

서비스 로직 작성

Post Service 프로젝트 생성

프로젝트 생성 – post project

<https://start.spring.io/> 접속



Project

☒ Gradle - Groovy ☐ Gradle - Kotlin ☐ Maven

Language

☒ Java ☐ Kotlin ☐ Groovy

Spring Boot

☐ 4.0.0 (SNAPSHOT) ☐ 3.5.1 (SNAPSHOT) ☒ 3.5.0 ☐ 3.4.7 (SNAPSHOT)

☐ 3.4.6 ☐ 3.3.13 (SNAPSHOT) ☐ 3.3.12

Project Metadata

Group

Artifact

Name

Description

Package name

Packaging ☒ Jar ☐ War

Java ☐ 24 ☐ 21 ☒ 17

Dependencies

ADD DEPENDENCIES... CTRL + B

Spring Web WEB

Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.

Cloud LoadBalancer SPRING CLOUD ROUTING

Client-side load-balancing with Spring Cloud LoadBalancer.

Eureka Discovery Client SPRING CLOUD DISCOVERY

A REST based service for locating services for the purpose of load balancing and failover of middle-tier servers.

OpenFeign SPRING CLOUD ROUTING

Declarative REST Client. OpenFeign creates a dynamic implementation of an interface decorated with JAX-RS or Spring MVC annotations.

Spring Data JPA SQL

Persist data in SQL stores with Java Persistence API using Spring Data and Hibernate.

MySQL Driver SQL

MySQL JDBC driver.

Lombok DEVELOPER TOOLS

Java annotation library which helps to reduce boilerplate code.

Validation I/O

Bean Validation with Hibernate validator.

Post 프로젝트 기본 세팅

- ✓ 파일 > 프로젝트 구조 > SDK 확인
- ✓ application.yml, application-local.yml 등 설정 분리
 - application.properties는 삭제
- ✓ active profiles 지정
 - Community 버전: VM 옵션에 -Dspring.profiles.active=local 입력
 - Ultimate 버전: 활성 프로파일에 local 입력
- ✓ 기본 코드 세팅
 - ApiResponseDto (응답 메시지 정규화)
 - ApiError, ClientError 등 Api Exception
 - ApiCommonAdvice (에러 응답 처리)

Post 프로젝트 설정

```
spring:
  application:
    name: backend-post
```

application.yml

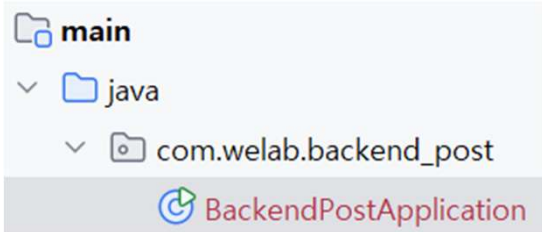
```
server:
  port: 8083

spring:
  datasource:
    url: jdbc:mysql://localhost:13307/post?serverTimezone=UTC&useSSL=true&autoReconnect=true&useUnicode=true&characterEncoding=utf-8
    username: user
    password: 1234
    driver-class-name: com.mysql.cj.jdbc.Driver
  hikari:
    connection-test-query: SELECT 1 # HikariCP 유효성 검사 추가
  jpa:
    hibernate:
      ddl-auto: create # 오직 테스트 환경에서만
    generate-ddl: true # 오직 테스트 환경에서만
    show-sql: true
    open-in-view: false

eureka:
  instance:
    prefer-ip-address: true
  client:
    register-with-eureka: true
    fetch-registry: true
    serviceUrl:
      defaultZone: http://localhost:8761/eureka/
```

application-local.yml

Post 프로젝트 코드 작성



```
@EnableDiscoveryClient
@EnableFeignClients
@SpringBootApplication
public class BackendPostApplication {

    public static void main(String[] args) {
        SpringApplication.run(BackendPostApplication.class, args);
    }

}
```