

# Software Requirement Specification

탄소를 JAVA라

by

김민지, 김찬용, 안상현, 양승빈, 윤시형, 임동준, 최경식  
Team1



Instructor: 이은석

Document Date: 2023.10.28

Faculty: Sungkyunkwan University

# Contents

1. Introduction
  - 1.1 Purpose
  - 1.2 Scope
  - 1.3 Definitions, Acronyms, and Abbreviation
  - 1.4 References
  - 1.5 Overview
2. Overall Description
  - 2.1 Product Perspective
    - 2.1.1 System Interfaces
    - 2.1.2 User Interfaces
    - 2.1.3 Hardware Interfaces
    - 2.1.4 Software Interfaces
    - 2.1.5 Memory Constraints
    - 2.1.6 Operations
  - 2.2 Product Functions
    - 2.2.1 사용자의 코드 입력
    - 2.2.2 데이터 송수신
    - 2.2.3 전송된 코드 실행
    - 2.2.4 탄소배출량 계산
    - 2.2.5 탄소배출량 데이터 분석
    - 2.2.6 사용자에게 분석 결과 전송
  - 2.3 User Classes and Characteristics
    - 2.3.1 User Classes
    - 2.3.2 User Characteristics
  - 2.4 Design and Implementation Constraints
  - 2.5 Assumptions and Dependencies
3. Specific Requirements
  - 3.1 External Interface Requirements
    - 3.1.1 User Interface
    - 3.1.2 Hardware Interface
    - 3.1.3 Software Interface
    - 3.1.4 Communication Interface
  - 3.2 Functional Requirements
    - 3.2.1 Use Case
    - 3.2.2 Use Case Diagram
    - 3.2.3 Data Flow Diagram
  - 3.3 Performance Requirements
    - 3.3.1 Static Numerical Requirement
    - 3.3.2 Dynamic Numerical Requirement
  - 3.4 Design Constraints
    - 3.4.1 Physical design constraints
    - 3.4.2 Standards compliance
  - 3.5 Software System Attributes
    - 3.5.1 Product Requirements
    - 3.5.2 Organizational Requirements
    - 3.5.3 External Requirements
  - 3.6 Organizing the Specific Requirements
    - 3.6.1 Context Model

- 3.6.2 Process Model
    - 3.6.3 Interaction Model
  - 3.7 System Architecture
  - 3.8 System Evolution
    - 3.8.1 Limitation and Assumption
    - 3.8.2 Evolution of Hardware and Change of User Requirements
- 4. Appendix
  - 4.1 Software Requirements Specification Standard
  - 4.2 Document History

## List of figure

Image 3.1: Use Case Diagram

Image 3.2: Data Flow Diagram

Image 3.3: Context Diagram

Image 3.4: Process Diagram

Image 3.5: System Architecture Diagram

# List of Table

Table 3.1: User Interface of input processing

Table 3.2: User Interface of main page

Table 3.3: Hardware Interface of applicable device for the system

Table 3.4: Software Interface of applicable device for the system

Table 3.5: Communication Interface of applicable device for the system

Table 3.6: Use Case of Code Input

Table 3.7: Use Case of Code Execute Code

Table 3.8: Use Case of Results Analysis

Table 3.9: Use Case of Confirm Results

# 1. Introduction

## 1.1. Purpose

본 문서는 '탄소를 JAVA라' 서비스를 위한 소프트웨어 요구사항 명세서이다. '탄소를 JAVA라' 서비스는 2023학년도 2학기 소프트웨어공학개론 (SWE3002-41) (이하 '교과목') 수업을 수강하는 대학생으로 이루어진 Team1이 고안하고 설계한 서비스이다. 해당 서비스가 제공하는 모든 기능과 시스템 구현상의 제약 조건을 명시하며, 요구사항을 명확하게 명시한다.

본 문서는 본 교과목의 Team1의 구성원과 교수 및 조교를 주요 독자로 설정한다. 교육 및 문서작성 등 비상업적 목적으로 본 문서를 열람하고 사용하는 것을 허용한다.

본 문서는 '탄소를 JAVA라' 서비스의 기능과 구현사항, 요구사항을 명확하게 명시함으로써 시스템의 개발 및 유지보수 과정에 참고할 수 있도록 제작되었다.

## 1.2. Scope

'탄소를 JAVA라' 서비스는 사용자로부터 코드를 입력 받고, 해당 코드를 실행했을 때 배출되는 탄소배출량을 계산하며, 코드가 실행된 환경을 함께 표시하는 서비스이다. GPU를 사용하지 않는 환경에서 실행 가능한 코드만을 대상으로 하며, 코드는 Java 코드로 한정한다. 이 시스템은 기후 변화와 환경 오염에 대한 인식 향상을 목적으로 하며, 특히 개발자들이 코딩을 할 때 환경적 영향을 고려하도록 돕는다.

## 1.3. Definitions, Acronyms, and Abbreviation

Acronyms & Abbreviation	Explanation
GPU	Graphics Processing Unit의 약자로, 컴퓨터 시스템에서 그래픽 연산을 빠르게 처리하여 결과값을 모니터에 출력하는 연산 장치이다.
RAM	Random Access Memory의 약자로, 프로그램이 실행되는 동안 필요한 정보를 저장하는 메모리이다.
UML	Unified Modeling Language의 약자로, 표준 모델링 언어이다.
ESG	기업의 비재무적 요소인 환경(Environment), 사회(Social), 지배구조(Governance)를 포함한 단어로, 기업의 친환경 경영, 사회적 책임, 투명한 지배구조 등을 의미한다.
IDE	Integrated Development Environment의 약자로, 공통된 개발자 툴을 하나의 그래픽 사용자 인터페이스로 결합하는 애플리케이션을 구축하기 위한 소프트웨어이다.

Terms	Definition
탄소배출량	특정 활동으로 인해 발생하는 탄소의 양(gram)을 나타내는 지표이다.
그린 코딩	환경 보호와 지속 가능한 개발을 강조하는 개발 방법론이다. 소프트웨어 개발 과정에서 환경 친화적인 방법을 채택하고, 에너지 효율적인 코드를 작성하며, 자원을 절약하는 것을 추구한다.

## 1.4. References

830-1998-1EEE Recommended Practice for Software Requirements Specifications

<https://ieeexplore.ieee.org/document/720574>

2022 Spring 41 class team 1

[https://github.com/skkuse/2022spring\\_41class\\_team1](https://github.com/skkuse/2022spring_41class_team1)

Green algorithms calculator

<http://calculator.green-algorithms.org/>

Green algorithms tool

<https://github.com/GreenAlgorithms/green-algorithms-tool>

## 1.5. Overview

본 소프트웨어 요구사항 명세서는 네 개의 주요 챕터로 구성된다. 첫 번째 챕터에서는 본 프로젝트의 결과물인 ‘탄소를 JAVA라’의 배경 및 중요성에 대해 설명하며, 해당 문서에서 사용되는 전문 용어 및 약어에 대한 정의를 제공한다. 두 번째 챕터에서는 사용자의 입력을 처리하는 시스템의 전반적인 인터페이스와 기능, 디자인 및 구현 제약사항 등 시스템 전반에 대한 개요를 자연어 형식으로 간략하게 밝힌다. 세 번째 챕터에서는 인터페이스, 기능, 성능 등에 대한 요구사항을 구체적으로 제시한다. 마지막 챕터에서는 본 요구사항 명세서를 작성하기 위해 참고한 기준이 되는 문서를 밝힌다. 팀의 모든 멤버는 본 문서 작성 및 프로그램 설계와 구현에 기여하였음을 밝힌다.

## 2. Overall Description

### 2.1. Product Perspective

현대 사회는 탄소배출로 인한 환경 오염과 기후 변화가 급속히 진행되고 있으며, 환경문제에 대한 중요성과 심각성에 대한 인식이 증가하고 있다. 최근 4차 산업시대의 핵심 분야로 대두된 소프트웨어 분야에서도 탄소배출을 최소화하기 위해 노력하고 있고, 탄소배출을 최소화하는 코딩 방법 역시 강조하고 있다.

본 서비스는 사용자가 코드를 입력하여 해당 코드를 실행시킬 때 얼마나 많은 전력을 사용하는지, 그리고 이로 인해 발생하는 탄소 배출량이 얼마나 되는지를 측정하는 서비스를 제공한다. 이렇게 측정된 결과는 사용자로 하여금 작성된 코드가 환경에 미치는 영향을 인식하고 탄소배출량 감축을 위한 코드 최적화를 진행할 수 있는 기회를 제공한다.

#### 2.1.1. System Interfaces

사용자는 웹 기반 인터페이스를 통해 **Java** 코드를 입력하며, 시스템은 입력된 코드를 분석하여 탄소배출량을 계산한다. 계산한 탄소배출량은 실행 환경 정보와 함께 표시한다. 또한, 발생한 탄소를 처리하기 위해 몇 그루의 소나무가 필요한지, 전기차와 같은 이동수단이 발생 시키는 탄소와 비교했을 때 비율이 어느 정도인지 등과 같은 탄소배출량 분석 결과를 시각화한다.

#### 2.1.2. User Interfaces

사용자는 웹사이트 기반의 **UI**를 통해 유저 인터페이스를 제공받는다. 코드 입력 영역에 **Java** 코드를 입력할 수 있으며, 다중 탭 기능을 활용하여 여러 개의 코드를 입력할 수 있다. ‘컴파일’ 버튼을 클릭하여 입력한 **Java** 코드에 대한 탄소배출량을 계산할 수 있다. 계산을 완료하면 결과 영역에 탄소배출량과 탄소배출 관련 시각화 정보, 그리고 코드 실행 환경 정보가 표시된다.

#### 2.1.3. Hardware Interfaces

본 시스템은 웹 애플리케이션으로, 기본적인 데스크탑 및 노트북 컴퓨터에서 작동한다. **1.0GHz** 이상의 프로세서와 **4GB** 이상의 **RAM**을 갖춘 기기에서의 사용을 권장한다. 인터넷 연결이 필수적으로 요구된다.

#### 2.1.4. Software Interfaces

본 시스템은 **Windows**, **MacOS** 및 **Linux** 환경에서 동작하며, 최신 버전의 **Chrome**, **Firefox**, **Edge** 등 주요 웹 브라우저와 호환되어야 한다.

#### 2.1.5. Memory Constraints

본 시스템은 **4GB** 이상의 **RAM**을 갖춘 환경에서의 사용을 권장한다.

#### 2.1.6. Operations

##### System Administrator

탄소배출량 계산: 코드의 탄소배출량을 계산한다. 입력된 **Java** 코드의 탄소배출량을 사용자가 확인할 수 있도록 한다.

### 2.2. Product Functions

#### 2.2.1 사용자의 코드 입력

사용자는 입력하는 **Java** 코드의 탄소 발생량과 전력 사용량을 확인하기 위해 웹사이트에 자신의 **Java** 코드를 입력한다. 사용자가 처음 ‘탄소를 **JAVA**라’에 접속하면 메인 페이지의 가장 위쪽에는 탄소 절감을 상징하는 북극곰 사진이 있고, 그 아래에 **Java** 코드를 입력할 수 있는 칸이 있다. 이 칸은 가로로 길게 설정되어 있어 사용자는 자신의 코드를 편하게 한 눈에 볼 수 있다.

사용자는 지정된 칸에 자신의 **Java** 코드를 입력하고, 컴파일 버튼을 통해 코드의 탄소 발생량, 전력 사용량을 계산하기 위한 다음 단계로 나아간다.



사용자의 **Java** 코드 입력을 위한 입력란은 하나 이상의 탭으로 구성되어 있다. 이 탭은 코드 입력란 위에 있는 버튼을 사용하여 추가·삭제·전환할 수 있으며, 여러 탭을 생성하여 다중 클래스가 필요한 코드의 경우에도 별도의 추가작업 없이 각 클래스의 코드를 입력할 수 있다.

### 2.2.2 데이터 송수신

사용자가 입력란에 **Java** 코드를 입력하면, 웹사이트는 서버에 **Java** 코드에 대한 정보를 전송한다. 이후 서버는 사용자에게 계산의 결과값을 반환한다. 사용자가 수신하는 정보에는 **Java** 코드와 코드가 실행되는 서버 정보와 탄소배출량 및 데이터 분석 결과가 포함되어 있다. 만약 여러 개의 탭을 이용해 코드를 입력했을 경우, 웹사이트는 각 탭의 코드를 종합하여 한 번에 전송한다.

### 2.2.3 전송된 코드 실행

서버는 웹사이트로부터 전송받은 코드를 실행한다. 이 때 두 가지 경우를 고려해야 한다.

#### 1) 코드가 오류 없이 실행 가능할 경우

서버는 코드를 실행하여 코드를 실행 과정에서 소요된 런타임을 측정하며 측정된 런타임을 서버의 하드웨어 정보와 결합하여 코드의 탄소배출량을 계산하는 데에 사용한다.

#### 2) 코드에 오류가 존재하여 실행 불가능할 경우

서버는 사용자에게 오류가 존재하여 코드를 실행할 수 없음을 알리는 팝업창을 띄우고, 사용자는 오류를 수정하여 다시 입력란에 코드를 제출해야 한다.

### 2.2.4 탄소배출량 계산

서버는 웹 사이트로부터 전송받은 **Java** 코드가 발생시키는 탄소배출량을 계산한다. 탄소배출량의 계산 과정에서는 컴파일 시간 및 서버 정보를 활용한다.

### 2.2.5 탄소배출량 데이터 분석

서버는 계산된 탄소배출량을 사용자가 더 직관적으로 받아들일 수 있도록 코드의 탄소배출량이나 전력사용량을 실생활과 연관지어 분석한다. 예를 들어, 현재 입력된 **Java** 코드가 발생시키는 탄소배출량과 같은 양의 탄소를 배출하며 가솔린 자동차가 이동할 수 있는 거리로 환산한다. 전력사용량의 경우 전기자동차가 같은 양의 전기를 사용하며 이동할 수 있는 거리로 환산할 수 있다. 같은 방식으로 휴대폰 충전 횟수, 에어컨 가동 시간 등으로도 환산할 수 있다.

### 2.2.6 사용자에게 분석 결과 전송

서버는 탄소배출량 계산 결과, 데이터 분석 결과와 서버 정보를 다시 사용자에게 전송하고, 사용자는 이를 직접 확인하며 자신의 코드가 환경에 어떤 영향을 미치는지 파악할 수 있다. 웹사이트의 결과창에는 탄소배출량과 하드웨어 정보, 데이터 분석 결과가 수치와 도표 형태로 표현되어 사용자가 자신의 코드를 쉽게 평가할 수 있다.

## 2.3. User Classes and Characteristics

### 2.3.1. User Classes

#### 1) 개발자, 학생

현재 전 세계에서 많은 에너지가 IT 분야에서 소모되고 있으며, 특히 최적화되지 않은 프로그램들로 인해 매우 비효율적으로 에너지가 낭비되고 있다. 또한 소프트웨어가 점점 많은 시스템에 관여하고 있기 때문에 그들의 기반이 되는 과정에서의 효율적인 코딩은 필수적이다. 이러한 상황에서 프로그래밍을 배우는 많은 사람들은 에너지를 절감할 수 있도록 효율적인 개발 방법을 숙지해야 한다.

#### 2) 기업

현재 존재하는 기업들은 경영 대부분의 과정을 소프트웨어로 지탱하고 있다. 대규모의 데이터가 송수신되는 상황에서 이들을 한꺼번에 처리하기 위한 에너지 사용이 결코 적지 않을 것이다. 수익을 최대화하고 비용을 최소화해야 하는 기업은 이 프로그램을 통해 탄소배출량과 전력사용량을 파악하여 많은 비용을 절감할 수 있다. 또한 **ESG** 경영이 필수적인 현대에는 친환경적인 경영을 뒷받침하는 도구로써 이 프로그램을 사용할 수 있다.

### 2.3.2. User Characteristics

#### 1) 개발자, 학생

개발자들은 본 웹 사이트를 통해 다양한 코드를 입력해 봄으로써 어떤 형태의 코드가 탄소를 더 많이 배출하는지, 어떤 형태의 코드가 환경 친화적인지를 습득할 수 있다. 또한 전력 사용량과 탄소 배출량을 실생활과 연관지은 결과창을 통해 그린 코딩의 필요성을 실감할 수 있다.

#### 2) 기업

기업은 결과창에서 숫자로 나타난 탄소 배출량 수치를 이용해 그들이 사용하고 있는 코드가 어느 정도의 에너지 비용을 발생시키고 있는지, 그리고 어떤 방식으로 코드를 변경해야 비용을 절감할 수 있는지 정확하게 파악할 수 있다. 또한 대규모의 소프트웨어를 운용하는 기업은 다중 탭을 활용하여 구조화된 프로그램을 입력할 수 있으며, 이를 통해 소프트웨어의 에너지 사용량 등 많은 정보를 얻을 수도 있다.

### 2.4. Design and Implementation Constraints

이 시스템은 본 문서에 기재된 요구 사항과 제약사항을 바탕으로 개발된다. 본 문서에 명시되어있지 않은 요구사항과 제약사항은 개발자의 재량에 의해 조정할 수 있다. 아래의 사항들은 프로그램을 구현할 때의 준수 사항이다.

- 사용자가 이해하기 쉬운 방향으로 시스템을 설계한다.
- 성능이 입증된 기술을 사용한다.
- 가능한 공개된 오픈소스를 사용한다.
- 사용자가 이용하기에 부족하지 않은 용량의 코드를 다룰 수 있도록 한다.
- 오류 발생의 경우 빠른 수정이 가능하도록 프로그램을 설계한다.
- 사용자가 효율적인 그린 코딩을 할 수 있도록 유도하는 방식으로 프로그램을 설계한다.
- 사용자가 해당 프로그램 이외에 다른 프로그램을 설치할 필요가 없도록 설계한다.

### 2.5. Assumptions and Dependencies

본 프로그램은 다음과 같은 환경을 가정하고 개발하며 탄소 배출량 계산을 진행하므로 이와 다른 환경에서 이 프로그램을 작동하면 원하는 결과가 도출되지 않을 수 있다.

- 1.0GHz 이상의 프로세서
- 4GB 이상의 RAM
- Windows, MacOS, Linux 환경
- 최신 버전의 Chrome, Firefox, Microsoft Edge 등 주요 웹 브라우저
- 인터넷 연결

## 3. Specific Requirements

### 3.1. External Interface Requirements

#### 3.1.1. User Interface

이름	마우스 및 키보드를 통한 입력 처리
목적/내용	시스템 사용자가 키보드 및 마우스의 입력을 통해 시스템에 명령 전달
입력 주체/출력 목적지	사용자/Windows 기반의 컴퓨터 기기
범위/정확도/허용 오차	<ul style="list-style-type: none"><li>• 범위: 텍스트 박스와 버튼 및 탭</li><li>• 정확도: 유저의 마우스 및 키보드 입력에 따른 정확도</li><li>• 허용 오차: 해당 없음</li></ul>
단위	버튼 클릭/키보드 입력
시간/속도	비정기적인 사용자의 입력/즉각적인 사용자 명령 수행
타 입출력과의 관계	입력 내용에 따라 클라이언트에서 처리 또는 서버로 명령 요청
화면 형식 및 구성	해당 없음
윈도우 형식 및 구성	해당없음
데이터 형식 및 구성	JAVA 코드
명령 형식	서버 송신 명령
종료 메시지	해당 없음

Table 3.1: User Interface of input processing

이름	모니터를 통한 메인 화면 출력
목적/내용	시스템 사용자에게 제공하는 인터페이스
입력 주체/출력 목적지	클라이언트/사용자
범위/정확도/허용 오차	<ul style="list-style-type: none"><li>• 범위: 사용자가 조정한 화면 크기</li><li>• 정확도: 유저의 마우스 및 키보드 입력에 따른 정확도</li><li>• 허용 오차: 해당 없음</li></ul>
단위	화면
시간/속도	사용자의 입력에 따른 화면 전환 및 서버 응답 대기
타 입출력과의 관계	사용자의 입력을 위한 인터페이스로서 출력 후 사용자의 입력 대기

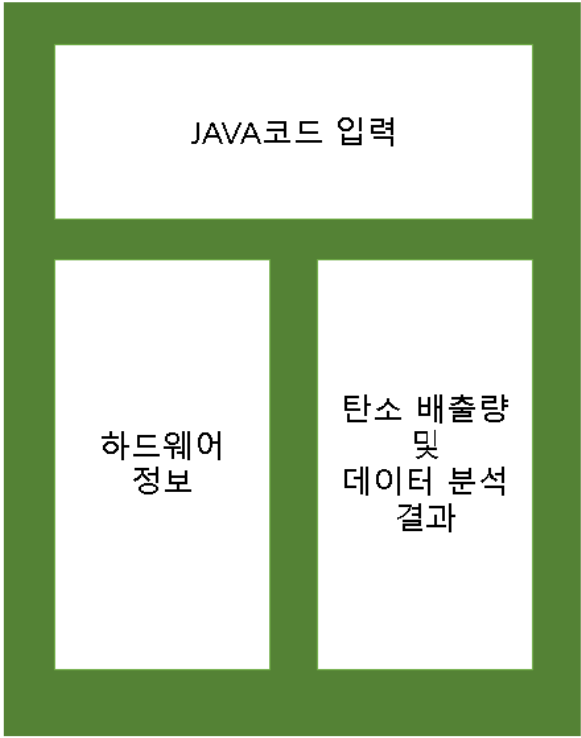
화면 형식 및 구성	 <ul style="list-style-type: none"> <li>개발할 웹사이트는 3가지의 구역(상단의 <b>Java</b> 코드 입력 구역, 좌측 하단의 하드웨어 정보 구역, 우측 하단의 탄소 배출량 및 데이터 분석 결과 구역)으로 구성</li> <li>사용자는 <b>Java</b> 코드 입력 구역을 통해 <b>Java</b> 코드를 입력하거나 버튼을 눌러 코드를 서버로 송신할 수 있음</li> <li>사용자로부터의 버튼 입력이 발생했을 시, 서버 정보를 좌측 하단부에 띄우며 우측 하단부에는 사용자가 입력한 <b>Java</b> 코드의 탄소 배출량과 산출된 탄소 배출량을 기반으로 진행된 데이터 분석 결과를 제시함</li> </ul>
윈도우 형식 및 구성	<ul style="list-style-type: none"> <li>다중 <b>tab</b>과 <b>Java</b> 코드를 작성하는 입력 창</li> <li><b>Linear layout</b> 형식으로 하드웨어 정보 구성</li> <li><b>Grid</b> 형식으로 탄소 배출량 및 데이터 분석 결과 구성</li> </ul>
데이터 형식 및 구성	이미지, 텍스트
명령 형식	해당 없음
종료 메시지	해당 없음

Table 3.2: User Interface of main page

### 3.1.2. Hardware Interface

이름	시스템에서 사용 가능한 디바이스
목적/내용	키보드, 마우스를 사용한 사용자의 입력 및 화면 출력
입력 주체/출력 목적지	사용자/서버
범위/정확도/허용 오차	<ul style="list-style-type: none"> <li>범위: 입출력이 가능한 전자기기 (컴퓨터, 스마트폰 등)</li> <li>정확도: 해당 없음</li> <li>허용 오차: 해당 없음</li> </ul>

단위	해당 없음
시간/속도	사용자의 입력에 해당하는 처리
타 입출력과 관계	해당 없음
화면 형식 및 구성	해당 없음
윈도우 형식 및 구성	해당 없음
데이터 형식 및 구성	해당 없음
명령 형식	코드
종료 메시지	해당 없음

Table 3.3: Hardware Interface of applicable device for the system

### 3.1.3. Software Interface

이름	웹 사이트
목적/내용	페이지 업데이트
입력 주체/출력 목적지	해당 없음
범위/정확도/허용 오차	<ul style="list-style-type: none"> <li>• 범위: Chrome, Edge, Firefox, Safari와 같은 웹 브라우저</li> <li>• 정확도: 해당 없음</li> <li>• 허용 오차: 해당 없음</li> </ul>
단위	해당 없음
시간/속도	새로 고침에 따른 즉각적인 처리
타 입출력과 관계	해당 없음
화면 형식 및 구성	웹 브라우저를 통한 웹 사이트 출력
윈도우 형식 및 구성	해당 없음
데이터 형식 및 구성	해당 없음
명령 형식	해당 없음
종료 메시지	해당 없음

Table 3.4: Software Interface of applicable device for the system

### 3.1.4. Communication Interface

이름	호스트 서버 - 사용자
목적/내용	각 사용자에서 호스트 서버에 접속을 요청하고, 사용자가 입력한 코드를 호스트 서버에서 각 사용자에게 전달받고 이에 해당하는 결과를 제공
입력 주체/출력 목적지	사용자와 호스트 서버
범위/정확도/허용 오차	해당 없음

단위	패킷
시간/속도	최소 10Mbps 이상
타 입출력과의 관계	해당 없음
화면 형식 및 구성	해당 없음
윈도우 형식 및 구성	해당 없음
데이터 형식 및 구성	<ul style="list-style-type: none"> <li>• Java code</li> <li>• 코드 실행 명령</li> </ul>
명령 형식	HTTP POST
종료 메시지	해당 없음

Table 3.5: Communication Interface of applicable device for the system

## 3.2. Functional Requirements

### 3.2.1. Use Case

이름	코드 입력
Actor	웹 사이트 방문자
Description	사용자가 탄소배출량을 측정할 코드를 입력하는 과정이다.
Normal Course	<ul style="list-style-type: none"> <li>• 사용자는 웹사이트에 접속한다.</li> <li>• 사용자는 제공된 탭에 <b>Java</b> 코드를 입력한다. 다중 탭으로 여러 클래스로 이루어진 입력도 가능하다.</li> </ul>
Precondition	• 사용자는 올바른 <b>Java</b> 코드를 입력해야 한다.
Post Condition	• 코드가 계속 입력 칸에 남아있게 해서, 결과 확인 후에도 어떤 코드에 의한 탄소배출량인지를 확인할 수 있어야 한다.
Assumption	해당 없음

Table 3.6: Use Case of Code Input

이름	코드 실행
Actor	서버
Description	웹사이트에서 서버로 코드를 송신해 서버에서 해당 코드를 실행하는 과정이다.
Normal Course	<ul style="list-style-type: none"> <li>• 사용자는 코드를 입력한 후, <b>Compile</b> 버튼을 클릭한다.</li> <li>• 서버는 사용자로부터 받은 코드를 실행한다.</li> </ul>
Precondition	• 서버는 정상적으로 실행되는 <b>Java</b> 코드를 수신받았을 경우, 코드를 정상적으로 실행해야 한다.

	<ul style="list-style-type: none"> <li>컴파일되지 않는 <b>Java</b> 코드를 수신받았을 경우, 올바른 오류 메시지를 반환해야 한다.</li> <li>코드가 실행되는 동안, 실행되는 시간을 측정해야 한다.</li> </ul>
Post Condition	<ul style="list-style-type: none"> <li>컴파일하는 과정에서 서버에 생성된 자바파일과 클래스파일들을 모두 삭제해야 한다.</li> </ul>
Assumption	해당 없음

Table 3.7: Use Case of Code Execute Code

이름	결과 분석
Actor	서버
Description	서버가 코드를 실행한 후 측정된 결과와 서버 환경을 바탕으로 탄소배출량을 계산하는 과정이다
Normal Course	<ul style="list-style-type: none"> <li>컴파일이 완료되면, 서버의 시스템 환경을 얻는다.</li> <li>컴파일 결과와 시스템 환경을 종합해서 전력소모량과 탄소배출량을 계산한다.</li> <li>계산된 결과를 사용자에게 송신한다.</li> </ul>
Precondition	<ul style="list-style-type: none"> <li>서버 컴퓨터가 사용하는 프로세서의 정보가 서버의 데이터에 저장되어 있어야 한다.</li> </ul>
Post Condition	<ul style="list-style-type: none"> <li>에러가 발생한 경우에 오류 메시지를 송신해야 한다.</li> <li>정상적으로 실행된 경우 계산된 결과를 사용자에게 송신해야 한다.</li> </ul>
Assumption	해당 없음

Table 3.8: Use Case of Results Analysis

이름	결과 확인
Actor	웹 사이트 방문자
Description	사용자가 코드를 실행한 결과를 확인하는 과정이다.
Normal Course	<ul style="list-style-type: none"> <li>서버에서 분석이 끝나면 웹페이지는 결과를 수신해 자동으로 페이지를 업데이트한다.</li> <li>페이지에 나타난 분석 결과를 확인한다.</li> </ul>
Precondition	<ul style="list-style-type: none"> <li>사용자가 페이지에 접속된 상태여야 한다.</li> </ul>
Post Condition	해당 없음
Assumption	해당 없음

Table 3.9: Use Case of Confirm Results

### 3.2.2. Use Case Diagram

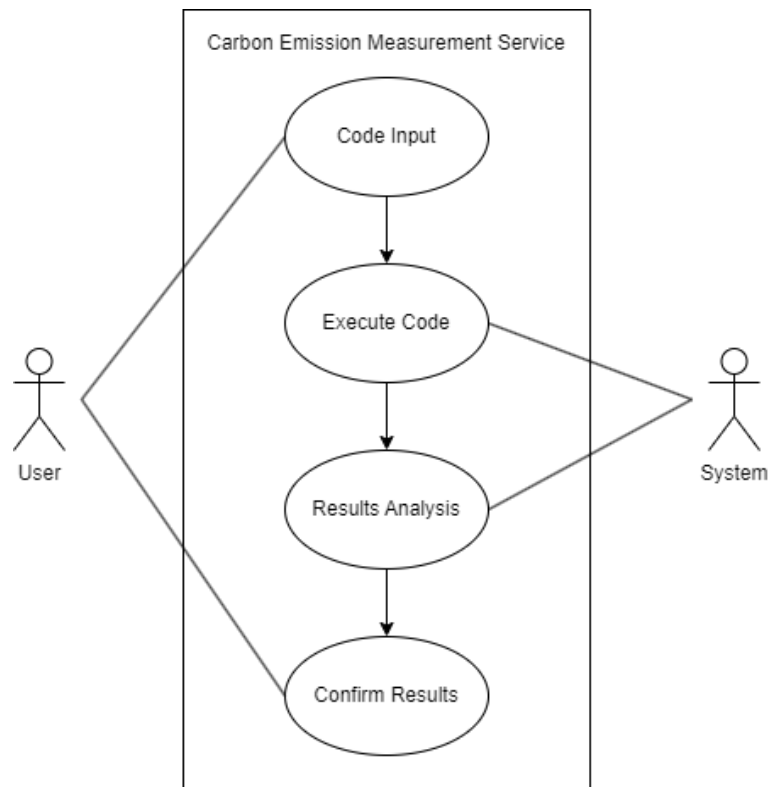


Image 3.1: Use Case Diagram

### 3.2.3. Data Flow Diagram

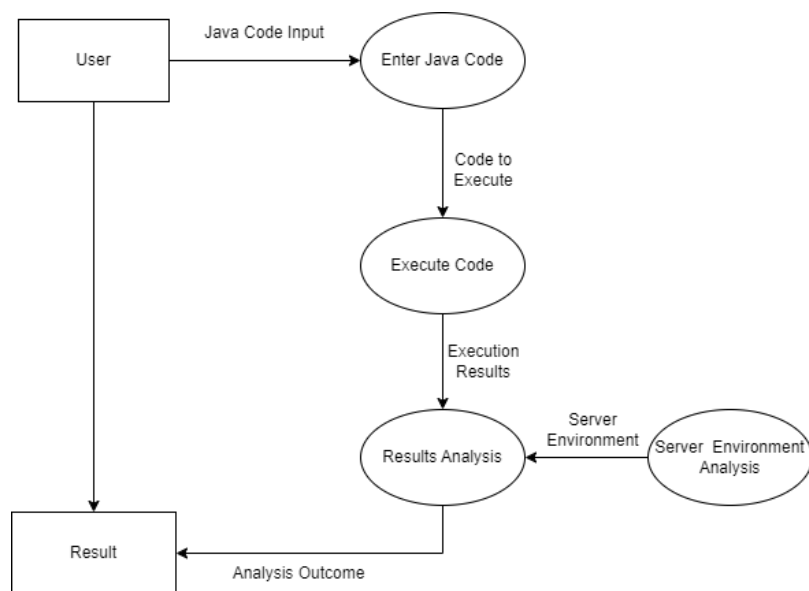


Image 3.2: Data Flow Diagram

### 3.3. Performance Requirements

아래는 본 시스템의 성능 요구사항에 관한 것이다. 예측에 기반한 내용이며 실제 구현시에 달라질 수 있다.



### 3.3.1. Static Numerical Requirement

- 이 시스템은 사용자가 자신의 코드를 입력하고, 입력한 코드를 바탕으로 발생하는 탄소배출량 시각화를 지원한다. 해당 사용자는 입력할 코드를 가지고 있어야 한다.
- 시스템은 1GHz 이상 프로세서, 4GB RAM 이상의 데스크탑 Windows/MacOS/Linux 환경이어야 한다.
- 10Mbps 이상의 인터넷 연결 속도 환경을 요구한다. 웹브라우저 사용이 가능해야 하며, 웹브라우저 사용시 원활하게 동작해야 한다.

### 3.3.2. Dynamic Numerical Requirement

- 시스템은 동시에 최소한 10명 사용자의 접속을 유지할 수 있어야 한다. 각 웹페이지는 동시에 최소한 10명의 사용자가 접속하여도 10초 이내 page load를 완료할 수 있어야 한다.
- 사용자가 Java code tab 변경시 화면이 5초 이내로 표시되어야 한다.
- 1초 이내의 Runtime을 가지는 Java code 실행 시, 시스템 과부하 없이 작동해야 한다. 여러 사용자가 접속하더라도 5초 이내로 실행 완료되어야 한다.
- 1초 이상의 Runtime을 가지는 Java code 실행 시, 10초 이내에 실행을 완료해야 한다. 여러 사용자가 접속하더라도 10초 이내로 실행 완료되어야 한다.
- Java code 오류 발생 또는 10초 이상의 실행 시, 실행을 중지하고 11초 이내로 이를 사용자에게 알려주어야 한다.
- Java code 탄소배출량 시각화는 12초 이내에 완료해야 한다.

## 3.4. Design Constraints

### 3.4.1. Physical design constraints

시스템의 목적은 이용자의 Java code의 탄소배출량을 측정하는 것이므로 데스크탑, 노트북, 모바일 스마트폰을 이용할 것을 권장한다. 시스템은 여러 탭에 나누어진 Java code를 로컬 컴퓨터로 이동시켜 컴파일이 가능한 환경을 조성해야 한다.

### 3.4.2. Standards compliance

웹사이트는 웹 어플리케이션으로 React를 통해 개발되며, React 코딩 표준(JavaScript Standard style) 및 HTML 표준을 따른다. 변수 이름은 Camel Case를 따른다.

서버는 Python과 FastAPI를 통해 개발되며, Python 코딩 표준을 따른다. 변수 이름은 Snake Case를 따른다.

## 3.5. Software System Attributes

아래는 본 시스템의 몇 가지 비기능적 요구사항에 관련된 내용이다. 비기능적 요구사항은 제품 요구사항, 조직상의 요구사항, 외부적인 요구사항으로 나누어 기술하였다.

### 3.5.1. Product Requirements

제품 요구사항에서는 본 시스템의 실행 시간 중에 시스템이 어떻게 작동해야 하는지에 대해 설명한다. 본 시스템은 아래의 요구사항을 만족해야 한다.

#### Usability Requirements

본 시스템은 사용자의 Java code를 컴파일하고 탄소배출량을 시각화해야 하기 때문에, 사용자 Java code의 오류를 공지하고, 오류가 없다면 컴파일 할 수 있어야 한다. 다중 Class를 지원하기 때문에, 여러 class로 구성된 Java code 입력을 지원해야 한다. 또한 여러 탭에 나누어진 Java code를 컴파일 할 수 있어야 한다. 탄소배출량 시각화는 실생활에 주로 사용되는 것을 예시로 제시해서 누구나 쉽게 이해할 수 있어야 한다.

### Performance Requirements

여러 Class를 가진 Java code의 실행이 가능해야 한다. Java code를 입력하고 탄소배출량을 시각화하기까지 12초 이내에 완료해야 한다. 코드의 오류나, Runtime이 10초 이상일 경우에 11초 이내로 실행을 중지하고 사용자에게 공지해야 한다.

### Security Requirements

다른 사용자의 코드는 확인할 수 없으며, 입력한 코드에 관한 기록은 서버에 남지 않는다. 사용자는 자신과 관련된 제한된 데이터만을 볼 수 있으며, 탄소배출량을 계산하는 서버에 직접적으로 접근할 수 없어야 한다.

### 3.5.2. Organizational Requirements

조직상의 요구사항은 사용자 및 개발자가 속한 조직의 정책과 절차에 의해 발생하는 요구사항들에 관한 설명이다.

### Environmental Requirements

Java code를 입력할 수 있는 데스크탑, 노트북, 모바일 스마트폰 웹 브라우저 환경에서 접속을 권장한다.

### Operational Requirements

시스템은 사용자가 원하는 때 언제든지 자유롭게 접속하여 Java code를 입력하고, 탄소 배출량 시각화를 진행할 수 있다.

### 3.5.3. External Requirements

외부적인 요구사항에서는 시스템의 외부적 요인으로 인해서 발생하는 다양한 요구사항들에 관해서 설명한다.

### Safety / Security Requirement

시스템은 사용자가 입력한 코드에 대해서 어떠한 정보도 다른 사용자가 알 수 없도록 해야 한다.

### Regulatory Requirements

본 시스템은 오로지 사용자의 Java code를 탄소배출량 계산을 위한 목적으로만 사용함을 밝히고, 추가적인 사용자에게 관한 정보는 이용하지 않는다. 법에 따라 사용자의 Java code를 복제해서는 안된다. 시스템은 국제 개인 정보 보호 표준에 따라 개발되어야 한다.

## 3.6. Organizing the Specific Requirements

이 구간에서는 Unified Modeling Language(UML) 및 표 형식 기반의 그래픽 표기법을 사용하여 시스템 모델을 설명한다. 시스템 모델은 시스템, 서브 시스템 간의 관계를 설명한다.

### 3.6.1. Context Model

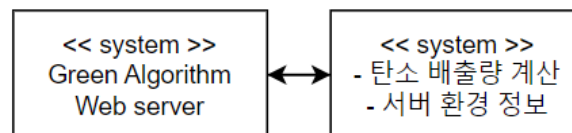


Image 3.3: Context Diagram

### 3.6.2. Process Model

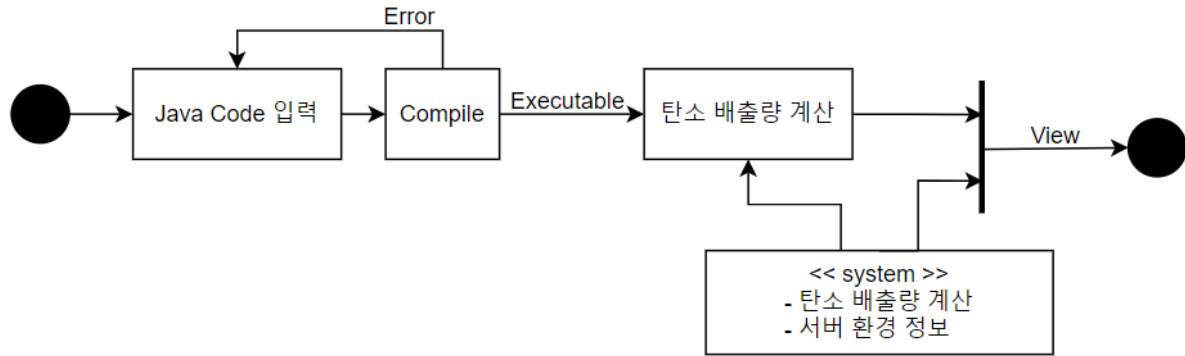


Image 3.4: Process Diagram

### 3.6.3. Interaction Model

- Use Case Diagram 참고

## 3.7. System Architecture

이 절에서는 개발될 시스템 아키텍처에 대한 **high-level** 개요를 제시한다. 이는 각각의 서브 시스템과 그 구성요소를 밝히고 서브 시스템 간의 통신이 어떻게 이루어지는 지 명시한다.

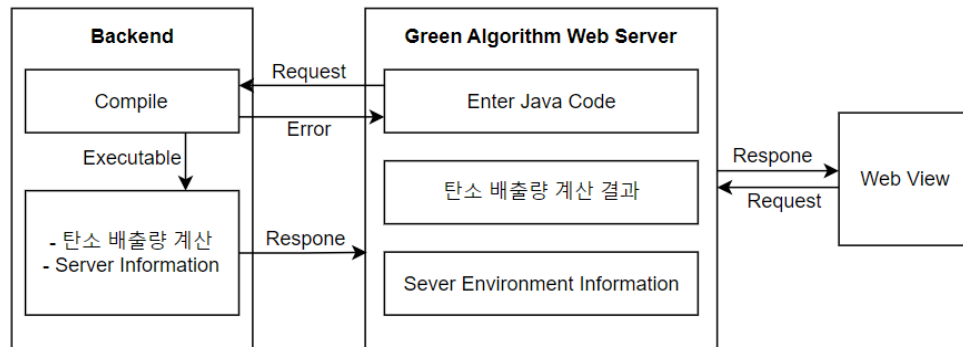


Image 3.5: System Architecture Diagram

## 3.8. System Evolution

이 절에서는 시스템 도입 이후 생길 수 있는 사용자 요구사항 변화나 하드웨어 환경 변화, 또는 시스템에 대한 기본 가정의 변화 등 예상할 수 있는 주요한 변화를 설명한다. 이를 통해 시스템의 결점을 파악할 수 있으며, 시스템 설계자는 향후 시스템의 개선이 필요할 경우 면밀하게 그 제약사항을 파악하여 유지보수 수요에 대응할 수 있다.

### 3.8.1. Limitation and Assumption

본 시스템은 사용자가 웹사이트에 접속하여 **Java** 코드를 서버에 전송하고, 서버는 이를 실행하여 탄소배출량을 계산한 뒤 실행결과를 웹사이트에 다시 반환하는 경우를 가정하고 있다. 여기에서 가장 중요한 시스템은 **Java** 코드를 실행하는 서버 시스템이다. 사용자는 **Java**만을 사용해 탄소배출량 계산을 시도해야 하며, 다른 프로그래밍 언어는 지원하지 않는다. 사용자는 작성한 코드를 서버에 전송하고 서버를 통해 탄소배출량을 계산해야 한다. 서버 시스템은 지정된 특정 서버만을 활용해 탄소배출량을 계산한다. 사용자는 반환받은 예상 탄소배출량을 몇몇 탄소배출행위에서 발생하는 탄소 배출량과 비교하여 함께 확인할 수 있다.

### 3.8.2. Evolution of Hardware and Change of User Requirements

다른 프로그래밍 언어를 통해 탄소배출량을 계산하고 싶은 사용자를 위해 **Java** 이외 언어에 대한 컴파일 환경이나 인터프리터 환경을 확충해야 한다. 또한 **GPU** 등 다양한 구동 환경을 지원할 수 있도록

확장성을 지원하여야 한다. 사용자가 원하는 환경에서 자신이 작성한 코드를 구동할 수 있도록 사용자 개발 환경을 지원하여야 한다. 사용자가 주어진 탄소배출행위 이외의 탄소배출원과 자신의 코드가 얼마나 탄소를 배출하는지 그 정도를 비교할 수 있도록 웹페이지가 사용자가 스스로 탄소배출원의 특성을 설정하여 제시할 수 있도록 기능을 제공하여야 한다. 본 시스템은 위와 같은 기능을 고려하여 발전해 나갈 것이다.

## 4. Appendix

### 4.1. Software Requirements Specification Standard

본 요구사항 명세서는 IEEE 표준(IEEE Std 830-1998 - IEEE Recommended Practice for Software Requirements Specifications)을 기반으로 작성하였다. 다만 요구사항을 기술하는 데 필요하지 않은 일부 구성요소는 활용하지 않았다.

### 4.2. Document History

Document History			
Date	Version	Description	Writer
2023-10-24	V 1.00	Introduction, Product Perspective	양승빈
2023-10-24	V 1.00	Product Functions, User Classes and Characteristics, Design and Implementation Constraints, Assumptions and Dependencies	윤시형
2023-10-24	V 1.00	External Interface Requirements	김민지
2023-10-24	V 1.00	Functional Requirements	김찬용
2023-10-24	V 1.00	Performance Requirements, Design Constraints, Software System Attributes	안상현
2023-10-24	V 1.00	Organizing the Specific Requirements, System Architecture	최경식
2023-10-24	V 1.00	System Evolution, Appendix	임동준
2023-10-27	V 1.01	Formatting	김민지/양승빈