

소프트웨어공학개론 SWE-3002_41
기말 발표

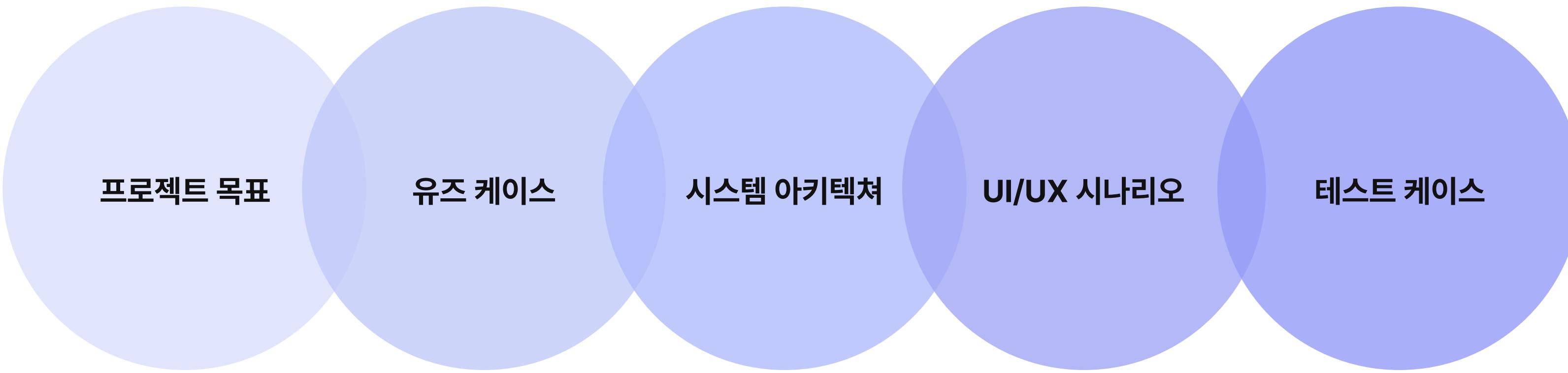
당신을 위한 코드 주치의, 코닥

CODOC

4조

김다한
김민재
이동신
진서영
최지애
한종승

목차



프로젝트 목표

유즈 케이스

시스템 아키텍쳐

UI/UX 시나리오

테스트 케이스

프로젝트 목표



취약점이나 코드 스타일에 대한 피드백을 받기가 어려워요



문제 풀이 후 어려움을 느낀 부분에 대해 정리하는 습관을 들이지 못해서 계속 같은 실수를 반복해요



오답인 경우에 대한 해설을 찾기가 어려워요



어떤 문제에 취약한지 객관적으로 분석하기 어려워요

4조

CODOC

프로젝트 목표

유즈 케이스

시스템 아키텍쳐

UI/UX 시나리오

테스트 케이스

프로젝트 목표

CODOC

LLM의 코드 분석을 통해
약점을 진단해주고 성취도를 높이는
사용자 맞춤형 코딩 학습 플랫폼

4조

CODOC

프로젝트 목표

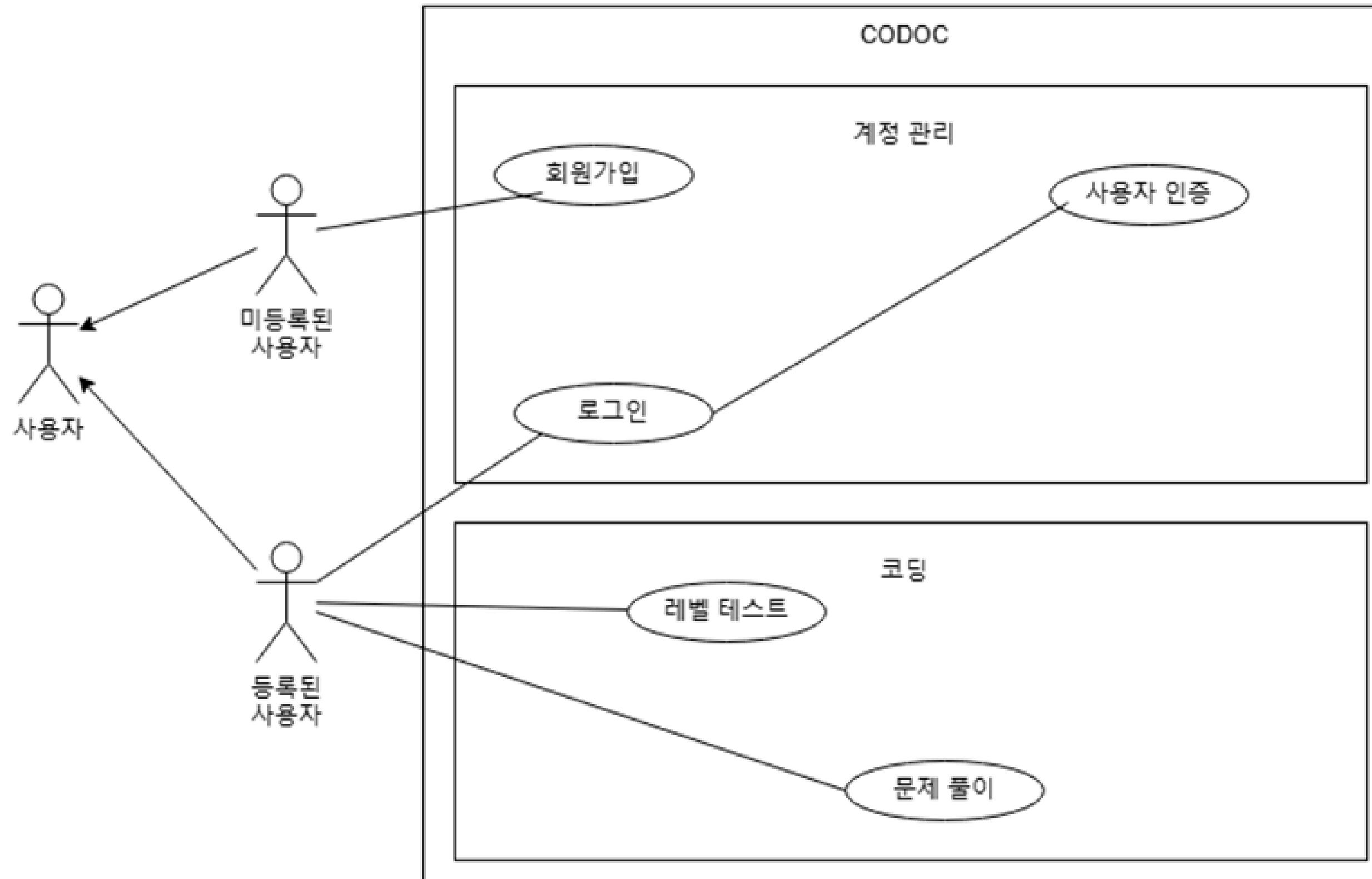
유즈 케이스 - 사용자

유즈 케이스

시스템 아키텍쳐

UI/UX 시나리오

테스트 케이스



4조

CODOC

프로젝트 목표

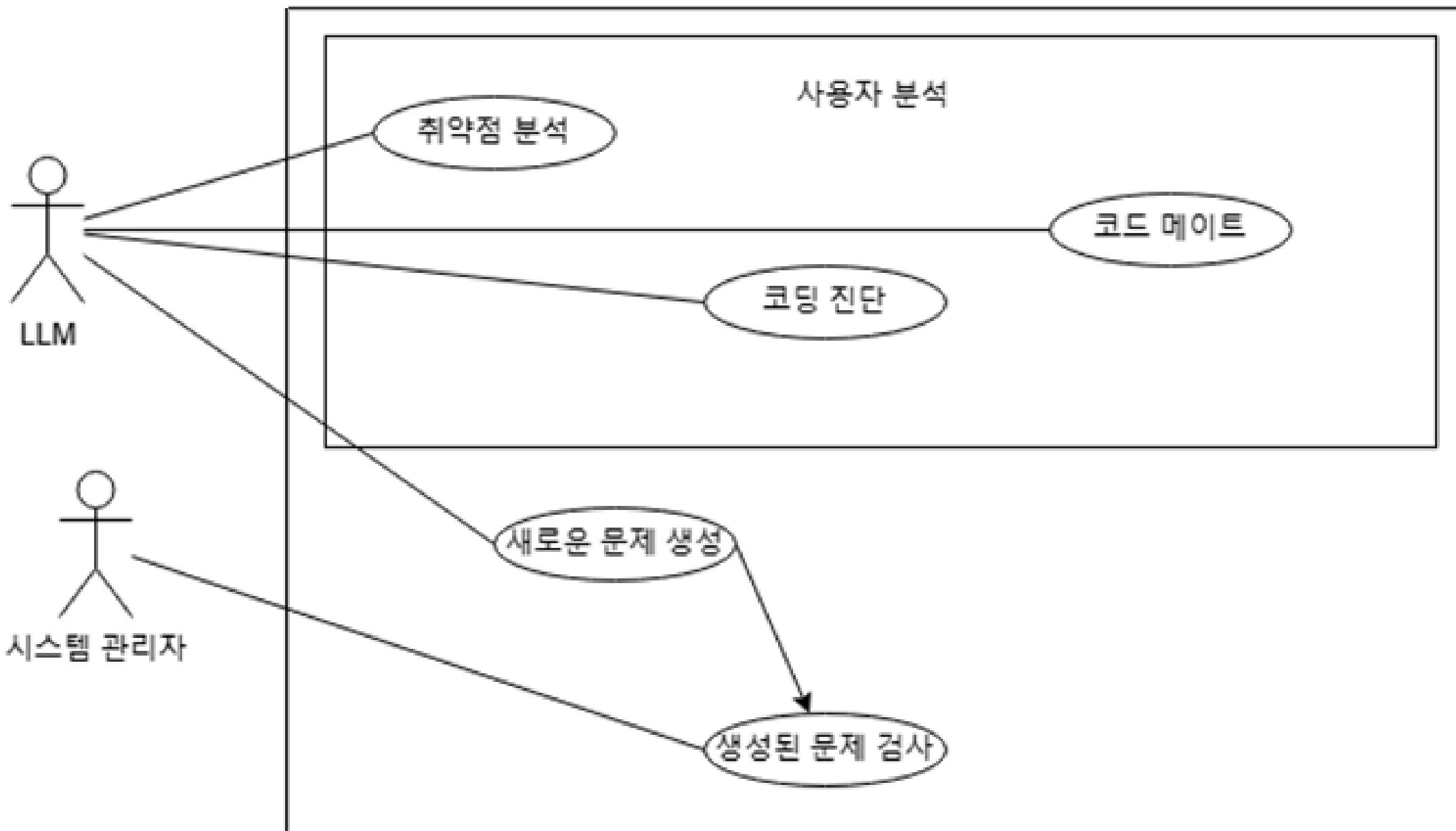
유즈 케이스 - LLM / 관리자

유즈 케이스

시스템 아키텍쳐

UI/UX 시나리오

테스트 케이스



4조

CODOC

프로젝트 목표

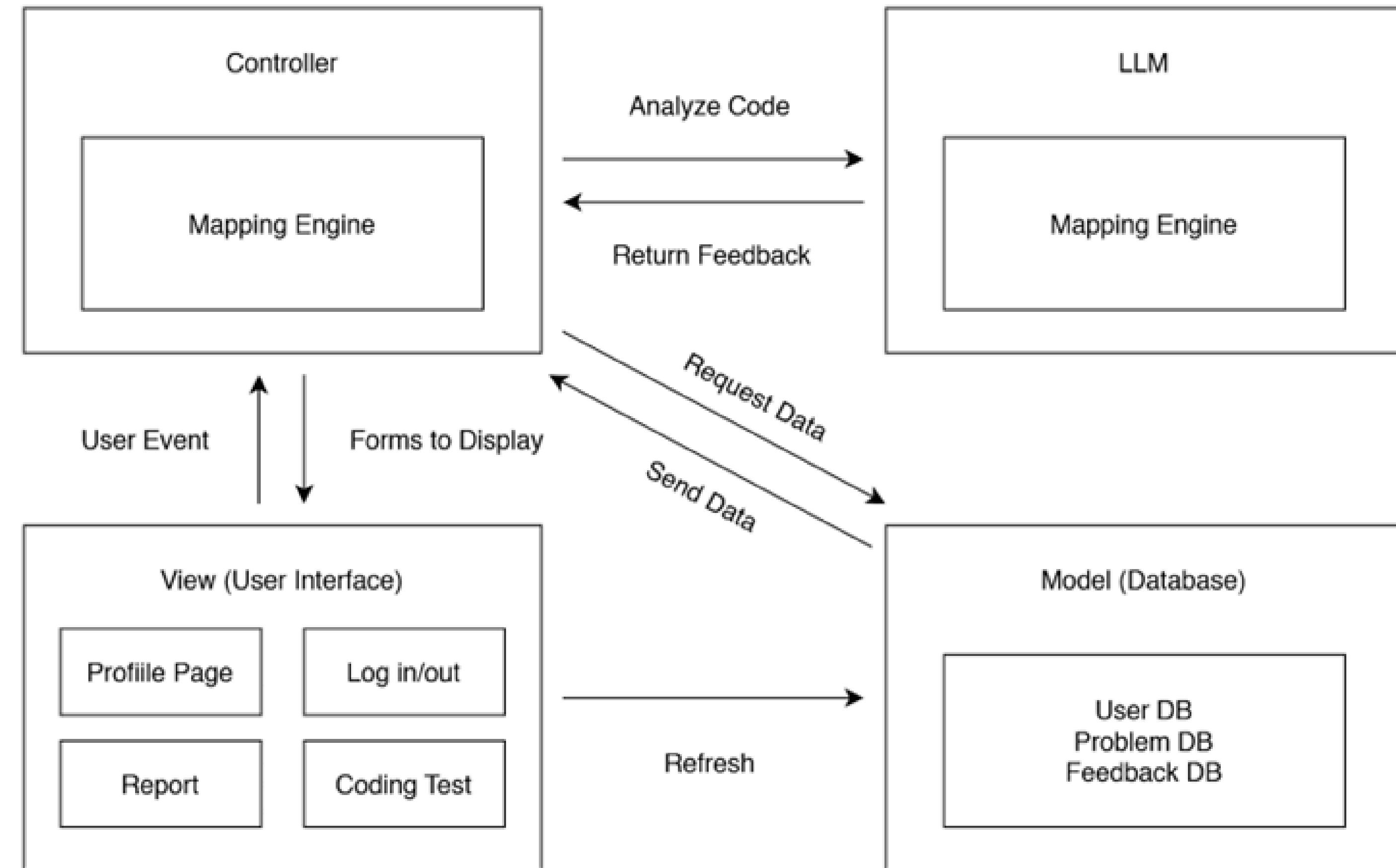
시스템 아키텍쳐

유즈 케이스

시스템 아키텍쳐

UI/UX 시나리오

테스트 케이스



4조

CODOC

프로젝트 목표

사용자 관리 (로그인 / 회원가입)

유즈 케이스

시스템 아키텍쳐

UI/UX 시나리오

테스트 케이스

4조

CODOC

로그인

아이디
CODOC1234

비밀번호

로그인하기

비밀번호 찾기 | 회원가입하기

회원가입

CODOC과 함께 코딩 고수가 되어봐요!

이름
김코닥

이메일 주소
email gmail.com

비밀번호
비밀번호

본인이 생각하는 레벨
1 2 3 4 5

주 사용 프로그래밍 언어
Python

가입하기

계정이 있으신가요? [로그인하기](#)

회원가입

CODOC과 함께 코딩 고수가 되어봐요!

이름
김코닥

이메일 주소
Codoc1234 gmail.com

비밀번호
codocodoc

본인이 생각하는 레벨
1 2 3 4 5

주 사용 프로그래밍 언어
Python

가입하기

계정이 있으신가요? [로그인하기](#)

프로젝트 목표

유즈 케이스

시스템 아키텍쳐

UI/UX 시나리오

테스트 케이스

4조

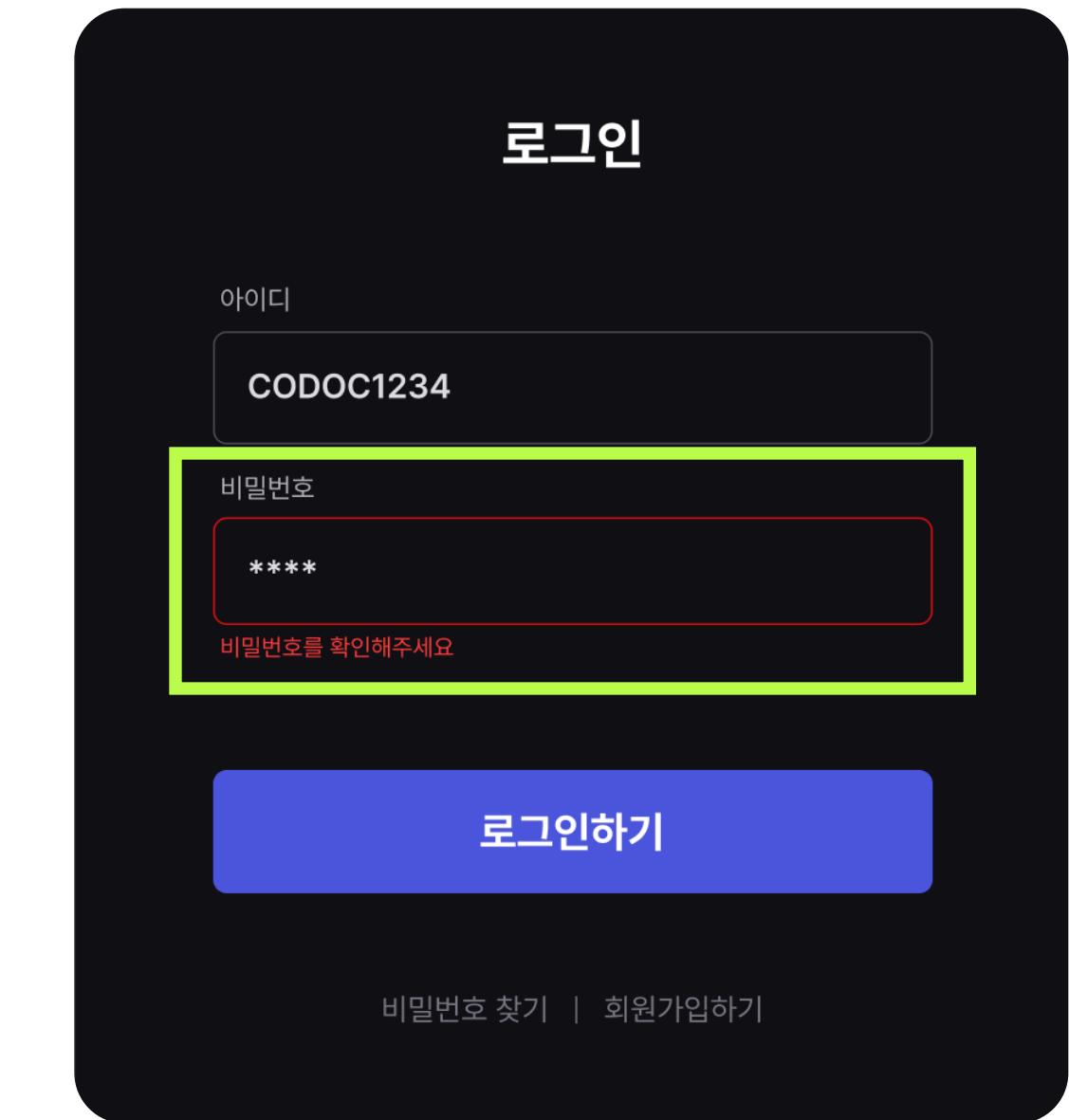
CODOC

로그인 테스트 케이스

Requirement : 로그인/로그아웃 요청은 2초 이내에 완료되어야 한다.

Reliability

```
// 로그인 요청  
IF 이메일 입력 없음 THEN  
    RETURN Error("이메일 입력 필요")  
  
IF 유효하지 않은 이메일 THEN  
    RETURN Error("유효하지 않은 이메일")  
  
user = FIND_USER_BY_EMAIL(이메일)  
  
IF user == null THEN  
    RETURN Error("존재하지 않는 이메일입니다")  
  
IF 비밀번호 입력 없음 THEN  
    RETURN Error("비밀번호 입력 필요")  
  
IF user.password != 입력된 비밀번호 THEN  
    RETURN Error("비밀번호를 다시 확인해주세요")  
  
RETURN "로그인 성공"
```



잘못된 비밀번호 입력 시 출력 화면

프로젝트 목표

유즈 케이스

시스템 아키텍쳐

UI/UX 시나리오

테스트 케이스

4조

CODOC

로그인 테스트 케이스

Requirement : 로그인/로그아웃 요청은 2초 이내에 완료되어야 한다.

Performance

```
// 로그인 요청  
start_time = 시작 시각 측정  
로그인 프로세스  
end_time = 종료 시각 측정
```

```
time = end_time - start_time
```

```
IF start_time == null OR end_time == null THEN  
RETURN Error("시간 측정 실패")
```

```
IF time < 0 THEN  
RETURN Error("time 측정 오류")
```

```
IF time > 2 THEN  
RETURN Error("시간 초과")  
RETURN "성공"
```

```
// 로그아웃 요청  
start_time = 시작 시각 측정  
로그아웃 프로세스  
end_time = 종료 시각 측정
```

```
time = end_time - start_time
```

```
IF start_time == null OR end_time == null THEN  
RETURN Error("시간 측정 실패")
```

```
IF time < 0 THEN  
RETURN Error("time 측정 오류")
```

```
IF time > 2 THEN  
RETURN Error("시간 초과")  
RETURN "성공"
```

프로젝트 목표

레벨테스트

회원가입 후 새로운 사용자의 코딩 실력 수준을 평가하여 학습 시작 수준을 결정해 맞춤형 기초 문제를 제공하는 기능

유즈 케이스

시스템 아키텍쳐

UI/UX 시나리오

테스트 케이스

CODOC

코딩 테스트 코딩 진단서 학습 노트

한 레벨 당 3 문제 제출

0/3 → 아래 레벨로 내려가서 테스트 진행
1 or 2/3 → 해당 레벨 확정
3/3 → 다음 레벨로 올라가서 테스트 진행

Python 3

히스토리

레벨 4 - 2번 문제
레벨 4 - 1번 문제

레벨 3 - 3번 문제

레벨 3 - 2번 문제

레벨 3 - 1번 문제

문제

팰린드롬이란, 앞에서부터 읽었을 때와, 뒤에서부터 읽었을 때가 같은 문자열이다.
모든 문자열이 팰린드롬이 아니기 때문에 다음과 같은 4가지 연산으로 보통 문자열을 팰린드롬으로 만든다.

1. 문자열의 어떤 위치에 어떤 문자를 삽입 (시작과 끝도 가능)
2. 어떤 위치에 있는 문자를 삭제
3. 어떤 위치에 있는 문자를 교환
4. 서로 다른 문자를 교환

1, 2, 3번 연산은 마음껏 사용할 수 있지만, 마지막 연산은 많아야 한 번 사용할 수 있다.
문자열이 주어졌을 때, 팰린드롬으로 만들기 위해 필요한 연산의 최솟값을 출력하는 프로그램을 작성하시오.

풀이

코드를 작성하세요

실행하기 제출하기

입력

첫째 줄에 문자열이 주어진다. 영어 소문자로만 이루어져 있고, 길이는 최대 50이다.

출력

첫째 줄에 문제의 정답을 출력한다.

실행 결과

실행 결과가 여기 표시됩니다

4조

CODOC

프로젝트 목표

코딩 테스트

사용자들이 문제에 대해 코드를 작성하고 이에 대한 피드백을 제공받는 기능

유즈 케이스

시스템 아키텍쳐

UI/UX 시나리오

테스트 케이스

4조

CODOC

The screenshot shows the CODOC platform interface. At the top, there is a navigation bar with the CODOC logo and links for '코딩 테스트', '코딩 진단서', and '학습 노트'. The main area is titled '사용할 언어 선택 기능' (Language Selection Function). It includes a dropdown menu set to 'Python 3' (highlighted with a green box), a '도움받기' (Help) button (also highlighted with a green box), and two buttons at the bottom right labeled '실행하기' (Run) and '제출하기' (Submit) (both highlighted with a green box). On the left, there is a '문제' (Problem) section containing text about palindromes and a list of 4 tasks. Below it is an '입력' (Input) section with sample input text. On the right, there is a '풀이' (Solution) section with Python code for solving the palindrome problem. A large green box highlights the entire right-hand sidebar area.

문제

팰린드롬이란, 앞에서부터 읽었을 때와, 뒤에서부터 읽었을 때가 같은 문자열이다.

모든 문자열이 팰린드롬이 아니기 때문에 다음과 같은 4가지 연산으로 보통 문자열을 팰린드롬으로 만든다.

1. 문자열의 어떤 위치에 어떤 문자를 삽입 (시작과 끝도 가능)
2. 어떤 위치에 있는 문자를 삭제
3. 어떤 위치에 있는 문자를 교환
4. 서로 다른 문자를 교환

1, 2, 3번 연산은 마음껏 사용할 수 있지만, 마지막 연산은 많아야 한 번 사용할 수 있다.

문자열이 주어졌을 때, 팰린드롬으로 만들기 위해 필요한 연산의 최솟값을 출력하는 프로그램을 작성하시오.

입력

첫째 줄에 문자열이 주어진다. 영어 소문자로만 이루어져 있고, 길이는 최대 500이다.

출력

첫째 줄에 문제의 정답을 출력한다.

풀이

```
# DP 베이스 초기화
dp = [[0] * n for _ in range(n)]

for length in range(2, n + 1):
    for i in range(n - length + 1):
        j = i + length - 1
        if s[i] == s[j]:
            dp[i][j] = dp[i + 1][j - 1]
        else:
            dp[i][j] = min(dp[i + 1][j], dp[i][j - 1]) + 1

return dp[0][n - 1]
```

실행 결과

실행 결과가 여기 표시됩니다.

사용할 언어 선택 기능

Python 3

도움받기

실행하기 제출하기

코드 메이트

도움이 필요하시면 알려 주세요!
함께 문제를 해결해 드릴게요. 😊

프로젝트 목표

유즈 케이스

시스템 아키텍쳐

UI/UX 시나리오

테스트 케이스

4조

CODOC

코딩 테스트 테스트 케이스

Requirement : 코드메이트의 응답 시간은 3초 이내에 이루어져야 한다

Function test_codemate_response_time:

Initialize the CodeMate system

Set the maximum response time to 3 seconds

Step 1: 사용자 입력 모니터링 테스트

Simulate user input in the input field

IF CodeMate fails to detect the input without delay:

 RETURN "실패: CodeMate가 사용자 입력을 즉시 감지하지 못했습니다."

ELSE

 PRINT("감지중")

Step 2: 트리거 작동 테스트

Simulate a trigger event (e.g., user request or code error)

Start measuring response time when the trigger is detected

IF CodeMate fails to detect and activate the trigger correctly:

 RETURN "실패: CodeMate 트리거가 올바르게 작동하지 않았습니다."

ELSE

 PRINT("작동시작")

Step 3: 코드 개선 및 오류 수정 제안 테스트

Pass sample code to CodeMate for analysis

Generate CodeMate Recommendation

Stop measuring response time when the response is generated

Calculate the measured response time

IF response time exceeds 3 seconds:

 RETURN "실패: CodeMate의 응답 시간이 3초를 초과했습니다."

RETURN "SUCCESS: CodeMate response time test passed."

END Function

프로젝트 목표

유즈 케이스

시스템 아키텍쳐

UI/UX 시나리오

테스트 케이스

테스트 케이스

코딩 테스트 테스트 케이스

CODOC

코딩 테스트 코딩 진단서 학습 노트

문제

팰린드롬이란, 앞에서부터 읽었을 때와, 뒤에서부터 읽었을 때가 같은 문자열이다. 모든 문자열이 팰린드롬이 아니기 때문에 다음과 같은 4가지 연산으로 보통 문자열을 팰린드롬으로 만든다.

1. 문자열의 어떤 위치에 어떤 문자를 삽입 (시작과 끝도 가능)
2. 어떤 위치에 있는 문자를 삭제
3. 어떤 위치에 있는 문자를 교환
4. 서로 다른 문자를 교환

1, 2, 3번 연산은 마음껏 사용할 수 있지만, 마지막 연산은 많아야 한 번 사용할 수 있다.

문자열이 주어졌을 때, 팰린드롬으로 만들기 위해 필요한 연산의 최솟값을 출력하는 프로그램을 작성하시오.

풀이

Python 3

```
# DP 네이밍 조기화
dp = [[0] * n for _ in range(n)]

for length in range(2, n + 1):
    for i in range(n - length + 1):
        j = i + length - 1
        if s[i] == s[j]:
            dp[i][j] = dp[i + 1][j - 1]
        else:
            dp[i][j] = min(dp[i + 1][j], dp[i][j - 1]) + 1

return dp[0][n - 1]
```

Code Mate

s[i]와 s[j]가 다를 때 두 가지 경우를 생각해 보시면 좋을 것 같아요. 각각의 경우에서 팰린드롬을 만들기 위해 필요한 연산 수를 비교해 보세요. 그리고 더 적은 쪽을 선택한 후에 1을 더해주시면 됩니다. 더 궁금한 점이 있으시면 언제든지 말씀해 주세요. 같이 고민해 드릴게요!

실행하기 제출하기

코드메이트 작동 시 출력 화면

4조

CODOC

프로젝트 목표

유즈 케이스

시스템 아키텍쳐

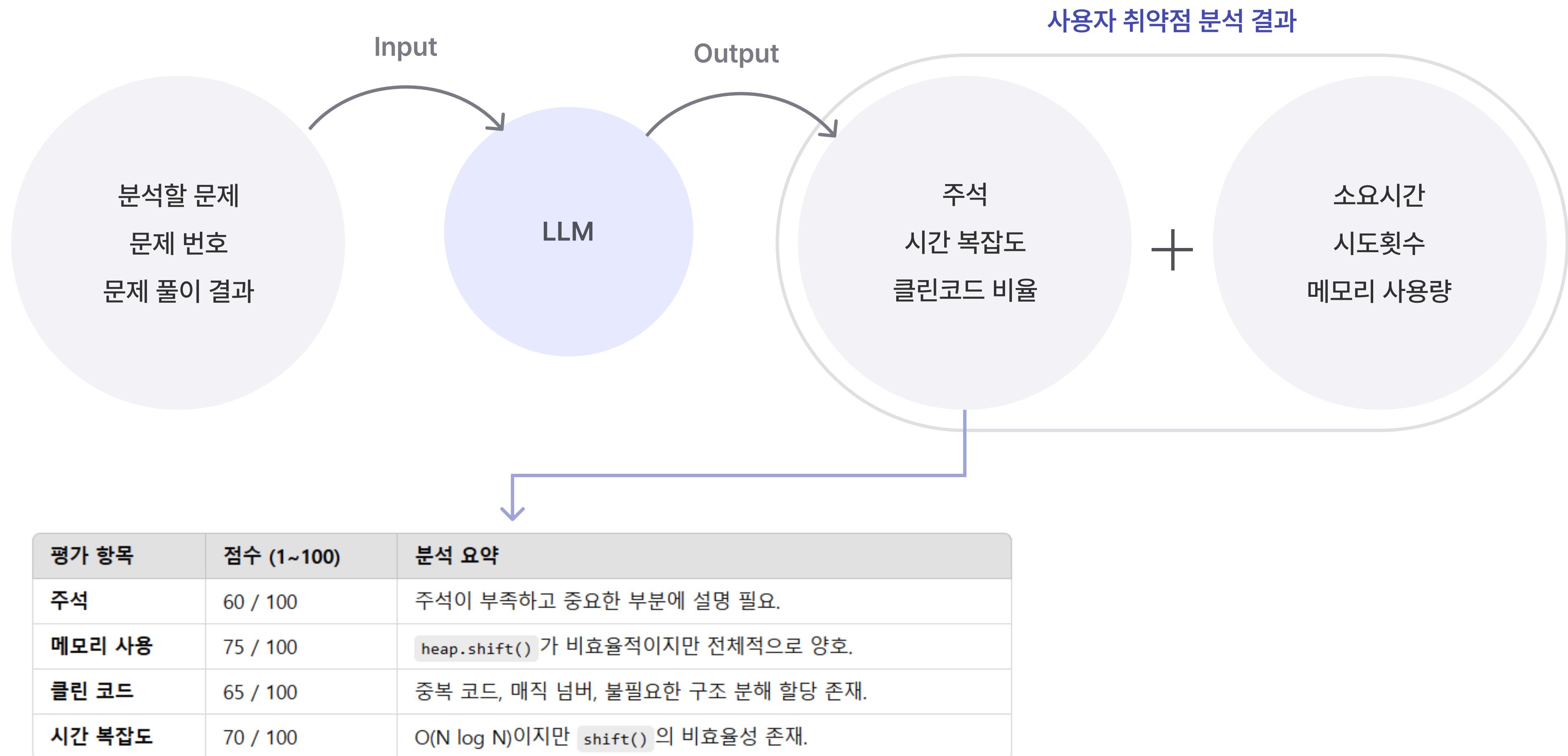
UI/UX 시나리오

테스트 케이스

4조
CODOC

사용자 취약점 분석

사용자가 작성한 코드의 개선점을 6가지 측면에서 측정해서 보여주고 그에 따른 추천 문제를 제공하는 기능



LLM을 이용한 코드 분석 예시

프로젝트 목표

코딩 진단서

사용자의 학습 성취도와 코딩 역량을 시각화하여 제공해 사용자의 성과와 진행 상황 확인을 돋는 기능

유즈 케이스

시스템 아키텍쳐

UI/UX 시나리오

테스트 케이스

테스트 케이스

4조
CODOC



프로젝트 목표

코딩 진단서

사용자의 학습 성취도와 코딩 역량을 시각화하여 제공해 사용자의 성과와 진행 상황 확인을 돋는 기능

유즈 케이스

시스템 아키텍쳐

UI/UX 시나리오

테스트 케이스

4조
CODOC



프로젝트 목표

학습 노트

사용자가 과거에 풀었던 기록을 확인할 수 있는 페이지로, 코드 복습 및 코드 최적화 아이디어를 제공

유즈 케이스

시스템 아키텍쳐

UI/UX 시나리오

테스트 케이스

The screenshot shows the CODOC application interface. At the top, there is a navigation bar with tabs: 'CODOC' (highlighted), '코딩 테스트', '코딩 진단서', and '학습 노트'. On the left, there's a sidebar with sections for 'UI/UX 시나리오', '테스트 케이스', and '시스템 아키텍쳐'. The main area displays a '히스토리' (History) section with several solved problems listed:

- 레벨 3 - 문자열 알고리즘
팰린드롬이란, 앞에서부터 읽었을 때와, 뒤에서부터 읽었을 것이다.
- 레벨 3 - 수학적 계산
린드롬이 아니기 때문에 다음과 같은 4가지 자열을 팰린드롬으로 만든다.
두 개의 정수가 주어졌을 때, 그들...
- 레벨 3 - 스택 / 문자열 처리
주어진 문자열에서 괄호('(', ')', '{', ...')' 위치에 어떤 문자를 삽입 (시작과 끝도 가능)
- 레벨 2 - 문자열 처리
입력이 "hello"라면 출력은 "olleh"이다.
- 레벨 2 - 배열 및 리스트 처리
주어진 정수 배열에서 짝수와 홀수만 추출하는 프로그램을 작성하시오.

Below the history, there is a note: '이 주어진다. 영어 소문자로만 이루어져 있다.' (This is given. It consists only of lowercase English letters.) and a placeholder '정답을 출력한다.' (Outputs the answer).

In the center, there is a '풀이 코드' (Solution Code) section for a problem in Python 3:

```
#include <iostream>
#include <vector>

using namespace std;

vector <int> findPrimes(int n) {
    vector <int> primes;
    for (int num = 2; primes.size() < n; num++) {
        bool isPrime = true;
        for (int P : primes) {
            if (num % P == 0) {
                isPrime = false;
                break;
            }
        }
        if (isPrime) {
            primes.push_back(num); // 소수 추가
        }
    }
}
```

To the right, there is a '관련 자료 및 참고 링크' (Related Materials and References) section with a '소스' (Sources) subsection:

- [C/C++] 소수 구하기 (for 문, 재귀, 에라토스테네스의 체)
- [C++] n번째 소수 찾기
- [C++] 소수 최적화 코드
- [C++] 소수 알고리즘 개선

A green callout box highlights this section with the text '코드의 어느 부분에 관한 기록인지 표시' (Indicates which part of the code is recorded).

On the far right, there is a '코드 메이트 기록' (Code Mate Log) section with hints:

- [힌트 1] 소스코드 최적화 제안
- [힌트 2] 기본적인 에러 처리 추가
- [힌트 3] 시간 복잡도 개선 필요
- [힌트 4] n번째 소수 구하기 알고리즘 개선

4조

CODOC

프로젝트 목표

사용자의 회원가입 및 계정 정보 저장은 각각 2초 이내로 완료되어야 한다.

유즈 케이스

시스템 아키텍쳐

UI/UX 시나리오

테스트 케이스

4조

CODOC

```
FUNCTION SIGN_UP(email, nickname, password, level, language):
    # 유효성 검사
    IF email IS EMPTY OR nickname IS EMPTY OR password IS EMPTY THEN
        RETURN Error("모든 필드를 입력해주세요.")
    END IF

    # 이메일 유효성 검사
    IF email IS NOT VALID THEN
        RETURN Error("유효하지 않은 이메일")
    END IF

    # 중복 체크
    IF USER_EXISTS(email) THEN
        RETURN Error("이미 존재하는 이메일입니다.")
    END IF

    # 사용자 객체 생성
    user = NEW User(email, nickname, password, level, language)

    # 사용자 ID 생성
    user.create_id()

    # 처리 시간 측정 시작
    START TIMER

    # 데이터베이스에 사용자 정보 저장
    SAVE_USER_TO_DATABASE(user)

    # 처리 시간 측정 종료
    END TIMER

    # 처리 시간 확인
    IF TIMER_DURATION > 2 SECONDS THEN
        RETURN Error("요청 처리 시간이 2초를 초과했습니다.")
    END IF

    # 성공 메시지 반환
    RETURN "회원가입 성공"
END FUNCTION
```

프로젝트 목표

유즈 케이스

시스템 아키텍쳐

UI/UX 시나리오

테스트 케이스

4조
CODOC

사용자의 회원가입 및 계정 정보 저장은 각각 2초 이내로 완료되어야 한다.

Register Case 1 - Missing field

Success case

Input	"swe@skku.edu", "율전이", "1", "beginner", "C"
Output	회원가입 성공

Failure case

Input	"", "율전이", "1", "beginner", "Python" "swe@skku.edu", "", "1", "beginner", "Python" "swe@skku.edu", "율전이", "", "beginner", "Python" "swe@skku.edu", "율전이", "1", "", "Python" "swe@skku.edu", "율전이", "1", "beginner", "" "", "율전이", "", "beginner", "Python" 등
Output	회원가입 성공

Register Case 2 - Invalid email field

Failure case

Input	"swe.naver", "", "5", "beginner", "Python"
Output	유효하지 않은 이메일입니다

Register Case 3 - Duplicated email field

Failure case

Input	"db_inuse@email.edu", "", "5", "beginner", "Python"
Output	이미 존재하는 이메일입니다

Register Case 4 - Time out

Failure case

Input	"swe@skku.edu", "아돌프 블레인 찰스 데이비드 얼 프레더릭 제럴드 하버트 어빈 존 캐네스 로이드 마틴 니로 올리버 폴 퀸시 랜돌프 셔먼 토머스 엉커스 빅터 월리엄 저크시스 앤 시 주스 볼페슬레겔슈타인하우zenbe르거도르프포어알테른바렌ge비센하프트샤페르스베센샤페바렌볼게플레ge운트조르크팔티히카이트베슈첸폰안그라이펜두르하이르라우프기리히파인데벨헤포어알테른츠볼프타우젠타야레스포어안디에르사이넨반더에르스테르뎀엔슈데어라움시프게브라우흘리히트알스자인우어슈프룽폰크라프트게슈타르트자인랑에파르트힌츠비센슈테른아르티그라움아우프데어주헤나흐디슈테른벨헤ge하프트베본바르플라네텐크라이", "1", "beginner", "Python"
Output	요청처리 시간이 2초를 초과했습니다

프로젝트 목표

CODOC 시스템은 최대 10,000명의 사용자 계정을 관리할 수 있어야 한다

유즈 케이스

시스템 아키텍쳐

UI/UX 시나리오

테스트 케이스

4조

CODOC

Function test_user_management:

 Initialize the CODOC system
 Set maximum users to 10,000

FOR each user ID from 1 to maximum users:

 Add a user with the ID to the system

Verify the total number of users in the system is 10,000

IF not:

 RETURN "FAIL IN ADDING USER FOR maximum users "

FOR each user ID from 1 to maximum users:

 Read the user with the ID from the system
 Verify the user exists

FOR each user ID from 1 to maximum users:

 Delete the user with the ID from the system

Verify the total number of users in the system is 0

IF not:

 RETURN "FAIL IN DELETING USER FOR maximum users"

 RETURN "테스트 성공: User management functions correctly for maximum users."

END Function

프로젝트 목표

유즈 케이스

시스템 아키텍쳐

UI/UX 시나리오

테스트 케이스

4조
CODOC

CODOC 시스템은 최대 10,000명의 사용자 계정을 관리할 수 있어야 한다

User Management Case - Num Of User

Success case

Input	Set maximum users to 5000
-------	---------------------------

Output	성공: User management functions correctly for 5000
--------	--

Success case

Input	Set maximum users to 1
-------	------------------------

Output	성공: User management functions correctly for 1
--------	---

Success case

Input	Set maximum users to 10000
-------	----------------------------

Output	성공: User management functions correctly for 10000
--------	---

Failure case

Input	Set maximum users to 10001
-------	----------------------------

Output	"FAIL IN ADDING USER FOR 10001 users "
--------	--

Failure case

Input	Set maximum users to 20000
-------	----------------------------

Output	"FAIL IN ADDING USER FOR 20000 users "
--------	--

Failure case

Input	Set maximum users to -1
-------	-------------------------

Output	"FAIL IN DELETING USER FOR -1 users "
--------	---------------------------------------

Failure case

Input	Set maximum users to 0
-------	------------------------

Output	"FAIL IN ADDING/DELETING USER FOR 0 users "
--------	---

프로젝트 목표

문제를 데이터베이스에서 불러오는 시간은 3초 이내여야 한다

유즈 케이스

시스템 아키텍쳐

UI/UX 시나리오

테스트 케이스

Reliability

problem_key = 사용자가 선택한 문제의 고유 키
DB에 problem_key와 고유 키가 일치하는 문제 요청
problem = DB가 보내준 problem의 내용을 저장하는 dictionary

IF problem == null
RETURN Error ("문제를 불러오지 못했습니다")

문제 내용 출력

Performance

start_time = 시작 시간 측정
DB에 문제 내용 요청
end_time = 종료 시간 측정
time = end_time - start_time

IF start_time == null OR end_time == null THEN
RETURN Error("시간 측정 실패")

IF time < 0 THEN
RETURN Error("time 측정 오류")

IF time > 2 THEN
RETURN Error("시간 초과")
RETURN "성공"

4조

CODOC

프로젝트 목표

문제 채점 결과는 2초 이내에 데이터베이스에 저장되고 사용자에게 알려져야 한다

유즈 케이스

시스템 아키텍쳐

Reliability

```
IF 문제 채점 결과 == null OR 문제 채점 결과 IS EMPTY THEN  
RETURN Error("채점 결과 없음")
```

UI/UX 시나리오

DB_저장_결과 = 데이터베이스에 문제 채점 결과 저장

테스트 케이스

```
IF DB_저장_결과 IS FAIL THEN  
RETURN Error("데이터베이스 저장 실패")
```

알림_전송_결과 = 사용자에게 저장 완료 알림

```
IF 알림_전송_결과 IS FAIL THEN  
RETURN Error("알림 전송 실패")
```

RETURN "채점 결과 저장 성공"

Performance

start_time = 시작 시각 측정

데이터베이스에 문제 채점 결과 저장
사용자에게 저장 완료 알림

end_time = 종료 시각 측정

time = end_time - start_time

```
IF start_time == null OR end_time == null THEN  
RETURN Error("시간 측정 실패")
```

```
IF time < 0 THEN  
RETURN Error("time 측정 오류")
```

```
IF time > 2 THEN  
RETURN Error("시간 초과")  
RETURN "성공"
```

4조

CODOC

프로젝트 목표

새로운 문제 생성 시간은 30초 이내로 설정되어야 한다

유즈 케이스

시스템 아키텍쳐

Reliability

```
prob_remain = DB에 저장되어 있는 문제들 중 안 풀린 문제 개수  
IF prob_remain < 50  
    LLM으로 새로운 문제 생성
```

UI/UX 시나리오

테스트 케이스

Performance

```
start_time = 시작 시간 측정  
문제 생성  
end_time = 종료 시간 측정  
time = end_time - start_time  
IF start_time == null OR end_time == null THEN  
    RETURN Error("시간 측정 실패")  
IF time < 0 THEN  
    RETURN Error("time 측정 오류")  
IF time > 30 THEN  
    RETURN Error("시간 초과")  
RETURN "성공"  
\\
```

4조

CODOC

프로젝트 목표

사용자 레벨 반영은 문제 풀이 후 3초 이내에 이루어져야 한다

유즈 케이스

시스템 아키텍쳐

Reliability

highest_level = 각 레벨 당 20개 이상의 문제를 해결한 레벨
중에서 가장 높은 레벨

Performance

문제 해결 과정 DB의 사용자 코딩 히스토리에 저장
start_time = 시작 시간 측정

UI/UX 시나리오

user_level = 현재 사용자의 레벨
if most_level == user_level
 user_level 유지
 한단계 위 레벨의 문제 추천
else if most_level > user_level
 user_level = most_level

사용자 레벨 계산 프로세스

end_time = 종료 시간 측정
time = end_time - start_time

테스트 케이스

IF start_time == null OR end_time == null THEN
RETURN Error("시간 측정 실패")

IF time < 0 THEN
RETURN Error("time 측정 오류")

IF time > 2 THEN
RETURN Error("시간 초과")
RETURN "성공"

4조

CODOC

프로젝트 목표

유즈 케이스

시스템 아키텍쳐

UI/UX 시나리오

테스트 케이스

4조

CODOC

코딩테스트의 각 테스트 케이스 실행은 3초 이내에 수행되어야 한다

Reliability

```
test_cases = FIND_TESTCASE_BY_PROBLEM_ID(problem_id)
```

```
IF test_cases == null OR test_cases.length == 0 THEN  
RETURN Error("테스트 케이스 없음")
```

```
실패한_테스트_케이스_목록 = [ ]
```

```
FOR test_case IN test_cases
```

```
    실행_결과 = 실행(사용자 작성 코드,test_case.input)
```

```
    IF 실행_결과 == test_case.output THEN  
        실패한_테스트_케이스_목록.append(test_case)
```

```
    IF 실패한_테스트_케이스_목록.length > 0 THEN  
        RETURN Error("테스트 실패", 실패한_테스트_케이스_목록)
```

```
ELSE
```

```
    RETURN "테스트케이스 실행 성공"
```

Performance

```
test_cases = FIND_TESTCASE_BY_PROBLEM_ID(problem_id)
```

```
IF test_cases == null OR test_cases.length == 0 THEN  
RETURN Error("테스트 케이스 없음")
```

```
실패한_테스트_케이스_목록 = [ ]
```

```
FOR test_case IN test_cases  
    start_time = 시작 시각 측정
```

```
    실행_결과 = 실행(사용자 작성 코드, test_case.input)
```

```
    IF (현재_시각 -start_time) > 5초 THEN  
        RETURN Error("테스트 케이스 실행 시간 초과",test_case)
```

```
    IF 실행_결과 == test_case.output THEN  
        실패한_테스트_케이스_목록.append(test_case)
```

```
    IF 실패한_테스트_케이스_목록.length > 0 THEN  
        RETURN Error("테스트 실패", 실패한_테스트_케이스_목록)
```

```
ELSE
```

```
    RETURN "테스트케이스 실행 성공"
```

프로젝트 목표

Security - 사용자 ID 및 비밀번호 암호화

유즈 케이스

시스템 아키텍쳐

```
// 회원가입 혹은 비밀번호 변경  
// 비밀번호 입력 성공 이후  
hashed_password = HASH(입력된 비밀번호)
```

UI/UX 시나리오

```
IF hashed_password == null THEN  
RETURN Error("비밀번호 암호화 실패")  
SAVE_USER_PASSWORD(user_id, hashed_password)
```

RETURN "비밀번호 암호화 및 저장 완료"

테스트 케이스

```
// 로그인 시도  
// 비밀번호 입력 성공 이후  
IF HASH_MATCHES(user.password, 입력된 비밀번호) == False THEN  
RETURN Error("이메일 또는 비밀번호가 잘못되었습니다")
```

RETURN "로그인 성공"

4조

CODOC

프로젝트 목표

Security - 외부 시스템의 데이터베이스 접근을 차단

유즈 케이스

시스템 아키텍쳐

```
// 데이터베이스 접근 차단  
IF 외부 IP 요청 THEN  
RETURN Error("외부 접근이 차단되었습니다.")
```

UI/UX 시나리오

```
// 예외 처리  
IF 요청 IP == null THEN  
RETURN Error("유효하지 않은 접근")
```

테스트 케이스

```
// 내부 접근 허용  
IF 내부 네트워크 요청 THEN  
GRANT ACCESS  
RETURN "내부 네트워크 접근 허용"
```

```
// 외부 접근 요청  
request_ip = GET_REQUEST_IP()
```

```
IF request_ip NOT IN 허용된 내부 IP 목록 THEN  
LOG("외부 접근 시도 감지: " + request_ip)  
RETURN Error("외부 접근이 차단되었습니다.")
```

4조

CODOC

감사합니다

Q&A