**캡스톤 디자인 I**

**종합설계 프로젝트**

|  |  |
| --- | --- |
| 프로젝트 명 | Online algorithm battle service |
| 팀 명 | 알고리즘 서비스 연구소 |
| 문서 제목 | 계획서 |

|  |  |
| --- | --- |
| **Version** | 1.0 |
| **Date** | 2020-03-22 |

|  |  |
| --- | --- |
| **팀원** | 최명서 (조장) |
| 최지욱 |
| 박호준 |
| 우현웅 |
| 강수련 |
| 칼리드 |

**문서 정보 / 수정 내역**

|  |  |
| --- | --- |
| **Filename** | 계획서-한판고.doc |
| **원안작성자** | 최명서, |
| **수정작업자** | 최명서, 최지욱, 박호준, 우현웅, 강수련 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 수정날짜 | 대표수정자 | Revision | 추가/수정 항목 | 내 용 |
| 2020-01-02 | 최명서 | 1.0 | 최초 작성 |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**목 차**

[**1** **개요** 4](#_Toc347412182)

[1.1 프로젝트 개요 4](#_Toc347412183)

[1.2 추진 배경 및 필요성 4](#_Toc347412184)

[**2** **개발 목표 및 내용** 5](#_Toc347412185)

[2.1 목표 5](#_Toc347412186)

[2.2 연구/개발 내용 6](#_Toc347412187)

[2.3 개발 결과 7](#_Toc347412188)

[2.3.1 결과물 목록 및 상세 사양 7](#_Toc347412189)

[2.3.2 시스템 기능 및 구조 7](#_Toc347412190)

[2.4 기대효과 및 활용방안 7](#_Toc347412191)

[**3** **배경 기술** 8](#_Toc347412192)

[3.1 기술적 요구사항 8](#_Toc347412193)

[3.2 현실적 제한 요소 및 그 해결 방안 9](#_Toc347412194)

[3.2.1 하드웨어 9](#_Toc347412195)

[3.2.2 소프트웨어 9](#_Toc347412196)

[3.2.3 기타 9](#_Toc347412197)

[**4** **프로젝트 팀 구성 및 역할 분담** 10](#_Toc347412198)

[**5** **프로젝트 비용** 10](#_Toc347412199)

[**6** **개발 일정 및 자원 관리** 11](#_Toc347412200)

[6.1 개발 일정 11](#_Toc347412201)

[6.2 일정별 주요 산출물 12](#_Toc347412202)

[6.3 인력자원 투입계획 13](#_Toc347412203)

[6.4 비 인적자원 투입계획 14](#_Toc347412204)

[**7** **참고 문헌** 15](#_Toc347412205)

# **개요**

## 프로젝트 개요

본 프로젝트는 일정수준의 코딩이 가능하고 기본적인 알고리즘 공부가 된 사람들에게 게임에 대한 알고리즘을 작성하고 다른 사람들과 경쟁해 볼 수 있는 환경을 제공하는 웹사이트를 개발하는 프로젝트이다. 프로젝트는 총 세개의 서버로 나뉘어져 있으며, 클라이언트 서버, api서버, 유저들의 코드를 대전시키는 core서버가 있다.

## 추진 배경 및 필요성

근 몇 년간 AI 열풍이 불면서 알고리즘에 대한 관심도 높아졌다. 초중고 교육 현장에서는 sw교육 열풍이 불고 있고 전공자를 비롯한 많은 사람들이 백준과 프로그래머스 같은 알고리즘 트레이닝 사이트를 통해 알고리즘 트레이닝을 시작했다.



<그림 > 소프트웨어 교육 현황

하지만 이 사람들 중에 꾸준하게 알고리즘 트레이닝을 하는 사람들은 얼마나 될까. 시작한 사람들에 비해 적은 것은 자명하다. 그렇다면 이렇게 적은 수의 사람들만 꾸준하게 하는 이유는 무엇일까. 우리 팀의 결론은 지속적인 동력이 부족하다는 것이고 이러한 동력을 재미와 경쟁심으로 채워나갈 수 있다고 판단했다.

재미와 경쟁심을 동력으로 이용한 가장 효과적인 방법이 무엇인지 생각해 봤을 때 게임을 가장 쉽게 떠올릴 수 있었다. 단순히 문제에 대한 알고리즘을 짠 후에 테스트 케이스에 맞는지 채점 받는 자기만족에서 끝나는 것이 아닌 재미가 있는 게임, 다른 사람들의 알고리즘과의 경쟁을 통해 재미와 경쟁심을 동력으로 제공함으로써 지속적인 알고리즘 트레이닝을 할 수 있는 환경을 제공하기로 하였다.

### 기술 시장 현황

1. 국내

* 백준 온라인 저지(Baekjoon Online Judge)



<그림 > 백준 온라인 저지

아마 CS전공 학생이라면 한 번쯤은 "백준 풀었다**"**는 어휘를 사용해본 경험이 있을 것이다. 그 정도로 백준은 이미 국내에서 엄청나게 유명한 사이트이고, 단순한 문제 제공을 넘어서 모의대회 주최나 그룹 랭킹 등 다양한 기능도 제공해주고 있는 플랫폼이다.

특징

* 국내 최대 규모의 알고리즘 사이트(약 2만 문제 보유)
* 표준 입·출력 기반 풀이 작성
* 약 50가지 프로그래밍 언어 지원
* 단계별, 카테고리별 분류된 문제 제공
* 소요 시간, 메모리 사용 등 평가 결과 제공
* 개인, 학교/기관 등 랭킹 제공
* 국내외 알고리즘 대회 기출문제 제공
* 커스텀 그룹 형성이 가능해 자체적인 대회 주최 가능
* 프로그래머스
* 

<그림 > 프로그래머스

프로그래머스는 최근 떠오르고 있는 알고리즘 사이트이다. 프로그래머스는 단순히 알고리즘 문제를 제공해주는 것을 넘어, 국내 기업의 코딩 테스트를 진행하기 위한 플랫폼을 제공해주고 있다. 일례로 카카오, 네이버, 라인, NHN 등 국내 IT 기업의 신입사원 공채 코딩 테스트가 프로그래머스에서 진행되었다. 최근에는 "웹 개발자 챌린지" 등의 이름으로 수많은 스타트업들이 프로그래머스를 통해 개발자 채용을 진행하고 있다.

특징

* 네이버, 카카오 등 국내 기업의 코딩테스트 환경 제공
* 솔루션 함수 작성 기반 문제 풀이(입출력 기반 X)
* 단계별 분류된 문제 제공
* 코딩 테스트에 특화된 플랫폼
* 12가지 프로그래밍 언어 제공
* 개인 랭킹 제공
* 가시적인 테스트 케이스 제공
* 백준에 비해 문제 수가 적음

1. 해외

* 리트 코드(Leet Code)

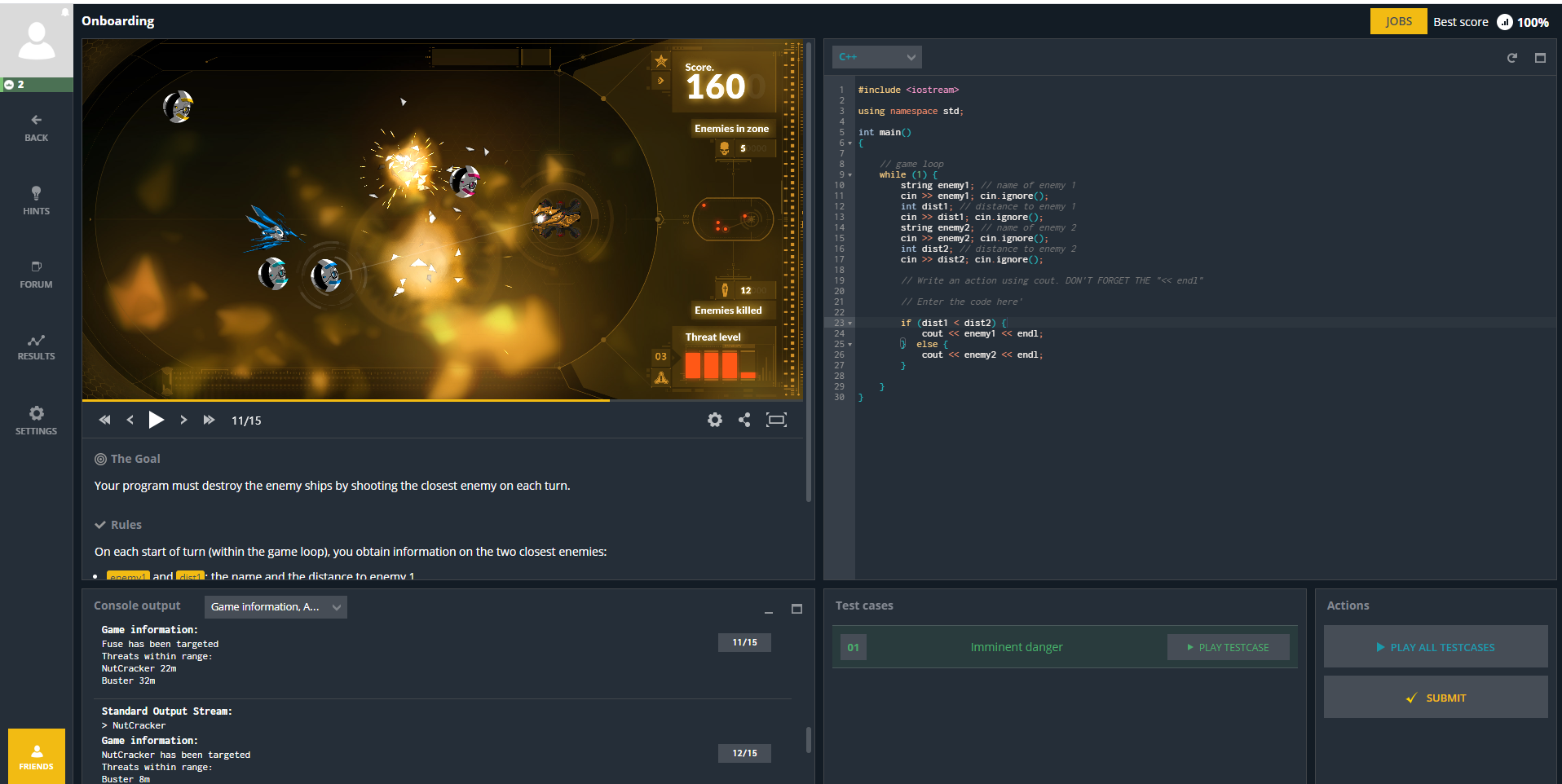


<그림 > 리트코드

리트코드는 세계적으로 유명한 알고리즘 사이트 중 하나이다. 문제의 양이 방대하고 종류도 다양해서 국내 및 해외에서 알고리즘 트레이닝을 위한 사이트로 많이 활용되고 있다.

특징

* 구글, 페이스북, 아마존 등 세계적인 IT 기업의 코딩 테스트 및 면접 기출문제 보유
* Weekly Contest 온라인 참가 가능
* 문제가 전부 영어
* Easy, Normal, Hard 문제 난이도 제공
* 각 문제에 대한 Discuss 가능
* 문제 해결 시 전체 풀이 대비 내 풀이의 성능 평가 제공
* Coding game

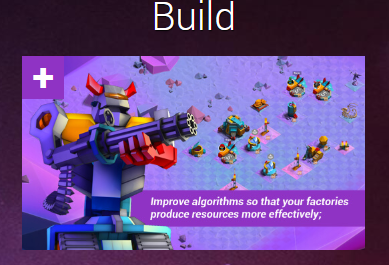


<그림 > coding game 실행 화면

Coding game은 코딩게임을 제공하는 서비스이다. 게임에 코딩을 접목하였고 25개의 프로그래밍 언어를 지원하여 더 재밌는 알고리즘 트레이닝을 할 수 있게 도와준다.

특징

* 코딩에 게임을 접목함으로써 실제적이고 자극적인 코딩 도전을 제공(Be developer-friendly)
* 많은 수요(2020년 3월 21일 기준 CodinGame 회원수 약 1,809,495명 집계)
* 코딩을 지식을 조금 갖고 있어야 서비스 이용 가능
* 2,000 개 이상의 게임 카테고리를 제공
* Empire of code



<그림 > Empire of code 사이트

Empire of code는 전략, 전술을 바탕으로 진행하는 게임에 코딩이 혼합된 서비스이다.

Python, JacaScript를 기반으로 건물들을 짓고 상대방과 싸워 이기는 것이 목표이다..

코딩 기술의 유무에 관계없이 게임을 할 수 있지만 코딩을 할 줄 아는 것이 확실한 이점이 된다.

특징

* 코딩 입문자도 가능한 코딩게임 서비스
* 타 코딩게임 사이트에 비해 실제 게임에 근접한 UI와 Ux로 더 흥미로운 서비스 이용 가능
* Python, JavaScript 코딩 기술스택만 지원, CodinGame과 같은 사이트에 비해 부족함

# **개발 목표 및 내용**

## 목표

본 프로젝트는 알고리즘 트레이닝에 지속가능한 동력을 제공하기 위한 웹사이트 제작을 목표로 한다. 상기 목적을 달성하기 위해 다음과 같은 기능을 갖는다.

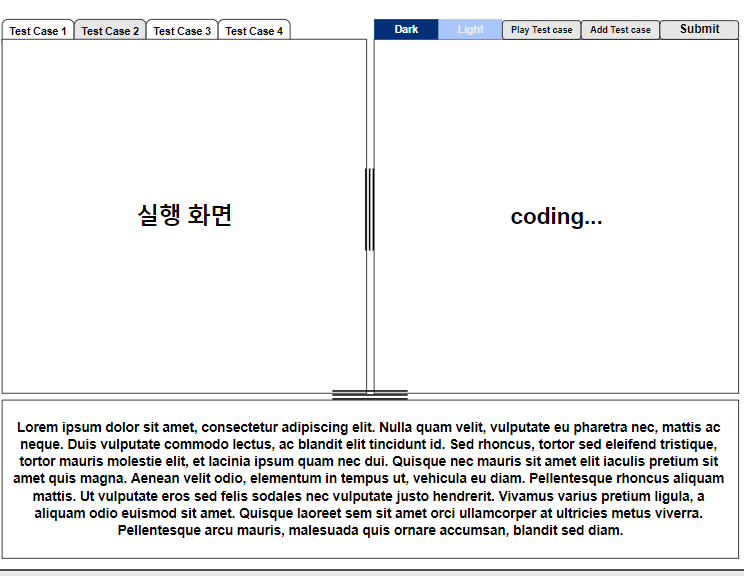
* 유저와 유저가 한 게임을 두고 알고리즘 제출한 후에 대전하는 기능
* 유저가 대전 기록의 리플레이를 보는 기능
* 유저가 자신의 코드와 대전할 수 있는 기능
* 관리자가 문제를 쉽게 추가할 수 있는 기능

## 연구/개발 내용

### 클라이언트 서버

다음과 같은 특징에 의해 JaveScript 프레임 워크로는 React/Redux를 선정하게 되었다.

* 가장 거대한 커뮤니티
* 컴포넌트의 가독성이 높고 간단하여 쉬운 유지보수 가능
* UI 수정 및 재사용 용이
* 버전 호환성 좋음
* 타 프레임워크와 혼용 가능
* Redux와 Hooks를 통한 state관리



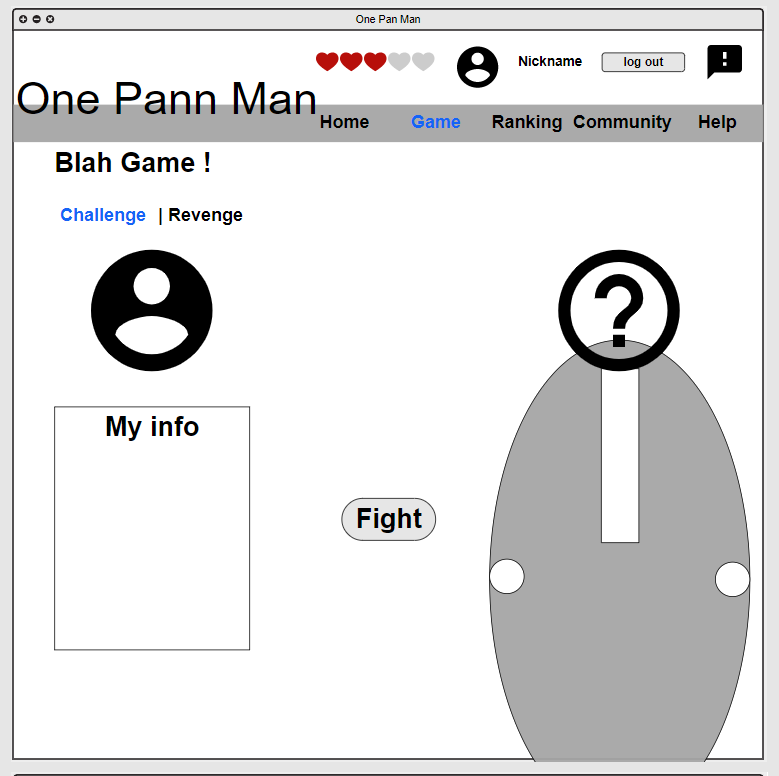
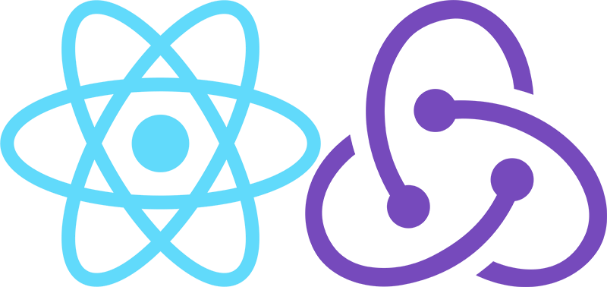


그림 대전 화면 예시



<그림 > Material UI, React, Redux 로고

React의 디자인 프레임워크로는 Material UI를 채택하였다. Material UI는 Bootstrap UI와 유사한 디자인 프레임워크지만 현재 공식 사이트 자체에 잘 정리된 문서, 참고 가능한 방대한 양의 데이터가 github를 비롯한 웹에 존재한다. 특히 React 환경에 특화가 되어있다는 점에서 이를 채택하였다.



<그림 > Phaser 로고

리플레이 페이지에서는 Phaser3 프레임 워크를 이용하여 실제로 게임을 하는 듯한 생동감 있는 화면을 보여주도록 한다. Phaser3는 HTML.5게임을 만들기 위한 오픈소스 프로젝트로써 데스크톱 및 모바일 용 무료 소프트웨어 2D게임 프레임워크이다.

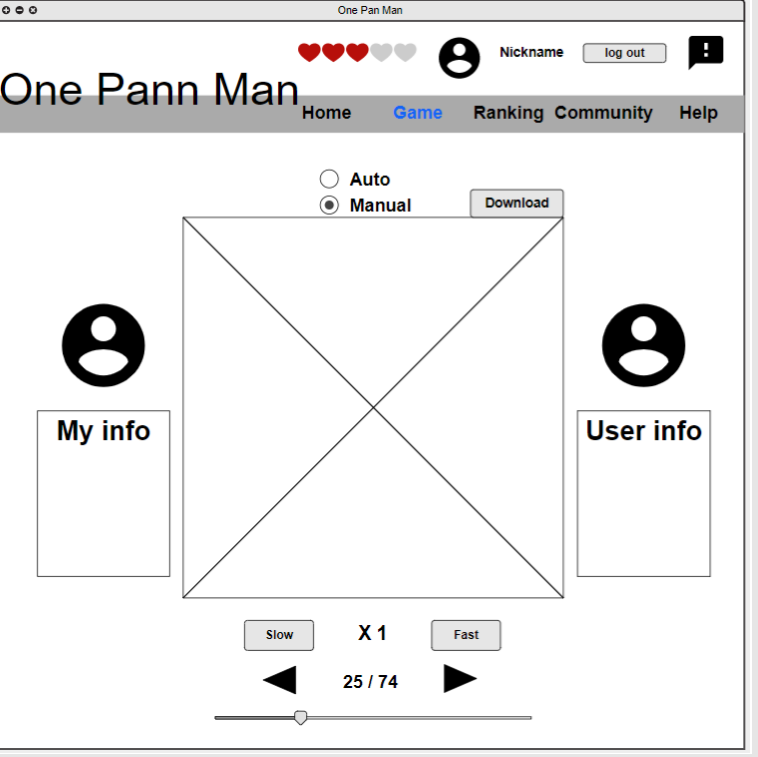
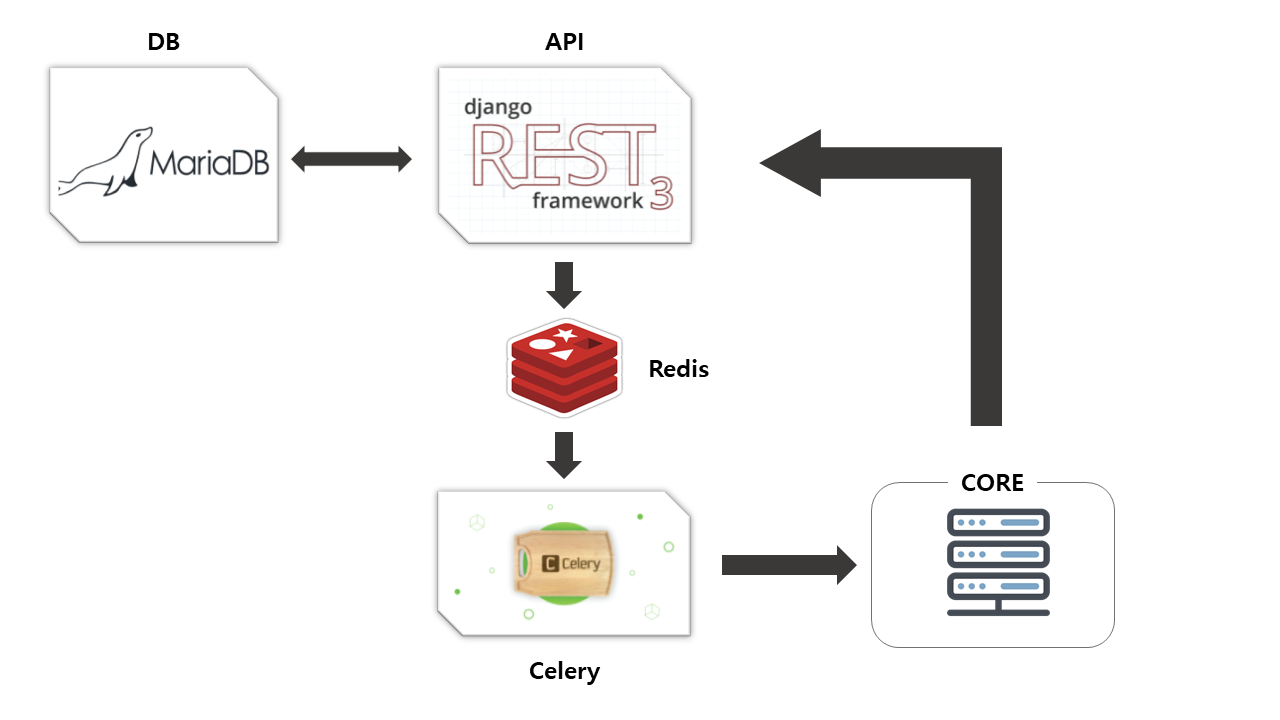


그림 리플레이 화면 예시

### API 서버



<그림 > API 서버와 Core 서버의 통신 구조

본 프로젝트에서는 대전 서비스 뿐만 아니라 유저들의 코드 데이터 및 게임 데이터관리도 중요한 요소이다. 이를 위해 API서버를 따로 개발하여 데이터 관리를 용이하게 한다.

API 서버를 개발하는 도구로는 Django 프레임워크의 Django Rest Framework 모듈을 사용하여 RESTAPI를 구현한다. Database데이터가 정형화 데이터에 적합하기 때문에 RDB인 mariaDB를 사용한다. 유저들의 코드를 컴파일 하고 실행하는 과정과 같이 응답을 받기 오래 걸리는 작업이 필수적으로 들어가야 하기 때문에 보다 더 나은 웹 서비스를 제공하기 위해 비동기적으로 작업을 처리하기로 결정하였다. 이 과정을 쉽게 하도록 도와주는 파이썬 프레임워크인celery를 사용하여 core서버와 통신한다. celery에 작업을 전달하기 위해 브로커로는 In-memory데이터 저장 장치인 Redis를 사용한다. 또한 팀 내에서 API 문서 공유는 온라인 API문서 자동화 오픈소스 swagger / redoc을 사용한다.

### Core서버

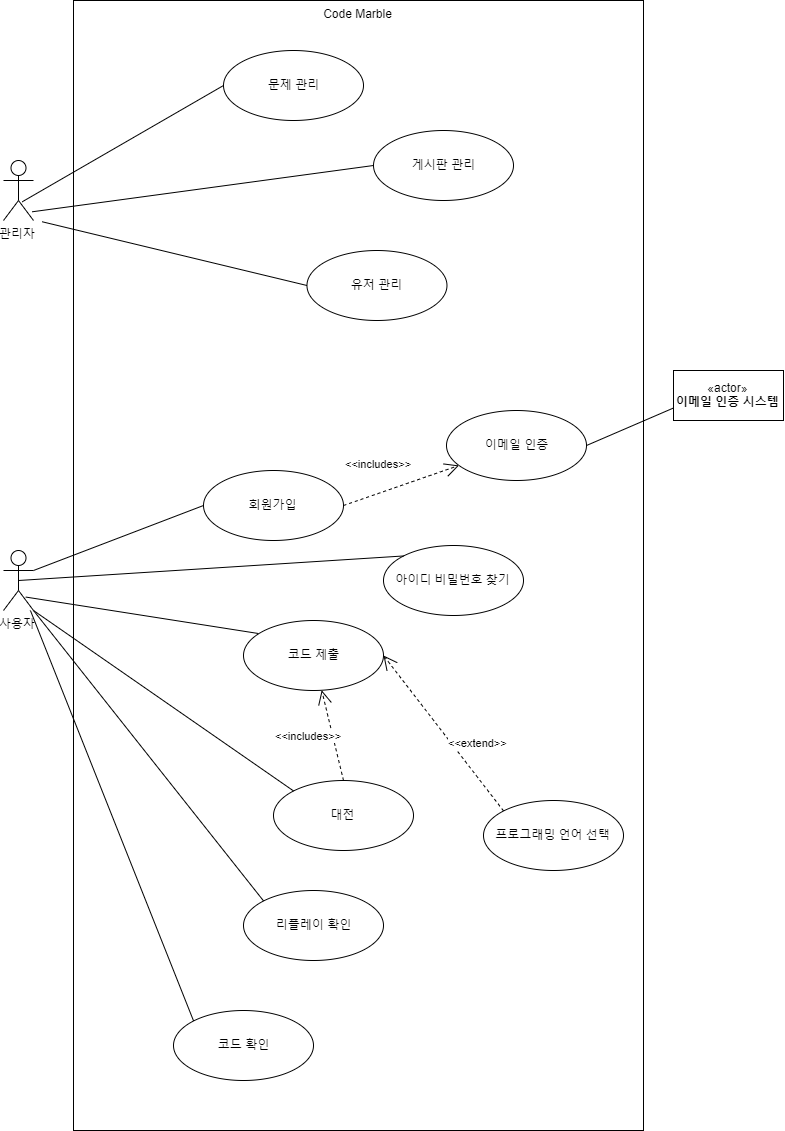
유저의 코드를 컴파일하고 실행하여 대전을 하는 과정이 응답을 받기까지 오래 걸리는 상황을 예상해서 비동기적으로 처리하기로 결정하여 만든 서버이다.

이 서버에서는 유저들의 코드를 컴파일 하고 대전을 하는 로직이 실행되는데 유저들의 코드가 오류가 있을 경우를 대비해 Docker를 사용하여 Docker안에서 Core프로그램이 실행된다.

게임의 규칙들을 크게 착수, 액션, 엔딩 3가지로 나누고 각각의 규칙 안에서 작은 규칙들로 나누어서 게임마다 규칙을 조합하여 유저의 착수를 확인하고 액션을 취하고 엔딩을 검사하는 로직을 진행한다. 유저의 코드에서 일어나는 오류를 최대한 자세하게 알려주기 위해 상황마다 자세하게 오류를 출력하게 구현할 예정이다.

## 개발 결과

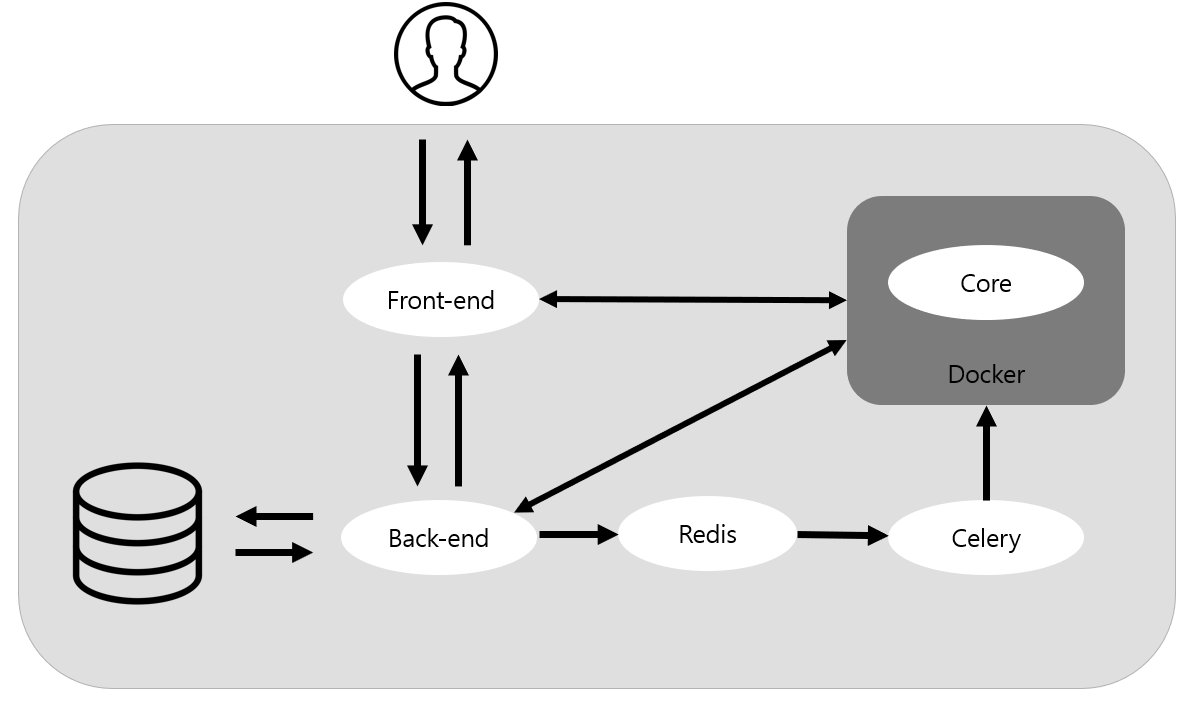
### 시스템 기능 요구사항

****

### 시스템 비기능(품질) 요구사항

|  |  |  |
| --- | --- | --- |
| **비 기능적인 요구사항**  **(Non-Functional Requirements)** | **내용** | **중요도 (Importance)** |
| NFR1 – 보안성 | 등록된 사용자만이 시스템이 접근할 수 있어야 한다. | 상 |
| NFR2 – 확장성 | 프론트엔드와 api서버로 나누어서 개발한다. | 중 |
| NFR3 – 품질 | 이 시스템의 성능은 동시 접속자 수 1000명일 때, 초당 10000트랜잭션을 처리할 수 있어야 한다. | 중 |
| NFR4 – 하드웨어 스펙 | 서버는 4TB HDD를 사용한다. | 상 |

### 시스템 구조



### 결과물 목록 및 상세 사양

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 대분류 | 소분류 | 기능 | 형식 | 비고 |
| 파일 | 문제 제출 | 유저가 문제를 제출한다. | 모듈 |  |
| 게임 대전 | 유저끼리 대전한다. | 모듈 |  |
| 리플레이 | 유저가 대전한 게임의 리플레이를 본다. | 모듈 |  |
| 게임 생성 | 관리자가 문제를 생성한다. | 모듈 |  |
| 대전 프로그램 | 유저가 낸 코드를 대전시킨다. | 모듈 |  |
| Docker imgae | 대전 프로그램 실행할 가상환경 | 모듈/Docker |  |
| UI | 웹페이지 | 사용자들이 사용할 web page interface를 제공한다 | 모듈 |  |
| Pdf viewer | 등록된 문제를 보여준다. | 모듈 |  |

## 프로젝트 형상 관리

## 기대효과 및 활용방안

* 본 프로젝트로 만든 웹사이트를 통해 기존의 알고리즘 트레이닝을 지속하지 못하는 유저들이 알고리즘에 대한 경쟁심과 재미를 얻어 알고리즘 트레이닝을 지속하는 효과를 기대한다.
* 알고리즘을 잘한다 하여도 실 구현 능력은 뒤떨어질 수도 있다고 생각했기 때문에 게임에 관한 알고리즘을 구현하는 과정에서 논리적 사고력과 자신의 생각을 구현할 수 있는 능력을 향상시키는 효과도 기대한다.
* 다른 유저와의 대전만이 아닌 자신의 코드와 대전 해볼 수 있는 기능도 추가시켜 자신의 알고리즘의 장단점을 찾아볼 수 있는 효과도 기대한다.
* 관리자 측면에서는 문제를 게임 규칙을 착수, 액션, 엔딩으로 대분류 하고 각각의 대분류 안에서도 규칙들을 소분류 하여 관리자가 원하거나 유저가 원하는 문제를 추가하기 쉽게 하여 손쉬운 관리를 기대한다.

# **배경 기술**

## 기술적 요구사항

1. 개발환경 - 운영체제

우분투 18.04, 맥OS, 윈도우10

1. 개발환경 – 컴파일 환경

우분투 18.04

1. 개발언어

Python 3.6, Javascript, html, css

1. 라이브러리

Django, Celery, Redis, numpy, yarn, Docker

1. 프레임워크

React, parser3

## 현실적 제한 요소 및 그 해결 방안

### 하드웨어

유저의 코드를 실행 시켜야 하므로 많은 변수가 존재한다. 따라서 core서버를 따로 두어 비동기적을 처리한다.

### 소프트웨어

서비스상에서 모든 언어를 지원할 수 없다. 따라서 한정된 언어만 지원할 예정이다.

### 기타

현재 코로나 바이러스로 인해 프로젝트 회의 및 미팅에 차질이 일어나고 있다.

팀원 중 외국인 팀원은 본국으로 귀국하였고, 나머지 팀원들은 외출을 자제하다 보니 서로 간의 소통이 부족한 문제가 발생한다.

이를 화상회의 프로그램 zoom, 팀 협업도구 slack 등을 이용하여 최대한 온라인으로 소통하려 한다.

# **프로젝트 팀 구성 및 역할 분담**

| 이름 | 역할 |
| --- | --- |
| 최명서 | * Software Project Leader * Core설계 및 구현 * DB 설계 * Docker container 구현 * Documentation |
| 최지욱 | * Front-end 개발(UI 및 url 설계) * 리플레이 화면 구현 * Documentation |
| 박호준 | * Front-end개발(UI 및 url 설계) * Docker container 구현 * Documentation |
| 강수련 | * Front-end개발 * 프로젝트 관련 영상 기획 |
| 우현웅 | * DB 설계 및 구현 * API서버 설계 및 개발 * 프로젝트 관련 영상 기획 * Documentation |
| 칼리드 | * 귀국 |

# **프로젝트 비용**

|  |  |
| --- | --- |
| **항목** | **예상치 (MD)** |
| 클라이언트 서버 | 150 |
| Api 서버 | 50 |
| 개발 환경 설정 | 20 |
| Core서버 | 50 |
| 합 | 270 |

# **개발 일정 및 자원 관리**

## 개발 일정

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **항목** | **세부내용** | **1월** | **2월** | **3월** | **4월** | **5월** | **6월** | **비고** |
| 요구사항분석 | 요구 분석 |  |  |  |  |  |  |  |
| 관련분야연구 | 주요 기술 연구 |  |  |  |  |  |  |  |
| 관련 시스템 분석 |  |  |  |  |  |  |  |
| 설계 | 시스템 설계 |  |  |  |  |  |  |  |
| 구현 | 코딩 및 모듈 테스트 |  |  |  |  |  |  |  |
| 테스트 | 시스템 테스트 |  |  |  |  |  |  |  |
| 피드백 후 반영 |  |  |  |  |  |  |  |  |
| 최종 테스트 |  |  |  |  |  |  |  |  |

## 일정별 주요 산출물

|  |  |  |  |
| --- | --- | --- | --- |
| 마일스톤 | 개요 | 시작일 | 종료일 |
| 계획서 작성 | 프로젝트 아이디어 선정  프로젝트 자료 수집  개발 환경 구축  **산출물** :   1. 프로젝트 수행 계획서 2. 프로젝트 기능 일람표 | 2020-01-02 | 2020-01-10 |
| 요구사항 분석 및 관련 분야 연구 | 요구사항 분석  관련 분야 연구  산출물 :   1. 요구사항 명세서 | 2020-01-10 | 2020-01-31 |
| 설계 완료 | 시스템 설계 완료  **산출물 :**   1. 시스템 설계 사양서 | 2020-02-03~ | 2020-02-16 |
| 1차 구현 완료 | 문제 제출, 대전, 리플레이 기능 구현 완료  **산출물** :  알고리즘 대저 웹페이지 및 1차 구현 소스 코드 | 2020-02-17 | 2020-03-31 |
| 테스트 | 구현한 기능 테스트  **산출물 :**   1. 피드백 사항들 | 2012-04-01 | 2012-04-23 |
| 중간 보고 | 문제 제출, 대전, 리플레이 기능 구현 완료  **산출물 :**   1. 프로젝트 1차 중간 보고서 2. 프로젝트 진도 점검표 3. 1차분 구현 소스 코드 | 2020-04-20 | 2020-04-23 |
| 구현 완료 | 문제 추가, 커뮤니티, 자신의 코드와 대전, github연동 기능 구현 완료  **산출물:**   1. 피드백이 반영되고 2차 구현이 완료된 소스 코드 | 2020-04-27 | 2020-05-22 |
| 테스트 | 시스템 통합 테스트  **산출물:**   1. 2차 피드백 사항 | 2020-05-25 | 2020-06-05 |
| 온라인 평가용 자료 작성 | 온라인 평가용 자료 작성  **산출물:**  온라인 평가용 자료 | 2020-05-25 | 2020-06-08 |
| 최종 보고 | 최종 보고  **산출물** :  최종결과 보고서 | 2020-06-01 | 2020-06-18 |

## 인력자원 투입계획

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 이름 | 개발항목 | 시작일 | 종료일 | 총개발일(MD) |
| 전원 | 요구사항 분석 및 관련 분야 연구 | 2020-01-02 | 2020-01-31 | 75 |
| 최명서, 우현웅 | Back-end 설계 | 2020-02-03 | 2020-02-16 | 20 |
| 최지욱, 박호준, 강수련 | Front-end 설계 | 2020-02-03 | 2020-02-16 | 30 |
| 최명서 | Core설계 및 구현 | 2020-02-17 | 2020-03-31 | 30 |
| 우현웅 | API 설계 및 구현 | 2020-02-17 | 2020-03-31 | 30 |
| 최지욱 | Front-end 구현 및 리플레이 구현 | 2020-02-17 | 2020-03-31 | 30 |
| 박호준, 강수련 | Front-end 구현 | 2020-02-17 | 2020-03-31 | 60 |
| 전원 | 시스템 테스트 | 2020-04-01 | 2020-04-23 | 50 |
| 전원 | 피드백을 반영한 수정 및 2차 구현 | 2020-04-27 | 2020-06-05 | 100 |

## 비 인적자원 투입계획

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 항목 | Provider | 시작일 | 종료일 | Required Options |
| 개발용 PC 3대 | 알고리즘 연구실 | 2020-01-02 | 2020-06-05 |  |
| 개발용 노트북 3대 | 본인 소유 | 2020-01-02 | 2020-06-05 |  |
| 서버용 PC 1대 | 알고리즘 연구실 | 2020-01-02 | 2020-06-05 |  |

# **참고 문헌**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 번호 | 종류 | 제목 | 출처 | 발행년도 | 저자 | 기타 |
| 1 | 웹페이지 | Coding game | www.codinggame.com |  |  |  |
| 2 | 웹페이지 | 우리는 알고리즘의 주인이 아니다 | http://www.snunews.com/news/articleView.html?idxno=20981 |  |  |  |
| 3 | 웹페이지 | 알고리즘 사이트 비교 및 추천 | <https://readystory.tistory.com/120> |  |  |  |
| 4 | 웹페이지 | Empire of Code | <https://checkio.org/> |  |  |  |
| 5 | 웹페이지 | Front-End프레임워크 선택하기 | [https://medium.com/aha-official/%EC%95%84%ED%95%98-%ED%94%84%EB%A1%A0%ED%8A%B8-%EA%B0%9C%EB%B0%9C%EA%B8%B0-0-angular-react-vue-js-%ED%94%84%EB%A0%88%EC%9E%84%EC%9B%8C%ED%81%AC-%EC%84%A0%ED%83%9D-f797392118d](https://medium.com/aha-official/%EC%95%84%ED%95%98-%ED%94%84%EB%A1%A0%ED%8A%B8-%EA%B0%9C%EB%B0%9C%EA%B8%B0-0-angular-react-vue-js-%ED%94%84%EB%A0%88%EC%9E%84%EC%9B%8C%ED%81%AC-%EC%84%A0%ED%83%9D-f797392118d0)0 |  |  |  |
| 6 | 웹페이지 | Material-Ui Docs | https://material-ui.com/ |  |  |  |