**Test Case & Pseudocode**

Performance Test는 아래의 pseudocode로 진행한다.

텍스트, 스크린샷, 폰트, 라인이(가) 표시된 사진

자동 생성된 설명

**Figure 1**: Performance Test Pseudocode

**1 회원 가입**

**1.1 Test Case**

Description: 사용자가 자신의 정보를 바탕으로 본 서비스에 회원 가입한다.

Pre-condition: 사용자가 시스템에 등록되지 않은 상태이다.

<CASE 1 (정상 가입)>

Input: name=“이승민”, phone\_Number=”123-4567-8910”, email=”lsm&gmail.com”, id=”lsm”, password=”12341234”, domain\_Interested = [“Movie”, “Actor”]

Steps: 회원가입 페이지로 이동 -> 미등록 확인 -> 모든 정보를 위와 같이 입력 -> 입력된 전화번호로 본인 인증을 진행 -> 완료 버튼 클릭 -> 사용자 정보를 데이터베이스에 저장.

Output: "회원가입이 완료되었습니다.”

<CASE 2 (전화번호 형식 오류)>

Input: name=“이승민”, phone\_Number=”123-가나다라-8910”, email=”lsm&gmail.com”, id=”lsm”, password=”12341234”, domain\_Interested = [“Movie”, “Actor”]

Steps: 회원가입 페이지로 이동 -> 미등록 확인 -> 모든 정보를 위와 같이 입력 -> 오류 메시지 출력

Output: “형식에 맞지 않습니다. 다시 입력해주세요.”

**1.2. Pseudocode**

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명

**Figure 2**: User Registration Pseudocode

**2. Log-In/Logout**

**2.1 Test Case**

Description: 사용자가 서비스에 Log-In 하고 Log-Out 한다.

Pre-condition: Log-In을 위해서 사용자가 서비스에 가입되어 있어야 한다.

<CASE 1 (정상 로그인)>

Input: id=“lsm”, password=”12341234”

Steps: 로그인 화면에서 id, password 입력 -> 등록 확인 -> 비밀번호 확인 -> Log-In 성공

Output: “로그인 성공”

<CASE 2 (미등록 로그인)>

Input: id=“lsm”, password=”12341234444444”

Steps: 로그인 화면에서 id, password 입력 -> 미등록 확인 -> 오류 메시지 출력 -> 로그인 페이지로 Re-direction

Output: “입력한 정보가 올바르지 않습니다. 다시 한번 확인하여 주세요.”

<CASE 3 (정상 로그아웃)>

Input: id=“lsm”

Steps: 로그아웃 버튼 클릭 -> 로그인 확인 -> 로그아웃 -> 로그인 페이지로 Re-direction

Output: “로그아웃 완료”

**2.2 Pseudocode**

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명

**Figure 3**: User Login Pseudocode

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명

**Figure 4**: User Logout Pseudocode

**3.1 Test Case**

Description: 사용자가 자신의 프로필 정보(특히, 관심사)를 수정한다.

Pre-condition: 사용자가 Log-In 된 상태여야 한다.

<CASE 1 (관심 분야 업데이트)>

Input: id=”lsm”, domain\_interested = “Economics”

Steps: 프로필 버튼 클릭 -> 관심사 수정 버튼 클릭 -> 로그인 확인 -> 새로운 관심사 입력 -> 정보 업데이트

Output: “정보가 업데이트 되었습니다.”

<CASE 2 (관심 분야 형식 오류)>

Input: : id=”lsm”, domain\_interested = 111%@!@#

Steps: 프로필 버튼 클릭 -> 관심사 수정 버튼 클릭 -> 로그인 확인 -> 새로운 관심사 입력 -> 형식 오류 확인 -> 오류 메시지 출력

Output: “형식에 맞지 않습니다. 다시 입력해주세요.”

**3.2 Pseudocode**

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명

**Figure 5**: Update Profile Pseudocode

**4 문제 제출**

**4.1 Test Case**

Description: 사용자가 문제 풀이 후 작성한 코드를 제출한다.

Pre-condition: 작성한 코드는 비어 있지 않아야 한다.

<CASE 1 (코드 제출)>

Input: id=”lsm”, problem\_id = 1, code = “print(“hello\_world \n”)”

Steps: 제출 버튼 클릭 -> 제출 코드가 비어 있는지 확인 -> 서버에 제출 -> 제출된 코드 저장

Output: “제출 완료”

<CASE 2 (NULL 코드 제출)>

Input: id=”lsm”, problem\_id = 1, code = NULL

Steps: 제출 버튼 클릭 -> 제출 코드 NULL 확인 -> 오류 메시지 출력

**4.2 Pseudocode**

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명

**Figure 6**: Submit Code Process Pseudocode

**5 단계별 Hint 제공**

**5.1 Test Case**

Description: 사용자가 단계적으로 단계별 Hint 기능을 사용한다.

Pre-condition: 사용자가 같은 문제에 대해 문제 힌트 기능을 3회 이하 사용한 상태이다. 사용자가 LLM 생성 기능을 하루 20회 이하 사용한 상태이다.

<CASE 1 (2단계 요청)>

Input: id=”lsm”, problem\_id = 1

Steps: 힌트 요청 버튼 클릭 -> LLM 횟수제한 검사 -> 힌트 단계 (2단계) 식별 -> 힌트 단계 및 문제 정보를 LLM에 전달하고, 해당 힌트 생성 요청 -> 생성된 힌트를 받아 사용자에게 출력

Output: 생성된 2단계 힌트. Ex) “수의 범위가 명확하게 정의되어 있어, 배열이나 해시맵을 사용해 효율적으로 처리할 수 있습니다.”

<CASE 2 (4회 이상 요청)>

Input: id=”lsm”, problem\_id = 1

Steps: 힌트 요청 버튼 클릭 -> LLM 횟수제한 검사 -> 힌트 단계 범위 (1~3단계) 초과 -> 오류 메시지 출력 -> 관련 강의 및 자료 링크 탐색 -> 해당 링크 출력

Output: “더 이상 제공될 수 있는 힌트가 없습니다. 관련된 내용을 다시 학습하세요.”, 관련 링크. Ex) [www.xxxx.xxxx](http://www.xxxx.xxxx)

<CASE 3 (LLM 기능 20회 이상 요청)>

Input: id=”lsm”, problem\_id = 1

Steps: 힌트 요청 버튼 클릭 -> LLM 횟수제한 검사 -> 오류 메시지 출력

Output: “오늘 LLM 생성 기능을 최대(20회)로 사용하셨습니다. 내일 다시 시도해 주세요.”

**5.2 Pseudocode**

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명

**Figure 7**: Request Hint Pseudocode

**6 변형 문제 제공**

**6.1 Test Case**

Description: 사용자가 문제에 대한 변형 기능을 사용한다.

Pre-condition: 사용자가 해당 문제를 이미 풀었고, 같은 문제에 대해 변형 문제 제공 기능을 5회 이하 요청했다. 사용자가 LLM 생성 기능을 하루 20회 이하 사용한 상태이다.

<CASE 1 (변형 문제 생성)>

Input: id=”lsm”, problem\_id = 1

Steps: 사용자가 변형 문제 생성 버튼 클릭 -> LLM 횟수제한 검사 -> 문제 풀이 완료 인식 -> 5회 이하 요청 인식 -> LLM에 해당 문제 정보 및 변형 요청 전달 -> LLM이 생성한 변형 문제를 사용자에게 출력하고, 해당 문제의 테스트 코드 및 케이스를 호스트 서버에 저장

Output: 생성된 변형 문제, 해당 문제의 테스트 코드 및 테스트 케이스

Ex)

-원래 문제

def add\_numbers(a, b):

"""두 수를 더한 결과를 반환합니다."""

return

-변형 문제

def multiply\_numbers(a, b):

"""두 수를 곱한 결과를 반환합니다."""

return

-테스트 코드 및 케이스

assert multiply\_numbers(2, 3) == 6

assert multiply\_numbers(0, 5) == 0

assert multiply\_numbers(-1, 4) == -4

<CASE 2 (사용자가 문제 풀이 미완료)>

Input: id=”lsm”, problem\_id = 1

Steps: 사용자가 변형 문제 생성 버튼 클릭 -> LLM 횟수제한 검사 -> 문제 풀이 미완료 인식 -> 오류 메시지 출력

Output: “아직 문제 풀이를 완료하지 않았습니다. 문제 풀이를 완료한 후, 요청하여 주시기 바랍니다.”

<CASE 3 (변형 문제 5회 초과 요청)>

Input: id=”lsm”, problem\_id = 1

Steps: 사용자가 변형 문제 생성 버튼 클릭 -> LLM 횟수제한 검사 -> 문제 풀이 완료 인식 -> 5회 초과 요청 인식 -> 오류 메시지 출력

Output: “더 이상 변형할 수 없습니다. 다른 문제를 시도해보세요.”

<CASE 4 (LLM 기능 20회 이상 요청)>

Input: id=”lsm”, problem\_id = 1

Steps: 사용자가 변형 문제 생성 버튼 클릭 -> LLM 횟수제한 검사 -> 오류 메시지 출력

Output: “오늘 LLM 생성 기능을 최대(20회)로 사용하셨습니다. 내일 다시 시도해 주세요.”

**6.2 Pseudocode**

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명

**Figure 8**: Generate Modified Problem Pseudocode

**7 맞춤형 문제 제공–관심 분야 관련 문제**

**7.1 Test Case**

Description: 사용자가 맞춤형 종합 문제 제공 기능을 사용한다.

Pre-condition: 사용자가 Log-In 되어 있고, LLM과 호스트 서버가 연결되어 있는 상태이며, 학습자의 관심 분야가 학습자의 개인정보에 등록되어 있어야 한다. 사용자가 LLM 생성 기능을 하루 20회 이하 사용한 상태이다.

<CASE 1 (정상 생성)>

Input: id=”lsm”, want\_alg = [“파일\_입출력”]

Steps: 사용자가 맞춤형 문제 생성 버튼 클릭 -> LLM 횟수제한 검사 -> 로그인 확인 -> 관심 분야 등록 확인 -> 사용자에게 정의된 여러 알고리즘들 중 선택 요청 -> 사용자가 관심 알고리즘을 선택하고, 이 외에 관심있는 알고리즘을 추가적으로 입력 -> 해당 정보들과 학습자의 관심 분야를 바탕으로 LLM에게 맞춤형 문제 및 테스트코드 & 테스트케이스 생성 요청 -> 생성된 문제를 사용자에게 출력 -> 호스트 서버에 테스트코드 & 테스트 케이스 저장

Output: 생성된 맞춤형 문제, 해당 문제의 테스트 코드 및 테스트 케이스

Ex) 관심 분야가 영화인 경우

- 맞춤형 문제

import json

def print\_movie\_titles(filename):

"""영화 목록 파일에서 제목을 출력하는 함수"""

return

-테스트 코드

def test\_print\_movie\_titles():

# 테스트용 영화 목록을 movies.json 파일로 저장

movies = [ {'title': '인터스텔라', 'year': 2014}, {'title': '어벤져스: 엔드게임', 'year': 2019}, {'title': '기생충', 'year': 2019} ]

with open('movies.json', 'w', encoding='utf-8') as file:

json.dump(movies, file, ensure\_ascii=False, indent=4)

# 출력 결과를 확인하기 위한 테스트

import io

import sys

captured\_output = io.StringIO()

sys.stdout = captured\_output

print\_movie\_titles('movies.json')

# 원래 출력으로 돌려놓기

sys.stdout = sys.\_\_stdout\_\_

# 예상된 출력과 비교

expected\_output = "인터스텔라\n어벤져스: 엔드게임\n기생충\n"

assert captured\_output.getvalue() == expected\_output

-테스트 케이스

INPUT:

movies.json 파일

OUTPUT:

인터스텔라

어벤져스: 엔드게임

기생충

<CASE 2 (관심 분야 미등록)>

Input: id=”lsm”, want\_alg = [“파일\_입출력”]

Steps: 사용자가 맞춤형 문제 생성 버튼 클릭 -> LLM 횟수제한 검사 -> 로그인 확인 -> 관심 분야 미등록 확인 -> 오류 메시지 출력

Output: "학습자의 관심 분야가 등록되어 있지 않습니다. 먼저 관심 분야를 선택하여 등록해주시기 바랍니다."

<CASE 3 (LLM 기능 20회 이상 요청)>

Input: id=”lsm”, want\_alg = [“파일\_입출력”]

Steps: 사용자가 맞춤형 문제 생성 버튼 클릭 -> LLM 횟수제한 검사 -> 로그인 확인 -> 오류 메시지 출력

Output: “오늘 LLM 생성 기능을 최대(20회)로 사용하셨습니다. 내일 다시 시도해 주세요.”

**7.2 Pseudocode**

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명

**Figure 9**: Generate Custom Problem Pseudocode