Green Coding
요구사항 명세서

소프트웨어공학개론

3조

김지수 배경엽 이동국 이유진 임현서 조준형

Contents

Ⅰ. Introduction

1.1 Purpose

본 문서는 세계의 프로그래머를 대상으로 한다. 코드의 탄소 배출량을 줄이는 refactoring 서비스를 제공하는 web-application의 requirement를 명세화하며 추후 발생할 maintainance를 위해 작성되었다.

1.2 Scope

그린 코딩은 유저에게 Java 코드를 입력받아 입력받은 코드의 실행 시간과 메모리 사용량을 측정한 뒤 탄소 발자국 공식에 따라 탄소 배출량을 계산하여 유저에게 보여주는 웹 서비스이다.

2개의 페이지로 구성되어 있으며, 첫번째는 메인으로서 코드의 입력과 수정된 코드의 출력을 보여주며, 알고리즘 분석에 따라 불필요한 코드 패턴을 하이라이팅하고, 탄소 배출량을 시각화하여 보여준다. 두번째는 게시판으로서 유저들이 코드를 공유하고 싶을 때 코드가 어떤 식으로 수정이 되었는 지 특정 패턴에 따라 분류해놓은 게시물을 보여준다.

1.3 Definitions, acronyms, and abbreviations

탄소 발자국**(**탄소 배출량**):** 활동, 제품, 회사 또는 국가가 대기에 추가하는 온실가스의 총량을 비교할 수 있게 해주는 계산된 값 또는 지수.

carbon footprint(탄소 발자국) = energy needed * carbon intensity,

energy needed = runtime * **PUE** * **PSF**

* (power draw for cores * usage + power draw for memory)

PUE: Power Usage Effectiveness. 데이터센터에서 운영하는 데 필요한 추가 에너지(냉방, 조명 등).

PSF: Pragmatic Scaling Factor. 여러 개의 동일한 실행(예: 테스트 또는 최적화)을 고려하는 데 사용

1.4 References

Lannelongue, L., Grealey, J. & Inouye, M., ‘Green Algorithms: Quantifying the Carbon Footprint of Computation’, Advanced Science, 8(12), 2021., https://doi.org/10.1002/advs.202100707

The IEEE standard for requirements documents, https://ifs.host.cs.st-andrews.ac.uk/Books/SE9/Web/Requirements/IEEE-standard.html

생활 속 탄소발자국 계산기, https://www.kcen.kr/tanso/home.green

1.5 Overview

본 요구사항 명세서는 네 챕터로 구성되어 있다.

첫 번째 챕터는 소개글로서 소프트웨어공학개론 3팀이 제안하는 Green Coding 시스템과 요구사항 명세서의 목적과 시스템이 제공하는 서비스를 설명한다. 또한 본 문서에서 사용하는 용어 및 약어에 대한 보충 설명을 포함하고 있다.

두 번째 챕터는 전반적인 설명으로서 시스템과 요구 사항에 영향을 미치는 일반적인 요소를 설명한다. 예를 들어 시스템의 개발 동기/효과 및 주요 기능에 대한 요약, 유저의 유형과 일반적인 특성, 제약 사항들을 포함한다.

세 번째 챕터는 외부 인터페이스 요구사항으로서 시스템의 세부 요구사항 및 특성에 대한 설명이다. 입출력에 대한 정보, 처리하는 기능들, 시스템 또는 시스템과 사람 간의 상호 작용에 적용되는 요구사항, 데이터베이스에 저장될 논리적 요구사항, 디자인 제약사항, 시스템 특성들을 포함한다.

4번째 챕터에서는 본 문서를 작성하기 위해 참고한 참조 문헌들의 목록이다.

Ⅱ. Overall Description

2.1 product perspective

“Green Coding”은 이산화탄소로 지구 온난화가 가속화되고 있는 현재에 탄소 배출량을 줄이기 위한 하나의 활동으로서 지속 가능한 코드 개발을 위해 개발하는 소프트웨어의 탄소 배출량을 측정하여 가시화하는 프로그램이다.

웹 서비스에 코드를 입력하면 백엔드에서 코드의 저장 및 탄소 배출 패턴 탐지를 위한 리팩토링 작업이 일어나고, 검출된 패턴을 수정하여 다시 사용자에게 전달한다.

여기서 코드의 리팩토링은 가독성 및 유지보수성 향상과 성능 개선 등 많은 효과를 볼 수 있다는 이점이 있다. 특히 여러 번 사용되는 프로그램에 대해 코드의 리팩토링을 사용한 횟수만큼 배로 효과를 볼 수가 있다. 리팩토링의 장점을 살려 웹 서비스에 접목시켜 코드를 입력하면 자동적으로 특정 패턴을 검출하여 코드를 효율적으로 수정한다. 웹 서비스의 개략적인 과정은 다음과 같다.

2.1.1. System Interfaces

웹 서비스에 코드를 입력하면 서버에서 코드의 저장 및 탄소 배출 패턴 탐지를 위한 리팩토링 작업이 일어나고, 검출된 패턴을 수정하여 다시 사용자에게 전달한다.

개선된 코드와 개선 이전의 코드는 사용자의 허가를 받아 데이터베이스에 저장하여 게시판에 등재할 수 있다.

유저들이 웹 서비스를 통해 감소한 탄소 배출량은 하루 단위로 데이터베이스에 누적 저장되며 웹 화면을 통해 확인할 수 있다.

2.1.2 User Interfaces

유저의 Java 코드를 웹사이트에 입력을 하고 버튼을 누르면 현재 코드와 불필요한(탄소 배출량이 많은) 패턴을 수정한 코드의 탄소 배출량을 비교하여 패턴을 수정했을 때 얼마만큼의 탄소를 감축 했는지 쉽게 확인이 가능하다. 코드의 수정된 부분을 색, 표시 등 하이라이팅하여 보여주어 쉽게 코드를 분석할 수 있다. 또한 국가별로 웹 서비스 이용자 수를 보여주어 친환경 문제에 대한 참여를 독려할 수 있다. 마지막으로 자신의 코드를 공개하고 싶거나 다른 사람의 코드를 공유받고 싶을 때 게시판 페이지에서 특정 패턴별로 코드가 어떻게 검출이 되었는 지 코드 템플릿으로 확인할 수 있다.

2.2 Product functions

Function		Condition	Description
국가별 이용자 수	등록	메인 페이지에서 사용자의 Java 코드 입력 후 리팩토링 버튼 클릭	웹 서비스를 이용하는 사용자의 국가를 조사하여 코드 리팩토링을 이용한 경우 해당 사용자의 국가 이용자 수를 더한다. 데이터베이스에서 해당 정보가 저장이 되는 기능이다.
	출력	웹에 접속할 때	데이터베이스에 저장된 국가별 이용자 수 정보를 가져와 그래프로 표현하여 웹에 접속하는 사용자들로 하여금 출력하여 보여주는 기능이다. 국가별 이용자 수는 사용자들에게 친환경 코드 리팩토링 과정에 참여를 촉진할 수 있다.
알고리즘 패턴 분석		메인 페이지에서 사용자의 Java 코드 입력 후 리팩토링 버튼 클릭	사용자의 코드에서 불필요하게 탄소 배출량을 증가시키는 패턴을 검출한다. 예를 들어, 중첩된 if문이 3개 이상 존재하는 경우, arraylist 객체의 사이즈를 반복문의 조건에서 계속 구하는 경우, 반복문 내에서 중복적으로 객체(Object) 선언하는 경우, 그리고 추가적인 패턴을 추가하여 Buggy.java 파일에서 불필요 패턴을 찾아내는 기능이다.
수정된 코드 출력			위 알고리즘 패턴 기능을 수행한 후 수정된 Fixed.java 파일을 사용자로 하여금 보여주는 기능이다. 수정된 파일은 arraylist 객체의 사이즈를 반복문 밖으로 꺼내어 선언하는 등 수정 전 코드와 같은 기능을 하면서 효율적으로 패턴을 사용하는 코드를 출력하는 과정을 가진다.
패턴 하이라이팅			위 알고리즘 패턴 기능을 통해 패턴을 검출하여 패턴이 시작되는 코드 라인부터 끝나는 라인까지 데이터를 저장하여 코드의 패턴 라인을 색, 밑줄 등으로 표시하는 기능이다. 예를 들어, 수정 전 코드에서 불필요한 패턴을 빨간색으로 표시하고, 수정 후 코드에서 수정된 패턴을 초록색으로 표시하여 패턴을 찾는 디버깅을 할 때 빠르고 쉽게 변화를 인식할 수 있도록 한다.

탄소 절감량 시각화			사용자 코드의 수정 전과 수정 후의 탄소 배출량을 비교하여 리팩토링으로 인해 탄소 절감량을 구해서 시각적인 자료를 포함하여 사용자로 하여금 탄소 절감량에 대해 와닿기 쉽고 리팩토링의 효과를 이해할 수 있도록 출력하여 보여주는 기능이다. 예를 들어, 코드의 탄소 절감량이 5kg일때 승용차가 38km 이동 시 탄소 배출량과 동일하다는 자료를 보여주고, 코드의 탄소 절감량이 160kg일때 사람 1명이 월간 탄소 배출량(세계 평균)과 동일하다는 자료를 보여줄 수 있다. 탄소 발자국 계산식과 여러 자료를 이용해 추정치로서 보여줄 수 있으며, 탄소 절감량의 크기에 따라 더 광범위하게 수치를 표현할 수 있는 기능이다.
게시판	등록	사용자의 Java 코드 입력(자신의 코드 공개 활성화) 후 버튼 클릭	자신의 코드를 게시판에 공유하고 싶을 때, 코드를 입력하고 버튼을 클릭할 때 공개 활성화를 표시하여 수정 전 코드와 수정 후 코드에 대한 정보를 웹 이용자 모두에게 보여줄 수 있는 기능이다. 게시판에 해당 코드 템플릿이 게시가 될 때, 특정 패턴별로 분류하여 코드 템플릿의 유지보수를 높이고 관리가 쉽게 가능하다.
	출력	게시판 페이지로 넘어가는 버튼 클릭	메인 페이지에서 게시판 페이지로 넘어가면 특정 패턴별로 분류된 코드 템플릿을 출력하는 기능이다. 코드 템플릿은 데이터베이스에 저장된다. 사용자들로 하여금 게시된 코드 템플릿은 다른 사용자들의 리팩토링 과정에 도움을 줄 수 있다.

2.2.1 코드 입력과 출력

User의 코드를 입력 받고 탄소 배출 절감 코드를 출력하는 기능이다.

User는 코드 input box를 통해 코드를 텍스트 혹은 파일 형식으로 입력할 수 있으며, 코드 output box를 통해 개선된 코드의 변화 위치와 복사 기능을 사용할 수 있다.

2.2.2 정보 시각화

탄소 감소량을 시각화하여 User의 참여를 독려하는 기능이다.

탄소 배출 코드 탐지 알고리즘이 User의 코드의 탄소 배출량을 얼마나 줄였는가를 그 양에 따라 비행기, 자전거, 샤워 등 다양한 비교군을 통해 나타낸다.

하루 동안 웹 서비스를 사용한 유저들이 감소시킨 탄소 배출량의 누적 총량을 보여준다.

2.2.3 게시판

Output box에서 유저의 허가를 받아 코드를 저장하고 게시하는 페이지이다.

각 게시글의 제목은 사용된 알고리즘을 글머리에 제시한다.

각 게시글은 수정 전 코드와 수정 후 코드를 보여주며, 변화한 부분을 강조하여 나타낸다.

2.3 User classes and characteristics

2.3.1 개인, 소규모

JAVA 프로그래밍에 대한 지식을 알고 있지만 1인 혹은 작은 집단에 포함되어 코드에 대한 이해와 공부를 하는 개발자이다. 예를 들어, 프로그램을 만든 후 자신의 코드를 최적화하기를 원하는 학생 혹은 소규모 프로그래머가 있다. 또한 환경 분야에 관심을 가지고 코드가 가진 영향력에 대해 알고 싶어 하는 친환경 개발자도 포함된다.

2.3.2 대규모 집단, 기업

여러 분야의 이해관계자들이 모여 프로젝트를 진행하고 프로그램 개발자가 포함된 기업 혹은 단체나 환경 보호에 관심을 가지고 친환경 코드 리팩토링에 대한 분석을 하는 집단이다. 일반적으로 정해진 예산 속에서 프로젝트를 진행하기 때문에 프로그램 개발 중에 필요한 비용(시간 혹은 금액)을 줄이기 위해 그린 코딩을 이용할 수 있다. 또한 환경 보호 단체에서는 게시판에 게시된 코드 템플릿을 바탕으로 코드 리팩토링이 가져다 주는 탄소 절감량을 효과적으로 분석할 수 있다.

2.4 Design and implementation constraints

본 웹 서비스를 설계하면서 고려해야 할 제약 조건은 다음과 같다.

- 개발은 Window XP 이상 운영체제를 사용하는 사용자를 대상으로 한다.
- Input file 및 코드는 JAVA 언어를 사용해야 한다.
- 파일은 단일 파일을 받으며, 여러 파일이 포함된 폴더를 받을 수 없다.
- 시스템의 확장성 및 유지보수를 고려해야 한다.
- 사용자 친화적 UI를 구성하여 처음 본 사용자도 사용할 수 있어야 한다.
- 각 사용자가 코드를 입력하여 수정된 코드를 출력하는 과정이 독립적으로 이루어진다.
- 사용자가 웹 서비스를 이용할 때 가지는 제약 사항을 최소화한다.
- 시스템 처리 과정을 단순화하여 처리 시간이 너무 길지 않도록 한다.

- 저장된 데이터가 외부 영향에 의해 변하지 않아야 한다.불필요한 메모리의 사용을 줄인다.

2.5 Assumptions and Dependencies

웹 서비스는 input 코드의 탄소 배출 감량 기능을 제공하는 웹사이트다. 코드에 대한 기본적인 이해가 있으며 개발자 윤리를 따르는 사용자들이 사용한다고 가정한다. 사용자에 대해 아래와 같은 가정을 보장한다.

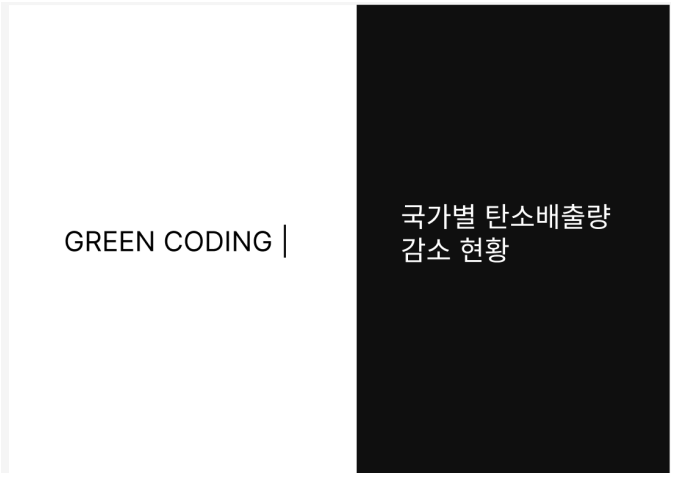
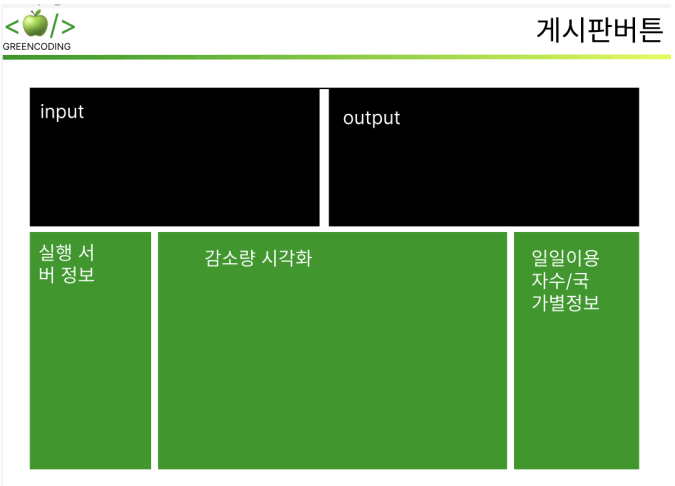
- 사용자는 JAVA 프로그래밍에 대한 선행 지식이 있다고 가정하며, 입력한 코드는 정상 작동을 보장한다.
- 사용자가 코드를 공개하여 게시판에 코드 템플릿을 게시할 때, 특정 패턴을 검출하여 정상적으로 수정이 완료된 코드 템플릿을 게시 한다.
- 프로그램의 영향을 끼칠 정도로 입력한 코드의 실행 시간이 길지 않도록 한다.
- 입력한 코드가 악의적으로 웹 서비스 혹은 서버에 직접적으로 개입하지 않도록 한다.
- 사용자의 참여는 최종적으로 코드의 탄소 배출량 감축을 목표로 한다.

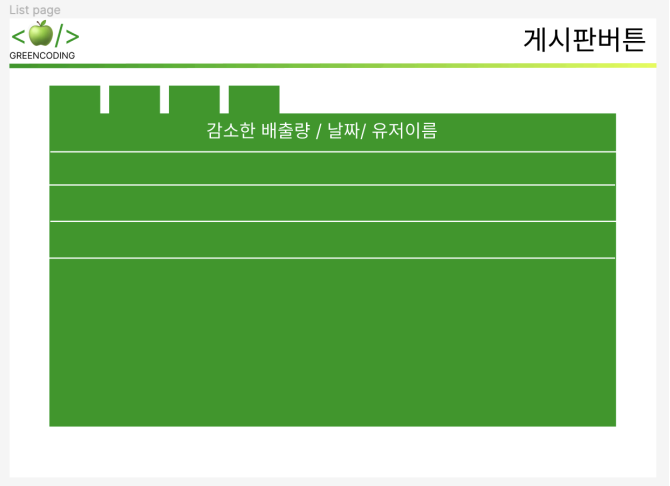
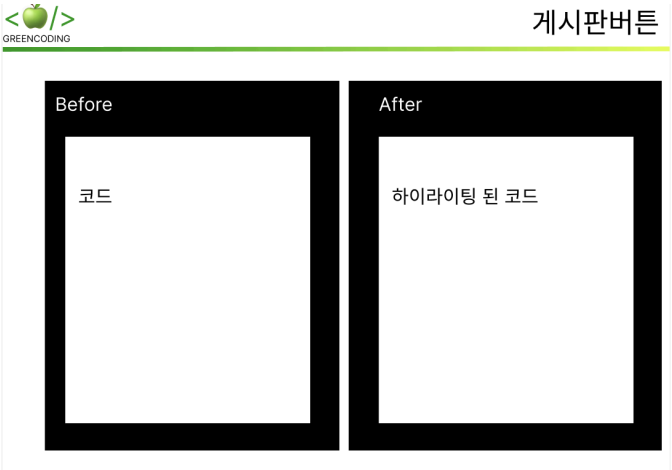
III. External Interface Requirements

3.1 External interfaces

3.1.1. User Interface

이름	마우스 및 키보드를 통한 입력 처리
목적/내용	시스템 사용자가 마우스 및 키보드의 입력을 통해 시스템에 명령 전달
입력 주체/출력 목적지	사용자/Linux 기반의 웹 서버
범위/정확도/허용 오차	- 범위: 화면에서의 버튼 개수에 따른 입력 범위 - 정확도: 사용자의 마우스 및 키보드 입력 정확도 - 허용 오차: 해당 없음
단위	버튼 클릭 및 키보드 입력
시간/속도	비정기적인 사용자의 입력/즉각적인 사용자 명령 수행
타 입출력과의 관계	입력 내용에 따라 클라이언트에서 처리 또는 서버로 명령 요청 및 응답 사용
화면 형식 및 구성	해당 없음
윈도우 형식 및 구성	해당 없음
데이터 형식 및 구성	main method가 있는 JAVA 코드
명령 형식	JAVA 코드 입력
종료 메시지	해당 없음

이름	모니터를 통한 메인 화면 출력
목적/내용	시스템 사용자에게 제공하는 인터페이스
입력 주체/출력 목적지	클라이언트/사용자 디스플레이 디바이스
범위/정확도/허용 오차	- 범위: 화면에서의 버튼 개수에 따른 입력 범위 - 정확도: 사용자의 마우스 및 키보드 입력 정확도 - 허용 오차: 해당 없음
단위	화면
시간/속도	사용자의 입력에 따른 화면 전환
타 입출력과의 관계	사용자의 입력을 위한 인터페이스 출력 후 사용자의 입력 대기
화면 형식 및 구성	 3.1.1.1 cover - 메인페이지 진입 전 간단한 웹페이지 소개 - 국가별 탄소배출량 감소 현황 공개 

	3.1.1.2 main page - input 창에 코드 입력 시 output 창에 하이라이팅 된 코드 출력 - 하단에 실행 서버 정보, 감소량 시각화, 일일 이용자수 및 국가별 정보 제공  3.1.1.3 code list page - 알고리즘별 탭 구분 - 코드 히스토리가 목록 형식으로 구성됨  3.1.1.4 detail page - 리스트 클릭 시 input, output 코드 보여줌
윈도우 형식 및 구성	- linear layout 형식으로 cover 구성 - Grid 형식으로 main page 구성 - table 형식으로 code list page 구성 - linear layout 형식으로 detail page 구성
데이터 형식 및 구성	이미지, 텍스트, JAVA 코드

이름	JAVA 코드 입력 및 탄소 배출량 출력 인터페이스
목적/내용	JAVA 코드 입력 및
입력 주체/출력 목적지	사용자/서버
단위	화면
화면 형식 및 구성	JAVA 코드 입력 창, 개선된 코드 출력 창, 코드 제출 버튼, 코드 복사 버튼, 코드 공개 여부 버튼, 탄소 절감량 시각화
데이터 형식 및 구성	JAVA 코드
명령 형식	제출 버튼 클릭에 따른 명령
종료 메시지	컴파일 실패 시: “컴파일 실패" 컴파일 성공 시: “게시판에 공유하시겠습니까?”

3.1.2. Hardware Interface

이름	클라이언트 I/O 디바이스
목적/내용	키보드, 마우스를 사용한 사용자의 입력
입력 주체/출력 목적지	사용자/서버
범위/정확도/허용 오차	해당 없음
단위	해당 없음
시간/속도	사용자의 입력
타 입출력과의 관계	입력받은 코드를 웹 서버로 전송
화면 형식 및 구성	해당 없음
윈도우 형식 및 구성	해당 없음
데이터 형식 및 구성	JAVA 코드
명령 형식	HTTP request
종료 메시지	해당 없음

3.1.3. Software Interface

이름	웹 사이트
목적/내용	화면 출력
입력 주체/출력 목적지	해당 없음
범위/정확도/허용 오차	chrome, edge, firefox, safari와 같은 웹 브라우저에서 사용 가능
단위	해당 없음
시간/속도	새로 고침 및 클릭에 따른 즉각적인 처리
타 입출력과의 관계	해당 없음
화면 형식 및 구성	웹 브라우저를 통한 웹사이트 출력
윈도우 형식 및 구성	해당 없음
데이터 형식 및 구성	해당 없음
명령 형식	해당 없음
종료 메시지	해당 없음

3.1.4 Communication Interface

이름	호스트 서버-클라이언트
목적/내용	각 클라이언트가 호스트 서버에 접속 요청, 사용자가 입력한 코드를 클라이언트에서 호스트 서버로 전달, 호스트 서버는 탄소 배출량 계산 및 코드 개선 후 결과를 클라이언트로 전달
입력 주체/출력 목적지	호스트 서버와 클라이언트
범위/정확도/허용 오차	해당 없음
단위	패킷
시간/속도	최소 10Mbps 이상
타 입출력과의 관계	해당 없음

데이터 형식	실행 가능한 JAVA 프로그래밍 코드를 포함한 JSON 데이터
명령 형식	Send() 콜에 의한 통신
종료 메시지	Close() 콜에 의한 소켓 종료

3.2 Function requirement

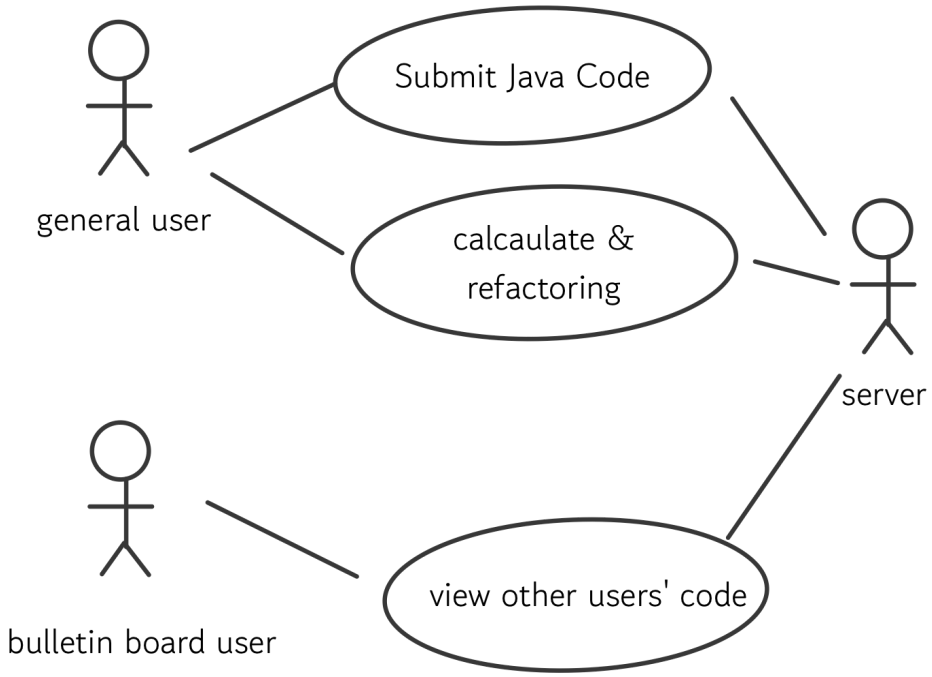
3.2.1. Use Case

use case name	submit JAVA code
actor	general user
description	사용자가 입력창에 JAVA 코드를 입력하고 계산된 탄소배출량과 리팩토링 된 코드를 출력창으로 확인하는 과정이다.
normal course	1. 모든 사용자는 웹 접속 시 커버 페이지를 스크롤하여 메인 페이지로 이동한다. 2. 사용자가 상단 좌측의 입력창에 본인의 JAVA 코드를 입력한 후 제출 버튼을 누른다. 3. 제출 버튼을 누르면 상단 우측의 출력창에 리팩토링 된 JAVA 코드가 하이라이팅과 함께 나타난다. 4. 사용자는 해당 코드를 게시판에 공유할 것인가에 대한 여부를 버튼 클릭을 통해 결정한다. a. 게시판 공유 시 i. 공개 사용자로 공유: 본인의 github 아이디를 추가로 입력한 후 공유한다. ii. 비공개 사용자로 공유: 익명으로 공유하기 버튼을 클릭한다. b. 게시판 공유 안 할 시 다음 단계로 이동한다. 5. 사용자는 하단에서 코드가 실행된 HW 환경과 감소된 탄소 배출량을 시각적 요소로 확인한다.
precondition	- 사용자는 웹에 접속하여 메인 페이지로 이동할 수 있어야 한다 - 사용자는 JAVA코드를 소유하고 있다
postcondition	- 사용자는 계산된 탄소 배출량과 실행 환경의 HW 정보를 확인한다
assumptions	- 사용자의 JAVA 코드는 컴파일이 가능하다 - 불가능할 경우 오류 메시지 출력 - 코드가 실행된 HW 환경은 서버 기준이다

use case name	view other users' code
actor	bulletin board user
description	게시판 사용자는 서버에서 반환된 결과를 확인하고, 자신의 코드를 게시할지 여부를 선택한다.
normal course	1. 사용자는 게시판 페이지에 접속한다 2. 사용자는 원하는 알고리즘 탭을 클릭한다. 3. 다른 사용자들이 남긴 코드를 간략화한 리스트를 테이블 형식으로 확인할 수 있다. 4. 사용자는 확인을 원하는 항목을 선택하여 세부 코드 페이지로 이동한다. 5. 해당 페이지에서 사용자는 input 코드, output코드 및 탄소 배출량을 확인한다.
precondition	- 다른 사용자가 입력한 코드가 테이블에 존재하여야 한다. - 존재하지 않을 시, 'no data in the table' 출력
post condition	사용자는 선택한 코드의 세부 정보를 확인한다.
assumptions	- 게시판에는 여러 사용자가 게시한 코드가 존재한다. - 게시판의 코드는 리팩토링과 탄소 배출량 계산이 완료된 코드이다. - 탄소 배출량은 서버의 HW 사양에 따라 계산된다

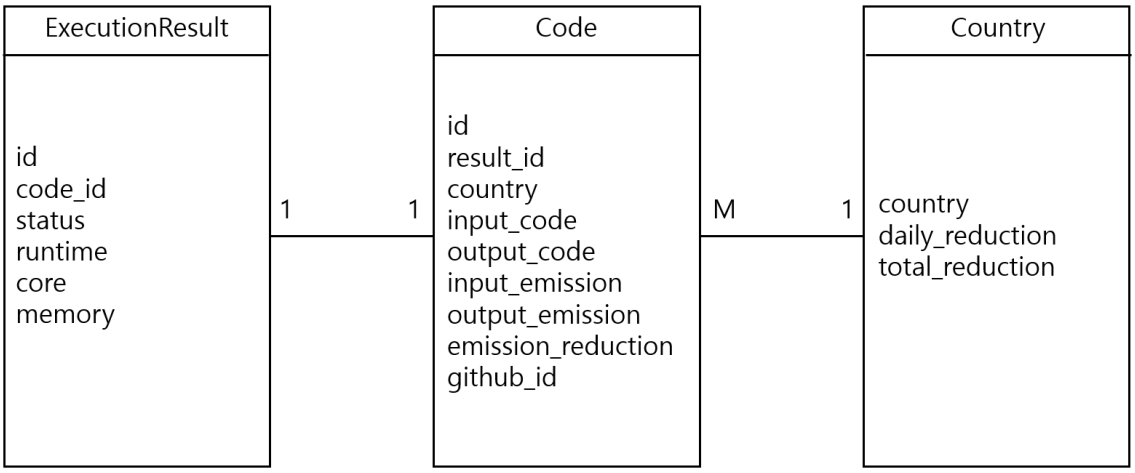
use case name	calculate carbon emission and refactoring code
actor	server
description	서버는 사용자가 제출한 코드를 기반으로 탄소 배출량을 계산한다
normal course	1. 사용자가 코드를 제출한다 2. 서버는 사용자가 제출한 코드를 분석하여 탄소 배출량을 계산하고 리팩토링한 결과를 반환한다
precondition	- 사용자가 코드를 제출해야 한다 - 서버에는 green algorithm이 존재하여야 한다
post condition	- 사용자의 input, output 코드, 탄소 배출량, 국가정보는 데이터베이스에 저장된다
assumptions	- 사용자가 제출한 코드는 서버에서 처리 가능한 형식이어야 한다

3.2.2. Use Case Diagram



3.2.3. Data Dictionary

3.2.3.1. E-R Diagram



3.2.3.2. Tables

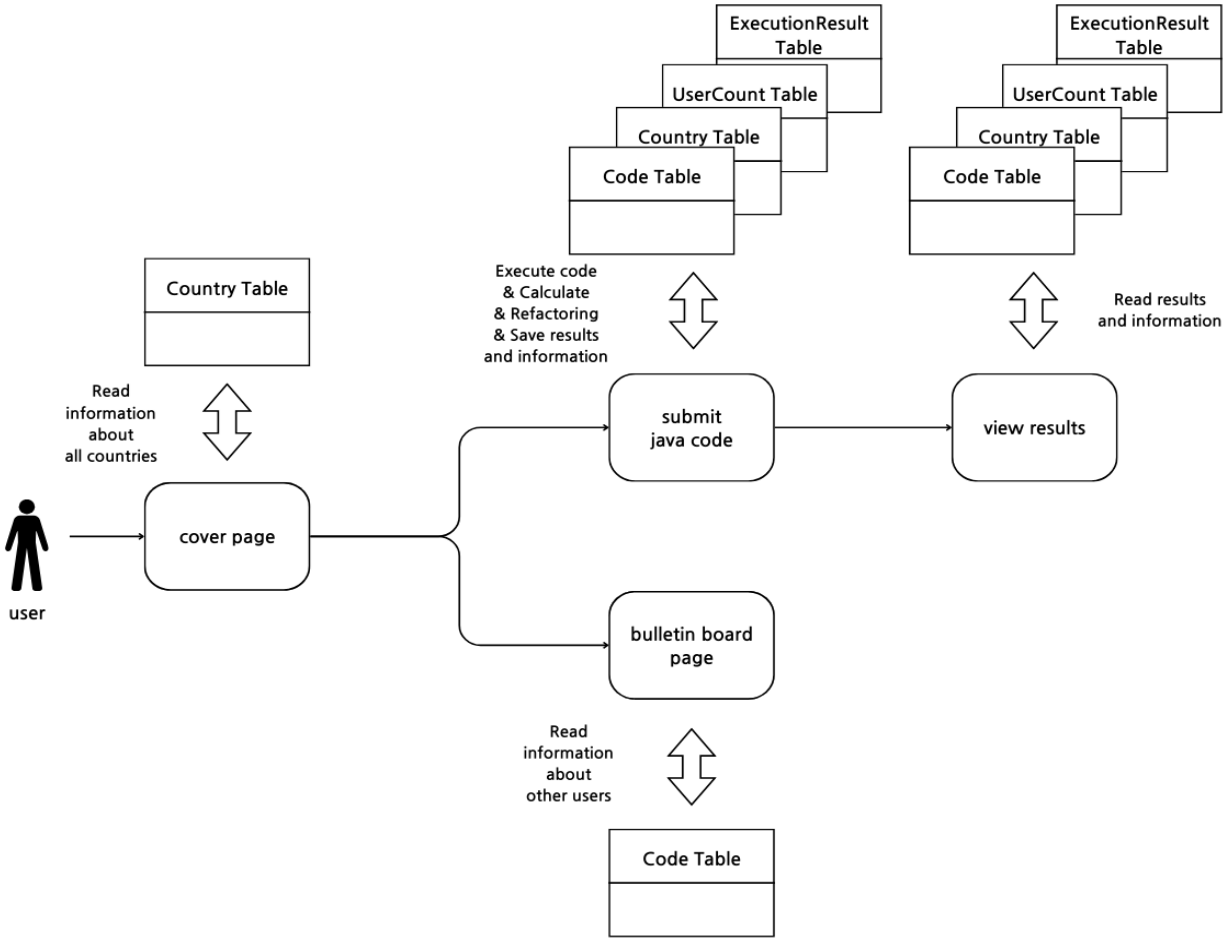
Code				
Field	Key	Domain	Constraint	Description
id	PK	int	Not Null	Entity id
result_id	FK	int	Not Null	Execution result
country	FK	char(100)	Not Null	User’s nationality
input_code		text	Not Null	User’s Java code
output_code		text	Not Null	Improved Java code
input_emission		float	Not Null	Carbon emission amount of user’s java code
output_emission		float	Not Null	Carbon emission amount of improved java code
emission_reduction		float	Not Null	Carbon emission reduction amount
github_id		char(100)		Github id

ExecutionResult				
Field	Key	Domain	Constraint	Description
id	PK	int	Not Null	Entity id
code_id	FK	int	Not Null	
status		varchar(255)	Not Null	(“SUCCESS”, “COMPILE_ERROR”, “RUNTIME_ERROR”, “TIME_EXCEEDED”, “MEMORY_EXCEEDED”
runtime		BigInt	Not Null	Cpu time for execution (ms)
core		int	Not Null	The number of cores
memory		BitInt	Not Null	Memory usage (Byte)

UserCount				
Field	Key	Domain	Constraint	Description
date	PK	date	Not Null	
daily_user		int	Not Null	Count for daily users
total_user		BigInt	Not Null	Count for total users

Country				
Field	Key	Domain	Constraint	Description
country	PK	char(100)	Not Null	
daily_reduction		float		Daily carbon emission reduction for each country
total_reduction		float		Total carbon emission reduction for each country

3.2.4. Data Flow Diagram



3.3 Performance requirements

아래는 본 시스템의 성능 요구사항을 예측하여 기술한 것으로, 실제 구현 시 달라질 수 있다.

3.3.1. Static Numerical Requirement

-시스템은 여러 명의 사용자가 동시에 코드를 입력하고 결과를 확인할 수 있도록 지원해야 한다. 주 사용자는 코드로부터 배출되는 탄소의 양을 절감하려는 개발자이다.

- 시스템은 1.5GHz 이상의 프로세서, 2GB RAM 이상의 데스크탑 Windows 환경에서 부드럽게 동작해야 한다.

- 5Mbps 이상의 인터넷 연결 속도 환경을 요구한다. 운영체제는 Windows 10 이상이며, 웹 애플리케이션을 지원하는 환경에서 정상적으로 동작해야 한다.

- 사용자가 입력하는 코드의 프로그래밍 언어는 JAVA로 제한한다.

- 코드 실행 후 실행 환경을 나타내야 한다.

3.3.2. Dynamic Numerical Requirement

-시스템은 동시에 최소 30명의 사용자 접속을 유지할 수 있어야 한다. 각 웹 페이지는 동시에 최소 30명의 사용자가 접속하여도 부드럽게 동작해야 한다.

-최대 10000개의 입력 코드&개선된 코드&탄소 절감량 정보를 저장 및 관리할 수 있어야 한다.

- 사용자가 코드를 입력하면 실행부터 결과 확인까지 걸리는 시간은 1분 이내여야 한다. 결과는 5초 이내로 데이터베이스에 저장되고 이를 시각화하여 나타낼 수 있어야 한다.

- 사용자가 웹 페이지를 이동할 때 새로운 웹 페이지가 5초 이내로 표시되어야 한다.

- 웹 사이트는 24시간 가동되어 사용자가 이용할 수 있어야 한다.

3.4 Logical database requirements

시스템은 FastApi에 기본적으로 설치되어 있는 SQLite 데이터베이스 서비스를 이용하여 데이터를 관리 한다. 해당 데이터베이스를 통하여 시스템 사용자들의 계정 정보를 저장한다. 사이트에 방문한 일일 사용자 횟수가 저장되며, 매일 누적 방문자 횟수에 추가된다.

- 웹 사이트를 통해 감소시킨 일일 탄소 배출량이 저장되며, 매일 누적 탄소 배출량 감소량에 추가된다.

- 사용자가 제출한 개선되기 전의 원본 코드와 개선된 이후의 개선버전 코드를 저장한다. 각각의 탄소 배출량 또한 저장될 수 있다.

- 각 국가별로 개선된 코드를 적용했을 때 발생하는 탄소 배출 감소량을 저장한다. 이 정보는 국가별로 통계를 내어 환경적 영향을 추적하는 데 사용될 수 있다.

3.5 Design constraints

3.5.1 Physical design constraints

시스템의 목적은 사용자가 웹 사이트상에서 전송한 코드에 대한 탄소배출량 검출, 개선코드 생산이므로 데스크탑 또는 노트북 기기를 이용할 것을 권장하며, 모바일 기기 환경은 권장하지 않는다.

3.6 Software system attributes

3.6.1 Reliability

- 사용자의 잘못된 입력 형식이나 예기치 못한 상황에 대응할 수 있어야 한다. 이를 위해 사용자 입력 데이터의 유효성을 확인하고, 오류 발생 시 적절한 안내 메시지를 표시하여야 한다. 오류 발생 시 로그를 통해 문제를 추적하고 개선할 수 있어야 한다.
- 사용자가 제출한 코드에 대한 탄소배출량은 그린 알고리즘에 기반하여 정확하고 일관적인 방식으로 계산되어야 한다. 이러한 계산 과정은 시스템에서 투명하게 이해할 수 있어야 하며, 사용자가 결과를 신뢰할 수 있도록 보장되어야 한다.

3.6.2 Availability

- 시스템 장애 발생으로 인한 재시작 시 사용자 경험이 영향을 받지 않도록 구성되어야 하며, 데이터를 손실 없이 복구할 수 있어야 한다. 시스템은 일정 시간 간격으로 데이터베이스의 체크포인트를 통해 데이터의 일관성을 유지해야 한다.

3.6.3 Security

- 사용자들은 인증과정 없이 코드입력란을 통해 텍스트를 서버로 전송할 수 있으므로, 서버로 보내는 코드에 포함된 악의적인 접근을 검사해야 한다.
- 시스템 관리자를 제외한 일반 사용자는 시스템 관리자만큼의 권한을 가질 수 없고, 사용자는 자신과 관련되고 제한된 데이터만을 볼 수 있으며, 시스템 데이터베이스에 직접적으로 접근할 수 없도록 보호되어야 한다.

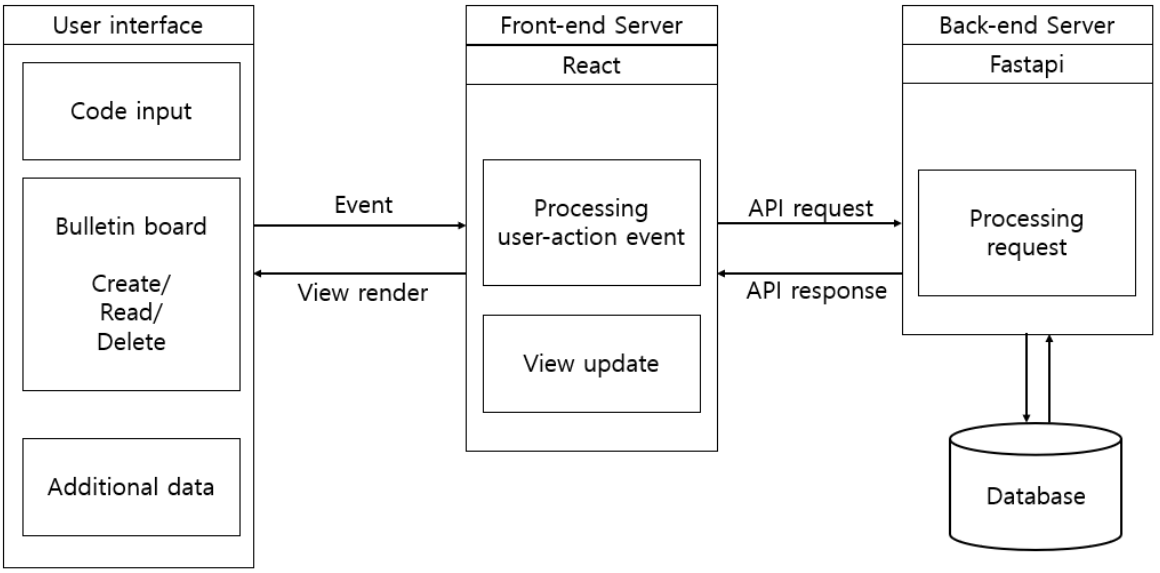
3.6.4 Maintainability

- 시스템은 일반화, 계층화되어야 하며, 각 모듈은 독립적으로 수정, 테스트되며 이를 통해 변경이 필요한 부분만 수정할 수 있어야 한다.
- 시스템 내부의 모듈 사이에는 명확한 인터페이스로 연결되며 복잡한 의존성을 최소화해야 한다.

3.6.5 Portability

- 웹 표준을 준수하여 웹 접근성 및 다양한 플랫폼에서의 호환성을 보장하기 위해 다양한 환경에서의 이식성을 확인하는 테스트를 수행한다.

3.7 System Architecture



3.8 System Evolution

3.8.1 Limitation and Assumption

시스템은 데스크탑 환경에서 지원되며 모바일 기기에서의 사용은 고려되지 않는다.

3.8.2 Evolution of Hardware and Change of User Requirements

개선코드를 공유할 수 있는 게시판 기능에서 사용자들의 익명성에 대한 요구가 변화할 수 있다. 사용자들이 익명성을 더 중요하게 생각하거나 그 반대일 수 있으므로, 이러한 변화에 유연하게 대응할 수 있는 시스템이어야 한다. 예를 들어, 추가적인 익명성 옵션을 제공하거나 익명성을 선택적으로 사용할 수 있도록 하는 기능을 추가할 수 있다.

IV. Appendixes

4.1 Requirements specification standards

본 요구사항 명세서는 IEEE/ANSI 830-1998 (IEEE, 1998)을 기준으로 작성되었다.