REST API(STS) Guide▱

MESIM Platform▱

R&D Center / ESB division

저작권 ©2007-2023

Metabuild, R&D Center

Chapter 01 REST API 환경 세팅 (STS) | 4

1.1 STS 최신 버전 설치 | 4

1.2 프로젝트 파일 압축 해제 | 4

1.3 Lombok 다운로드 및 설치(있으면 SKIP) | 5

1.4 Package Explorer 내에 프로젝트 Import | 7

1.5 resources/static/index.html 생성 후 추가 | 10

1.6 DB 설정 및 Application.properties내에 default값 추가 | 11

1.7 해당 스프링부트 프로젝트 서버 실행 | 12

Chapter 02 REST API 개발 | 14

2.1 Controller | 14

2.2 Service | 19

2.3 Mapper | 20

2.4 Model | 21

2.5 Mybatis(xml 쿼리문) | 23

Chapter 03 Swagger | 24

3.1 Swagger 접속 | 25

3.2 Swagger API 테스트 | 28

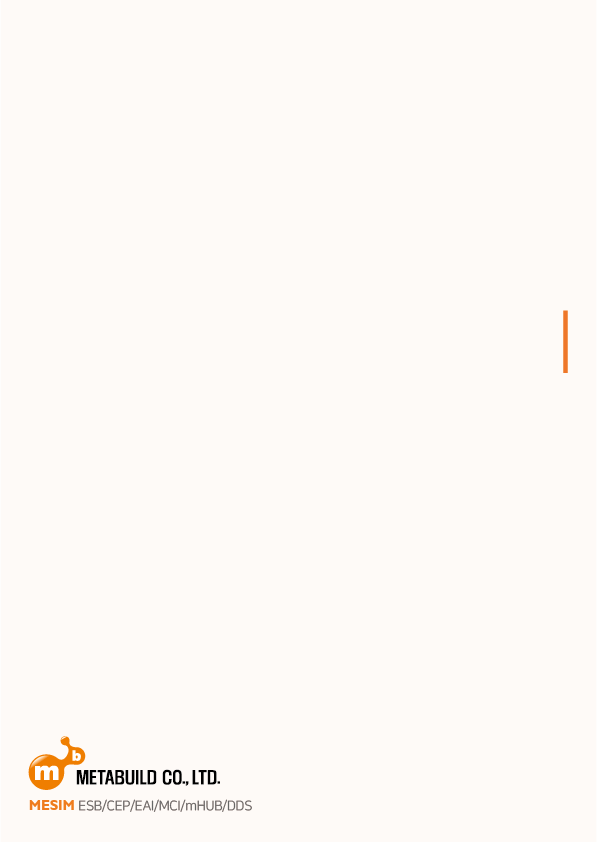
3.3 Swagger API Annotations | 29

**Chapter 01**

**REST API 환경 세팅(STS)**

1.1 STS 최신버전 설치

1.2 프로젝트 파일 압축 해제

1.3 Lombok 다운로드 및 설치(있으면 SKIP)

1.4 Package Explorer 내에 프로젝트 Import

1.5 resources/static/index.html 생성 후 추가

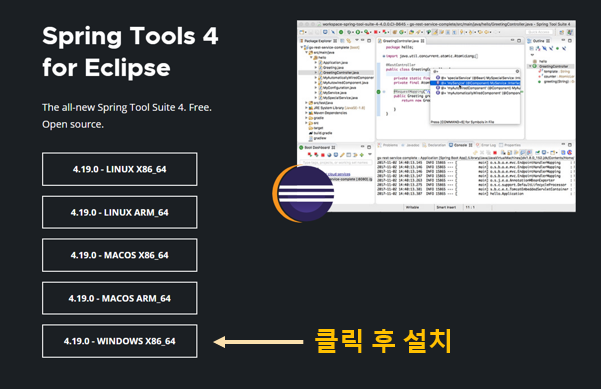
1.6 Application.properties내에 default값 추가 및 DB설정

1.7 해당 스프링부트 프로젝트 서버 실행

# REST API 환경 세팅(STS)

## STS 최신 버전 설치

1. STS 설치주소 : <https://spring.io/tools>

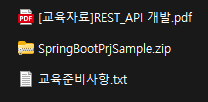


## 프로젝트 파일 압축 해제

1. 주어진 해당 파일 압축 해제



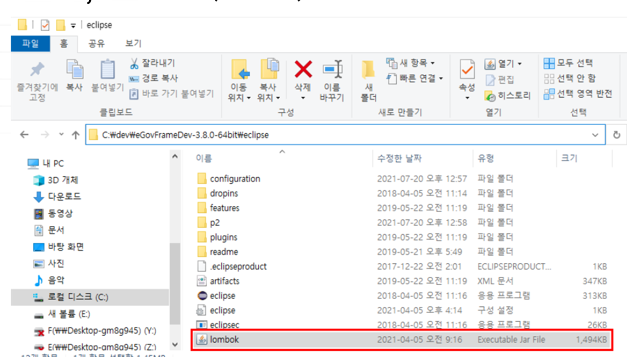
1. SpringBootPrjSample.zip 압축해제



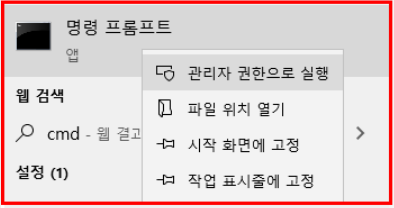
## Lombok 다운로드 및 설치(있으면 SKIP)

설치 주소 : <https://pncode.tistory.com/100>

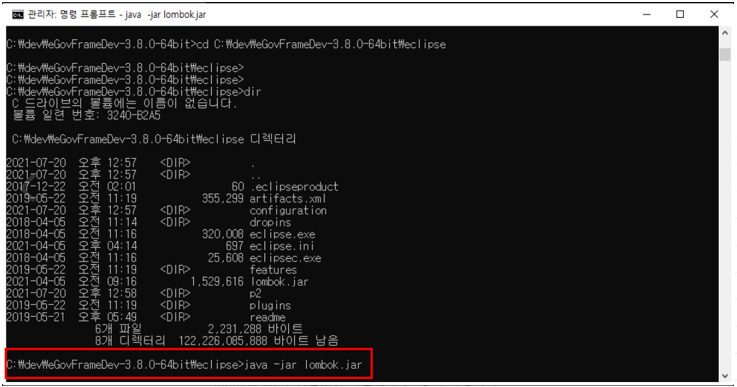
1. Lombok.jar 파일을 STS(이클립스) 파일이 있는 곳으로 이동



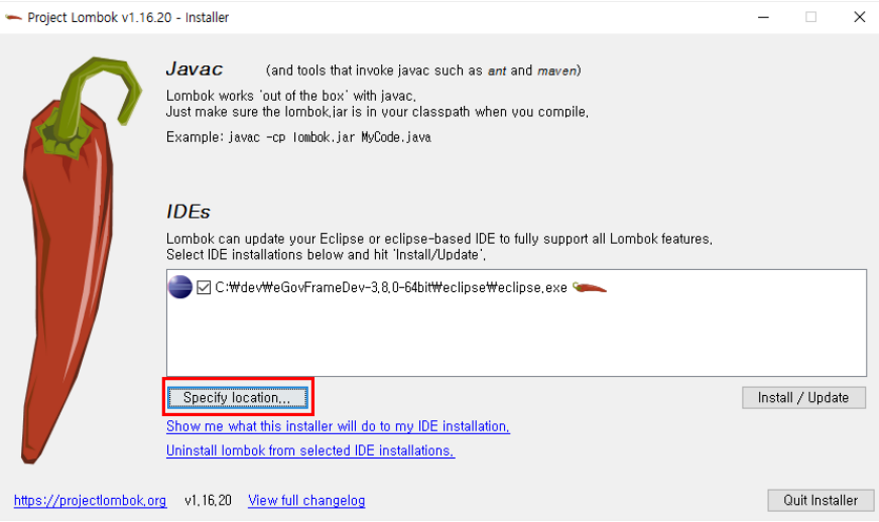
1. CMD(명령 프롬포트)를 관리자 권한으로 실행



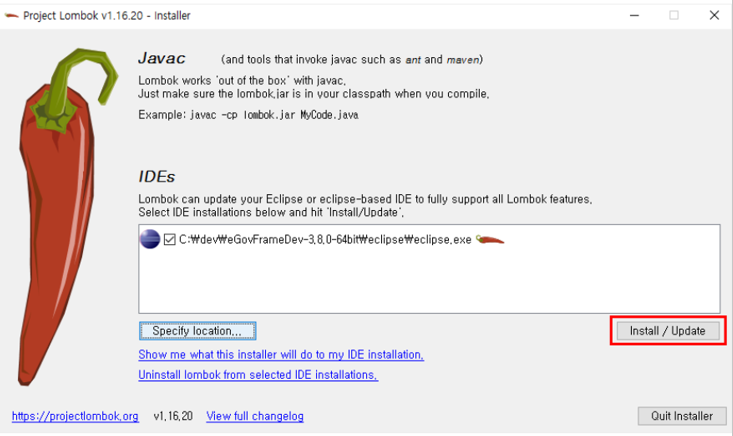
1. 이클립스가 있는 디렉토리 내에서 java -jar Lombok.jar 입력 후 다음 화면 실행



1. IDEs에 정상적인 eclipse.exe가 등장한다면 체크 후 Specify location… 클릭

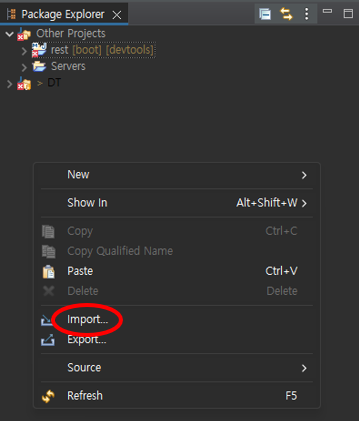


1. Install / Update 클릭하여 Lombok을 설치

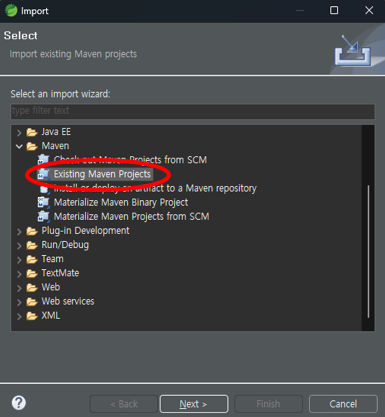


## Package Explorer 내에 프로젝트 Import

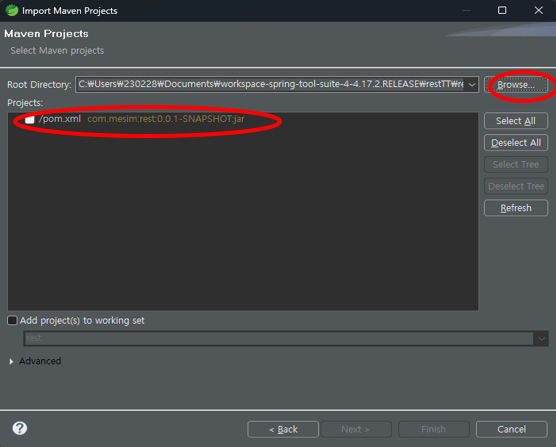
1. STS 내의 Package Explorer 빈 공간의 우클릭 – Import… 클릭



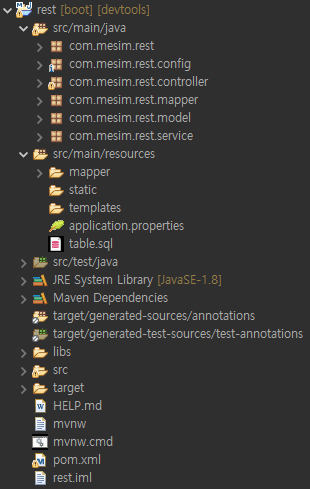
1. Maven프로젝트이기 때문에 Maven – Existing Maven Projects 선택, Next 클릭



1. Browse…에서 해당 rest 디렉토리 선택 – 프로젝트 선택 후 Finish 클릭

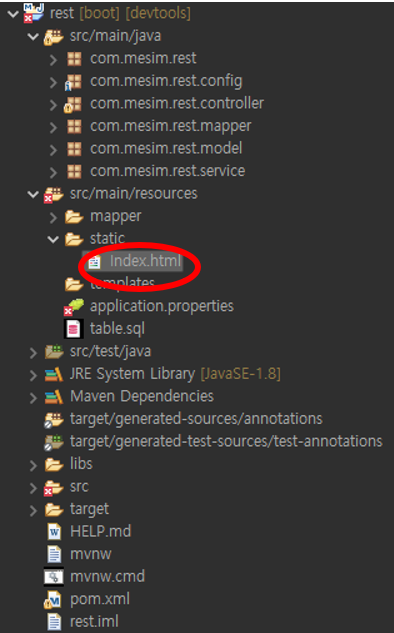


1. 위의 그림처럼 프로젝트가 Import 되었다면 정상

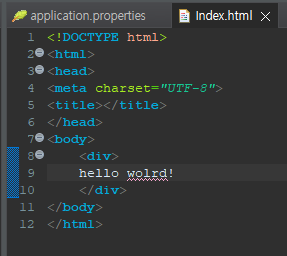


## Resources/static.index.html 생성 후 추가

1. Resources/static 우클릭 – New – Other - Html파일을 생성하여 해당 스프링부트 서버 기능을 시각화시켜줄 html이 기본적으로 있어야 하기 때문에



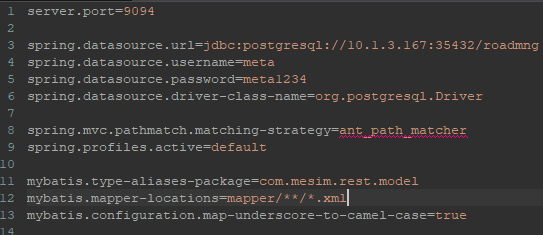
1. 다음과 같이 생성 후 저장.



## Application.properties내에 default값 추가 및 DB설정

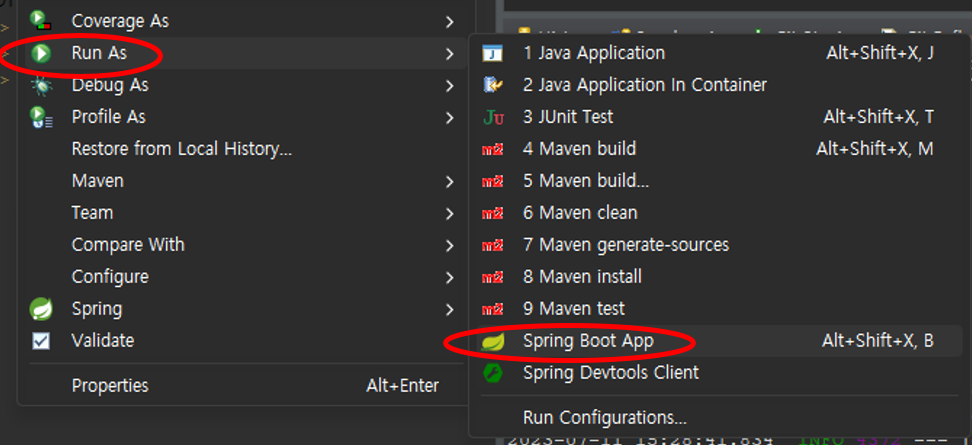
1. 프로젝트 내에 application.properties에 spring.profiles.active=default 추가 후,

3.4,5,6 라인에 연동해야 할 데이터베이스 정보 입력 후 저장



## 스프링부트 프로젝트 서버 실행

1. Package Explorer내에 프로젝트 우클릭 후 Run As -> Spring Boot App 클릭



**Chapter 02**

**REST API 개발**

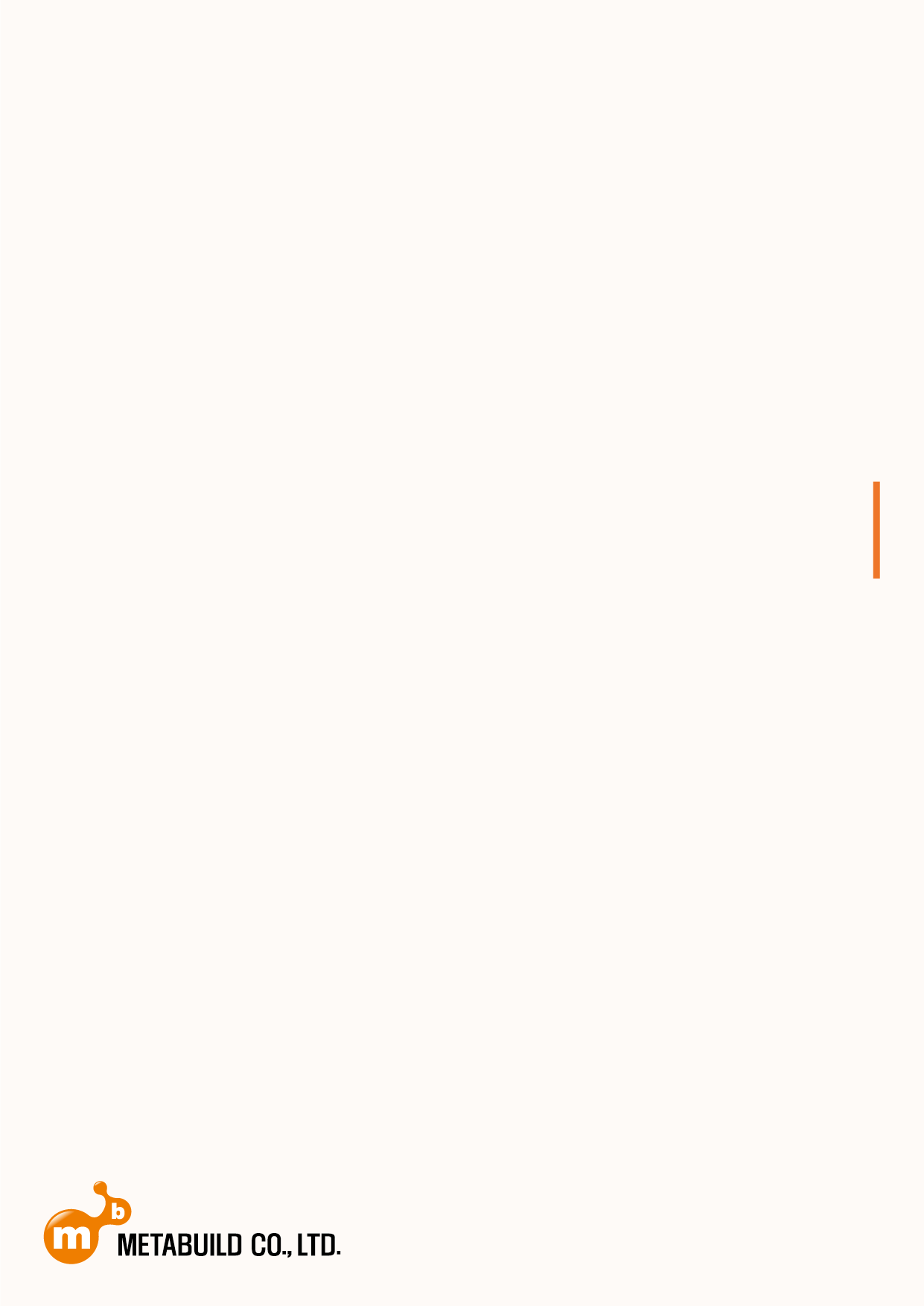
2.1 Controller

2.2 ServiceImpl

2.3 Mapper

2.4 Model

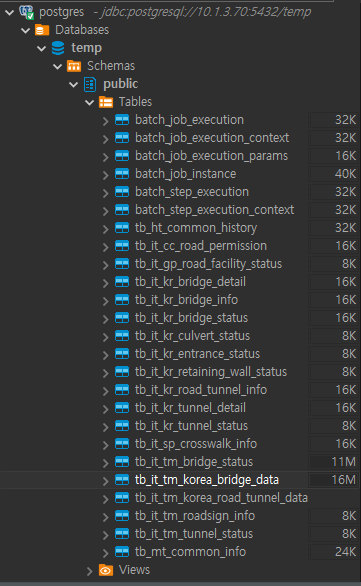
2.5 Mybatis(xml 쿼리문)



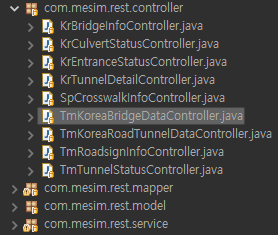
# REST API 개발

## Controller (다른 테이블도 동일하게 제작)

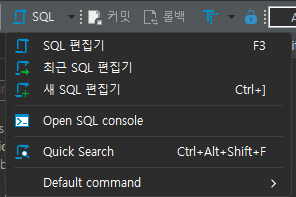
1. tb\_it\_tm\_korea\_bridge\_data에 해당하는 테이블의 데이터들을 GET으로 통신하여 조회하는 API를 만들기 위해서 첫번째로 DBeaver(DB 관리툴)내에 해당 테이블 이름과 정보들을 확인한다.



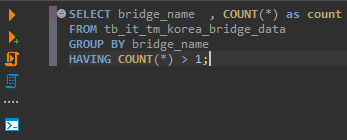
1. com.mesim.rest.controller내에 조회할 데이터가 들어있는 테이블 네이밍을 참고하여 파스칼 표기법으로 뒤에 ~Controller을 붙여서 java class를 생성한다.



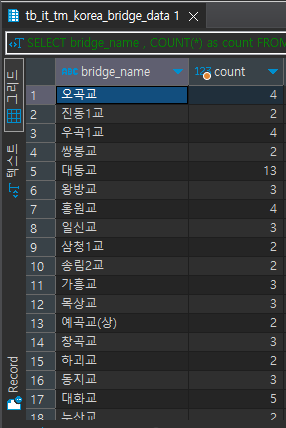
1. DBeaver에서 SQL편집기(F3)을 들어간다.



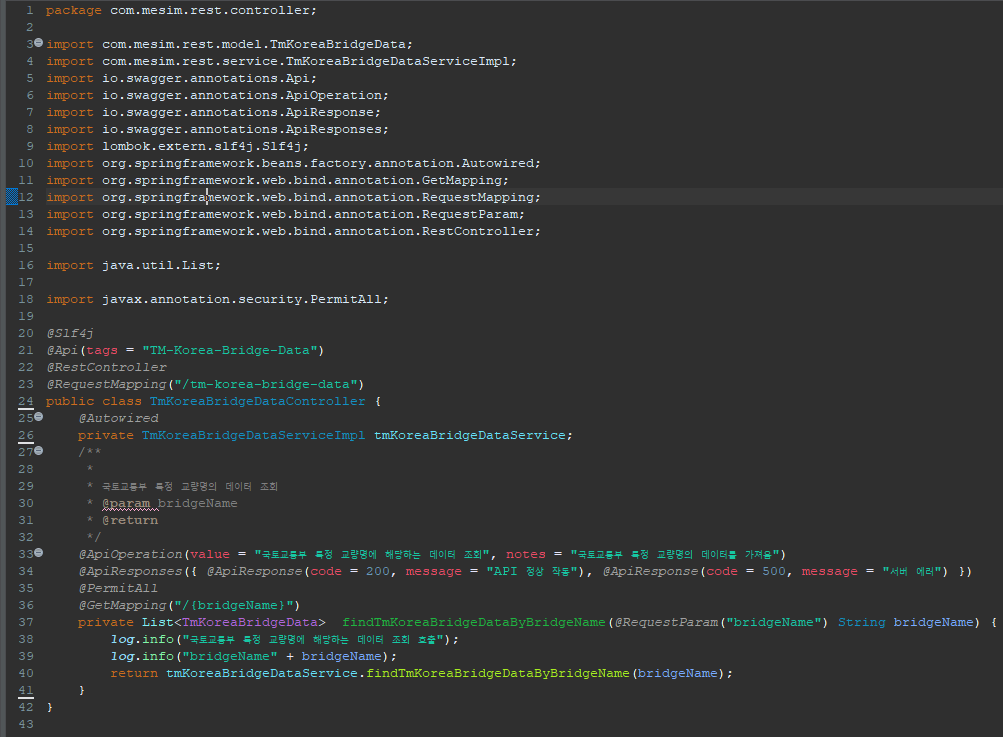
1. 기준 데이터로 설정한 컬럼 내에 데이터가 고유값으로 1개씩 존재하는지, 2개이상인지 확인하기 위해서 다음 SQL문을 실행한다.



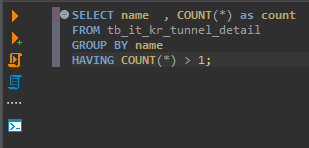
1. 해당 컬럼의 데이터 COUNT수가 1보다 높을 때 SQL문을 돌렸을 때 값이 나오기 때문에 값이 이렇게 나온다면 기준 컬럼이 고윳값1개가 아니라는 것을 확인한다.

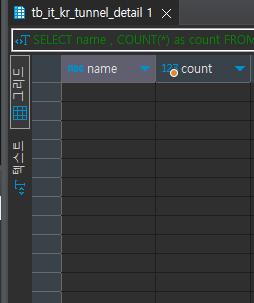


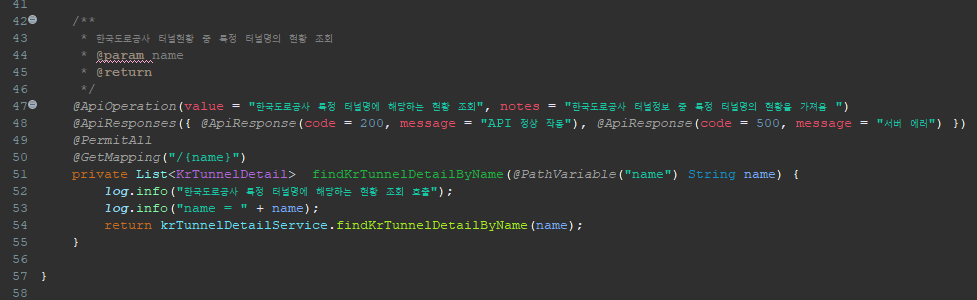
1. Controller파일을 열어서 기준 id값이 되는 컬럼을 bridge\_name으로 지정한 후 그것을 기준으로 조회하는 인터페이스 역할의 코드 작성한다. (고윳값 1개가 아닌 것은 @Requestparam을 사용한다)



1. 이런식으로 기준 컬럼 데이터값이 고유 1개의 값인게 확인된다면 다음과 @PathVariable을 사용한다.

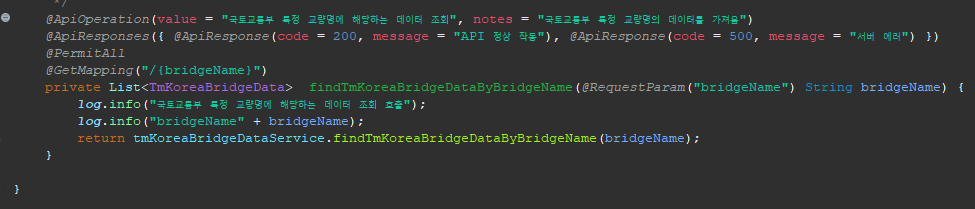




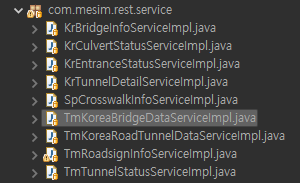


## Service (다른 테이블도 동일하게 제작)

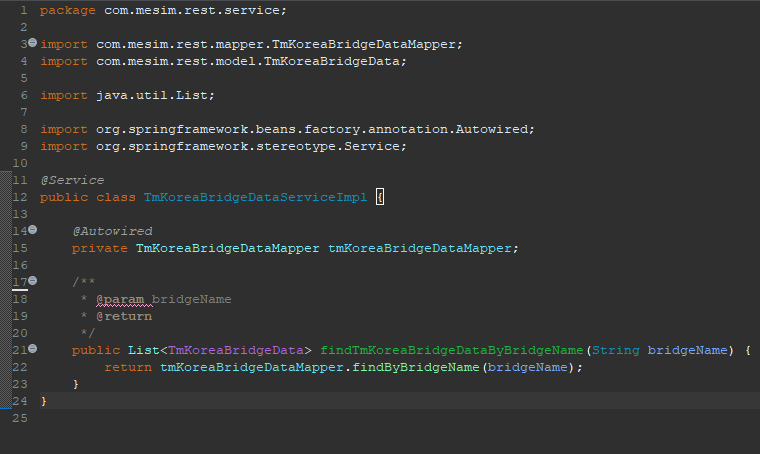
1. Controller에서 Service클래스가 있어야 return이 가능하기에 Service 생성 필요성을 인지한다.



1. com.mesim.rest.service내에 다음과 같이 해당 테이블의 Controller와 같은 파스칼 표기법 네이밍 뒤에 ~ServiceImpl을 붙여서 클래스를 생성한다.

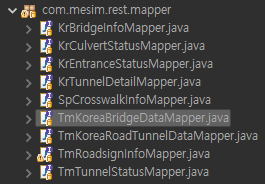


1. Service단에서는 Mapper로 return한다는 개념을 참고하여 코드를 작성한다.

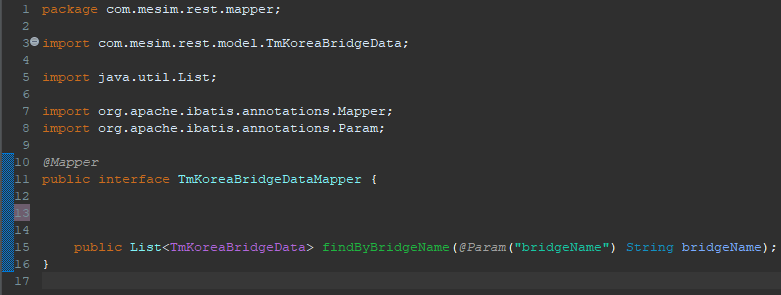


## Mapper (다른 테이블도 동일하게 제작)

1. com.mesim.rest.mapper내에 해당 테이블 파스칼 표기법 네이밍 뒤에 ~Mapper를 붙여서 class를 생성한다.



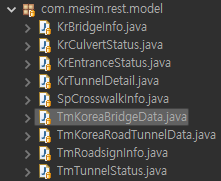
1. 다음과 같이 사진을 참고하여 Mapper에 해당하는 코드를 작성해준다.



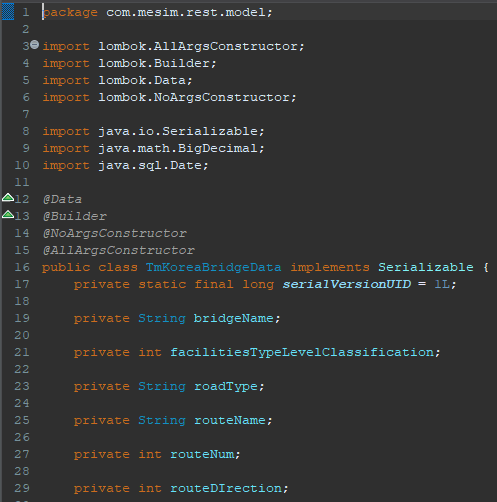
\*이때 mapper의 이름은 mybatis로 제작한 쿼리의 id와 같아야 한다.

## Model (다른 테이블도 동일하게 제작)

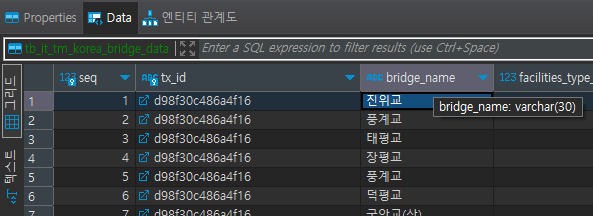
1. com.mesim.rest.mapper에 테이블 파스칼 표기법 네이밍 그대로 다음과 같이 class를 생성해준다.



1. 다음을 참고하여 해당 테이블 데이터에 해당하는 필드들을 캡슐화하여 객체로 표현하여 지정하는 코드들을 모두 작성해준다.

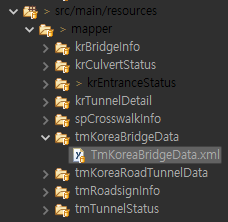


1. 데이터 필드 별 데이터 타입은 DBeaver에서 해당 필드에 마우스를 올려놓아서 확인한 뒤 작성해준다.

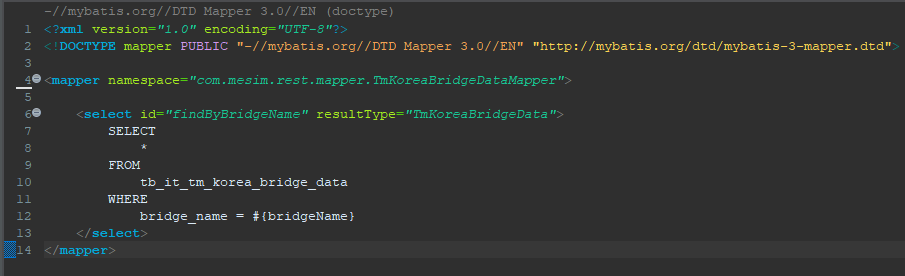


## Mybatis(Xml 쿼리문) (다른 테이블도 동일하게 제작)

1. src/main/resources/mapper에 다음과 같이 xml파일을 생성한다.



1. 다음을 참고하여 해당 테이블의 Mybatis(XML) 쿼리를 작성한다.



## 번외

### Model과 DTO

**Model과 DTO의 공통점과 차이점?**

**DTO**

Data Transfer Object인 DTO는 데이터 전달을 목적으로 하는 객체로, 특히 서비스 사이에서 데이터를 전송하거나 뷰와 컨트롤러 간의 데이터 전달에 사용된다. 일반적으로 여러 필드를 포함하기에 DB의 특정 엔티티와 1:1 or 1:N 관계를 맺을 수 있다. 주로 API 요청과 응답에서 활용된다. 또한 필요한 정보만을 담아서 불필요한 데이터 노출을 방지하고자 사용한다.

**Model**

데이터 논리적 구조를 표현하는 객체로, 기본적으로 DB와의 연계를 통해 데이터 CRUD 작업을 수행하는 역할을 담당한다. 일반적으로 DB의 테이블과 1:1로 매핑되어 실제 데이터 CRUD 작업을 수행한다.

**DTO와 Model의 공통점**

모두 데이터를 나타내는 객체이며 다양한 필드와 메소드를 가지고 있다. 주로 View와 Controller 사이에서 데이터 전달과 관리에 사용된다. 프로젝트에 따라 두 객체를 혼합한 상태로 사용하기도 하는데, 예를 들어 Model 객체를 DTO로 변환하여 Controller에서 View로 데이터를 전송하거나, 반대로 DTO를 Model로 변환하여 DB에 저장하는 용도로 사용하기도 한다.

### @Controller 와 @RestController 의 차이

@Controller  
API와 view를 동시에 사용하는 경우에 사용한다.  
대신 API 서비스로 사용하는 경우는 @ResponseBody를 사용하여 객체를 반환한다.  
view(화면) return이 주목적이다.

@RestController  
view가 필요없는 API만 지원하는 서비스에서 사용한다.  
Spring 4.0.1부터 제공  
@RequestMapping 메서드가 기본적으로 @ResponseBody 의미를 가정한다.  
data(json, xml 등) return이 주목적이다.

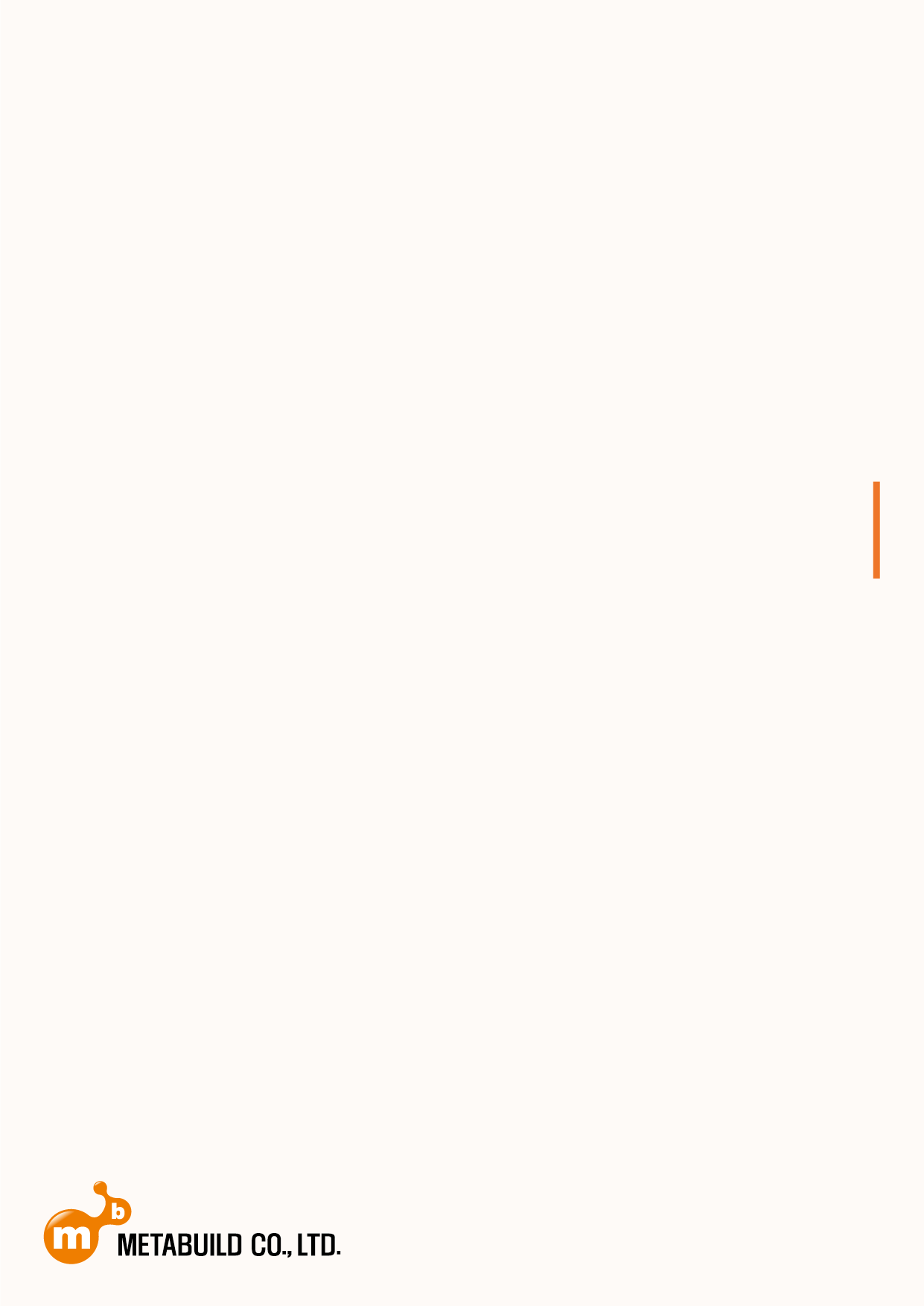
즉, @RestController = @Controller + @ResponseBody 이다.

**Chapter 03**

**Swagger**

3.1 Swagger 접속

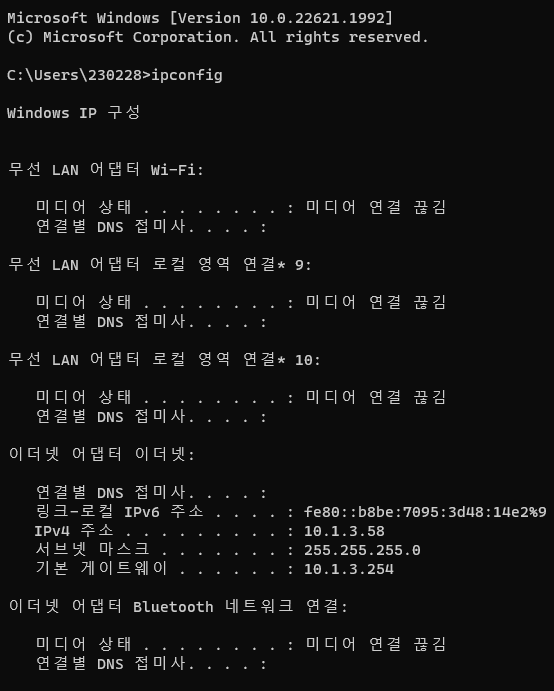
3.2 ServiceImpl



# Swagger

## Swagger 접속

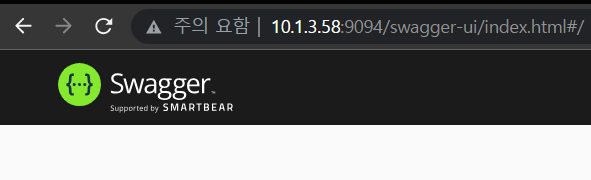
1. CMD(명령 프롬포트)를 켜서 ipconfig 입력 하여 컴퓨터 IP를 확인한다.



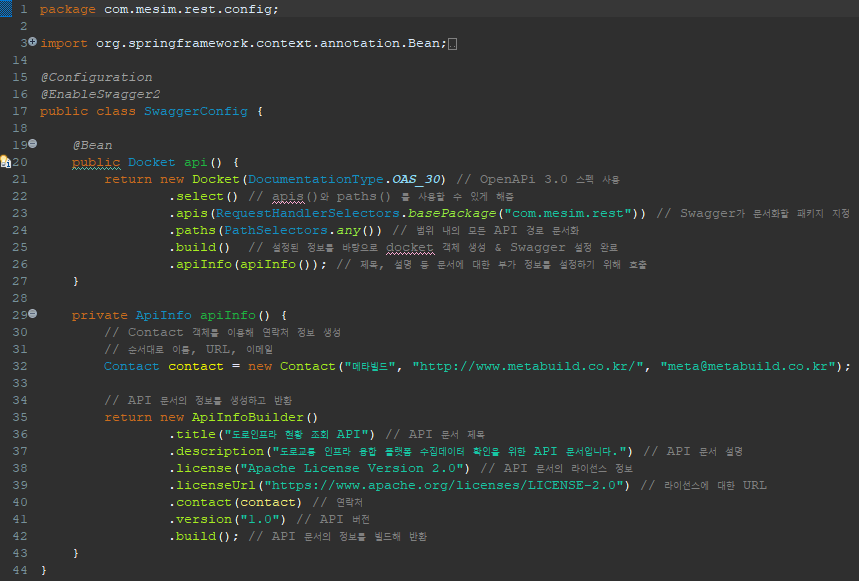
1. Sts 내에 application.properties내에 server.port 확인 후



1. 다음 사진과 같이 주소 입력 후 들어가서 정상적으로 해당 Ip와 Port에 해당하는 Swagger 웹이 등장하는지 확인한다. (Spring boot 서버 정상 기동 후 작동 가능)



1. Com.mesim.rest.config내에 SwaggerConfig.java 클래스 파일을 다음과 같이 작성하여

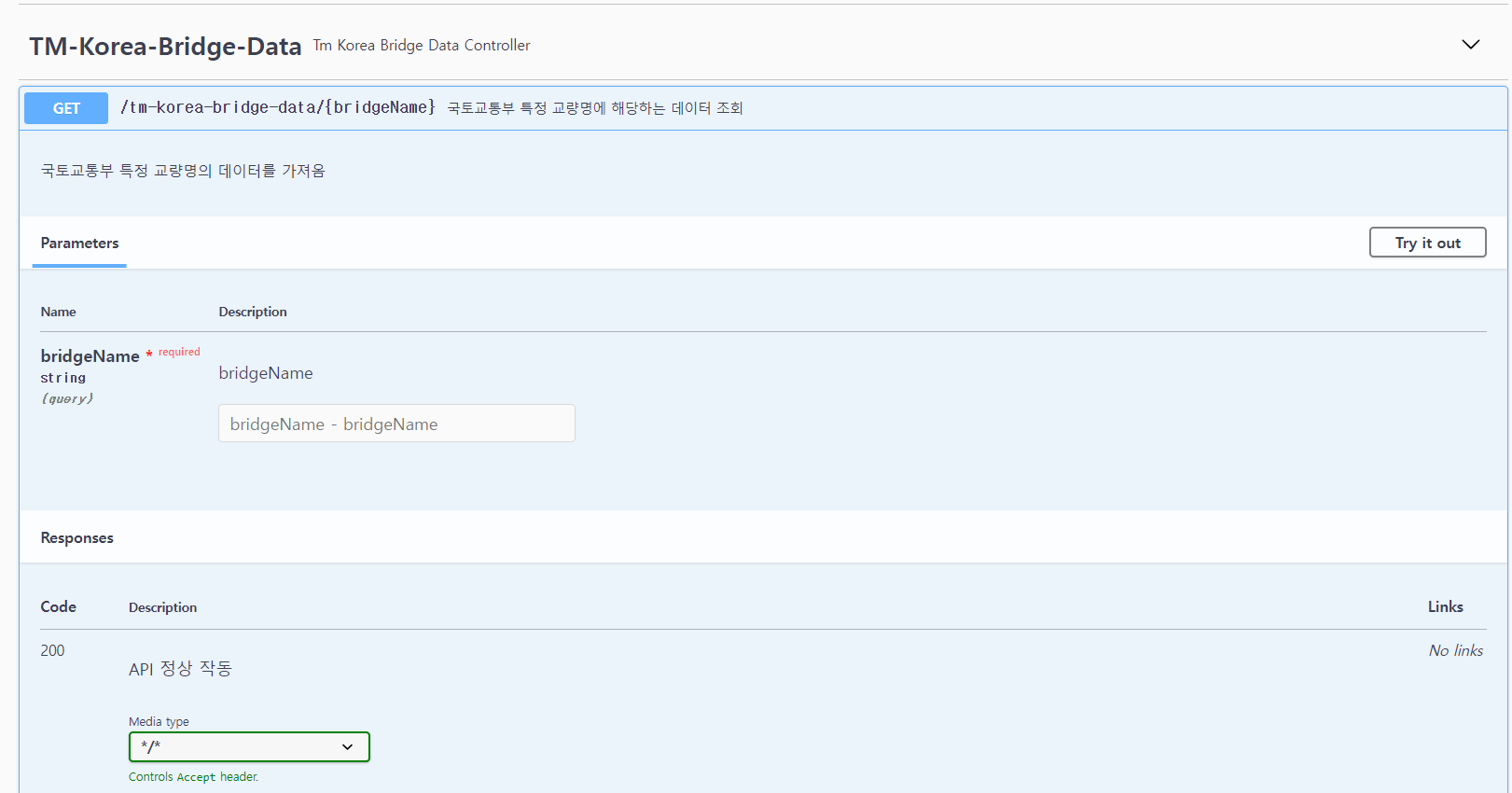


1. Swagger 웹 내에 다음과 같이 상단 API에 대한 네이밍 및 정보에 대한 info를 추가해준다.

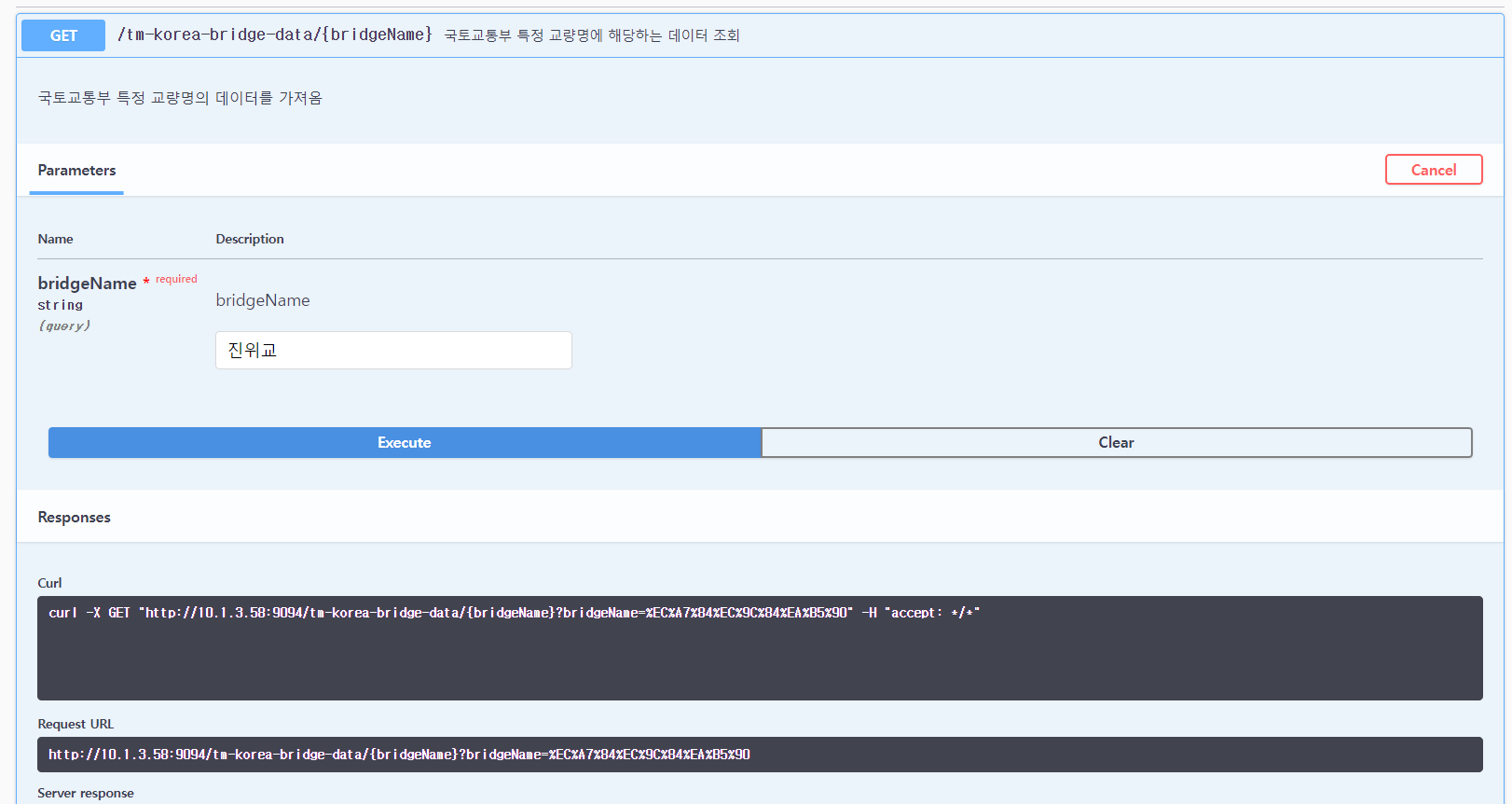


## Swagger API 테스트

1. 해당 Swagger 링크에서 REST API GET통신 할 Controller가 존재하는지 확인한 후, API 정상 작동인지 확인한다. (아래와 같이 통신한다면 정상)



1. Try it out을 클릭하여 해당 키 값인 (위의 사진에선 bridge\_name)필드의 데이터를 입력한후 정상적으로 데이터를 조회하는지 테스트를 한다. (아래와 같이 통신한다면 정상)

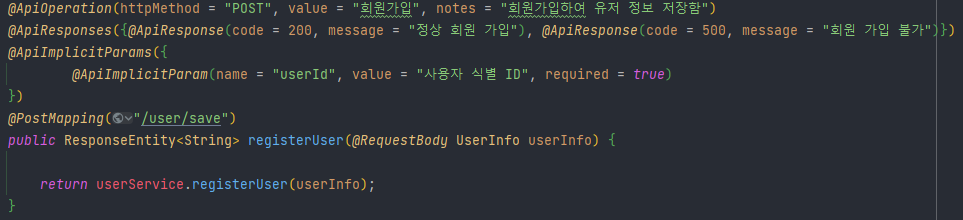


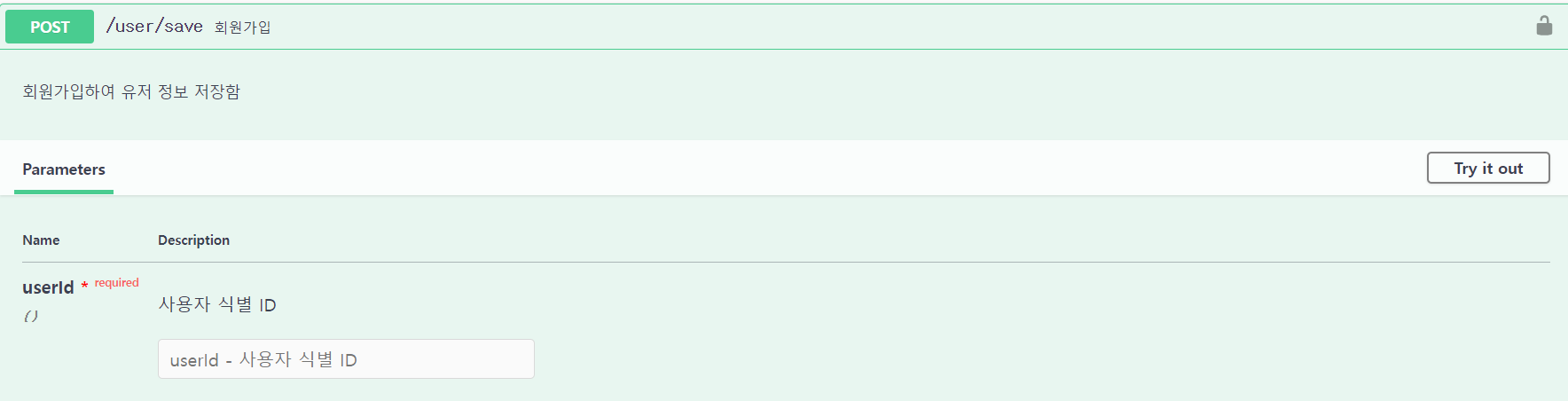
## Swagger Annotations

### @ApilmplicitParams

DTO(Model) 상세 파라미터의 설명을 추가하거나 필수 파라미터, Data Type, 파라미터 Type을 설정한다.

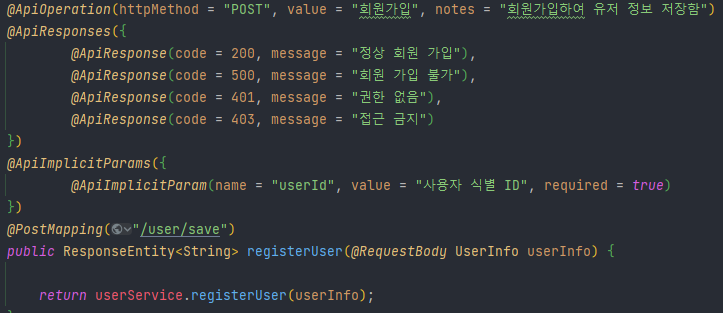
<<예시>>



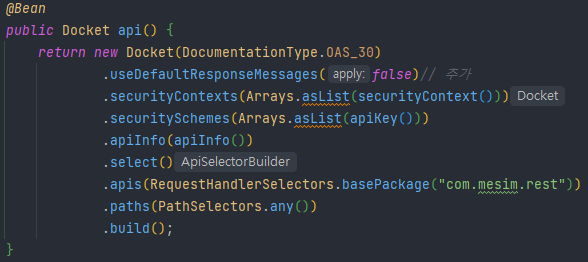


### @ApiResponses & @ApiResponse

해당 엔드포인트에 대한 반환값을 커스텀하여 입력할 수 있다. 다음과 같이 어노테이션을 작성하면 된다.



이때, SwaggerConfig.java에 다음과 같은 코드를 추가해준다.



### @ApiModelProperty

해당 프로퍼티에 대한 hidden, value, example 등을 작성할 수 있다.

그 예는 다음과 같다.

