# 요리용 코딩 언어 개발

## 착수 보고서


첫눈에

202255661 박혜연

201924419 김도엽

201924577 정종현

# 목차

# 과제 배경 및 목표

## 1. 과제 배경

최근 다양한 분야에서 텍스트 기반 프로그래밍 언어가 활용되고 있지만, 이는 프로그래밍에 익숙하지 않은 일반 사용자, 특히 요리와 같은 실생활 도메인에서의 접근성을 크게 떨어뜨리는 요인이 되고 있다. 요리라는 행위는 순서, 도구, 시간 등의 요소가 복합적으로 작용하며, 이를 구조화된 형태로 표현하는 데 어려움이 존재한다. 기존의 요리 레시피는 컴퓨터가 해석하기 어렵다.

그러므로 이 프로젝트에서는 Scratch 와 같은 블록형 코딩 시스템을 도입하여, 요리 과정을 시각적 블록 형태로 구조화하고, 드래그 앤 드롭 방식으로 조작 가능하도록 하여 누구나 손쉽게 요리 과정을 코딩할 수 있는 언어를 설계하고자 한다. 텍스트가 아닌 시각 중심의 조작 방식은 비개발자도 쉽게 학습 가능하고, 요리 과정을 보다 명확하게 순서화하고 재사용 가능한 데이터로 전환할 수 있다는 점에서 큰 의의를 갖는다.

## 2. 과제 목표

이 프로젝트에서는 요리용 코딩언어를 Scratch 방식의 블록 인터페이스로 구현하여, 사용자가 시각적으로 각 요리 단계와 흐름을 쉽게 인식하고 구성할 수 있도록 돕는 도구를 개발하는 것을 목표로 한다. 각 블록은 재료, 조리 동작, 조리 양, 조리 시간 등으로 구성되며, 사용자는 이를 드래그 앤 드롭하여 요리 과정을 설계할 수 있다.

이러한 시스템을 통해 사용자는 요리법을 단순한 텍스트가 아닌 블록 기반의 구조화된 코드 형태로 직관적으로 표현하고, 조리 순서를 시각적으로 이해하고 조작할 수 있다. 또한 이 시스템은 레시피 자동화, 공유, 수정을 쉽게 만들어줄 뿐만 아니라 이후 디지털 조리 도우미나 스마트 주방 시스템과의 연계 가능성도 열어준다. 궁극적으로는 비전문가도 손쉽게 사용할 수 있는 요리용 코딩언어 생태계 구축을 지향한다.

# 과제 구성

## 1. 요구 사항 분석

A. 사용자 친화적 UI : 블록 기반 인터페이스를 활용하여 사용자가 드래그 앤 드롭 방식으로 요리과정을 직관적으로 설계할 수 있어야 한다.

B. 요리용 코딩언어 개발 : 블록 UI에서 입력된 레시피 정보를 JSON 형식으로 구조화하여 백엔드에서 파싱, 저장, 재사용 가능하도록 한다. 블록 조립 방식과 JSON 데이터의 규칙을 통해 요리 과정을 코드화하여 순차적으로 실행, 조건처리, 반복 처리 등이 가능하도록 설계한다.

C. 레시피 생성 및 저장 : 블록 기반으로 생성한 레시피를 저장, 수정, 불러오기 가능하도록 한다.

D. 레시피 개인화: 로그인 인증을 통해 유저마다 작성하는 레시피가 구분되어 레시피의 개인화를 보장한다.

## 2. 시스템 기능 구성

A. 프론트엔드

F1. **블록기반 UI 구현**: 시각적 접근성을 높이기 위해 블록 기반으로 설계 및 드래그 앤 드롭 형태로 블록을 끌고 와 순서를 구조화하여 나타내는 기능을 구현한다.

F2. **회원 가입 및 로그인 기능 구현**: 유저별 자신만의 레시피를 만들어 저장하기 위해 유저 구분을 위한 기능을 구현한다.

F3. **사용자 정보 수정 기능 구현**: 사용자의 정보를 수정할 수 있는 페이지를 만든다.

F4. **만든 레시피 저장 및 불러오기 기능 구현**: 만든 레시피를 저장하고 불러오기 기능을 구현한다.

F5. **만든 레시피 수정 및 삭제 기능 구현**: 만든 레시피에 대해 수정이 필요하거나 삭제하는 기능을 구현한다.

F6. **자동저장 기능 구현**: 레시피를 작성 중 데이터가 날아가는 것을 방지하고자 일정 주기(예: 5초, 10초)마다 DB에 반영한다.

F7. **블록 모형 설계**: 각 블록 항목마다 어떻게 연결되는지 구조화되기 쉽게 논리 및 순서에 모순이 없도록 블록모형을 구상한다.

F8. **블록 카테고리화**: 재료 및 동작 등 순서에 필요한 블록을 분류해서 해당 블록에서의 기능을 명확히 구현한다.

- 재료 블록: 재료 블록 생성시 사용자에게 요구되는 정보를 기입하게 한다. (ex. 고체, 액체 등), 재료 블록의 경우 '만들기' 기능과 '찾기' 기능을 지원한다.
- 동작 블록: '만들기', '찾기' 기능을 지원하지 않는다.

B. 백엔드

B1. **회원 가입 및 로그인 기능 구현:** 사용자별 계정을 구분할 수 있도록 회원가입 및 로그인 기능을 제공한다. 로그인 시에는 인증 토큰을 발급하여 사용자 세션을 관리한다.

B2. **사용자 관리 기능 구현:** 사용자의 기본 정보(예: 이름, 이메일, 비밀번호 등)를 등록, 수정, 삭제할 수 있는 관리 기능을 제공한다. 관리자 계정은 일반 사용자 계정과 구분하여 권한별로 관리할 수 있도록 한다.

B3. **레시피 관리 기능 구현:** 사용자가 생성한 레시피 데이터를 백엔드에서 저장, 수정, 삭제, 조회할 수 있도록 API를 설계한다. 프론트에서 전달된 JSON 형식의 블록 데이터를 파싱하여 레시피로 저장한다.

B4. **블록 데이터 처리 및 검증 기능 구현:** 프론트에서 전달받은 블록 데이터를 파싱하여 실행 가능한 구조로 해석하고, 순서 및 논리 오류를 검증한다. 레시피의 유효성을 검증하고 그 결과를 프론트로 반환한다.

B5. **자동저장 및 버전 관리 기능 구현:** 프론트에서 일정 주기로 전송되는 레시피 데이터를 백엔드에서 자동 저장하고, 이전 버전 데이터를 관리할 수 있는 기능을 제공한다. 필요 시 특정 버전으로 되돌릴 수 있도록 한다.

B6. **관리자 기능 구현:** 관리자 계정으로 전체 사용자 관리 기능을 제공한다.

B7. **데이터베이스 연동:** 사용자 정보, 레시피 데이터, 블록 데이터 등을 MongoDB에 연동하여 효율적으로 관리한다.

## 3. 제약 사항

A. 재료의 이름은 한글로만, 시간과 양은 숫자로만, 단위는 블럭에 지정되어 있는 걸 이용한다. 특수기호는 기입 불가하다.

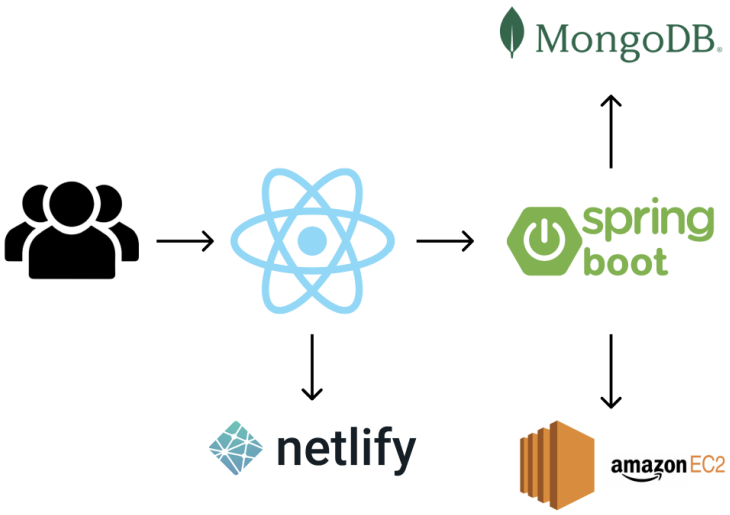
B. 모든 동작을 다 표현할 수 없으므로, 잘 쓰이는 동작들로만 구성한다.

C. 재료를 사용자가 추가하므로, 재료에 관한 오류 검증이 제한적이다.

## 4. 개발 도구

| 영역 | 선택한 기술 | 이유 |
|---|---|---|
| 프론트엔드 | React + Blockly | 시각적 블록 UI, 드래그 앤 드롭, 상태 관리에 강함 |
| 백엔드 | Java + Spring Boot | 강력한 엔터프라이즈 프레임워크, 보안·트랜잭션·데이터 처리에 강점 |
| 데이터베이스 | MongoDB | 유연한 문서형 DB, 빠른 개발과 Node.js/Java 모두 연동 가능 |
| 배포 (프론트) | Netlify | 정적 React 앱 배포에 최적화, 자동화 배포 지원 |
| 배포 (백엔드) | AWS EC2 | Spring Boot 서버 배포 가능, Java 환경 호환 |
| 협업, 문서화 | Notion, Velog, Figma, GIthub | 디자인 기획, 일정 관리, 보고서 협업용, 코드 관리 |

## 5. 개발 구성도



## 수행 계획

1. 역할 분담

| | | |
|---|---|---|
| 박혜연 | 백엔드 | 서버 개발, 배포 |
| 김도엽 | 백엔드 | 서버 개발, DB 연동 |
| 정종현 | 프론트엔드 | UI, 블록 시각화 |
| 공통 | 설계 | UI 설계, 기본 구성 설계 |

2. 개발 일정