Best Practice

세상을 바꿀 수 있는 아이디어를
실체화 하는 것을 꿈꾸는 개발자입니다.

학력

- 숭실대학교 IT대학 글로벌미디어학부(2016.03 ~)

이력

- 스타트업 겨울방학 인턴십 백엔드 개발자 (DINO STUDIO – 2021.12.29 ~ 2022.02)
- 숭실대학교 창업지원단 소속 온더브릿지 백엔드 개발자(2021. 04 ~ 2021.08)

동아리 및 커뮤니티

- 대학 연합 개발동아리 UMC 숭실대 지부 Server Member(2021.08 ~ 2021.11)
- Google Developers Student Club 숭실대 지부 Server/Cloud Member(2021.09~)
- 숭실대학교 개발동아리 유어슈 백엔드 팀원(2021.10~)

백엔드 개발자 오진호

기술 스택 : Java, Spring, Springboot, DataJPA

Email: ohjinho7@gmail.com
Github: https://github.com/ohjinhokor

DINO STUDIO
(스타트업 인턴십)

프로젝트 명 : 하이엔드(High-End)

- **프로젝트 기간**

  2021. 12. 29 ~ 진행 중

- **기술 스택**

  - Java, SpringBoot, DataJpa, MySql, Jwt
  - EC2, RDS, S3, CloudFront, CICD (Travis CI, Code Deploy, S3 이용)

- **프로젝트 개요**

  - 남녀의 만남을 주선하는 소개팅 어플입니다.
  - 특정 조건이 충족하는 사람만 서비스를 이용할 수 있습니다.

- **나의 구현 내용**

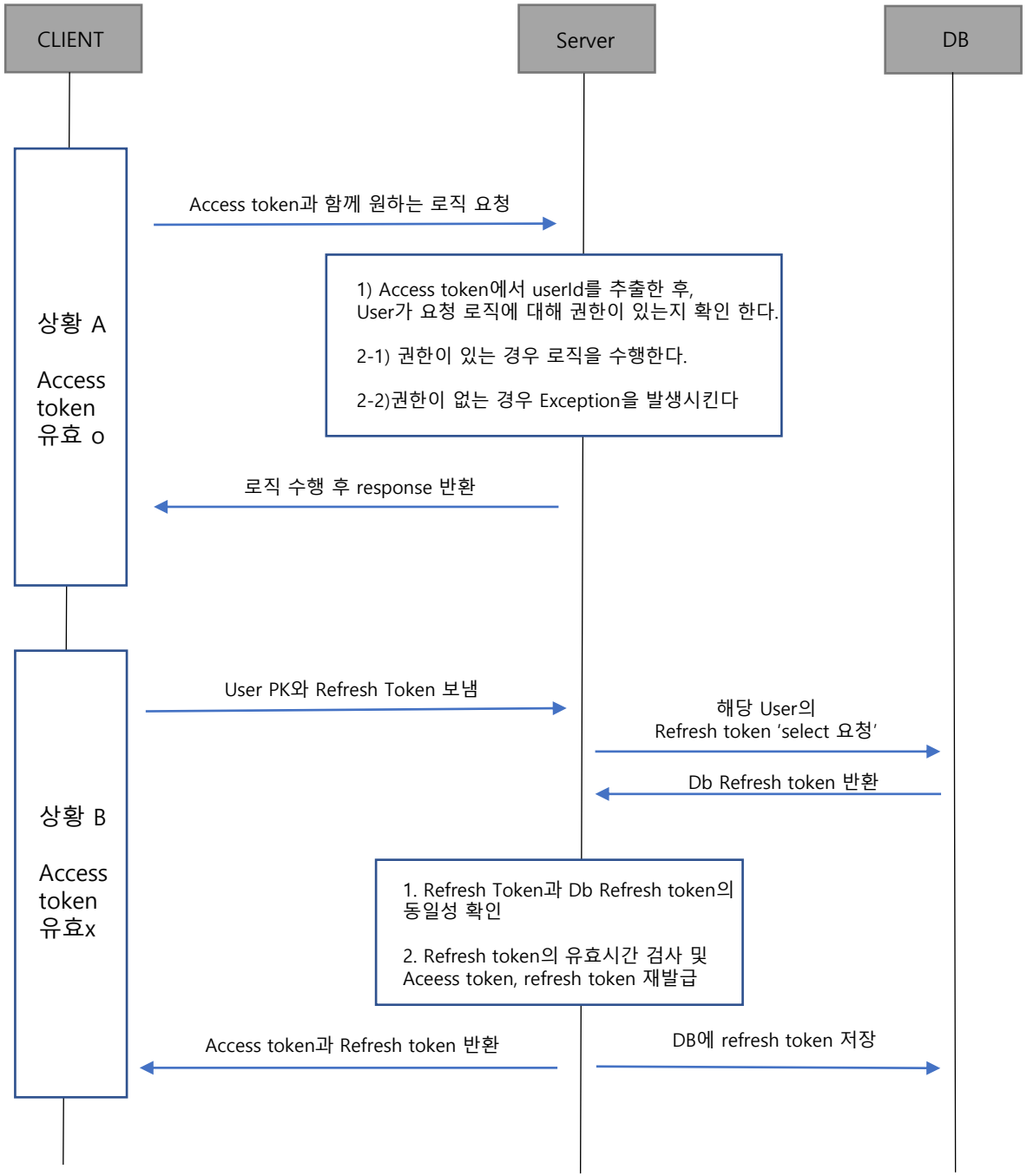
  ⊙ 서버 API 구현 및 핵심 비즈니스 로직 작성

  ⊙ AWS Cloud 서비스를 이용한 서버 구축 - EC2, RDS, S3, CloudFront

  ⊙ 스웨거를 이용한 API 문서 자동화

  ⊙ 프로젝트의 유일한 백엔드 개발자로 참여

  ⊙ 데이터베이스 설계 –> 백엔드 API 구현 –> 배포(CICD)의 모든 과정을 진행



ㅇ Interceptor, JWT를 사용한 보안

- 개인 정보가 많이 사용되는 서비스였기 때문에 권한이 필요한 모든 요청마다 token에서 user의 pk를 추출하여 요청 로직에 대한 권한을 확인하였습니다.

1. Controller로 들어가기 전 Interceptor에서 Token의 유효시간을 확인하는 작업을 진행합니다.
2. Token의 시간이 유효하다면, Interceptor에서 userId를 추출한 후 Controller로 userId 값을 넘겨줍니다.
3. 넘겨 받은 userId를 이용하여 User가 요청 로직에 대해 권한이 있는지 확인합니다. 예를 들어 특정 회원의 글을 조회하는 로직을 요청한다면 그 글이 해당 User에게 속해 있는 글인지 확인하는 절차를 거칩니다.
4. Refresh Token을 사용하여 보안을 높이고, 자동 로그인이 가능하도록 하였습니다.
5. 로그아웃을 하면 Refresh Token을 무효화 시킵니다.
6. Refresh Token의 저장위치는 DB로 하였습니다.

○ 아는 사람 피하기 기능

⊙ 기능에 필요한 테이블

1. User Table(User의 'Avoid Acquaintance'가 True이면 아는 사람 피하기 기능을 사용하는 것임)

2. AcquaintancePhoneNumber Table
 - User Table과 N : 1의 관계로 매핑된 테이블입니다. User의 핸드폰에 저장된 모든 연락처를 저장합니다.

3. AcquaintanceRelation Table (두 이용자가 아는 사이임을 나타내는 테이블)
 - User 대 User 의 N:N 관계를 풀어낸 테이블입니다. 즉 FK로 두 유저의 PK를 갖습니다.

⊙ 상황 1 – 아는 사람 피하기 기능 On

1) 신규 유저의 핸드폰에 저장된 모든 번호를 받아옵니다. 이후 AcquaintancePhoneNumber 테이블에 저장합니다.

2) 해당 번호로 가입한 기존 유저를 리스트로 받아옵니다.

3) 기존 유저들과 신규 유저가 아는 사이임을 AcquaintanceRelation 테이블에 저장합니다.

```
userRepository.findByPhoneNumber(phoneNumber).ifPresent(targetUser ->
{registerNewAcquaintance(user, targetUser);});
```

4) 이미 아는 사이로 등록된 관계인 경우 AcquaintanceRelation 테이블에 데이터를 추가등록하지 않습니다.

⊙ 상황 2 - 새로운 유저 가입

1) 새로운 유저 가입시 신규 유저의 전화번호를 통해 기존 유저와 아는 사이인지 확인한 후 리스트로 받아옴(AcquaintancePhoneNumble 테이블 이용)

```
"select a from AcquaintancePhoneNumber a where a.user =:user
and a.acquaintancePhoneNumber =:phoneNumber"
```

2) AcquaintanceRelation 테이블에 리스트에 있는 기존 유저와 신규유저가 아는 사이임을 등록합니다.

⊙ 상황 3 – 아는 사람 피하기 기능 Off

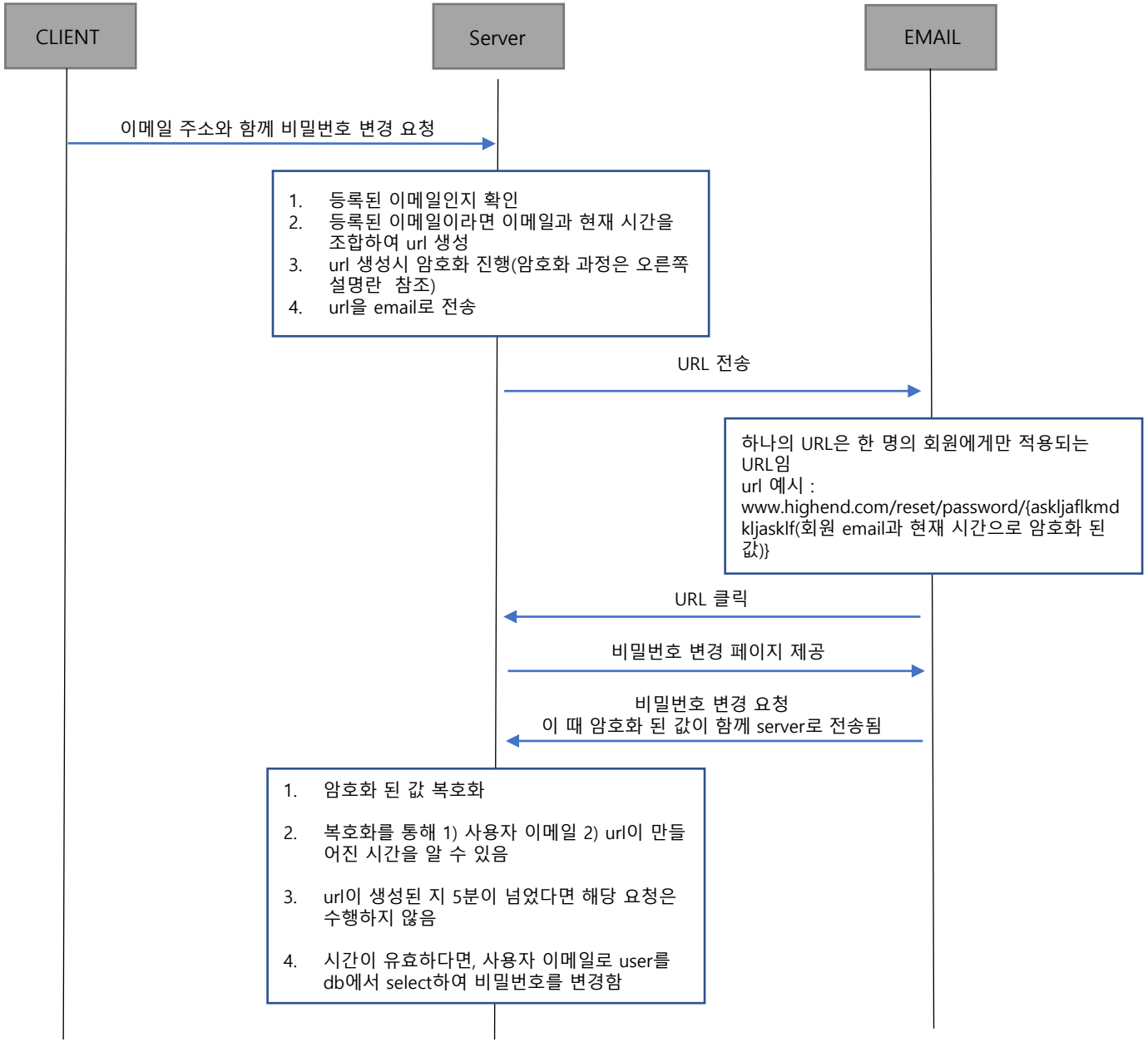
1) AcquaintancePhoneNumber 테이블에서 User와 연결되어있는 모든 Row를 삭제합니다.

2) 주의할 점 : 'AcquaintanceRelation' 테이블은 'AcquaintancePhoneNumber' 테이블과 달리 User와 연결되어 있는 모든 Row를 지우면 안됩니다.

 - UserA와 아는 사이라고 등록되어있는 UserB가 아는 사람 피하기 기능을 사용하고 있다면 Row을 삭제하면 안됩니다.

 - UserA와 아는 사이라고 등록되어있는 UserB가 아는 사람 피하기 기능을 사용하지 않는다면 해당 Row를 삭제할 수 있습니다.

```
acquaintanceRelationRepository
        .findMyAcquaintanceList(user)
        .stream()
        .filter(ac -> userRepository.findById(ac.getUserId()).isPresent())
        .filter(ac -> !ac.getAvoidingAcquaintance())
        .forEach(ac ->
acquaintanceRelationRepository.deleteAcquaintanceRelation(userRepository.findById(userId).orElseThrow(
()-> new NoSuchElementException("존재하지 않는 회원 pk입니다")),ac));
```



ㅇ 암호화

⊙ 암호화 방법

- 자바의 Cipher 클래스를 사용하였습니다.
- AES/CBC/PKCS5Padding

⊙ 암호화 이용 과정

1. 현재 시간과 User의 Email을 암호화합니다

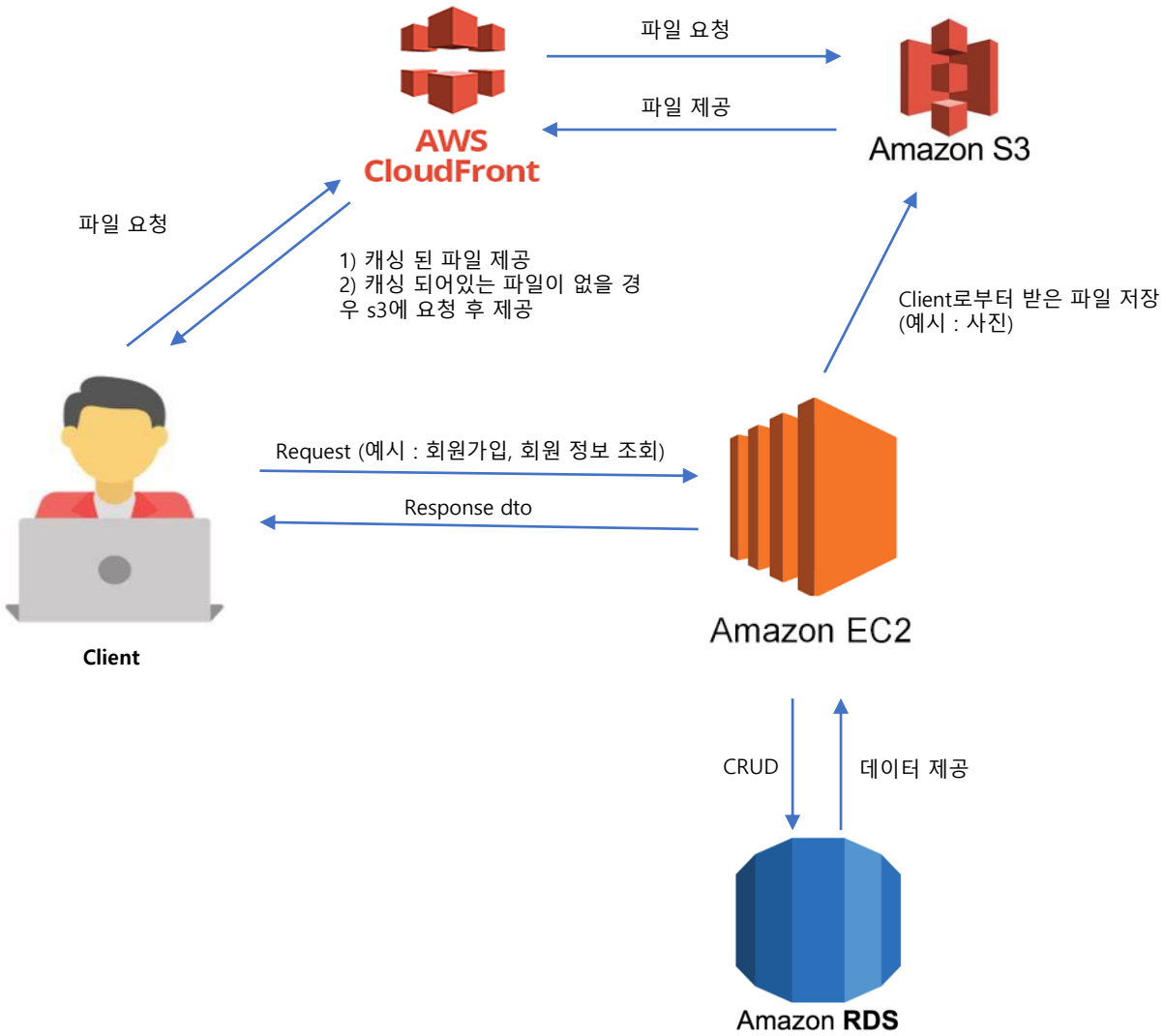
2. 암호화 된 값과 함께 url을 만든 후 User의 email로 전송합니다.

3. Client로부터 암호화 된 값과 함께 비밀번호 변경을 요청받습니다.

4. 암호화 된 값을 복호화 하여 url이 만들어진 시간과 Email을 얻어냅니다.

5. 유효시간을 확인 한 후 해당 User의 비밀번호를 변경합니다.

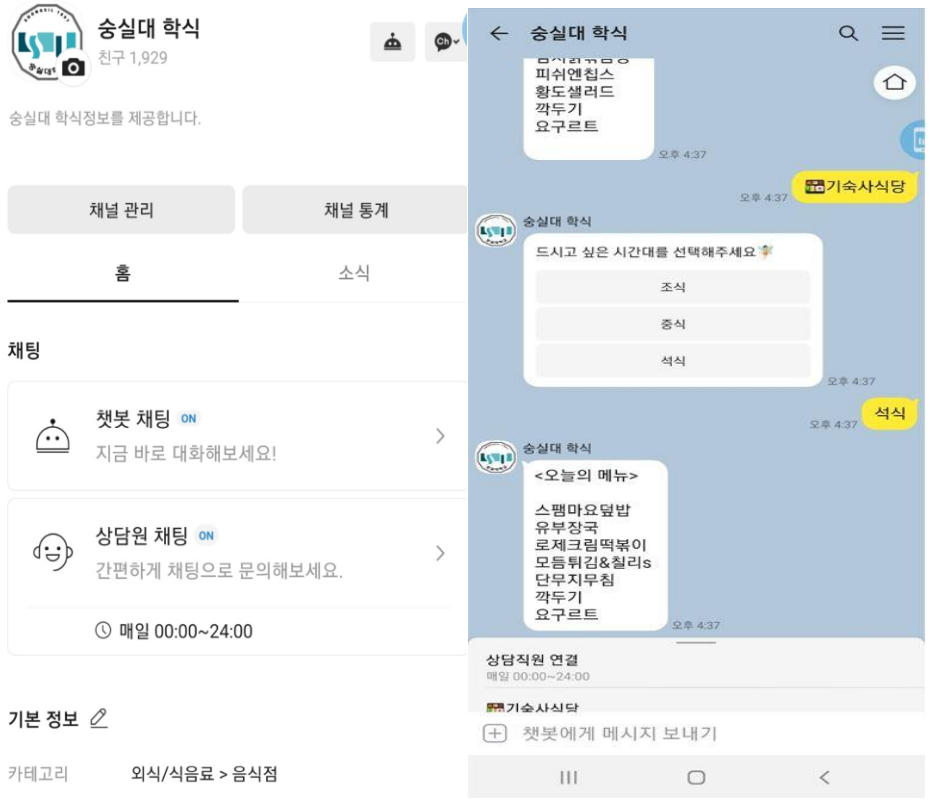
ㅇ 서버 구조도


파일 요청
파일 제공
AWS
CloudFront
Amazon S3
파일 요청
1) 캐싱 된 파일 제공
2) 캐싱 되어있는 파일이 없을 경우 s3에 요청 후 제공
Client로부터 받은 파일 저장
(예시 : 사진)
Request (예시 : 회원가입, 회원 정보 조회)
Response dto
Client
Amazon EC2
CRUD
데이터 제공
Amazon RDS

Side Project (소규모 프로젝트)

프로젝트 명 : 숭실대학교 학식 알림이

약 2000명의 사용자를 보유하고 있는 학식 알림이 서비스입니다.



프로젝트 기간

2021. 11 ~ 2021. 12

기술 스택

- Java, SpringBoot, DataJpa, Jsoup라이브러리, 카카오 API

- RDS, EC2

프로젝트 개요

1. 웹 크롤링을 통해 학식메뉴를 보여주는 카카오톡 챗봇입니다.

2. 자동화를 통해 성능을 개선하였습니다.

3. 서비스 기획 ~ 배포까지의 모든 프로세스에 참여하였습니다.

나의 구현 내용

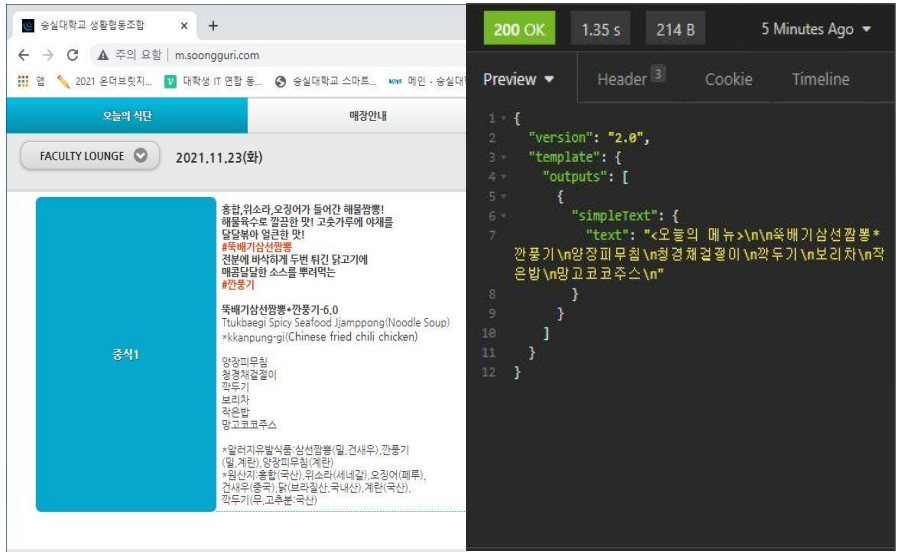
⊙ Jsoup 라이브러리를 이용한 웹 크롤링

⊙ 자동화를 통한 성능 개선

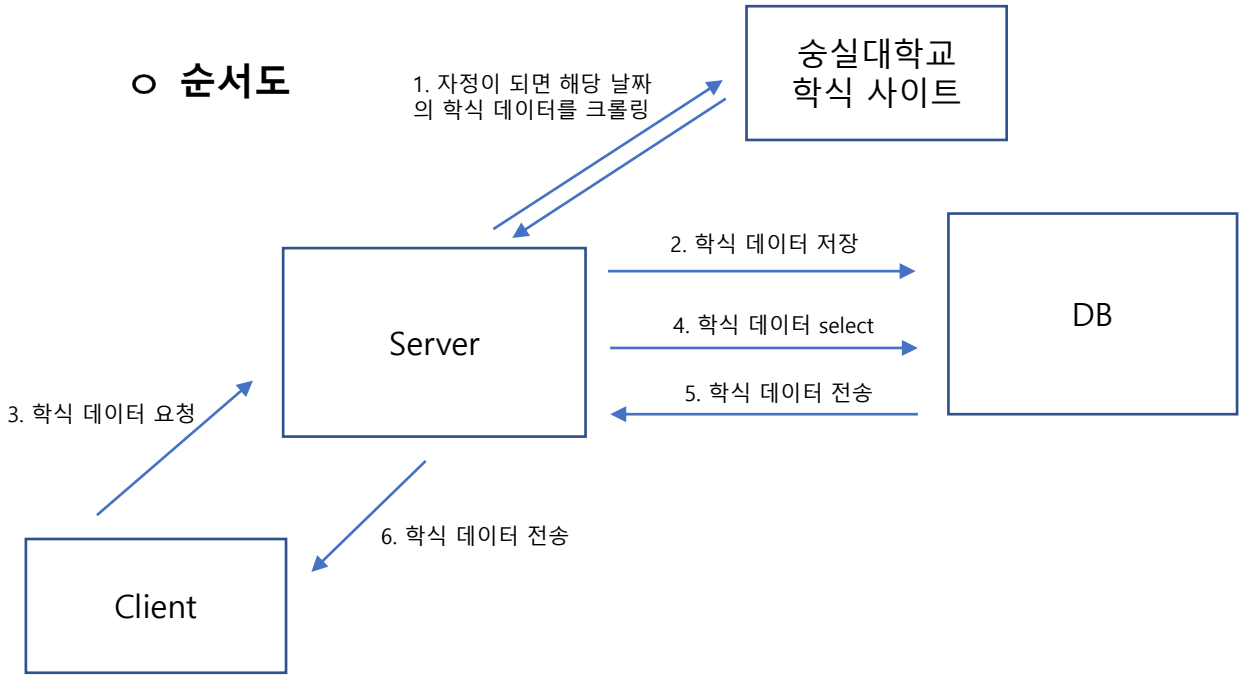
⊙ 카카오 오픈 빌더 API 활용

깃허브

URL : https://github.com/ohjinhokor/SoongsilHaksik



ㅇ 순서도



ㅇ Jsoup Library를 통한 웹 크롤링

학식 데이터를 추출한 후, Json형식의 response를 제공하는 API입니다.

1. URL을 통해 해당 사이트에 접근합니다.
2. 추출하고자 하는 데이터를 html코드에 맞게 설정합니다.
3. 데이터를 추출한 후 카카오 챗봇 API 형식에 맞는 'JSON Response'를 반환합니다.

성능 개선을 위한 Refactoring

문제가 되었던 부분: 요청이 들어올 때마다 크롤링을 하기 때문에 속도가 느림

<해결 방안>

1. 위의 문제를 해결하기 위해 @Scheduled 어노테이션을 사용하여 자동화 기능 추가
2. 자정이 지나면 자동으로 그 날의 학식 데이터를 데이터 베이스에 저장.
3. 이후 학식 정보를 요청하는 Request가 들어오면 데이터 베이스에 저장해 놓은 데이터를 반환 함

<리팩토링 결과> 수행 시간이 1561ms -> 213ms로 줄어드는 성능 향상을 보였습니다