성능과 안정성을 고려하여 아키텍쳐를 설계하고자 하는 정우진 개발자

교육사항
2022.07~2023.06 삼성청년 **SW**아카데미 수료
2016.03~2022.08 컴퓨터소프트웨어 학부 졸업

이메일 : **woojin1home@naver.com**
전화번호: **010-2030-6523**
Git-hub: https://github.com/gkgkgk78

목차

1 기술스택 목록

BE

- Spring Boot
- 비동기 통신
- Rabbit MQ
- Swagger
- Mysql
- Spring Security

협업 툴

- Gitlab
- Jira
- Notion
- MatterMost
- Webex

1

기술스택 상세

JAVA

- Backend 관련 API개발 및 로직 개발
- 객체지향의 특성을 이용하여 프로그램 개발
- MYBATIS를 활용하여 데이터베이스 연동 및 처리
- Spring Security 적용하여 인증 , 인가 처리
- 비동기 통신, RabbitMQ 적용하여 성능 개선

PYTHON

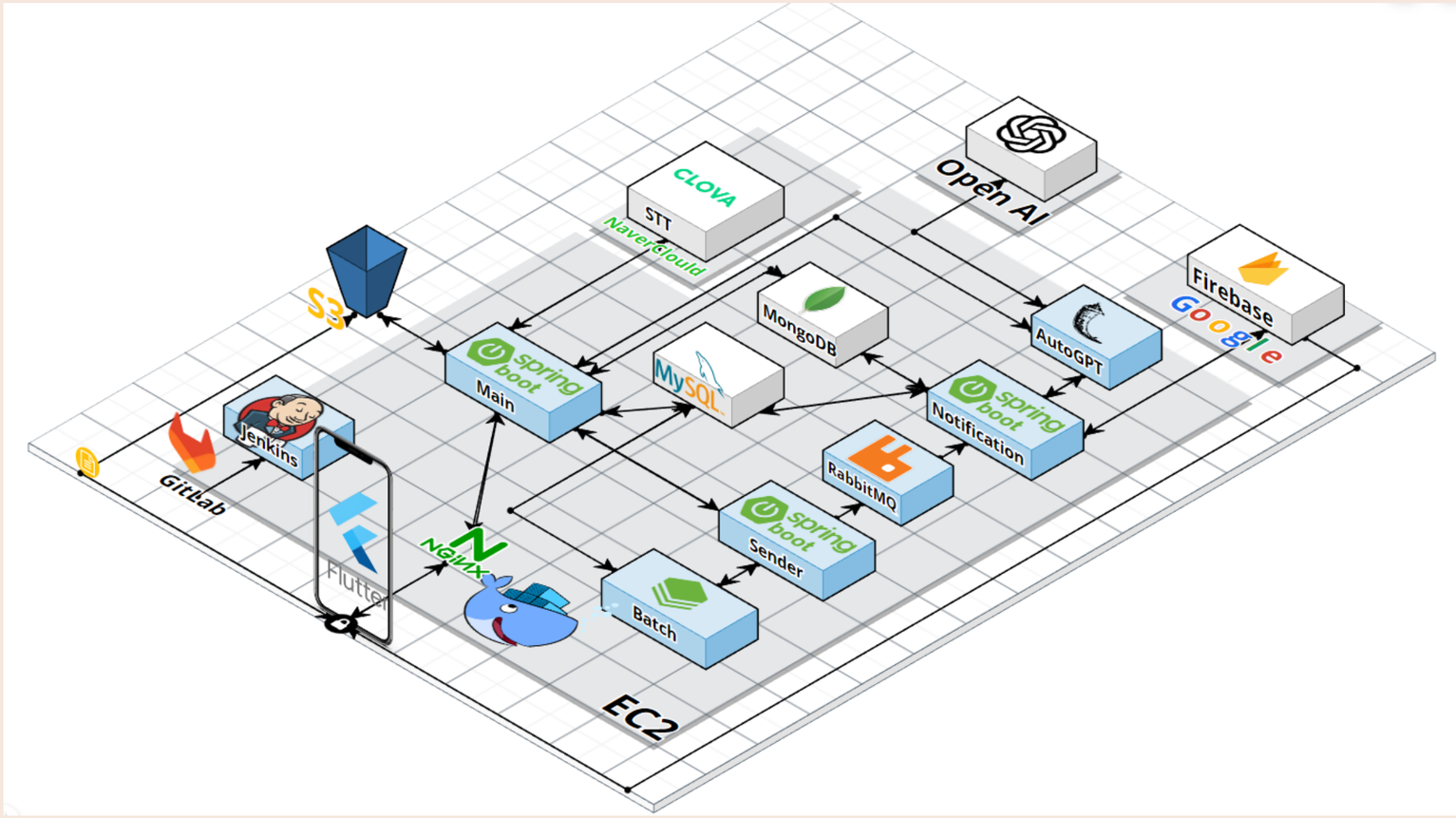
- Flask 서버 작성 및 API 개발
- Python 을 이용하여 기본 구현 및 알고리즘 구현

2-1

Moyeo

기간 : 2023.4.10 ~ 2023.5.26(6주)
인원 : 6인
깃주소: https://github.com/gkgkgk78/Moyeo.git

시스템 구성도



2-1

Moyeo

기간 : **2023.4.10 ~ 2023.5.26(6주)**
인원 : **6인**
깃주소: **https://github.com/gkgkgk78/Moyeo.git**

프로젝트 개요

- 스마트폰 어플을 이용하여 손쉽게 여행 을 기록하며 여행에 관련한 정보를 제공하여 주는 프로젝트

서비스 선정 이유

- 여행 기록을 손쉽게 기록하고 정리 할 수 있도록 하여 사용성 을 높이고자 하였습니다
- 손쉬운 동행 합류, 나가기 기능을 통하여 여러명의 사람들과 여행 기록을 공유 및 작성하도록 하였습니다

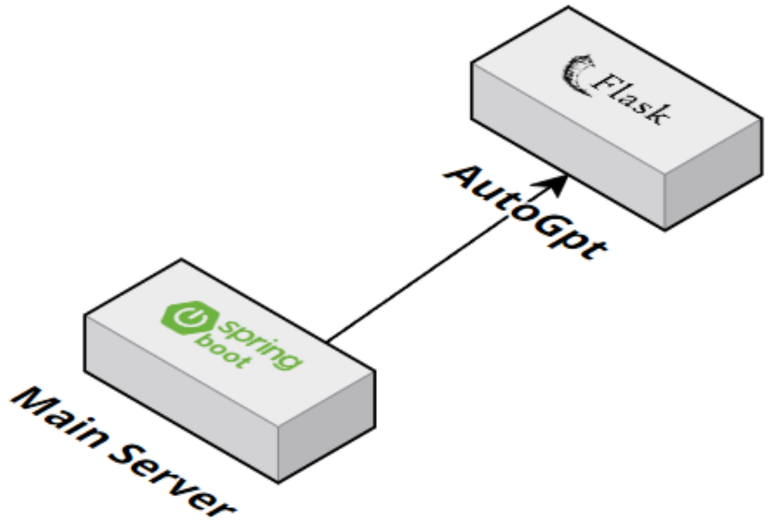
담당역할

- CLI기반인 AUTOGPT 를 FLASK의 API 형식으로 변환 및 CHATGPT 연동 작업
- 추가적으로 서버를 분리하여 비동기 NON-BLOCKING 형식으로 처리 작업 진행
- 푸시 알림 연동 작업 및 테스트 진행
- 안정적 서비스 제공 및 부하 분산을 위한 RABBIT MQ 적용

2-1

Moyeo

기간 : **2023.4.10 ~ 2023.5.26(6주)**
인원 : **6인**
깃주소: **https://github.com/gkgkgk78/Moyeo.git**



담당 기능 초기 구현

- 1.생성형 AI인 AUTO GPT는 CLI 기반 프로그램
- 2.프로그램 분석 후 FLASK 활용하여 API 화 진행
- 3.메인서버 관련 코드 작성

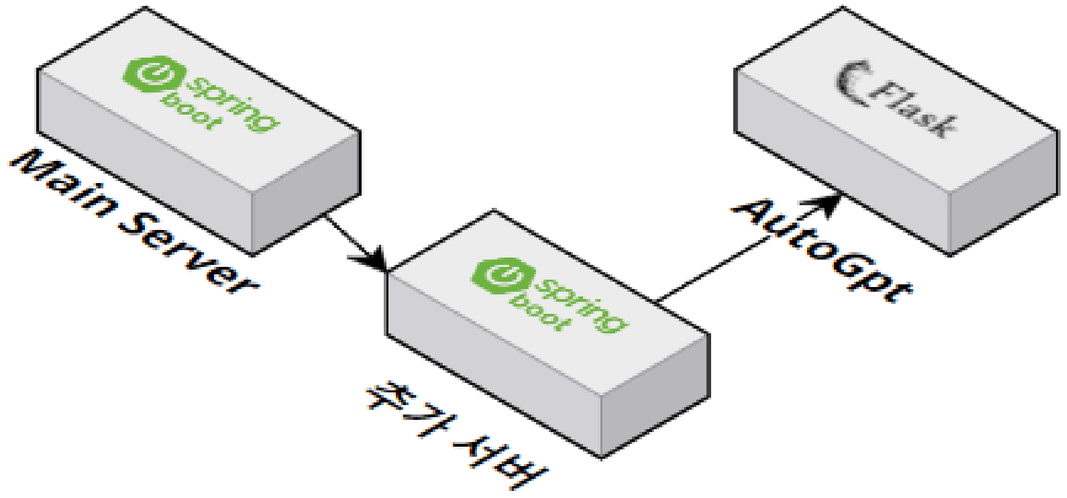
2-1

Moyeo

기간 : 2023.4.10 ~ 2023.5.26(6주)
인원 : 6인
깃주소: https://github.com/gkgkgk78/Moyeo.git

구현 시 문제 상황1

- 문제 상황
  - 구현한 기능은 여행 기록 등록시 + 챗봇 서비스 이용시 작동
    - 인터넷 검색 , 자가 추론을 하는 생성형 AI이다 보니 요청에 대한 응답 시간이 오래 걸렸습니다
    - 또한 메인 서버에서 동작 되다 보니 서버에 부담이 되었습니다
- 조치 사항
  - 1.챗봇 서비스와 관련된 사항은 Google FireBase를 활용한 푸시알람 처리
  - 2.비동기 방식을 적용하여 추가 Spring 컨테이너 에서 처리를 통해 메인 서버 부담 저하



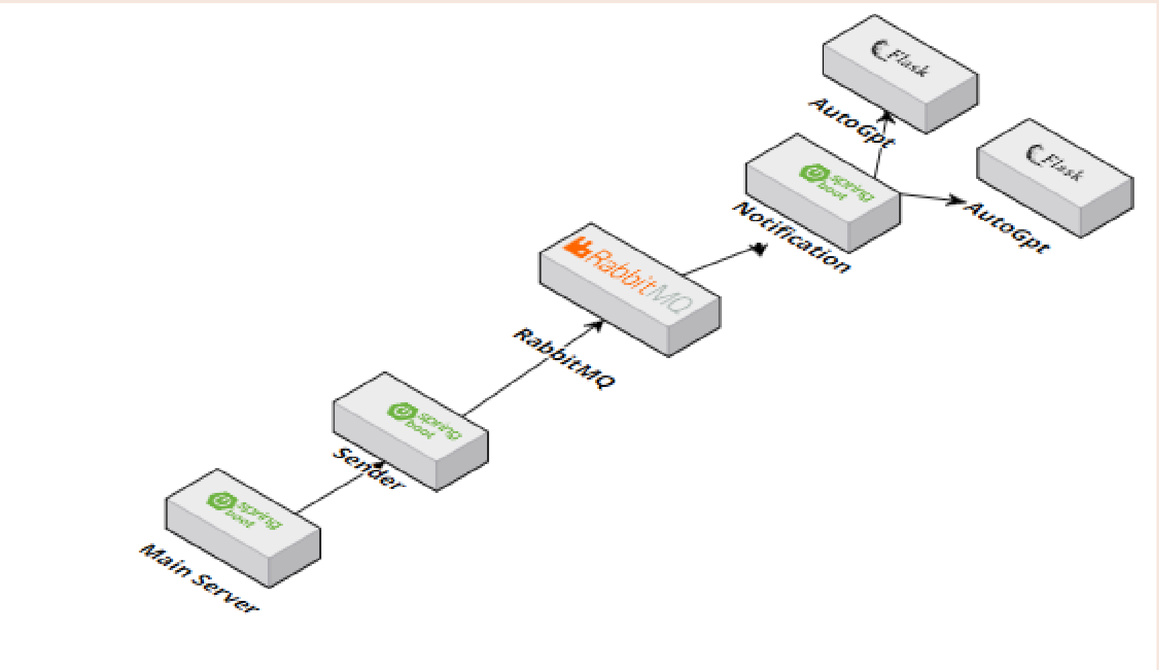
2-1

Moyeo

기간 : 2023.4.10 ~ 2023.5.26(6주)
인원 : 6인
깃주소: https://github.com/gkgkgk78/Moyeo.git

구현 시 문제 상황2

- 문제 상황
  - 핵심 서비스 이다 보니 많은 사용이 예상 되었습니다
  - 다른 서버에서 해당 로직을 처리 하여도 응답 시간 지연 문제 존재
- 조치 사항
  - 요청에 대한 처리 분할과 응답 속도를 개선 하고자 했습니다
    - 챗봇 서버 증설
    - 메시징 큐인 RABBITMQ 를 사용하여 ROUND ROBIN 방식을 적용해 부하 분산 처리



2-1

Moyeo

기간 : 2023.4.10 ~ 2023.5.26(6주)
인원 : 6인
깃주소: https://github.com/gkgkgk78/Moyeo.git

이슈 사항 요약

- 문제상황1 : 핵심 서비스 에서 최소 40초 최대 2분까지 걸리는 로직이 존재 하였습니다.
  - 해결방안: 오래 걸리는 로직을 독립적 병행 처리가 가능하다 판단 하였습니다.
  - 적용사항: 추가 Spring 서버를 두어 비동기 Non-Blocking 방식을 적용하여 분산 작업을 하고자 하였습니다.
- 문제상황2: 위 서비스에서 사용한 생성형 AI 인 AUTO-GPT 는 자가 추론을 하여 응답 시간이 오래 걸려 많은 사용자가 이용시 부하로 인해 성능상 문제가 될것으로 판단했습니다.
  - 해결방안: 메시징 큐 를 적용하여 성능과 안정성을 보장 할 수 있다고 판단 하였습니다.
  - 적용사항:AUTO-GPT 서버 증설 및 RABBIT-MQ를 적용하여 더 나은 사용자 경험을 제공 하고자 하였습니다.

프로젝트 성과 및 회고

- 팀 프로젝트의 경험을 바탕으로 팀장을 맡아 프로젝트를 진행 하며 일정관리를 통해 프로젝트 완수 하였습니다.
  - 프로젝트 참여율이 저조한 팀원과 개인 미팅 을 통해 더 나은 해결을 위해 노력 하였습니다.
  - 개인별 업무 수행도를 판단하여 팀원에게 알맞은 역할을 분배하여 팀의 성과를 더 향상 시키고자 하였습니다.
- 사용자의 사용경험 및 성능을 고려하며 서버 분리 및 안정적인 서비스 제공 위한 구조를 고려 하였습니다.

2-2

B+TREE

기간 : 2021.9.1~2021.9.27(3주)
인원 : 1인
깃 주소:https://github.com/gkgkgk78/B-Plus-tree.git

프로젝트 개요

- 데이터베이스의 인덱싱에 사용되는 알고리즘 중 하나인 B+Tree를 구현 하였습니다.

담당역할

- Tree 의 Insert, Delete, Range Search 를 구현하였으며 Tree의 Balance를 유지하며 구현 하였습니다.

프로젝트 성과 및 회고

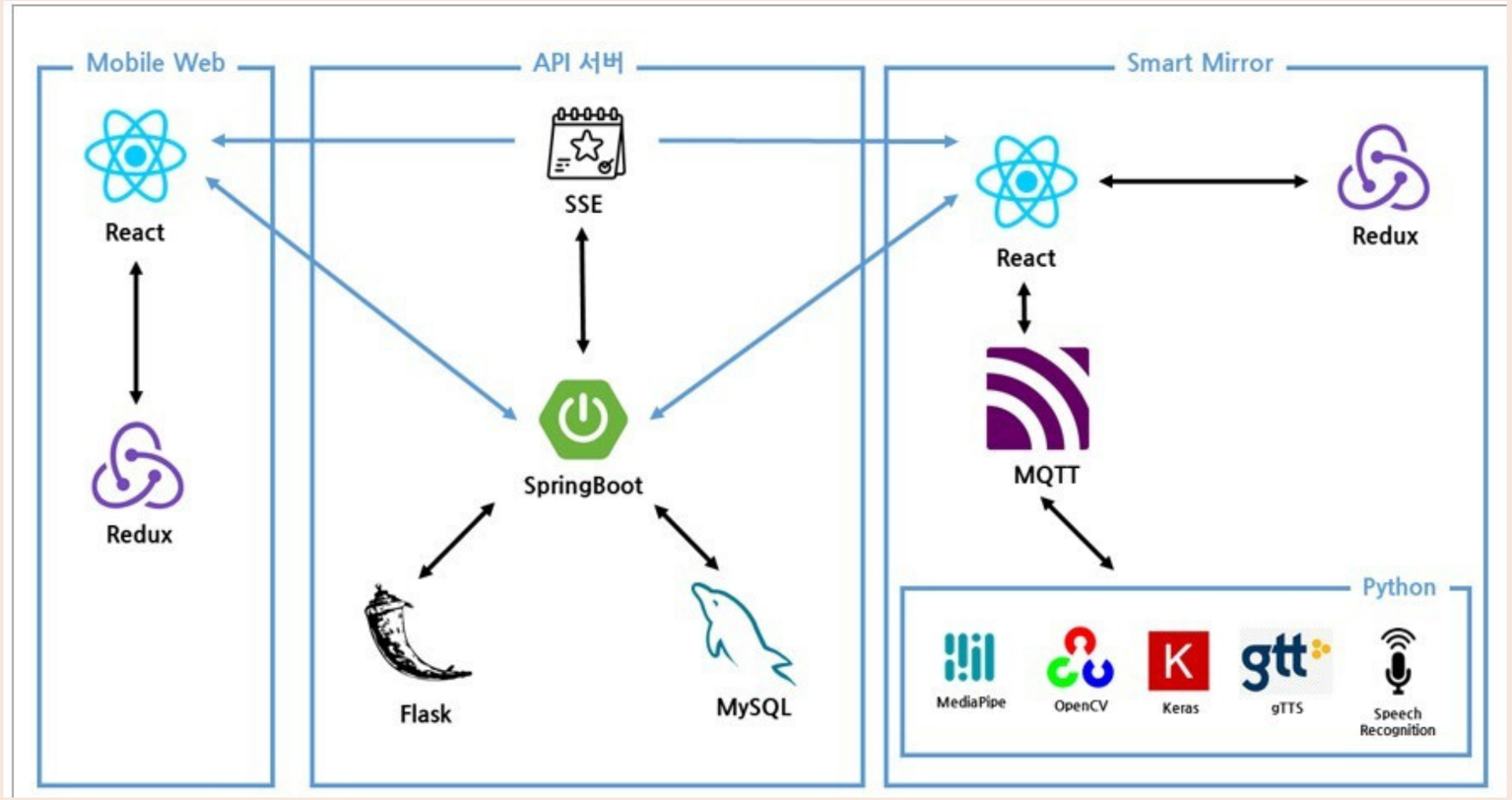
- B TREE 를 성공적으로 해내지 못했던 경험을 성공으로 바꾸고자 다시 공부를 하여, B+TREE 를 구현해 낼 수 있었습니다.
- 복잡한 로직을 가진 알고리즘을 구현하며 테스트 케이스 설계와 끈기있는 디버깅을 통해 구현능력을 기를 수 있었습니다.
- Balance를 가지는 Tree이기에 많은 데이터의 연산임에도 속도가 보장이 되어야 했습니다. 최대 100만개 까지의 Data를 1분 30초 내에 처리하도록 하였습니다

2-3

Famil_Link

기간 : 2023.01.03 ~2023.2.17(7주)
인원 : 6인
깃주소:https://github.com/gkgkgk78/Famil_link.git

시스템 구성도



2-3

Famil_Link

기간 : **2023.01.03 ~2023.2.17(7주)**
인원 : **6인**
깃주소:**https://github.com/gkgkgk78/Famil_link.git**

프로젝트 개요

- 현관앞에 위치하여 손쉽게 가족에게 영상편지를 보낼수 있는 스마트 미러 프로젝트.

서비스 선정 이유

- 바쁜 현대 사회에서 손쉽게 가족의 안부를 물으며 소통을 유도하기 위해 선정 하였습니다.

담당역할

- Spring Security 인증 인가 처리
- Spring File Upload & Download 처리
- Flask 서버에서 Teachable Machine 에서 학습한 얼굴 인증 처리 및 인증 인가 처리

2-3

Famil_Link

기간 : 2023.01.03 ~2023.2.17(7주)
인원 : 6인
깃주소:https://github.com/gkgkgk78/Famil_link.git

이슈 사항

- 문제상황: 얼굴 인증 정보 처리시 시간이 오래 걸리는 상황이 발생.
  - 해결방안: 얼굴 정보 불러오는 작업인 File I/O 작업을 최소화 하여 해결 가능하다 판단 하였습니다.
  - 적용사항: Python 의 Dictionary 자료구조를 이용하여 서버 최초 구동시 현재까지 등록된 얼굴 정보를 기록을 해두어 I/O작업을 최소한 으로 동작 하게 하였습니다. 추후 등록 되는 얼굴 정보도 Dictionary에 등록을 하는 API 를 만들어 발생 가능한 상황을 해결 하고자 하였습니다.

프로젝트 성과 및 회고

- 협업을 통해 프로젝트를 완성할 수 있었으며 처음 진행한 협업 프로젝트 이었지만 기간 내에 성공적 으로 완성을 해낼 수 있었습니다.
- 처음 사용해보는 스마트 미러 라는 기기에 서비스를 동작 시켜 보았습니다. 다양한 도메인에 대한 경험이 사용자에게 적절한 서비스를 제공해 줄 수 있음을 알게 되었습니다.

3

회고1-Index

기간 : **2023.07.06 ~2023.7.10**
인원 : **1인**
깃주소: **https://github.com/gkgkgk78/IndexTest**

회고 계기

- 지난 프로젝트들을 회고하며 보완점을 찾아 앞으로 더 나은 개발을 하고자 합니다.
- 문제점 :
  - NGRINDER를 적용해 핵심 API에 부하 테스트를 한 경험이 있습니다.
  - 성능을 개선하고자 캐싱 기능을 적용해 매번 DB에 접근 하지 않게 하여 성능을 향상 시키고자 했습니다.
- 해결방안
  - 회고하며 DB의 접근을 파악하는 것은 쿼리의 확인이 우선임을 깨닫고 로직을 작성한 쿼리의 시간을 측정하고자 했습니다.

3

회고1-Index

기간 : **2023.07.06 ~2023.7.10**
인원 : **1인**
깃주소: **https://github.com/gkgkgk78/IndexTest**

회고 후 조치

- 설계 과정
  - SELECT 연산 성능에 큰 영향을 끼치는 INDEX 를 적용해 QUERY EXPLAIN으로 테스트 하고자 했습니다.
- 실행 과정
  - 1. 로컬 DB에 각각 100만개의 데이터를 넣은 관계를 가진 테이블들을 만들었습니다
  - 2.각 WHERE, SELECT 문에서
    - INDEX가 걸린것과 걸리지 않은것의 측정 시간 비교
    - Clustered Index, Non-Clustered Index 를 생성해 QUERY EXPLAIN을 활용 하여 시간 측정을 했습니다.

3

회고2-단위 테스트

기간 : **2023.07.11 ~2023.7.20**
인원 : **1인**
깃주소:**https://github.com/gkgkgk78/TestCodePractice**

회고 계기 및 조치

- 문제점
  - 개발자는 기능 개발보다 안정적인 운영을 위해서 구현한 코드에서 에러를 찾아내는 작업에 더 많은 시간을 투자할 것으로 생각합니다.
  - 그전에는 직접 예외 케이스들을 넣어 테스트했지만, 앞으로 더욱 완성도 높은 코드를 위해 단위 테스트에 관해 공부하여 테스트 코드를 작성 했습니다.

- 조치
  - 단위 테스트 실행시 가짜 객체 주입을 통해 의존성을 떨어 뜨리고자 Mockito의 Mock 객체를 활용하여 단위 테스트 코드를 작성 했습니다.