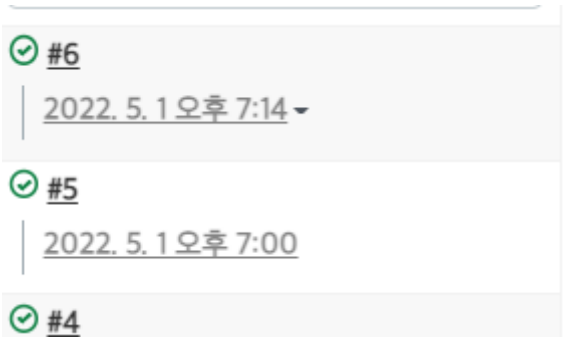
SpringBootBlog

1. 개발 환경
    - Intellij
    - Postman
    -
    - GitHub
    - Mysql
    -

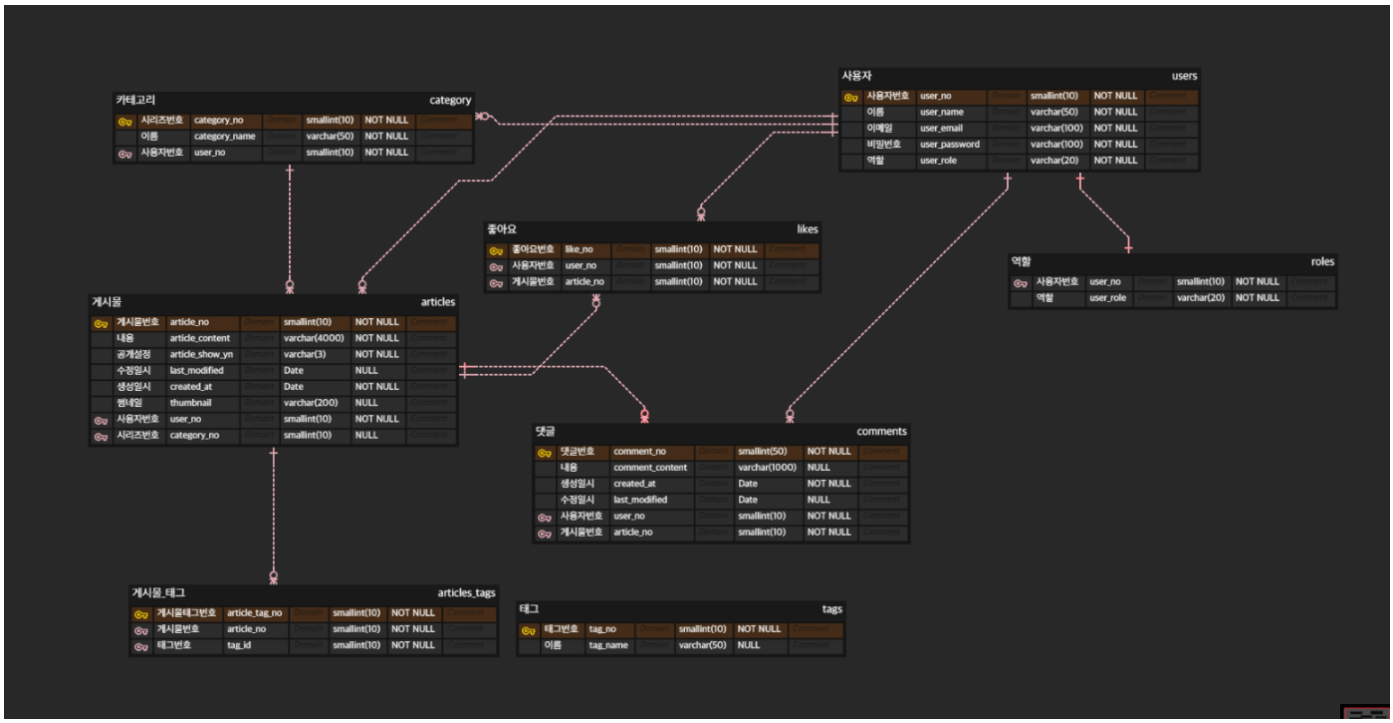
2. **사용기술**
    - 백엔드
        - **주요 프레임워크 / 라이브러리**
            - Java 11 openjdk
            - (안정된 거)SpringBoot 2.6.x
            - SpringBoot Security
            - Spring Data JPA
            - QueryDSL
        - **Build tool**
            - Gradle
        - **DataBase**
            - Mysql
        - **Infra**
            - AWS RDS
            - AWS S3
            - Jenkins
    - **프론트엔드**
        - Javascript
        - Html/CSS
        - thymeleaf
        - BootStrap 5
    - **라이브러리**
        - Lombok

- Toast Ui Editor
- Github-api

3. **진행상황**

> 1. 22/05/01
>
> - 젠킨스 연결 및 자동 배포 기능 구현완료 (github-webhook && ngrok > 로컬 url -> 임시 외부 url 로 변경)
>
> ```
> ngrok                                                                                                                    (Ctrl+C to quit)
>
> Session Status                online
> Account                       qkrtkdwns3410 (Plan: Free)
> Version                       3.0.3
> Region                        Japan (jp)
> Latency                       43.8077ms
> Web Interface                 http://127.0.0.1:4040
> Forwarding                    https://d020-222-112-97-146.jp.ngrok.io -> http://localhost:8080
>
> Connections                   ttl     opn     rt1     rt5     p50     p90
>                               164     0       0.00    0.00    0.07    5.70
>
> HTTP Requests
> -------------
>
> GET /static/14c0bbad/descriptor/hudson.tasks._ant.AntTargetNote/script.js              200 OK
> GET /adjuncts/14c0bbad/jenkins/management/AdministrativeMonitorsDecorator/resources.css 200 OK
> GET /job/blog_project/7/console                                                        200 OK
> GET /static/14c0bbad/descriptor/hudson.tasks._ant.AntOutcomeNote/style.css             200 OK
> GET /adjuncts/14c0bbad/org/kohsuke/stapler/bind.js                                     200 OK
> GET /adjuncts/14c0bbad/jenkins/management/AdministrativeMonitorsDecorator/resources.js  200 OK
> GET /static/14c0bbad/descriptor/hudson.tasks._ant.AntTargetNote/script.js              200 OK
> GET /adjuncts/14c0bbad/org/kohsuke/stapler/bind.js                                     200 OK
> GET /static/14c0bbad/descriptor/hudson.console.ExpandableDetailsNote/script.js         200 OK
> GET /adjuncts/14c0bbad/jenkins/management/AdministrativeMonitorsDecorator/resources.css 200 OK
> ```
>
> 
>

- ERD 다이어그램


카테고리
category
시리즈번호 category_no smallint(10) NOT NULL
이름 category_name varchar(50) NOT NULL
사용자번호 user_no smallint(10) NOT NULL
사용자
users
사용자번호 user_no smallint(10) NOT NULL
이름 user_name varchar(50) NOT NULL
이메일 user_email varchar(100) NOT NULL
비밀번호 user_password varchar(100) NOT NULL
역할 user_role varchar(20) NOT NULL
좋아요
likes
좋아요번호 like_no smallint(10) NOT NULL
사용자번호 user_no smallint(10) NOT NULL
게시물번호 article_no smallint(10) NOT NULL
역할
roles
사용자번호 user_no smallint(10) NOT NULL
역할 user_role varchar(20) NOT NULL
게시물
articles
게시물번호 article_no smallint(10) NOT NULL
내용 article_content varchar(4000) NOT NULL
공개설정 article_show_yn varchar(3) NOT NULL
수정일시 last_modified Date NULL
생성일시 created_at Date NOT NULL
썸네일 thumbnail varchar(200) NULL
사용자번호 user_no smallint(10) NOT NULL
시리즈번호 category_no smallint(10) NULL
댓글
comments
댓글번호 comment_no smallint(50) NOT NULL
내용 comment_content varchar(1000) NULL
생성일시 created_at Date NOT NULL
수정일시 last_modified Date NULL
사용자번호 user_no smallint(10) NOT NULL
게시물번호 article_no smallint(10) NOT NULL
게시물_태그
articles_tags
게시물태그번호 article_tag_no smallint(10) NOT NULL
게시물번호 article_no smallint(10) NOT NULL
태그번호 tag_id smallint(10) NOT NULL
태그
tags
태그번호 tag_no smallint(10) NOT NULL
이름 tag_name varchar(50) NULL

```
DROP TABLE IF EXISTS `users`;

CREATE TABLE `users` (
                `user_no`   smallint(10)  NOT NULL,
                `user_name`   varchar(50)  NOT NULL,
                `user_email`   varchar(100)    NOT NULL,
                `user_password`    varchar(100)    NOT NULL,
                `user_role`  varchar(20)  NOT NULL
);

DROP TABLE IF EXISTS `articles`;

CREATE TABLE `articles` (
                 `article_no`  smallint(10)  NOT NULL,
                 `article_content`  varchar(4000) NOT NULL,
                 `article_show_yn`    varchar(3)    NOT NULL,
                 `last_modified`    Date NULL,
                 `created_at`    Date NOT NULL,
                 `thumbnail`    varchar(200)    NULL,
                 `user_no`    smallint(10)  NOT NULL,
                 `category_no`  smallint(10)  NULL
);

DROP TABLE IF EXISTS `comments`;

CREATE TABLE `comments` (
                 `comment_no`    smallint(50)  NOT NULL,
                 `comment_content`    varchar(1000)  NULL,
                 `created_at`    Date NOT NULL,
                 `last_modified`    Date NULL,
                 `user_no`    smallint(10)  NOT NULL,
                 `article_no`  smallint(10)  NOT NULL
);

DROP TABLE IF EXISTS `category`;

CREATE TABLE `category` (
                 `category_no`  smallint(10)  NOT NULL,
                 `category_name` varchar(50)  NOT NULL,
                 `user_no`    smallint(10)  NOT NULL
);
```

22/05/03

> 스웨거 적용

```java
@Configuration
@EnableSwagger2
public class SwaggerConfig {

    private static final String API_NAME = "Blog API";
    private static final String API_VERSION = "0.0.1";
    private static final String API_DESCRIPTION = "Blog API 명세서";

    @Bean
    public Docket api() {
        // parameterBuilder : API 테스트시 모든 API에 전역 파라미터를 설정
        // 해당 소스는 모든 API 테스트시 HEADER 에 'Autorization' 이라는 값을 추가합니다.
        Parameter parameterBuilder = new ParameterBuilder()
            .name( name: HttpHeaders.AUTHORIZATION)
            .description( description: "Access Tocken")
            .modelRef( modelRef: new ModelRef( type: "string"))
            .parameterType( paramType: "header")
            .required( required: false)
            .build();

        List<Parameter> globalParamters = new ArrayList<>();
        globalParamters.add(parameterBuilder);

        return new Docket( documentationType: DocumentationType.SWAGGER_2)
            .globalOperationParameters( operationParameters: globalParamters) Docket
            .apiInfo( apiInfo: apiInfo()) //
```

코드가 지저분해 진다는 단점은 존재, 하지만 개별 기능에 대한 문서화가 가능한 장점이 있기에 도입.

테스트 코드 추가

JPA 엔티티 구성

22/05/10

1. 테스트 코드 작성

```java
@SpringBootTest
public class ArticlesRepositoryTest {

    private static final Logger logger = LoggerFactory.getLogger
      ( clazz: SpringBlogApplication.class);
    LocalDate now = LocalDate.now();

    @Autowired
    ArticlesRepository articlesRepository;

    @Test
    void save() {
        // 1. 게시물의 파라미터의 생성
        Articles params = Articles.builder()
                            .articleNo( articleNo: 1)
                            .articleContent( articleContent: "게시글의 내용입니다
                              .")
                            .articleTitle( articleTitle: "게시물의 제목입니다.")
                            .articleShowYn( articleShowYn: "Y")
                            .cdt( cdt: Date.valueOf( date: now))
                            .cid( cid: "박상준")
                            .userNo( userNo: 1)
                            .categoryNo( categoryNo: 1)
                            .build();


        System.out.println("params = " + params.getArticleContent());

        //2. 게시물 저장
        articlesRepository.save( entity: params);

        //3. 게시물 정보 조회
        if (articlesRepository.findById( id: 1)
                            .isPresent()) {
            Articles entity = articlesRepository.findById( id: 1)
                                        .get();
            assertThat( actual: entity.getArticleTitle()).isEqualTo( expected: "
```

```
        게시물의 제목입니다.");
        assertThat( actual: entity.getArticleContent()).isEqualTo
        ( expected: "게시글의 내용입니다.");
        assertThat( actual: entity.getCid()).isEqualTo( expected: "박상준");
    }
```

2.