사용자가 설치 해야 할 필수 프로그램은 없다. 또한 사용자가 프로그램을 실행하여 실패할 때마다 실패 지점(traceback)이 강조 표시되고 오류 정보가 터미널에 출력된다. 프로그램 실행에 실패한 줄도 빨간색으로 강조 표시되어 문제가 있는 코드 줄을 나타낸다.

2.2.6. Submission Results

학생 사용자는 해당 과제에 답변을 제출한 후 제출 결과를 볼 수 있다. 제출 결과는 전체 점수와 유사도 비율 등이 백분율로 표시된다.

2.2.7. Code Explanation

학생 사용자는 OpenAI Codex를 사용하여 소스 코드가 수행하는 작업에 대한 설명을 받을 수 있다. 이 부분은 제출 결과를 조회할 때 별도의 탭으로 구분되어 있다.

2.2.8. Related References

학생 사용자는 해결하려는 문제와 관련된 참조 목록을 볼을 수 있다. 권장 연습 문제 섹션, 비디오 참조 섹션 및 추가 학습 자료 섹션에 유용한 참고자료 링크들이 표시된다.

2.3. User Characteristics

2.3.1. System Administrator

시스템 관리자는 시스템이 장기간 중단되지 않고 의도한 대로 작동하는지 확인한다. 오류가 발생하거나 버그가 감지되지 않는 한 시스템 관리자는 일반적으로 시스템과 상호 작용하지 않는다.

2.3.2. Instructor

교수자 사용자는 강의를 진행하고 과제를 내는 사용자이다. 이 사용자는 프로세스가 최소한의 상호 작용으로 강의와 과제를 생성 및 삭제할 수 있어 번거롭지 않고 편리하게 진행할 수 있다. 이 과정에서 명확하고 직관적인 UX 요소가 포함된 UI를 통해 작업할 수 있다. 직관적인 UI/UX를 통해 교수자 사용자가 강의 진행 과정함에 있어 발생할 수 있는 오류의 가능성을 줄이면, 본인에게 큰 이점이 된다.

2.3.3. Student

학생 사용자는 강의를 수강하고 과제를 해결하는 사용자이다. 본 서비스는 대학 기관에서 사용하기 위해 개발되고 있기 때문에 학생 사용자는 주로 대학생이 될 것이다. 본 시스템에서 제공하는 IDE, 자동 채점, 코드 분석 등 여러 기능을 통해 강의 및 과제를 수행함에 있어서 도움을 받을 수 있고, 더 나아가 역량을 기를 수 있다.

2.4. Operating Environment

본 시스템은 PC 내 웹 브라우저 환경에서 운영되도록 구현되었으며, 지원되는 운영체제는 아래와 같다.

- Mac OS
- Linux
- Windows XP
- Windows 7
- Windows 8
- Windows 10
- Windows 11

2.5. Design and Implementation Constraints

본 시스템은 본 문서에 기재된 요구 사항을 바탕으로 고안되고 구현된다. 본 문서에 명시되지 않은 세부 조건들은 개발자의 재량에 의해 구현되고 설계될 수 있다.

- 가능한 오픈소스를 주로 사용
- 외부 API 및 오픈소스 사용 시, 안정성, 보안성이 검증된 서비스를 사용할 것

- 서버는 유저의 모든 입력을 신뢰하지 않을 것
- 유저 비밀번호는 해시 함수를 통해 암호화 되어 저장되고 활용될 것
- 서비스 운영에 불필요한 유저 정보를 요구하지 않을 것
- 시스템 관리자의 Acceptability를 고려하여 설계 할 것
- 유저의 Usability를 고려하여 편리하게 사용할 수 있는 인터페이스를 구축할 것
- 추후 시스템 확장 용이를 위해 유지보수 가능하도록 설계할 것

2.6. Assumptions and Dependencies

본 문서에서 명시된 시스템은 웹 브라우저를 통해 제공되는 것과, 그 중 Chrome 브라우저에 가장 최적화 되도록 제공된다는 것을 전제한다.

본 문서에서 명시된 시스템은 유저의 구동 환경이 아래 운영체제, 브라우저, 하드웨어, 네트워크 스펙이 충족되었다고 가정한다.

1. 1GHz 이상 32비트(x86) 또는 64비트(x64) 프로세서 (Core I5 이상)
2. 1GB RAM(32비트) 또는 2GB RAM(64비트)
3. 16GB 사용 가능한 하드 디스크 공간(32비트) 또는 20GB(64비트)
4. WDDM 1.0 이상 드라이버와 DirectX 9 그래픽 디바이스

본 문서에서 서술한 구동 환경을 만족하지 못한 환경에서는 정상 동작을 보장할 수 없다.

3. Interface Requirements

3.1. User Interfaces

Table 3. Use Interface of Landing page – Unregistered

이름	랜딩 및 메인 페이지 – 로그인 전
목적 및 설명	로그인 전 및 미등록 사용자는 문제를 클릭하여 해당되는 문제를 찾을 수 있으며, 로그인 전 및 미등록 사용자는 '로그인' 버튼을 클릭하여 로그인 페이지로 진입이 가능하다. 상단의 '공지사항', '도움말' 버튼으로 도움을 얻을 수 있다.
입력 소스, 출력 대상	클라이언트, 호스트 서버 호스트서버, 클라이언트
범위, 정확도, 오차 범위	페이지 전체 정렬 알고리즘의 정확도 정렬 알고리즘의 오차 범위를 따름
구성 단위	화면 (1920 * 1080)
시간 및 속도	사용자가 해당 항목을 클릭하였을 때/ 클라이언트와 서버간 통신 속도
기타 입출력 관계	N/A
화면 형식과 구성	상단 '공지사항', '도움말' 버튼은 사이트 정보를 알려주는 페이지로 링크된다. 헤더에는 '문제', '공지사항', '도움말', '강의' 등의 항목이 들어간다. 바디에는 사이트 이름과 사이트에 대한 설명이 있다. 좌측 상단의 '톱니바퀴' 모양을 통해 사이트를 설정하는 방법을 고를 수 있다. 우측 상단의 '로그인' 버튼을 통해서 로그인 페이지로 연결이 가능하다. 상단 배너를 클릭 시 '활성화' 모드가 되며, 배너 아래에 보라색 줄이 생성된다.

이름	랜딩 및 메인 페이지 – 로그인 전
데이터 유형	이미지, 텍스트, 위젯
명령 유형	각 위젯마다 대응되는 실행 코드 값
종료 메시지	N/A

Figure 1. Design of Landing page for unregistered

Figure 2. Design of Landing page for registered

Table 4. Use Interface of Landing page – Registered

이름	랜딩 및 메인 페이지 – 로그인 후
목적 및 설명	사용자는 문제를 클릭하여 강의 상세 페이지로 진입할 수 있다. 사용자는 이후 세부적인 페이지에 진입이 가능해 진다. 사용자는 '로그아웃' 버튼을 클릭하여 해당 사이트에서 로그아웃 할 수 있다. 상단의 '공지사항', '도움말' 버튼으로 도움을 얻을 수 있다.
입력 소스, 출력 대상	클라이언트, 호스트 서버 호스트 서버, 클라이언트
범위, 정확도, 오차 범위	페이지 전체 정렬 알고리즘의 정확도 정렬 알고리즘의 오차 범위를 따름
구성 단위	화면 (1920 * 1080)
시간 및 속도	사용자가 해당 항목을 클릭하였을 때/ 클라이언트와 서버간 통신 속도
기타 입출력 관계	N/A
화면 형식과 구성	우측 상단의 원형 이미지는 'GitHub'에서 추출한 이미지이다. 우측 상단의 원형 이미지 이후의 글자는 'GitHub'에서 가져온 아이디 (Nickname)이다. 상단 '공지사항', '도움말', '강의' 버튼은 사이트 정보를 알려주는 페이지로 링크 된다. 헤더에는 '문제', '공지사항', '도움말', '강의', '게시판', '그룹'의 항목이 들어간다 바디에는 사이트 이름과 사이트에 대한 설명이 있다. 좌측 상단의 '톱니바퀴' 모양을 통해 사이트를 설정하는 방법을 고를 수 있다. 상단 배너를 클릭시 '활성화' 모드가 되며, 배너 아래에 보라색 줄이 생성된다.
데이터 유형	이미지, 텍스트, 위젯

이름	랜딩 및 메인 페이지 – 로그인 후
명령 유형	각 위젯마다 대응되는 실행 코드 값
종료 메시지	N/A

Table 5. User Interface of Login page

이름	로그인 페이지
목적 및 설명	사용자는 Github 계정을 통해 로그인 또는 회원가입을 할 수 있다.
입력 소스, 출력 대상	클라이언트, 호스트 서버 (메인 사이트) 호스트 서버, Github 서버, 호스트 서버
범위, 정확도, 오차 범위	N/A
구성 단위	화면 (1920 * 1080)
시간 및 속도	N/A
기타 입출력 관계	사용자 계정 정보를 Github API와 연동하여 그 값을 받는다.
화면 형식과 구성	1. '로그인' 버튼을 클릭 시 연동된 API 로그인 화면으로 이동한다.
데이터 유형	암호화된 API 키, 쿼리
명령 유형	'로그인' 버튼에 해당하는 명령
종료 메시지	가입 절차 완료 시 ''체크' 표시와 함께 메인 페이지로 돌아감.

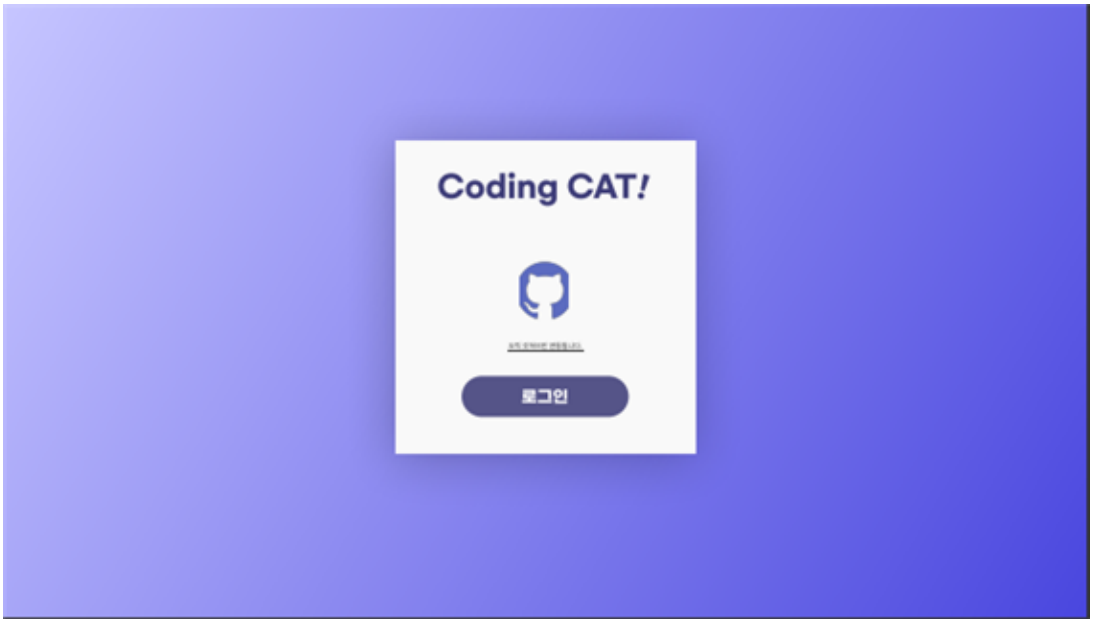

Figure 3. Design of Login page

Table 6. User Interface of Lecture page

이름	강의 선택 페이지
목적 및 설명	사용자는 메인 페이지 좌측에 가장 크게 설정되어 있는 box에서 '과목'(혹은 '강의) 명을 확인 할 수 있고, 각 과목에 할당된 과제들을 확인할 수 있다. 각각의 과제 우측에는 과제를 완료해야 하는 기간까지의 D-time을 확인 가능하며, 24시간 이내로 제출이 완료해야 한다면, '스톱워치' 이미지가 붉게(#F95656) 바뀐다.
입력 소스, 출력 대상	클라이언트, 호스트 서버 호스트 서버, 클라이언트
범위, 정확도, 오차 범위	N/A
구성 단위	화면 (1920 * 1080)
시간 및 속도	N/A
기타 입출력 관계	사용자 로그인 정보를 바탕으로 해당 사용자 데이터베이스에 업데이트

이름	강의 선택 페이지
화면 형식과 구성	상단 헤더에서 '문제'배너가 활성화됨. '강의 목록' 바 하단에서 강의 목록을 확인 가능함. 메인 화면 좌측에서 과제가 생성된 강의 목록을 확인 가능함. 각 강의마다 할당된 과제와 과제 제목을 분류된 바에서 확인 가능함. 각 과제마다 할당된 시간을 우측의 '스톱워치' 이미지가 확인 가능함. 각 과제마다 할당된 시간이 24H (1D) 이하로 줄어들시 우측의 '스톱워치' 이미지가 붉은색 (#F95656)으로 바뀜. 할당된 과제를 '제출'버튼을 통해 완료시 해당 과제가 흑백으로 물들며 비활성화 표시 및 클릭 불가 상태가 됨.
데이터 유형	텍스트, 이미지
명령 유형	'과제' 혹은' '시간' 클릭 시 해당 과제 페이지로 이동.
종료 메시지	N/A

Figure 4. Design of Lecture page

Table 7. User Interface of coding page

이름	코딩 페이지
목적 및 설명	사용자는 좌측 상단 '문제/참조&제약사항'을 통해서 문제의 내용을 확인 가능하다. 사용자는 테스트 케이스를 통해 문제의 input과 output을 확인가능하다. 이후, pass or fail 칸을 통해 pass 인지, fail인지 확인가능하다. 코드 입력 란을 통해 직접 코딩이 가능하다. 이후 '실행', '채점', '제출' 버튼을 클릭하면 이후 창으로 넘어간다.
입력 소스, 출력 대상	클라이언트 / 호스트 서버 호스트 서버 / 클라이언트
범위, 정확도, 오차 범위	해당 실습 페이지의 사용자의 실습 코드/ 각 언어 인터프리터와 사용자의 실습 코드에 따른 정확도/ 각 언어 인터프리터와 사용자의 실습 코드에 따른 오차
구성 단위	화면 (1920 * 1080)
시간 및 속도	사용자가 '실행', '채점', '제출' 버튼을 눌렀을 때/ 실행 코드의 복잡도 및 클라이언트와 서버 간의 통신 속도
기타 입출력 관계	해당 코드를 실행시키는 인터프리터의 결과를 사용자에게 표시 해당 인터프리터가 추출한 결과를 바탕으로 채점결과를 사용자에게 표시. 해당 인터프리터가 추출한 결과와 사용자가 입력한 결과를 바탕으로 수정 방안을 출력 해당 인터프리터가 추출한 결과와 제출한 결과를 비교하여 전체적인 코드의 표절률을 비교하여, 우측 상단에 표절률 출력 해당 인터프리터가 추출한 결과를 바탕으로 가독성, 효율, 기능 점수를 매겨 해당 과제에서 할당된 배율에 맞춰서 점수를 출력. 해당 인터프리터가 추출할 수 없거나, 코드 내에서 오류가 발생할 경우 오류코드를 생성하고 코드 입력 버튼 옆에 '경고'표시 생성.
화면 형식과 구성	페이지 헤더 중 가장 좌측에는 LECTURE PAGE로 나가는 버튼과 설정을 담당하는 '톱니바퀴' 버튼이 있다.

이름	코딩 페이지
화면 형식과 구성	페이지 헤더 중 1의 내용을 제외한 좌측 부분에는 해당 강의는 무엇인지와 해당 강의를 완료해야하는 시간이 적혀 있다. 페이지 헤더 중 가운데에는 현재 수행 중인 assignment가 무엇인지에 대해 적혀있다. 페이지 헤더 중 우측 부분에는 차례대로 '복사'버튼, '초기화'버튼, '불러오기'버튼, '내려받기 버튼이 있으며, 우측의 4개의 버튼을 통해 원하는 상황에서 저장을 할 수 있다. 좌측 상단에는 문제/ 참조 & 제약사항 페이지로 문제, 참조 그리고 제약사항과 관련된 텍스트가 생성되며, 사용자는 이를 통해서 코드를 입력하게 된다. 좌측 하단의 페이지에서는 테스트 케이스가 생성되며 input과 output을 통해 사용자는 pass or fail을 보고 이를 참조하여 코드가 작성 가능하다. 우측 페이지는 코드 입력 페이지로 텍스트가 입력 가능하다. 우측 페이지의 우측 상단에는 '실행', '채점', '체출' 버튼을 통해 원하는 상태의 화면으로 바뀐다. '실행' 버튼을 누를시 코드 입력 내부 창이 분할되어, 좌측은 코드를 그대로 출력하고, 우측은 실행 결과를 출력한다. '채점'버튼을 누를 시 코드 입력 내부 창이 분할되어, 총점을 출력한다. '제출'버튼을 누를 시 코드 입력 내부 창이 분할되어 좌측은 수정 방안을 우측은 제출결과를 각각 출력한다. '제출'후에는 상단의 그래프에서 출제자가 임의로 설정한 배율에 따라 각 항목의 점수와 총 점수를 확인가능하다. '체출'후에는 '가독성', '효율', '기능'으로 나뉜 버튼을 통해 각각의 점수를 체크 가능하며, 스크롤을 통해 코드 설명과 관련 자료의 확인을 할 수 있다.
데이터 유형	텍스트, 이미지
명령 유형	'실행', '채점', '제출', '가독성', 효율', '기능' 버튼에 해당되는 명령
종료 메시지	'해당 파일이 최종 제출되었습니다.'

Figure 5. Design of Coding page

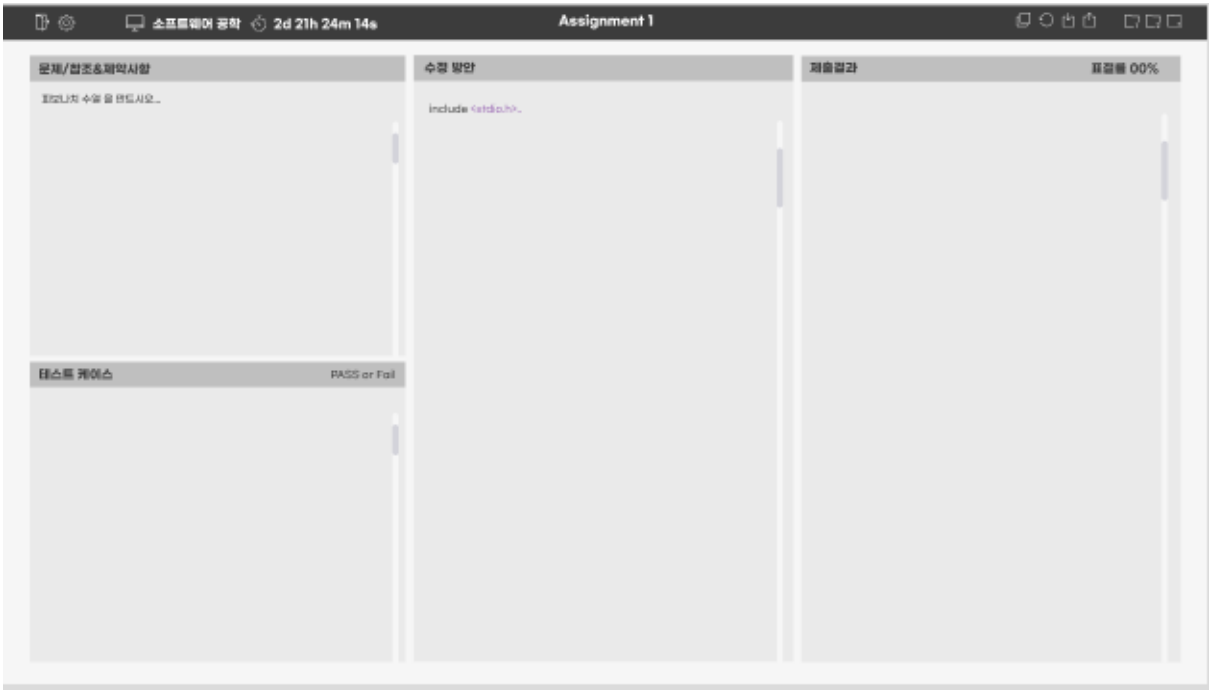

Figure 6. Design of Coding page

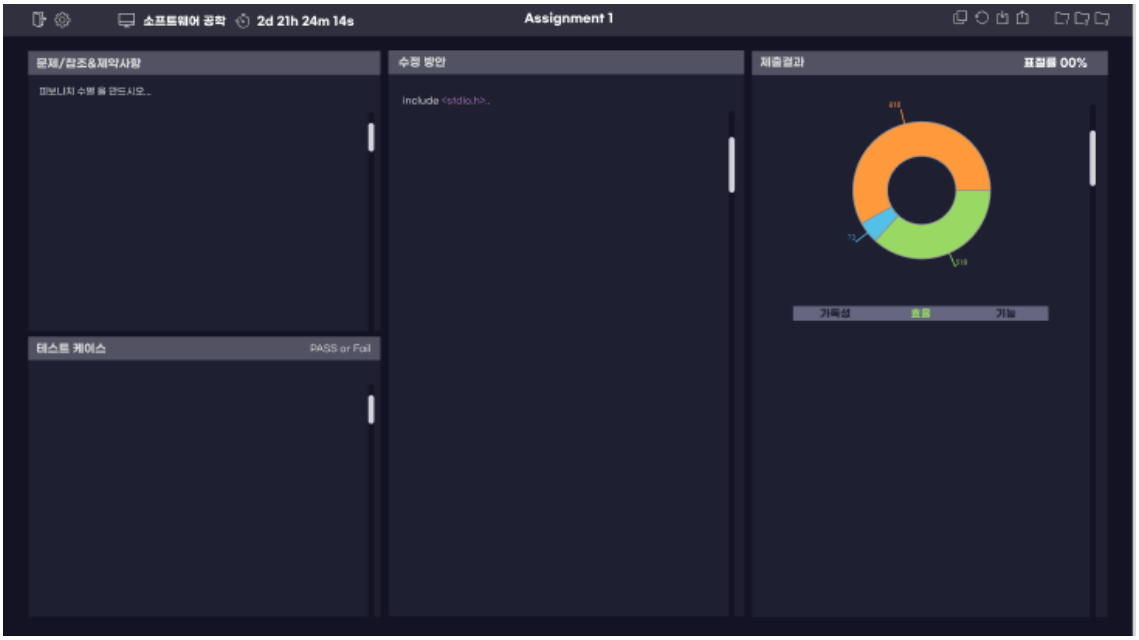

Figure 7. Design of Coding page

Table 8. User Interface of setting page

이름	설정 페이지
목적 및 설명	사용자는 배경 색 설정을 통해 배경을 '화이트' 또는 '블랙' 중에 선택이 가능하다. 사용자는 코드 에디터 설정을 통해 '언어'와 '테마'를 선택 가능하다. 이후 설정이 완료되면 '설정 완료' 버튼을 통해 해당 설정을 저장한다.
입력 소스, 출력 대상	클라이언트, 호스트 서버 호스트 서버, 클라이언트
범위, 정확도, 오차 범위	N/A
구성 단위	화면 (1920 * 1080)
시간 및 속도	사용자가 '설정 완료' 버튼을 눌렀을 때/ 클라이언트와 서버 간의 통신 속도
기타 입출력 관계	N/A

이름	설정 페이지
화면 형식과 구성	메인 화면 중 좌측 상단에 '톱니바퀴' 이미지를 통해 현재 페이지가 설정 페이지임을 알린다. 메인 화면 중 중앙 상단 배경 색 라인에 '흰색' 동그라미 버튼과 '검정색' 동그라미 버튼을 두어 배경색을 '화이트 모드' 또는 '블랙 모드' 중 하나로 변경할 수 있게 한다. 메인 화면 중 중앙 하단 부분에 코드 에디터 설정 박스를 두어 '언어'와 '테마'를 선택 가능하게 한다. '언어'와 '테마'는 dropbox를 이용해 하단에 나타나는 항목 중 선택이 가능하게 한다. 메인 화면 중 최하단에 위치한 '설정 완료' 버튼을 통해 저장이 가능하다.
데이터 유형	텍스트, 이미지
명령 유형	'설정 완료' 버튼에 해당되는 명령
종료 메시지	'해당 설정이 저장되었습니다.'

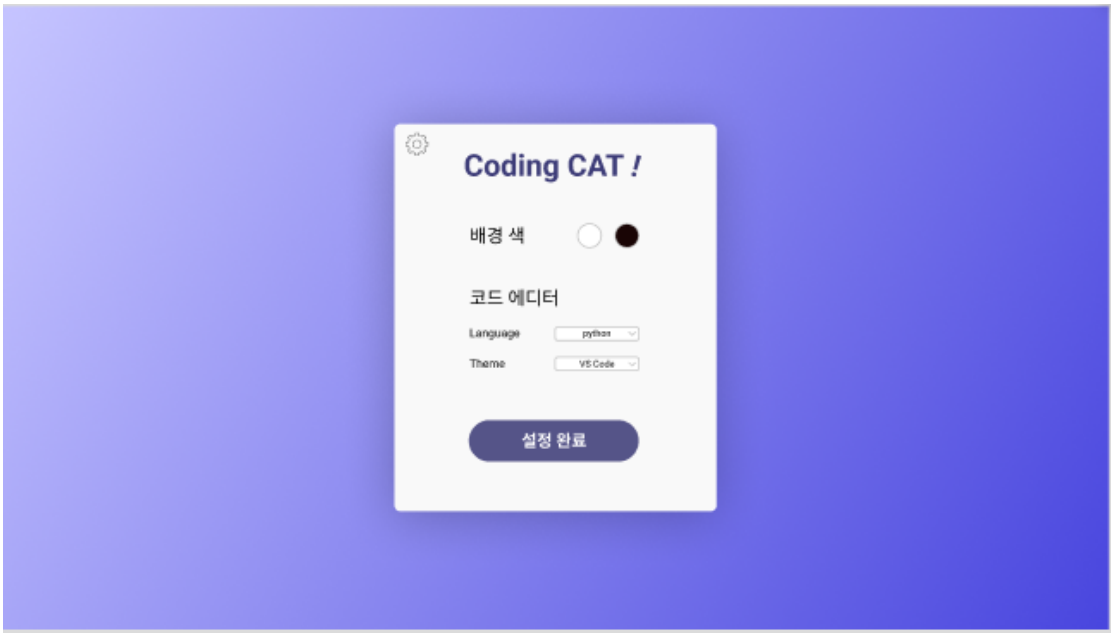

Figure 8. Design of setting page

3.2. Hardware Interfaces

Table 9. Hardware Interface of Client

이름	Web Browser
목적 및 설명	HTTP를 통한 네트워크 통신이 가능하고 모니터, 키보드, 마우스 등 기본 입출력이 가능한 기기로 사용자는 웹 브라우저를 통해 이 시스템을 이용할 수 있다.

Table 10. Hardware Interface of Server

이름	AWS Cloud Server
목적 및 설명	아마존의 클라우드 컴퓨팅 서비스를 이용하여 서버를 구축한다.
구성	Amazon EC2 인스턴스를 구동하여 Front-end 서버와 Back-end 서버를 동작시킨다. Back-end 서버에는 데이터베이스 SQLite가 내장되어 있다.
스펙	NorthEast Asia Region / i3 x86_64 Arch vCPU 2.5GHz 1 Core 2개 / Memory 1GB / Storage 8GB

3.3. Data Management Interfaces

Table 11. Data Management Interface of DB

이름	Relational Database SQLite3
목적 및 설명	사용자 데이터 보관 및 처리를 위한 데이터베이스 엔진
입력 소스	데이터 처리를 위한 질의문
출력 대상	질의에 대한 결과 처리 및 응답
유형	Relational Database SQL

3.4. Software Interfaces

Table 12. Software Interface of Client

이름	사용자 소프트웨어 인터페이스	
목적 및 설명	사용자가 서비스를 이용하기 위해 필요한 소프트웨어 인터페이스이다.	
Chrome Browser Latest Ver.	설명	사용자가 본 서비스를 이용하기 위한 기본 환경
	입력	서비스 데이터를 마우스, 키보드 등의 입력 장치로 입력
	출력	HTML, CSS, JS 렌더링 페이지
Python 3.9 ~	설명	사용자가 서비스가 제공하는 기능을 체험해보기 위한 환경
	입력	사용자 입력 Python 코드
	출력	실행 결과 및 오류 보고
SSH Client Latest Ver.	설명	서비스 관리자가 서버 관리를 위한 통신 환경
	입력	서버 Shell 입력
	출력	서버 Shell 출력

Table 13. Software Interface of Server

이름	서버 구축 소프트웨어 인터페이스	
목적 및 설명	사용자에게 코딩 테스트를 위한 서비스를 제공하고, OAuth 2.0 로그인을 처리하며, Stateless Credential 처리를 위해 서버가 필요로하는 소프트웨어 인터페이스이다.	
React	설명	서비스 UI/UX 제공을 위한 Front-end 프레임워크
	입력	기능 구현을 위한 코드 파일과 사용자 입력
	출력	HTML, CSS, JS 렌더링 페이지
Redux	설명	서비스 기능을 위한 상태 관리 라이브러리

이름	서버 구축 소프트웨어 인터페이스	
	입력	전역으로 관리할 상태 데이터
	출력	관리 대상 상태 데이터
Jest	설명	Javascript 코드의 동작 검증을 위한 라이브러리
	입력	검증 대상 코드와 동작 문맥 데이터
	출력	검증 성공 여부
Django	설명	서비스 기능 동작을 위한 Back-end 프레임워크
	입력	기능 구현을 위한 코드 파일과 사용자 입력 데이터
	출력	JSON 형식의 REST Form
Django AllAuth	설명	OAuth 2.0 기반 사용자 인증을 위한 라이브러리
	입력	OAuth 제공 서비스에서 식별할 수 있는 식별자와 비밀값
	출력	OAuth 제공 서비스의 인가 코드
DRF Simple JWT	설명	JWT 기반 사용자 인증을 위한 라이브러리
	입력	사용자 인증 정보
	출력	Access Token과 Refresh Token
DRF Spectacular	설명	서버가 제공하는 API를 자동 문서화 하기 위한 라이브러리
	입력	기능 API별 요청 및 응답 스펙 기술
	출력	Swagger 혹은 Reoc 형태로 API 문서 반환
Unittest	설명	Python 코드의 동작 검증을 위한 라이브러리
	입력	검증 대상 코드와 동작 문맥 데이터
	출력	검증 성공 여부

이름	서버 구축 소프트웨어 인터페이스	
Copydetect	설명	사용자의 제출 코드에 대한 표절 여부 판단을 위한 라이브러리
	입력	사용자의 입력 코드
	출력	표절 여부
Multimetric	설명	사용자의 제출 코드에 대한 효율성 수준 제공을 위한 라이브러리
	입력	사용자의 입력 코드
	출력	코드의 효율성
Pylama	설명	사용자의 제출 코드에 대한 가독성 수준 제공을 위한 라이브러리
	입력	사용자의 입력 코드
	출력	가독성 수준
OpenAI Codex	설명	사용자의 제출 코드에 대한 설명 제공을 위한 라이브러리
	입력	사용자의 입력 코드
	출력	구현 언어, 코드의 분량 등 다양한 정보
SSH Demon	설명	서버 관리를 위해 서비스 관리자의 SSH 접근을 위한 통신 환경
	입력	콘솔 창의 표준 입력
	출력	콘솔 창의 표준 출력

4. Functional Requirements

4.1. Use Case

Table 14. Function Use Case

No.	Name of Function	Function Description
F1	로그인 및 회원가입	사용자가 자신의 Github 계정을 통해 로그인 및

No.	Name of Function	Function Description
	하기	회원가입을 할 수 있어야 한다.
F2	landing page 보여주기 및 강의, 과제 선택하기	강의 및 과제 목록을 보고 원하는 강의와 과제를 선택할 수 있어야 한다.
F3	main page 보여주기	강의, 과제, 남은 시간, 문제, 테스트 케이스, 코드 에디터, 채점 결과 화면 등을 볼 수 있어야 한다.
F4	과제 이동하기	과제 이름을 클릭해 드롭다운 과제 목록을 통해 과제를 이동할 수 있어야 한다.
F5	테스트케이스 검증하기	테스트케이스 검증을 통해 output 여부에 따라 Pass/Fail 결과가 표시되며, Fail일 경우 학생 코드의 output이 표시된다.
F6	코드 실행하기	입력한 코드를 실행하여 그 결과가 나오고, 실행 결과가 에러일 경우 에러 확인을 할 수 있어야 한다.
F7	코드 중간 저장하기	입력한 코드를 최대 3번 저장할 수 있어야 한다.
F8	저장한 코드 불러오기	중간 저장한 코드 중 원하는 코드를 불러올 수 있어야 한다.
F9	코드 실시간 저장하기	입력한 코드가 실시간으로 저장되어 브라우저를 재실행해도 이전에 작성한 코드가 저장되어야 한다.
F10	파일 불러오기	소스 코드 파일을 로컬에서 업로드하여 코드 에디터 상에 표시할 수 있어야 한다.
F11	코드 초기화하기	입력한 코드를 지우고 과제에서 제시하는 스켈레톤 코드로 되돌릴 수 있어야 한다.
F12	코드 복사하기	입력한 코드를 복사할 수 있어야 한다.
F13	코드 다운로드하기	입력한 코드를 다운로드할 수 있어야 한다.

No.	Name of Function	Function Description
F14	코드 채점하기	입력한 코드를 채점하여 통과 여부를 알려주고, 테스트 케이스에 대한 실행 결과를 보여주어야 한다.
F15	코드 제출하기 및 결과 분석하기	최대 3번 코드를 제출하여 기능 점수, 효율 점수, 가독성 점수, 코드 설명 등을 확인할 수 있어야 한다.

Table 15. (1) 로그인 및 회원가입 하기

이름	로그인 및 회원가입 하기
행위자 (Actor)	Student / Instructor
설명 (Description)	보안을 위해 해당 시스템은 사용자의 개인정보를 직접 저장하지 않고, GitHub을 통한 OAuth 방식으로 사용자가 계정에 로그인하여 본 서비스를 이용할 수 있도록 한다. 사용자 클라이언트가 Github OAuth에 POST 메시지를 전송한다. 이후 인증서버는 이에 대한 반응으로 유저의 정보를 제공하여 이 과정이 성공했을 경우 시스템으로의 접속을 허용한다.
정상 흐름 (Action)	1. 사용자가 로그인 버튼을 클릭 시, Github 로그인 페이지로 이동한다. 2. Github 로그인 페이지에 사용자가 Github 계정 및 암호를 입력하여 로그인을 수행한다. 3. Github으로부터 얻은 토큰을 서버에 넘겨준다. 4. 데이터베이스에 저장되어 있는 토큰일 경우, 사용자는 개인화된 landing page로 이동할 수 있다. 5. 데이터베이스에 저장되어 있지 않은 토큰일 경우, 데이터베이스에 해당 사용자를 저장하며 개인화된 landing page로 이동할 수 있다.
전제 조건 (Pre-condition)	웹 서비스가 github에 정상적으로 등록되어 있다. Github의 OAuth2.0 인증이 정상적으로 작동한다.

이름	로그인 및 회원가입 하기
사후 조건 (Post-condition)	사용자가 인증 받은 토큰을 서버에게 넘겨주고, 서버는 해당 토큰을 인증하고, id_token을 추출하여 사용자에게 넘겨준다. 저장되지 않은 사용자의 경우, 데이터베이스에 새로 저장한다.
가정 (Assumptions)	사용자의 인터넷 환경이 원활하다. Github 서버가 정상적으로 동작한다.

Table 16. (2) Landing Page 보여주기 및 강의, 과제 선택하기

이름	Landing Page 보여주기 및 강의, 과제 선택하기
행위자 (Actor)	Student
설명 (Description)	본 시스템에 로그인을 성공한 이후, 사용자가 본인이 수강하는 강의의 과제를 선택할 수 있다.
정상 흐름 (Action)	1. 학생이 로그인 이후, 본 학생이 수강하는 강의 목록을 보여준다. 2. 원하는 강의를 선택 시, 해당 강의에 따른 과제 목록을 보여준다. 3. 원하는 과제 선택 시, 해당 과제에 따른 main page로 이동한다.
전제 조건 (Pre-condition)	학생이 수강하는 강의 목록, 강의에 해당하는 과제 목록이 데이터베이스에 저장되어 있다.
사후 조건 (Post-condition)	과제 선택을 통해 main page로 이동할 수 있다.
가정 (Assumptions)	AWS에서 backend 프로그램이 정상적으로 실행 중이다.

Table 17. (3) Main Page 보여주기

이름	Main Page 보여주기
행위자 (Actor)	Student
설명 (Description)	강의, 과제, 남은 시간, 문제, 테스트 케이스, 코드 에디터, 채점 결과 화면 등을 볼 수 있어야 한다.
정상 흐름 (Action)	1. 학생이 선택한 과제에 해당하는 정보를 데이터베이스에서 가져온다. 2. 강의, 과제 명, 과제 남은 시간, 문제 및 제약사항, 테스트케이스, 코드 에디터 섹션, 각종 기능 버튼 섹션, 결과 섹션을 표시한다.
전제 조건 (Pre-condition)	강의, 과제에 해당하는 마감 시간, 문제 및 제약사항, 테스트케이스, 스켈레톤 코드 정보들이 데이터베이스에 저장되어 있다.
사후 조건 (Post-condition)	없음
가정 (Assumptions)	AWS에서 backend 프로그램이 정상적으로 실행 중이다.

Table 18. (4) 과제 이동하기

이름	과제 이동하기
행위자 (Actor)	Student
설명 (Description)	과제 이름을 클릭해 드롭다운 과제 목록을 통해 과제를 이동할 수 있어야 한다.
정상 흐름 (Action)	1. 과제 이름 클릭 시, 드롭다운 메뉴를 통해 해당 강의의 과제 목록을 보여준다. 2. 원하는 과제 클릭 시, 해당 과제에 따른 main page로 이동한다.

이름	과제 이동하기
전제 조건 (Pre-condition)	강의에 해당하는 과제 목록이 데이터베이스에 저장되어 있다.
사후 조건 (Post-condition)	과제 선택을 통해 해당 과제의 main page로 이동할 수 있다.
가정 (Assumptions)	AWS에서 backend 프로그램이 정상적으로 실행 중이다.

Table 19. (5) 테스트케이스 검증하기

이름	테스트케이스 검증하기
행위자 (Actor)	Student
설명 (Description)	테스트케이스 검증을 통해 output 여부에 따라 Pass/Fail 결과가 표시되며, Fail일 경우 학생 코드의 output이 표시된다.
정상 흐름 (Action)	1. 학생이 코드 에디터에 자신의 코드를 입력한다. 2. 각 테스트케이스의 검증 버튼 클릭 시, 테스트케이스의 input에 대해 코드가 실행된다. 3. 테스트케이스의 input에 따른 학생 코드의 output이 테스트케이스의 output과 일치할 경우 Pass, 일치하지 않을 경우 Fail을 표시한다. 4. Fail일 경우, 학생 코드의 output 결과도 같이 표시한다.
전제 조건 (Pre-condition)	각 테스트케이스의 input, output이 데이터베이스에 저장되어 있어야 한다.
사후 조건 (Post-condition)	없음
가정 (Assumptions)	AWS에서 backend 프로그램이 정상적으로 실행 중이다.

Table 20. (6) 코드 실행하기

이름	코드 실행하기
행위자 (Actor)	Student
설명 (Description)	입력한 코드를 실행하여 그 결과가 나오고, 실행 결과가 에러일 경우 에러 확인을 할 수 있어야 한다.
정상 흐름 (Action)	1. 학생이 코드 에디터에 자신의 코드를 입력한다. 2. 실행 버튼 클릭 시, 코드 에디터에 입력된 코드를 실행한다. 3. 실행 결과 에러가 없을 경우, 결과 섹션에 출력 결과를 제공한다. 4. 실행 결과 에러가 있을 경우, 결과 섹션에 에러 메세지를 제공하고, 에러 메세지에 따른 위치를 출력하며, 에러 라인을 하이라이트로 표시한다.
전제 조건 (Pre-condition)	없음
사후 조건 (Post-condition)	없음
가정 (Assumptions)	없음

Table 21. (7) 코드 중간 저장하기

이름	코드 중간 저장하기
행위자 (Actor)	Student
설명 (Description)	입력한 코드를 최대 3번 저장할 수 있어야 한다.
정상 흐름	1. 학생이 코드 에디터에 자신의 코드를 입력한다.

이름	코드 중간 저장하기
(Action)	2. 원하는 저장 공간의 버튼 클릭 시, 해당 저장 공간에 코드가 저장된다.
전제 조건 (Pre-condition)	데이터베이스에 과제 당 3개의 저장 공간이 존재한다.
사후 조건 (Post-condition)	저장 버튼 클릭 시, 데이터베이스의 해당 저장 공간에 코드가 저장된다.
가정 (Assumptions)	AWS에서 backend 프로그램이 정상적으로 실행 중이다.

Table 22. (8) 저장한 코드 불러오기

이름	저장한 코드 불러오기
행위자 (Actor)	Student
설명 (Description)	중간 저장한 코드 중 원하는 코드를 불러올 수 있어야 한다.
정상 흐름 (Action)	1. 중간 저장이 되어있는 저장 공간의 버튼을 클릭 시, 해당 저장 공간에 있는 코드를 코드 에디터로 불러온다. 2. 해당 저장 공간에 저장돼있는 코드를 삭제한다.
전제 조건 (Pre-condition)	데이터베이스의 해당 저장 공간에 코드가 저장되어 있어야 한다.
사후 조건 (Post-condition)	데이터베이스의 해당 저장 공간의 코드를 삭제한다.
가정 (Assumptions)	AWS에서 backend 프로그램이 정상적으로 실행 중이다.

Table 23. (9) 코드 실시간 저장하기

이름	코드 실시간 저장하기
행위자 (Actor)	Student
설명 (Description)	입력한 코드가 실시간으로 저장되어 브라우저를 재실행해도 이전에 작성한 코드가 저장되어야 한다.
정상 흐름 (Action)	1. 학생이 코드 에디터에 자신의 코드를 입력한다. 2. 코드가 일정 시간마다 저장된다. 3. 브라우저 종료 혹은 과제 이동 등으로 메인 화면을 벗어난다. 4. 브라우저를 재실행 시, 이전에 작성한 코드가 코드 에디터에 표시된다.
전제 조건 (Pre-condition)	일정 시간 마다 frontend에 코드 에디터의 코드가 저장된다.
사후 조건 (Post-condition)	없음
가정 (Assumptions)	없음

Table 24. (10) 파일 불러오기

이름	파일 불러오기
행위자 (Actor)	Student
설명 (Description)	소스 코드 파일을 로컬에서 업로드하여 코드 에디터 상에 표시할 수 있어야 한다.
정상 흐름 (Action)	1. 파일 업로드 버튼을 눌러 파일 탐색기를 불러온다. 2. 업로드 하고자 하는 소스 코드 파일을 선택한다. 3. 해당 파일의 코드를 코드 에디터 상에 표시한다.

이름	파일 불러오기
전제 조건 (Pre-condition)	업로드 하고자 하는 파일이 학생의 pc에 존재해야 한다.
사후 조건 (Post-condition)	없음
가정 (Assumptions)	없음

Table 25. (11) 코드 초기화하기

이름	코드 초기화하기
행위자 (Actor)	Student
설명 (Description)	입력한 코드를 지우고 과제에서 제시하는 스켈레톤 코드로 되돌릴 수 있어야 한다.
정상 흐름 (Action)	1. 학생이 코드 에디터에 자신의 코드를 입력한다. 2. 초기화 버튼을 클릭 시, 코드 에디터의 코드가 삭제되고 스켈레톤 코드를 표시한다.
전제 조건 (Pre-condition)	해당 과제의 스켈레톤 코드가 데이터베이스에 저장되어 있다.
사후 조건 (Post-condition)	없음
가정 (Assumptions)	AWS에서 backend 프로그램이 정상적으로 실행 중이다.

Table 26. (12) 코드 복사하기

이름	코드 복사하기
행위자 (Actor)	Student
설명 (Description)	입력한 코드를 복사할 수 있어야 한다.
정상 흐름 (Action)	1. 학생이 코드 에디터에 자신의 코드를 입력한다. 2. 복사 버튼을 클릭 시, 코드 에디터의 코드가 클립보드에 저장된다.
전제 조건 (Pre-condition)	없음
사후 조건 (Post-condition)	복사한 코드가 클립보드에 저장된다.
가정 (Assumptions)	없음

Table 27. (13) 코드 다운로드하기

이름	코드 다운로드하기
행위자 (Actor)	Student
설명 (Description)	입력한 코드를 다운로드할 수 있어야 한다.
정상 흐름 (Action)	1. 학생이 코드 에디터에 자신의 코드를 입력한다. 2. 다운로드 버튼을 클릭 시, 코드 에디터의 코드를 파일로 변환하여 저장한다.
전제 조건 (Pre-condition)	없음

이름	코드 다운로드하기
사후 조건 (Post-condition)	저장한 코드 파일이 학생의 pc에 다운로드된다.
가정 (Assumptions)	없음

Table 28. (14) 코드 채점하기

이름	코드 채점하기
행위자 (Actor)	Student
설명 (Description)	입력한 코드를 채점하여 통과 여부를 알려주고, 테스트 케이스에 대한 실행 결과를 보여주어야 한다.
정상 흐름 (Action)	1. 학생이 코드 에디터에 자신의 코드를 입력한다. 2. 채점 버튼을 클릭 시, 테스트케이스를 통해 코드 에디터의 코드를 실행한다. 3. 각 테스트케이스별 Pass/Fail 여부와 점수를 제공한다. 4. 오픈 테스트케이스가 Fail 했을 경우, 테스트케이스 정보 및 결과를 제공한다. 5. 히든 테스트케이스가 Fail 했을 경우, 테스트케이스 정보를 미제공한다.
전제 조건 (Pre-condition)	해당 과제의 오픈 테스트케이스 및 히든 테스트케이스가 데이터베이스에 저장되어 있어야 한다.
사후 조건 (Post-condition)	없음
가정 (Assumptions)	AWS에서 backend 프로그램이 정상적으로 실행 중이다.

Table 29. (15) 코드 제출하기 및 결과 분석하기

이름	코드 제출하기 및 결과 분석하기
행위자 (Actor)	Student
설명 (Description)	최대 3번 코드를 제출하여 기능 점수, 효율 점수, 가독성 점수, 코드 설명 등을 확인할 수 있어야 한다.
정상 흐름 (Action)	1. 학생이 코드 에디터에 자신의 코드를 입력한다. 2. 제출 버튼 클릭 시, 기능, 효율, 가독성에 대해 채점이 이뤄진다. 3. 기능 점수 확인 시, 테스트케이스에 대한 채점 결과가 표시된다. 4. 효율 점수 확인 시, Line of Codes, Reservation Words, Data Flow Complexity, Control Flow Complexity에 대한 점수가 표시된다. 5. 가독성 점수 확인 시, mypy, pylint, eradicate, radon, pycodestyle에 대한 점수가 표시된다. 6. 표절율이 표시된다. 7. 정답 코드와 학생 코드간의 차이가 표시된다. 8. 코드 설명 및 관련 자료가 표시된다.
전제 조건 (Pre-condition)	정답 코드, 코드 설명, 관련 자료가 데이터베이스에 저장되어 있어야 한다. OpenAI Codex API를 통해 효율 점수, 가독성 점수, 표절율을 확인할 수 있어야 한다.
사후 조건 (Post-condition)	기능 점수, 효율 점수, 가독성 점수, 표절율이 데이터베이스에 저장된다.
가정 (Assumptions)	AWS에서 backend 프로그램이 정상적으로 실행 중이다. 효율 점수, 가독성 점수, 표절율을 계산하는 OpenAI Codex API가 정상적으로 동작한다.

4.2. Use Case Diagram

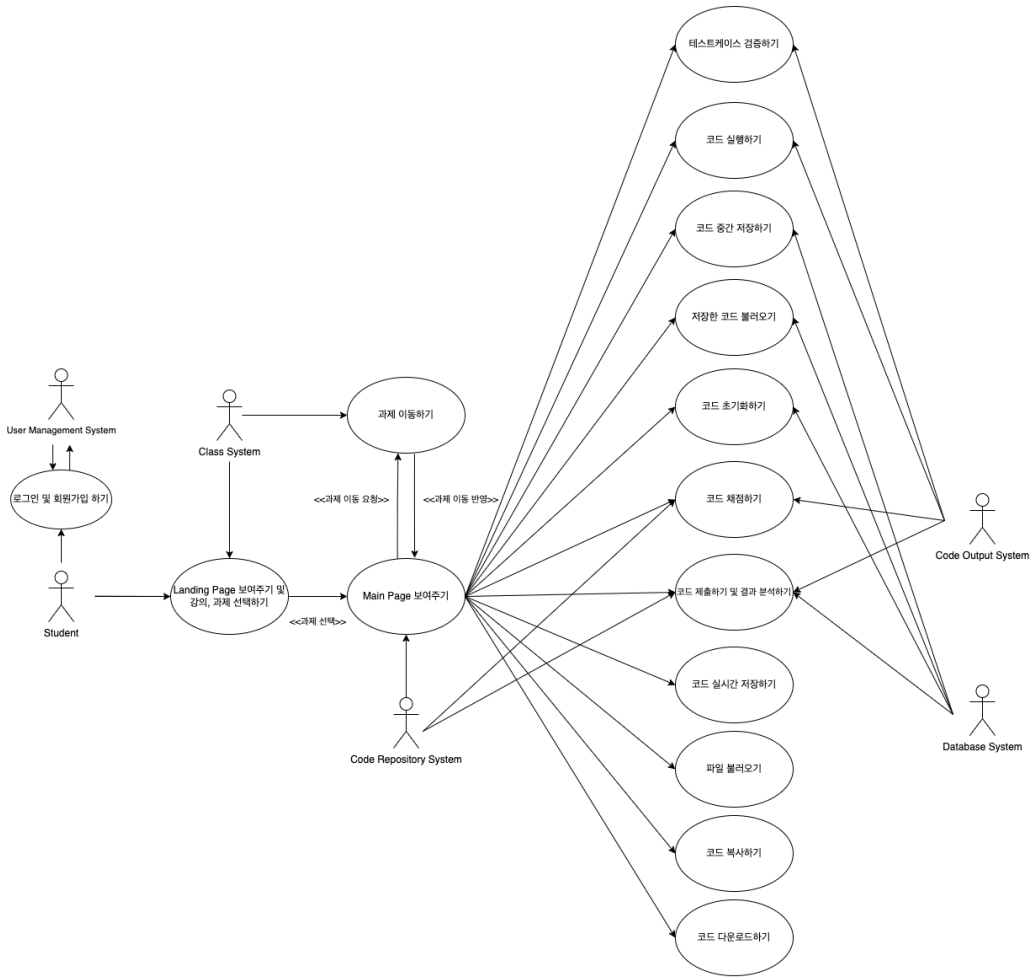

Figure 9. Use Case Diagram

4.3. Data Flow Diagram

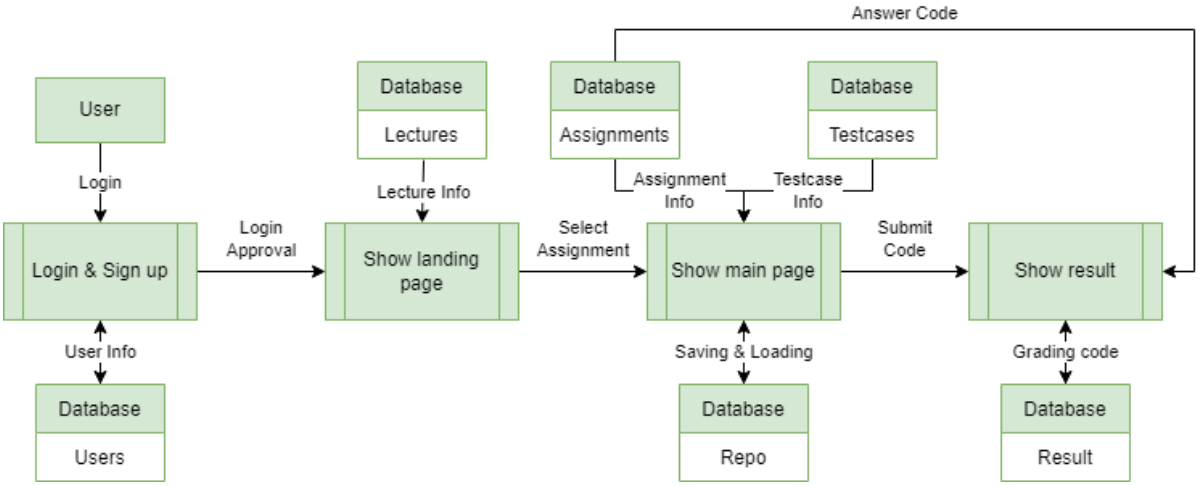

Figure 10. Data Flow Diagram

5. Product Requirements

본 항목에서는 본 시스템이 제공해야 할 사용성, 성능 등의 Non-functional requirements를 서술한다.

5.1. Usability Requirement

본 시스템은 사용자가 실제로 사용하기에 편한 코딩 테스트 시스템을 제공하는 것을 지향한다. 본 시스템은 웹 애플리케이션 기반으로 작성되었으므로, 사용자는 현대의 웹 브라우저를 통해 시스템을 사용할 것을 요구받는다. 하지만, 브라우저 이외의 사용자에 대한 별도의 요구 사항은 존재해서는 아니된다.

본 시스템에서 제공하는 코딩 테스트 환경은 개설된 교과목에 대해 교강사가 학생의 학습 및 평가를 할 수 있도록 제공하는 것이다. 따라서 본 시스템은 학생이 교강사가 설정한 과제들을 학생들은 과제의 의도에만 집중하여 문제를 해결할 수 있는 환경을 제공해야 한다.

5.2. Performance Requirement

본 시스템은 OAuth2.0 기반으로 사용자 로그인 기능을 구현하므로, 다중의 사용자가 시스템을 동시에 사용하는 것이 가능하다. 또한 본 시스템은 사용자의 요청을 다루는 서버가 AWS 클라우드 컴퓨팅 서비스에서 실행되기 때문에, 많은 사용자가 동시에 접속하는 경우에도 시스템의 심각한 성능 저하 없이 시스템을 사용할 수 있다.

단, 사용자의 악의적인 서비스 이용을 방지하기 위해, 사용자 한 명이 프로그램 코드를 동시에 실행하는 것을 허용하지 않고, 순차적으로 한 번에 하나의 코드만을 실행할 수 있도록 제한한다.

상기 요구사항이 충족된다면 사용자의 인터넷 연결이 원활하고 AWS 서버에 이상이 없다는 가정 하에, 모든 서비스 이용은 10초 이상의 지연이 발생하지 않아야 한다.

5.3. Space Requirement

사용자는 현대 웹 브라우저를 구동할 수 있는 최소 2 코어 이상의 CPU, 4GB 이상의 RAM, 10GB 이상의 디스크 여유 공간을 필요로 한다.

5.4. Security Requirement

본 서비스는 AWS 클라우드 컴퓨팅 서비스를 이용하여 배포되기 때문에, 배포된 서버의 Public IP를 알고 있다면 접근 가능하다. 그러나 서버에 과도한 API 요청을 하거나 악의적인 공격을 억제하기 위해 AWS의 보안 정책 중, Whitelist IP 정책을 채택하여 Back-end 서버에 접근 가능한 Host를 제한한다.

본 서비스는 토큰 기반 사용자 인증을 진행하며, 세션 및 브라우저 쿠키를 활용하지 않는다. 이에 따라 서버는 무상태성을 지원한다.

사용자의 정보를 토대로 수행되어야 하는 API에는 인증 토큰 검증이 필수적으로 요구된다. 만약 익명 사용자이거나 사용자 식별자가 일치하지 않는 경우 해당 요청은 401 HTTP 에러 코드를 반환하며 수행되지 않는다.

6. Organizational Requirements

본 항목에서는 사용자 집단 및 개발자 집단의 규제 및 절차에 의해 생성된 본 시스템에 대한 요구 사항들을 서술한다.

6.1. Environment Requirement

해당 시스템은 web 기술에 기반하여 코딩 테스트 서비스를 제공한다. 이 시스템을 이용하는 사용자는 PC를 통해 시스템에 접속하는 것을 상정하고 있으며, PC 환경에서의 올바른 작동을 보장한다. 따라서 이 시스템은 사용자가 PC를 통해 대표적인 PC 입력장치인 keyboard와 mouse를 사용해 시스템과 상호작용하는 것을 보장한다.

PC 이외의 환경에서의 올바른 작동은 보장되지 않을 수 있다. 또한 해당 시스템은 web 기술에 기반한 서비스이므로, 사용자가 web browser를 통해 서비스에 접속할 것을 전제로 한다.

사용자와 직접 상호작용하는 frontend는 Node.js, HTML, CSS에 기반한 React 라이브러리를 사용하여 개발한다.

Backend는 python에 기반한 django 라이브러리를 사용하며, 데이터베이스 관리 시스템으로 SQLite를 사용하여 개발한다.

해당 시스템을 개발하는 과정에 있어, 서로 다른 API가 작동하면서 충돌을 일으키지 않거나 라이선스 문제가 없을 것으로 판단되는 경우에 한하여 개발자는 오픈 소스 소프트웨어를 적극적으로 시스템에 포함할 수 있다. 단, 이 요구사항 명세서가 작성된 이후 발생한 모든 오픈 소스 API의 추가 및 변동에 대해서는 그 내용을 상세히 Git을 통해 VCS에 기록되어야 한다.

6.2. Operational Requirement

해당 시스템은 시스템에 접속하는 PC가 modern web환경에 있을 경우에는 React 라이브러리에 기반하여 사용자에게 동적인 웹서비스를 제공해야 한다. 이외에도, 해당 서비스는 본 문서가 다루는 요구사항 명세의 범위 내에서는 반드시 서비스 이용에 있어서 장애가 발생하여서는 아니된다.

해당 시스템은 코딩 테스트를 통해 학습을 하고자 하는 학생에 의해 사용되며, 모든 강의 및 과제가 끝날 때 까지 수행된다. 이를 위해 과제 정보와 그에 대한 테스트 케이스, 코드 설명, 부가적인 참고 자료가 주어진다.

해당 시스템의 사용자는 이 시스템을 통해 프로그래밍 언어 중 하나인 Python 언어에 대한 지식을 얻을 수 있어야 한다. 이 요구 사항은 해당 시스템이 문제 해결에

기반한 학습 수단을 사용자에게 제시하고, 사용자는 스스로 문제해결 과정을 수행하고 이에 대해 해당 시스템이 제공하는 피드백을 종합하여 Python 프로그래밍 언어에 숙련될 수 있어야 한다.

7. External Requirements

본 항목에서는 시스템의 개발 과정 및 운영, 그리고 system evolution에 걸쳐 시스템 외부적으로 발생하거나 발생할 수 있는 본 시스템에 대한 요구 사항들을 서술한다.

7.1. Regulatory Requirement

해당 시스템에 사용되는 주요 외부 API로는 Django, React, SQLite가 있다. 이들을 포함하여 본 시스템에 사용된 모든 외부 API들의 라이선스는 면밀히 검토되어야 하며, 라이선스간의 충돌은 절대로 허용하지 않는다. 라이선스 간 충돌이 발생한 경우, 대체하는 API를 적용하는 등의 방식으로 모든 라이선스 충돌은 시스템의 개발 과정에서 반드시 해결되어야 한다.

사용자에게 제공되는 코딩 테스트 문제들에 대해 대한민국 법령에서 공포한 저작권법에 관련하여 문제가 발생하지 않아야 한다. 본 시스템은 저작권법에 위배되는 문제들을 사용해서는 아니된다.

본 서비스는 GitHub 계정을 통한 OAuth 프로토콜을 사용한다. 따라서, 본 시스템은 OAuth 공급자인 GitHub의 사용 약관에 따라 서비스를 구현해야 한다.

7.2. Safety/Security Requirement

본 시스템은 대한민국 법령이 공포한 개인정보 보호법에 따라 본 시스템 사용자의 개인정보의 처리 및 보호를 수행한다. 본 시스템이 사용자의 개인정보를 사용함에 있어, 모든 사용자의 자유와 권리는 침해되어서는 아니된다.

본 시스템은 OAuth 프로토콜을 따르며, 클라이언트에서 서버로 유저 정보가 전송될 때 사용자의 개인정보는 비식별화된다. 더불어 사용자 인증 수단으로 엑세스 토큰 방식을 도입하여 비밀번호 유출에 대해 자유롭고, 이외에도 발생할 수 있는 MITM Attack 등의 성공 가능성이 낮다.

7.3. Ethical Requirement

사용자가 작성한 코드를 평가함에 있어서, 모든 사용자에 대해 동일한 평가기준을 적용하여 본 시스템이 제공하는 코딩 테스트의 공정성이 보장되어야 한다.

본 시스템이 추천하는 콘텐츠의 출처가 어떠한 이유에서라도 특정 집단으로 편향되어서는 아니된다. 이 시스템의 모든 사용자들은 반드시 평등한 학습의 기회를 가져야 한다.

8. Organizing System Flow

8.1. Context Model

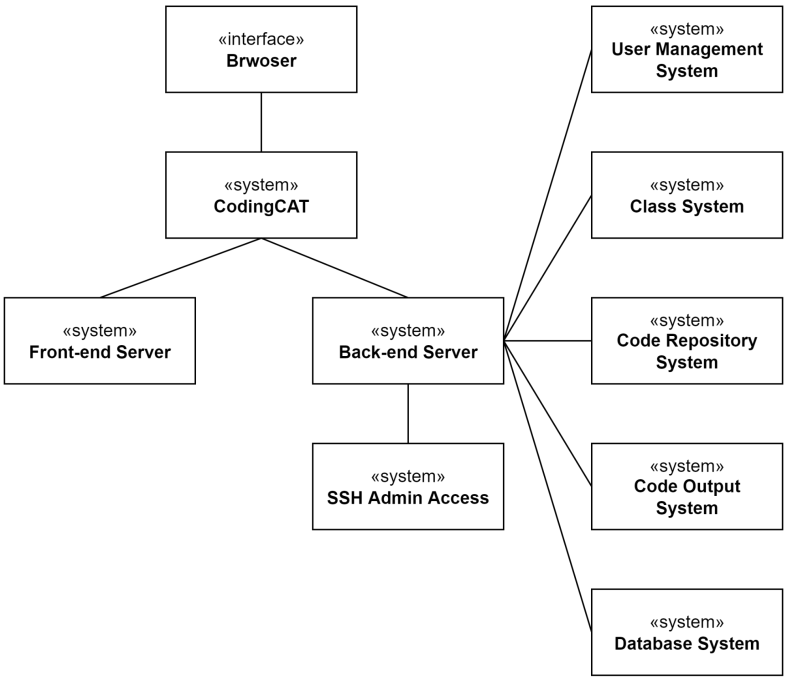

Figure 11. Context Model

8.2. Process Model

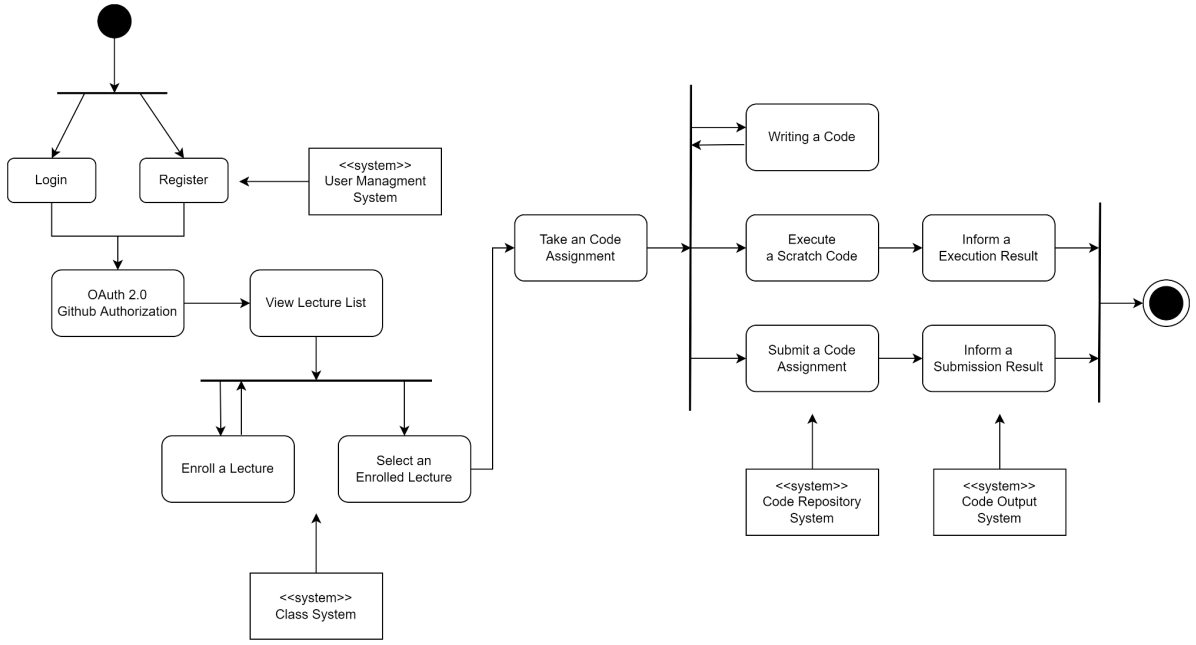

Figure 12. Process Model

8.3. Interaction Model

4.2 항목의 Use Case Diagram 형태로 표현되었다.

8.4. Behavior Model

8.4.1. Data Flow Diagram

4.3 항목의 Data Flow Diagram 형태로 표현되었다.

8.4.2. Sequence Diagram

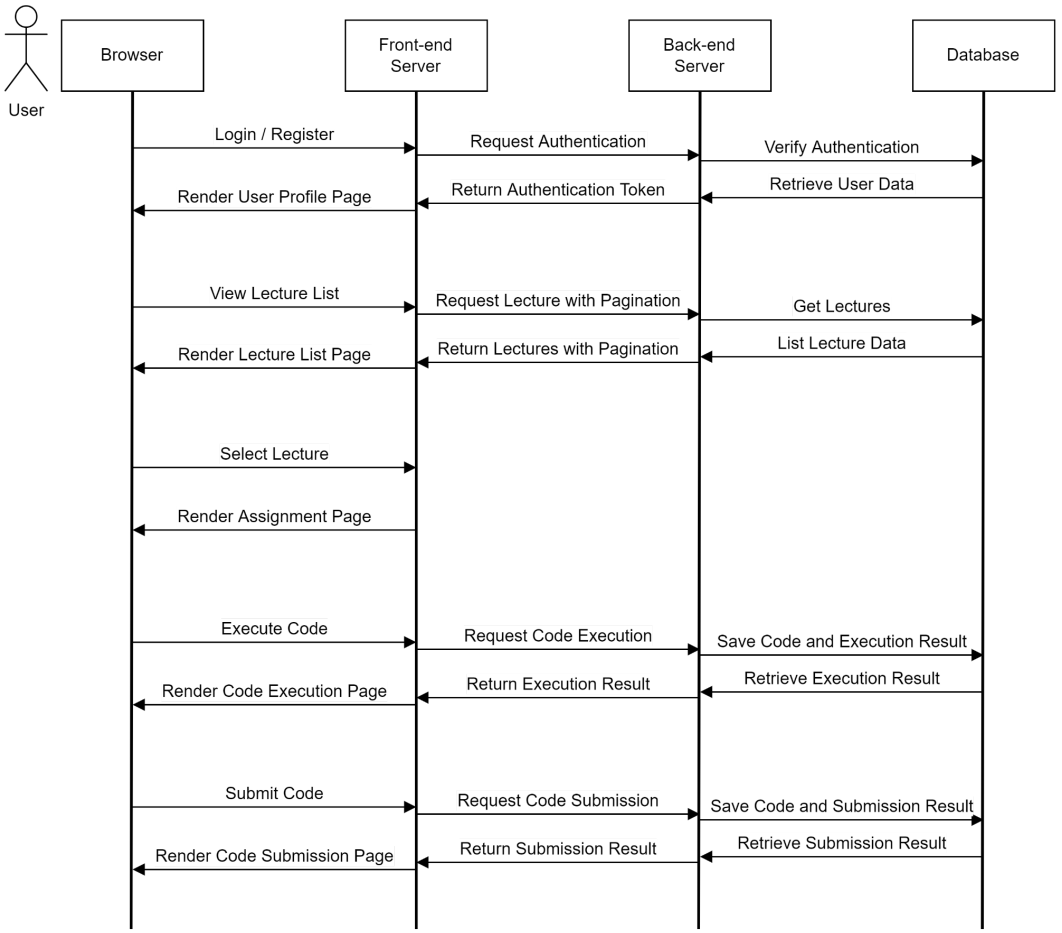

Figure 13. Sequence Diagram

9. System Architecture

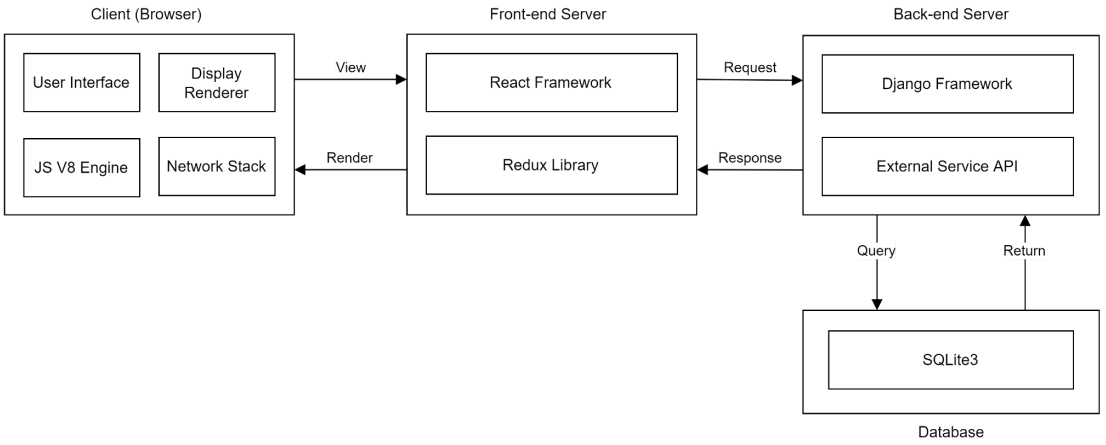

Figure 14. System Architecture

10. System Evolution

본 항목에서는 시스템이 추후 하드웨어 및 소프트웨어 개선, 사용자의 요구 사항 변경 및 추가, 기술적인 추세 변화, 그리고 동작 플랫폼 확장 등 발생할 수 있는 변동 사항을 서술한다. 이 항목은 향후 시스템의 유지보수 및 개선 방향성을 나타내기 때문에 시스템 엔지니어는 이를 고려하여 시스템을 설계하여야 한다.

10.1. Assumption and Limitation

본 서비스는 PC 환경의 웹 브라우저를 기반으로 사용자가 프로그래밍 언어를 이용하여 코딩 테스트를 진행할 수 있는 기능을 제공한다. 첫 출시 시점에는 주어진 강의와 과제, 그리고 연관 테스트 케이스만을 이용하여 Python 언어로 작성된 코드에 대한 코딩 테스트를 진행할 수 있다.

그러나 추후 특정 과제에 대해 사용자가 제안하는 테스트 케이스를 추가하거나, 강의 및 과제 등록 및 수강 여부를 동적으로 처리하는 등의 다양한 기능이 추가될 수 있다. 또한 Python 언어 뿐만 아니라 C++, Java, Javascript 등 타 언어로 작성된 코드에 대한 코딩 테스트 지원이 고려될 수 있다.

10.2. Evolutions of Hardware and Software

첫 출시 시점의 배포 환경은 AWS EC2 t2.micro 인스턴스를 기반으로 한다. 이는 최소한의 서버 구동 환경을 위한 리소스가 제공되므로, 추후 더 많은 사용자를 핸들링하고 그에 따른 데이터를 처리 및 보관을 하기 위해서는 더 높은 수준의 리소스가 필요하게 될 것이다. 이에 따라 AWS의 다른 인스턴스로 전환하거나, 혹은 GCP 등 타 클라우드 플랫폼으로 전환될 가능성이 있다.

서버의 주된 구성 요소로 React와 Django가 있다. 첫 출시 기준 React는 18.2 버전을, Django는 4.1 버전을 사용한다. 위의 프레임워크는 버전에 따라 구성 모듈이

새롭게 추가되거나 Deprecate되는 경우가 잦은 편으로, 시스템의 안정성과 유지보수 용이성을 위해 소프트웨어의 버전을 관리할 필요가 있다.

10.3. Diverse Platforms

현재의 목표 플랫폼은 PC 환경의 웹 브라우저 환경이다. 그러나 추후 서비스의 사용자 풀이 커지고 다양한 환경에서의 기능 제공이 요구 된다면, 모바일 환경의 웹 브라우저 환경을 지원하거나 Web App으로 Port하여 출시될 수 있다. 또한 Windows, MacOS 등 특정 운영체제 상에서 구동되는 Desktop Application으로의 확장도 이뤄질 수 있다.

11. Supporting Information

11.1. Software Requirement Specification

본 소프트웨어 요구 사항 명세서는 IEEE 권장 사항에 맞추어 작성되었다 (IEEE Recommended Practice for Software Requirements Specifications, IEEE-Std-830). 다만 본 서비스의 요구 조건을 용이하게 파악할 수 있도록 본래의 양식에서 일부 추가되거나 제외된 항목이 있다.

11.2. Document History

Table 30. Document History

Date	Version	Description	Write
2022.10.28	0.1	초안 작성 완료	박승호 외 5인
2022.10.30	1.0	최초 버전 발행	박승호 외 5인