



백엔드 개발자 오진호

Best Practice

세상을 바꿀 수 있는 아이디어를
실체화 하는 것을 꿈꾸는 개발자입니다.

송실대학교 IT대학 글로벌미디어학부(2016.03 ~)

송실대학교 HCI연구실 백엔드 개발자(2021.03 ~)

송실대학교 창업지원단 소속 온더브릿지 백엔드 개발자(2021. 04 ~ 2021.08)

대학 연합 개발동아리 UMC 송실대 지부 Server Member(2021.08 ~ 2021.11)

Google Developers Student Club 송실대 지부 Server/Cloud Member(2021.09~)

송실대학교 개발동아리 유어슈 백엔드 팀원(2021.10~)

기술 스택 : Java, Spring, Springboot, MVC, JPA

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

숭실대학교 창업지원단

On the Bridge

프로젝트 명 : 잇츠로드



○ 프로젝트 기간

2021. 04 ~ 2021. 08

○ 기술 스택

TypeScript, NestJs, Typeorm, MariaDb

○ 프로젝트 개요

음식점 모바일 오더 서비스 기능입니다.
손님이 음식점에 입장하여 QR코드를 통해 음식을 주문하면 가게 사장님이 카운터에서 주문한 음식을 바로 확인할 수 있는 서비스입니다.

상세 설명 : <https://maze-warlock-8d1.notion.site/965c208d24004a28a3e50bc30fe22d7c>

○ 나의 구현 내용

⦿ 음식점 등록 및 메뉴판 API 작성

⦿ 로그인 기능 구현

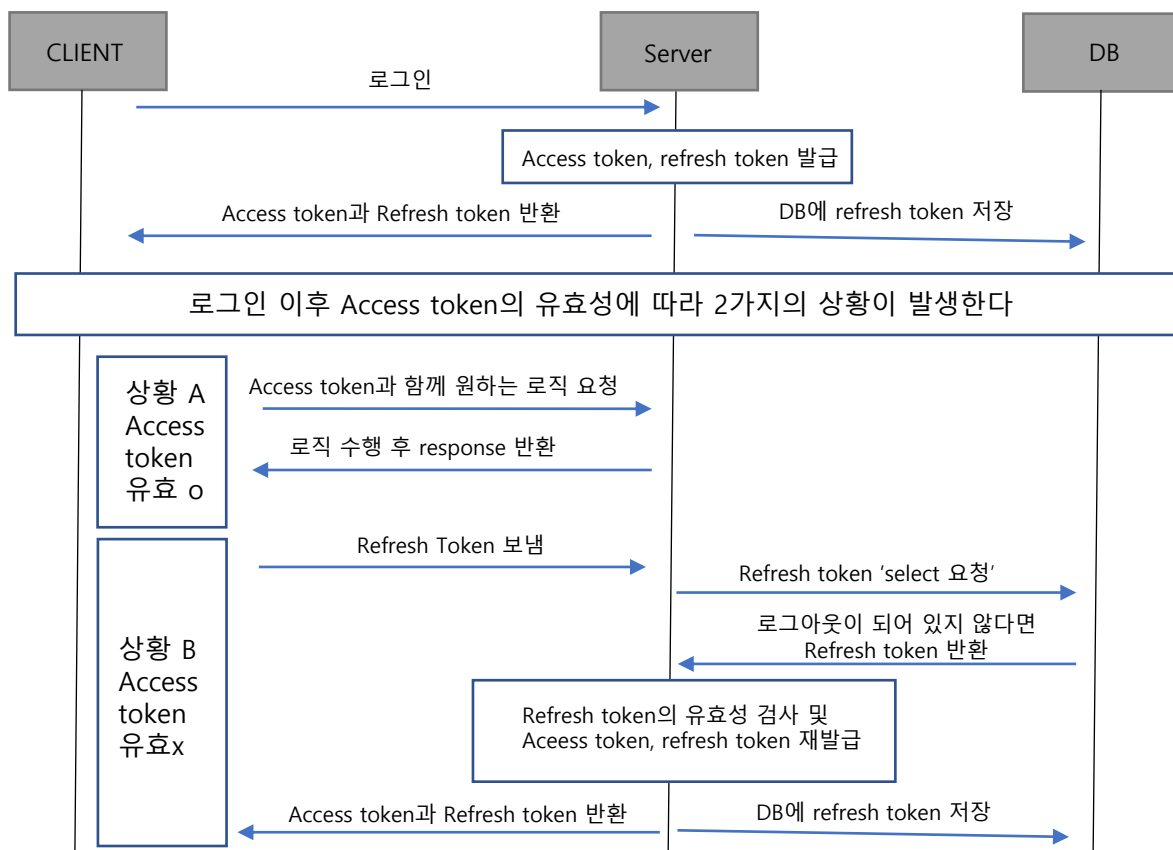
⦿ Git Fork, Git-Flow 적용



○ Validation

Http Header와 미들웨어를 사용하여 유효성 검사

1. HTTP header를 사용하여 클라이언트로부터 store_id와 JWT(토큰)을 받아옵니다.
2. JWT(토큰)에서 user_id를 추출하여 해당 user가 header에 들어있는 store_id를 pk로 갖는 store에 접근 권한이 있는지 확인합니다.
3. 권한이 있다면 user는 해당 음식점의 메뉴 삭제, 옵션 변경 등 음식점의 세부사항을 변경할 수 있습니다.
4. 권한이 없다면 해당 로직을 이용할 수 없습니다.



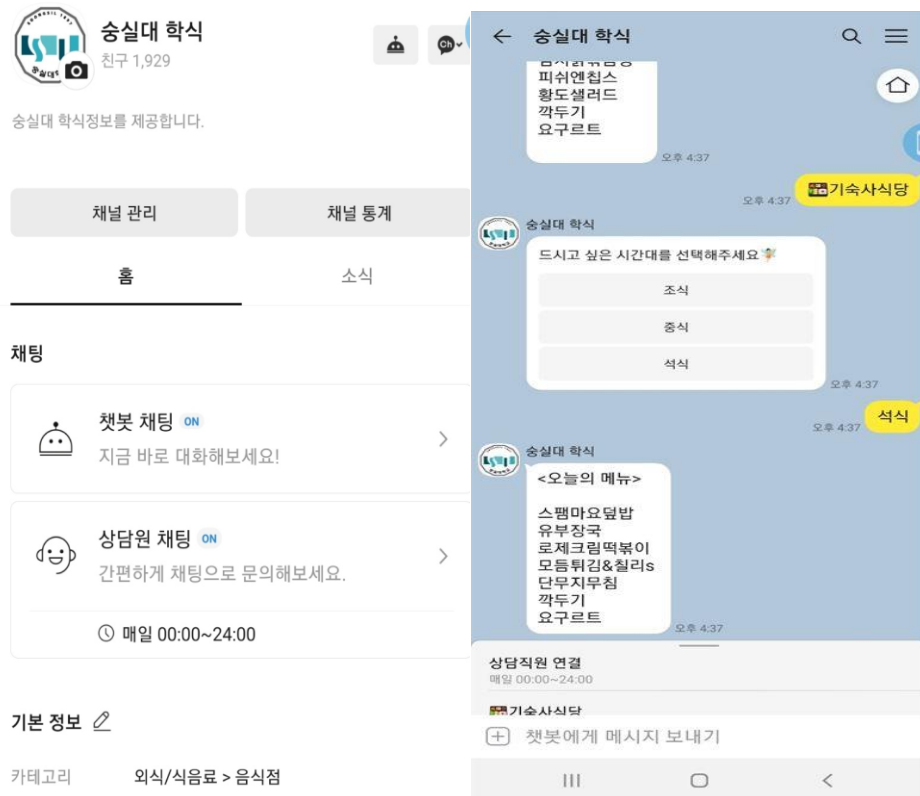
○ JWT 인증 사용

1. Refresh Token의 저장위치는 DB로 하였습니다
2. Response 값으로 Access token과 Refresh token을 반환합니다.
3. Refresh token의 유효성 검사는 토큰의 유효시간을 확인합니다.
4. 로그아웃을 하면 Refresh Token을 무효화 시킵니다.

Personal Project

프로젝트 명 : 송실대학교 학식 알림이

약 1900명의 사용자를 보유하고 있는 학식 알림이 서비스입니다.



프로젝트 기간

2021. 11 ~ 진행 중

기술 스택

Java, SpringBoot, Jsoup, 카카오 API

프로젝트 개요

1. 웹 크롤링을 통해 학식메뉴를 보여주는 카카오톡 챗봇입니다.
2. 자동화를 통해 성능을 개선하였습니다.
3. 서비스 기획 ~ 배포까지의 모든 프로세스에 참여하였습니다.

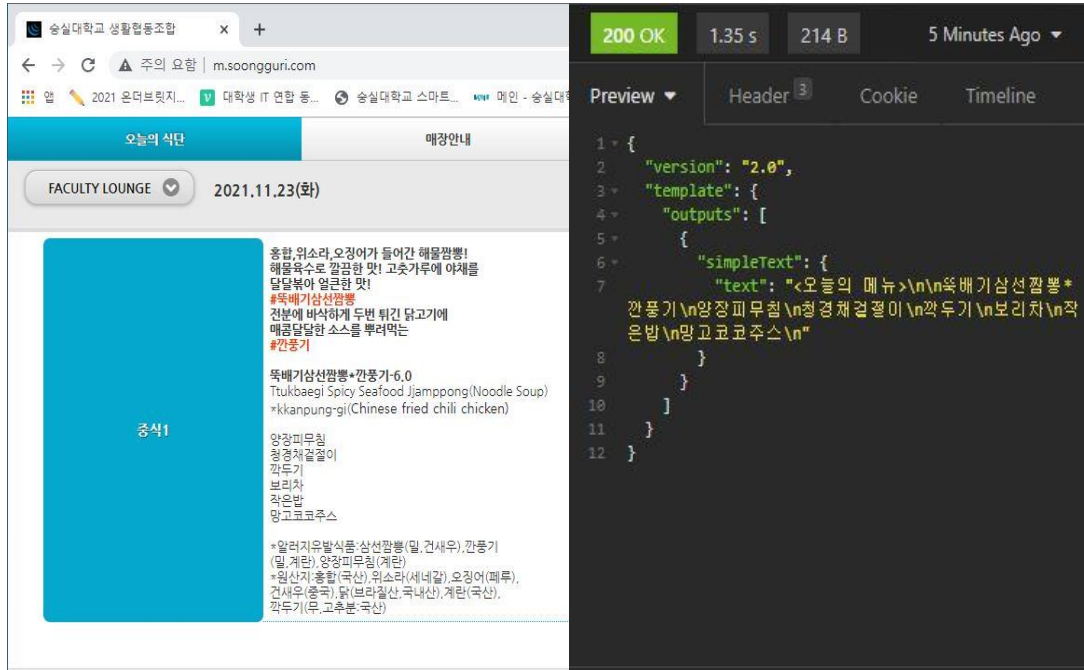
- 커피를 미리 결제하고 받아 갈 수 있는 원거리 결제 시스템을 개발 중에 있습니다.

나의 구현 내용

☞ Jsoup 라이브러리를 이용한 웹 크롤링

☞ 자동화를 통한 성능 개선

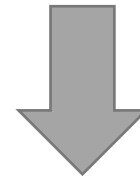
☞ 카카오 오픈 빌더 API 활용



○ 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

○ @Scheduled 어노테이션을 통한 자동화

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

수업 프로젝트

프로젝트 명 : 세상 소리(저시력자와 시각 장애인을 위한 뉴스 제공 서비스)



○ 프로젝트 기간

2021. 09 ~ 2021.12

○ 기술 스택

Java, SpringBoot, Jsoup, TTS 라이브러리, JsonObject 라이브러리

○ 프로젝트 개요

저시력자와 시각 장애인을 위해 뉴스를 제공하는 서비스입니다.

1. 뉴스를 음성으로 출력해주는 기능을 구현하였습니다.
2. 뉴스 내용을 점자로 변환한 후 Json형태의 파일로 제공합니다.

○ 나의 구현 내용

⊙ Jsoup 라이브러리를 이용한 웹 크롤링






⊙ TTS 라이브러리를 통해 뉴스 내용을 음성 파일로 변경 및 저장

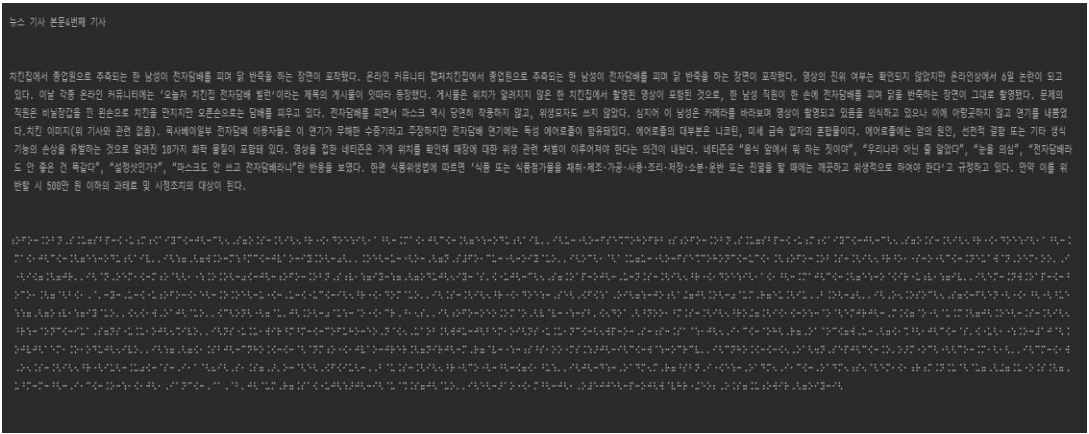
⊙ 뉴스 본문을 점자로 변환한 후 Json형식으로 저장


```
@Scheduled(cron = "0 0/59 * * * *")
public String socialNews(){
```

```
String path = "src/main/resources/static/social/social" + i + ".mp3";
new makeSound( apiKey: "9ccdfcd870e24163a3478032a26e2087",
    path,
    voiceNewsTopicWritings).makeTTS();
```

» NewsForDeafBlind » NewForBlind » src » main » resources » static » social

이름	#	제목	참여 음악가
 social0			
 social1			
 social2			
 social3			
 social4			



○ 웹 크롤링 및 음성 파일로 변환

뉴스 내용을 음성 파일로 저장하는 기능입니다.

1. @Scheduled 어노테이션을 통해 59분마다 뉴스 내용을 갱신합니다.
2. Jsoup 라이브러리를 통해 뉴스 데이터를 가져옵니다.
3. TTS 라이브러리를 통해 해당 데이터를 음성 파일로 변경 한 후 저장합니다.

○ 점자 변환 및 Json 형식의 파일로 저장

점자 프린팅을 위해 뉴스 내용을 점자로 변환한 후 그 내용을 Json형태의 파일로 제공합니다.

1. Jsoup 라이브러리를 통해 뉴스 데이터를 가져옵니다.
2. 모든 글자를 자,모음으로 분리합니다. ex) 진 -> 'ㅈ','ㅣ','ㄴ' 가 -> 'ㄱ','ㅣ'
3. 분리 된 글자들을 연결리스트에 저장합니다.
4. 연결리스트 안의 모든 글자들을 점자로 변환합니다(이 부분은 하드코딩으로 구현하였습니다.)
5. JsonObject 라이브러리를 이용해 정해진 형식의 Json파일로 저장합니다.
6. Json 파일을 통해 점자 데이터를 포함한 한글로 된 기사 제목과 내용, 기사 URL 등을 제공합니다.

○ 리팩토링 예정

세상 소리 프로젝트는 수업 시간에 처음 만난 학우분들과 진행한 프로젝트입니다.

객체지향에 익숙하지 않은 학우들도 있었기에 **절차 지향적**으로 코드가 작성되었습니다.

12월 이후 **객체 지향적인 아키텍처**로 리팩토링할 계획입니다.

개인 공부

Junit

```
class MemoryMemberRepositoryTest {
    MemoryMemberRepository repository = new MemoryMemberRepository();

    @AfterEach
    public void afterEach(){
        repository.cleanstore();
    }

    @Test
    public void save(){
        Member member = new Member();
        member.setName("example_name");

        repository.save(member);
        Member result = repository.findById(member.getId()).get();

        Assertions.assertEquals(member, result);
        assertThat(member).isEqualTo(result);
    }

    @Test
    public void findByName(){
        Member member1 = new Member();
        member1.setName("example_name2");
        repository.save(member1);

        Member member2 = new Member();
        member2.setName("example_name5");
        repository.save(member2);

        Member result = repository.findById(member1.getId()).get();

        assertThat(result).isEqualTo(member1);
    }

    @Test
    public void findAll(){
        Member member1 = new Member();
        member1.setName("spring1");
        repository.save(member1);

        Member member2 = new Member();
        member2.setName("spring2");
        repository.save(member2);

        List<Member> result = repository.findAll();
        assertThat(result.size(),isEqualTo(2));
    }
}
```

JPA

```
@Repository
@RequiredArgsConstructor
public class JpaMemberRepository implements MemberRepository {

    private final EntityManager em;

    @Override
    public Response createMemoTo save(DtoMemo dtoMemo) {
        em.persist(dtoMemo);

        Memo memo = new Memo(dtoMemo);
        Response createMemoTo = new Response createMemoTo(memo);

        return response createMemoTo;
    }

    @Override
    public Response updateMemoTo update(Long id, Request updateMemoTo memo) {

        Date now = new Date();
        DtoMemo updateMemo = em.find(DtoMemo.class, id);

        updateMemo.setTitle(memo.getTitle());
        updateMemo.setText(memo.getText());
        updateMemo.setUpdatedAt(now);

        Memo newMemo = new Memo(updateMemo);
        return new Response updateMemoTo(newMemo);
    }

    public void delete(Long id) {
        DtoMemo deleteMemo = em.find(DtoMemo.class, id);
        em.remove(deleteMemo);
    }

    @Override
    public Response showByPageMemoTo showByPage(Date searchDate, int page) {

        List<DtoMemo> memoList = em.createQuery("select m from DtoMemo m where m.createdAt > :searchDate", DtoMemo.class)
            .setParameter("searchDate", searchDate)
            .getResultList();

        Collections.reverse(memoList);
    }
}
```

예외 처리

```
@ExceptionHandler(value = {MethodArgumentNotValidException::class})
fun methodArgumentNotValidException(e : MethodArgumentNotValidException, request : HttpServletRequest) :
    val errors = mutableListOf<Error>()

    e.bindingResult.allErrors.forEach { errorObject ->
        val error = Error().apply { @NotNull
            this.field = (errorObject as FieldError).field
            this.message = errorObject.defaultMessage
            this.value = errorObject.rejectedValue
        }
        errors.add(error)
    }
    val errorResponse = ErrorResponse().apply { @NotNull ErrorResponse
        this.resultCode = "FAIL"
        this.httpStatus = HttpStatus.BAD_REQUEST.value().toString()
        this.httpMethod = request.method
        this.message = "요청에 오류가 발생하였습니다."
        this.path = request.requestURL.toString()
        this.timestamp = LocalDateTime.now()
        this.errors = errors
    }

    return ResponseEntity.status(HttpStatus.BAD_REQUEST).body(errorResponse)
}

@ExceptionHandler(value = {ConstraintViolationException::class})
fun constraintViolationException(
    e : ConstraintViolationException,
    request : HttpServletRequest
): ResponseEntity<ErrorResponse> {
    // 1. 에러 잡기
    val errors = mutableListOf<com.example.mvc.model.http.Error>()

    e.constraintViolations.forEach { @NotNull ConstraintViolation
        val error = com.example.mvc.model.http.Error().apply { @NotNull Error
            this.field = it.propertyPath.last().name
            this.message = it.message
            this.value = it.invalidValue
        }
        errors.add(error)
    }

    // 2. ErrorResponse
    val errorResponse = ErrorResponse().apply { @NotNull ErrorResponse
        this.resultCode = "FAIL"
        this.httpStatus = HttpStatus.BAD_REQUEST.value().toString()
        this.httpMethod = request.method
        this.message = "요청에 오류가 발생하였습니다."
        this.path = request.requestURL.toString()
        this.timestamp = LocalDateTime.now()
        this.errors = errors
    }
}
```

To. 스타트업

PPT에 있는 프로젝트에서는 JPA를 사용하지 않았지만, 개발 커뮤니티 과제와 개인 공부를 통해 꾸준히 Spring과 JPA를 학습하고 있습니다.

Pagination, 기본 CRUD, 테스트코드, Validation, 예외처리 등의 과제를 Spring, JPA, Junit을 통해 수행했습니다.



성숙한 백엔드 개발을 위해 네트워크 지식과 Infra 지식도 함께 학습하고 있습니다.

EC2, S3, RDS를 이용하여 배포를 진행한 경험이 있으며, 강의와 스터디를 통해 ELB, VPC 및 Subnet, Auto Scaling 등 AWS의 기능을 학습하였습니다.