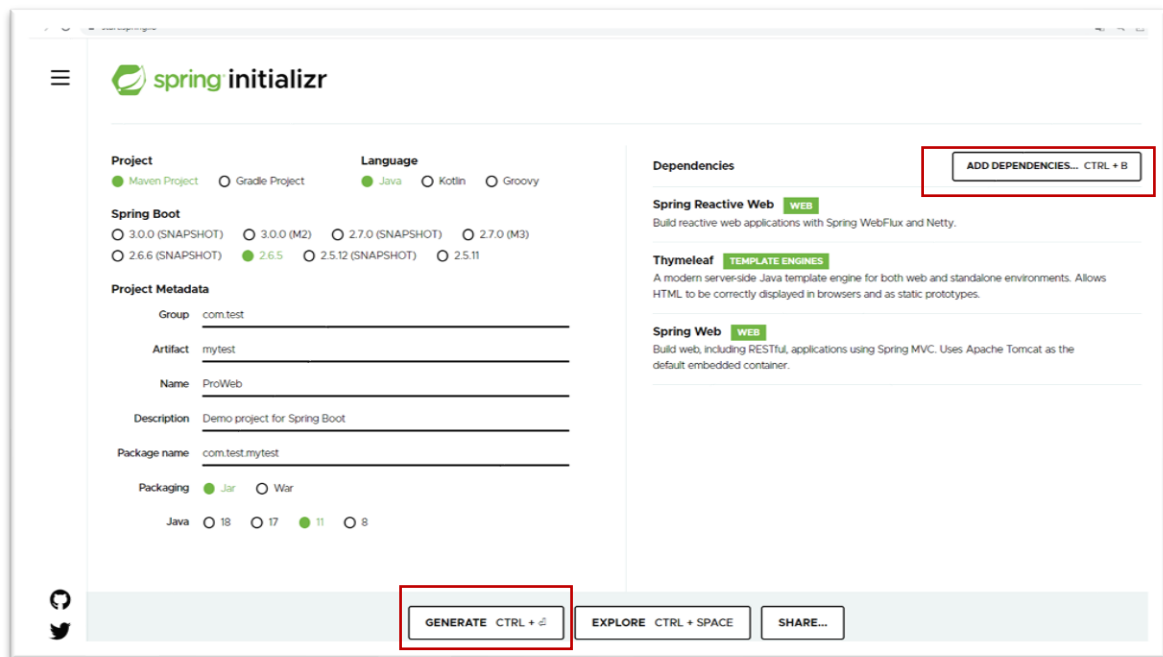**Spring-boot lab01 _ProWeb**

[STEP 01] start.spring.io 에서 아래와 같이 선택한다.

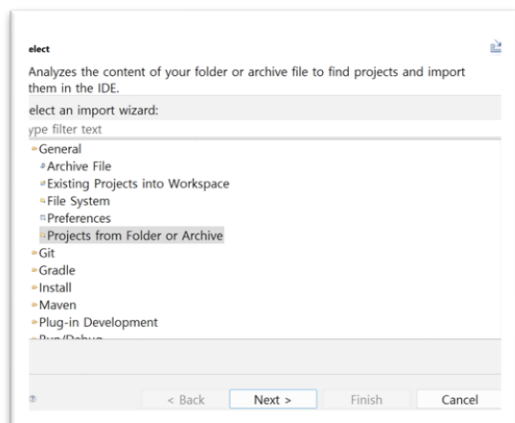1-1. ADD DEPENDENCIES 클릭 -> Dependencies를 추가한다.

1-2. Project Metadata를 화면과 같이 채운다.
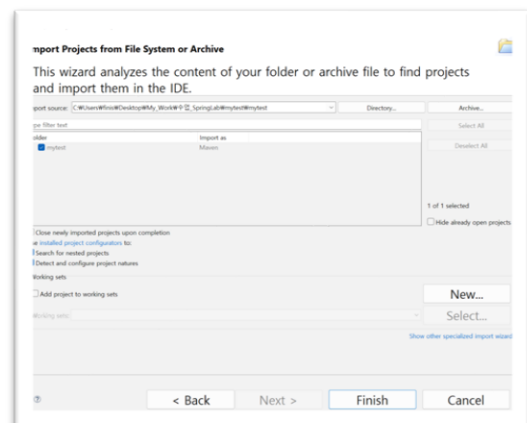
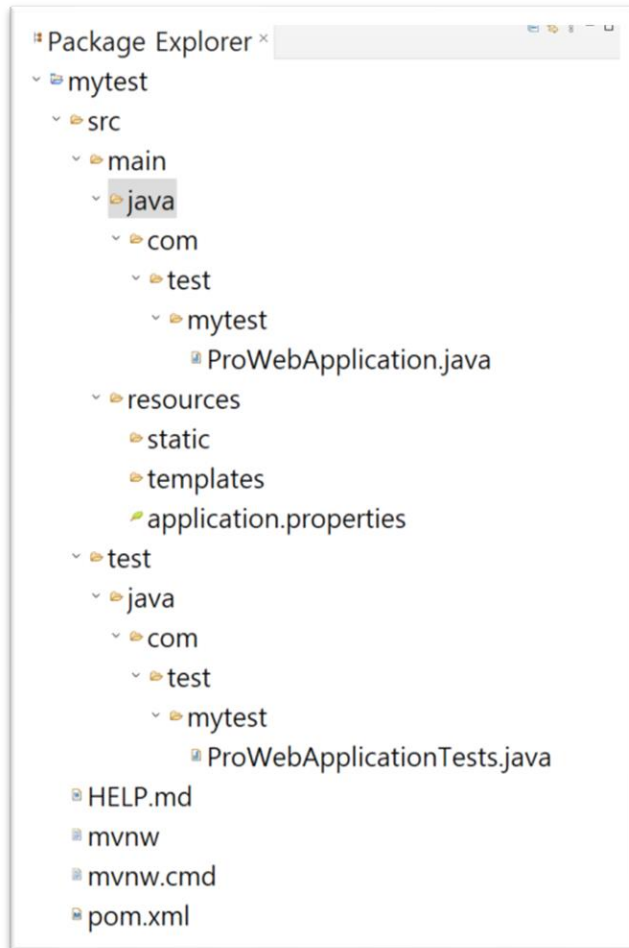1-3. GENERATE 클릭 -> 다운로드 한 후 압축을 푼다.

[STEP 02] 프로젝트를 임포트 한다.

2-1. STS에 임포트 한다.

->

2-2. 완성형 구조 폴더를 확인한다.

```
Package Explorer ×
mytest
  src
    main
      java
        com
          test
            mytest
              ProWebApplication.java
      resources
        static
        templates
        application.properties
    test
      java
        com
          test
            mytest
              ProWebApplicationTests.java
  HELP.md
  mvnw
  mvnw.cmd
  pom.xml
```

2-3. pom.xml 확인

```
*mytest/pom.xml ×

ependencies

ependencies

spring-boot-starter-thymeleaf (managed:2.6.5)
spring-boot-starter-web (managed:2.6.5)
spring-boot-starter-webflux (managed:2.6.5)
spring-boot-starter-test (managed:2.6.5)
reactor-test (managed:3.4.16)
```

```
*mytest/pom.xml ×
lverview
rtifact
roup Id : com.test
rtifact Id : mytest
ersion : 0.0.1-SNAPSHOT
ackaging : jar
Parent
roup Id : org.springframework.boot
rtifact Id : spring-boot-starter-parent
ersion : 2.6.5
elative Pa
Properties
java.version : 11                    Create...
                                     Remove
```
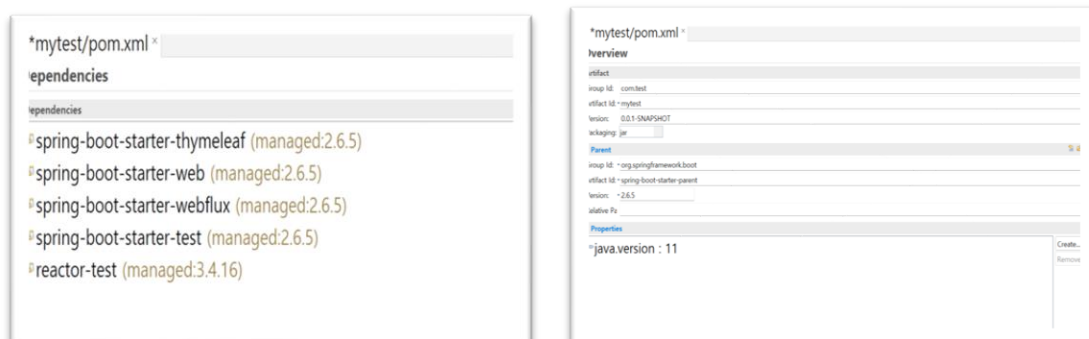
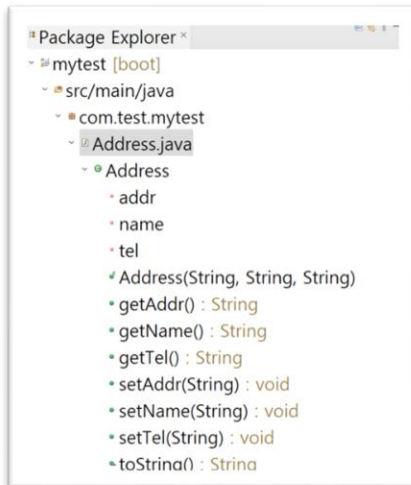**[STEP 03]** 클래스를 추가한다.

3-1. com.test.mytest.Address 추가 _일반적인 setter&getter (POJO)형식의 클래스
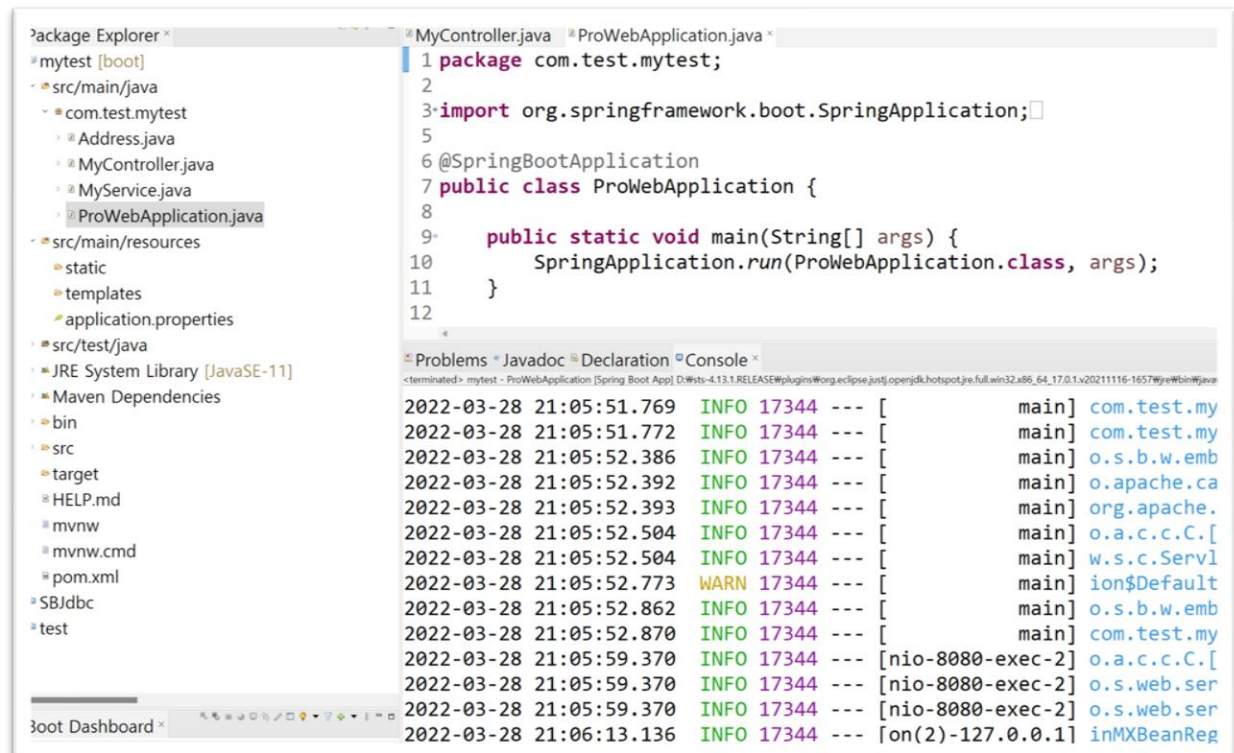

3-2 . com.test.mytest.MyService 추가

```java
package com.test.mytest;
import java.time.Duration;
import java.util.Arrays;
import java.util.List;
import java.util.Random;
import org.springframework.stereotype.Service;
import reactor.core.publisher.Flux;

@Service
public class MyService {
    private Random   my_ran=new  Random();

    public Flux<Address> getAllAddress() {
        return Flux.<Address>  generate(sink -> sink.next(randomAddress()))
                .delayElements(Duration.ofMillis(250));
    }

    private Address randomAddress() {
        return all.get(my_ran.nextInt(all.size()));
    }

    private List<Address>  all = Arrays.asList(
        new Address("Hong-gil dong","seoul","02-000,000"),
        new Address("jung-gil dong","inchon","031-000,000"),
        new Address("Pak-gil dong","pusan","051-000,000")
    );

}
```

3-3. com.test.mytest.MyController추가
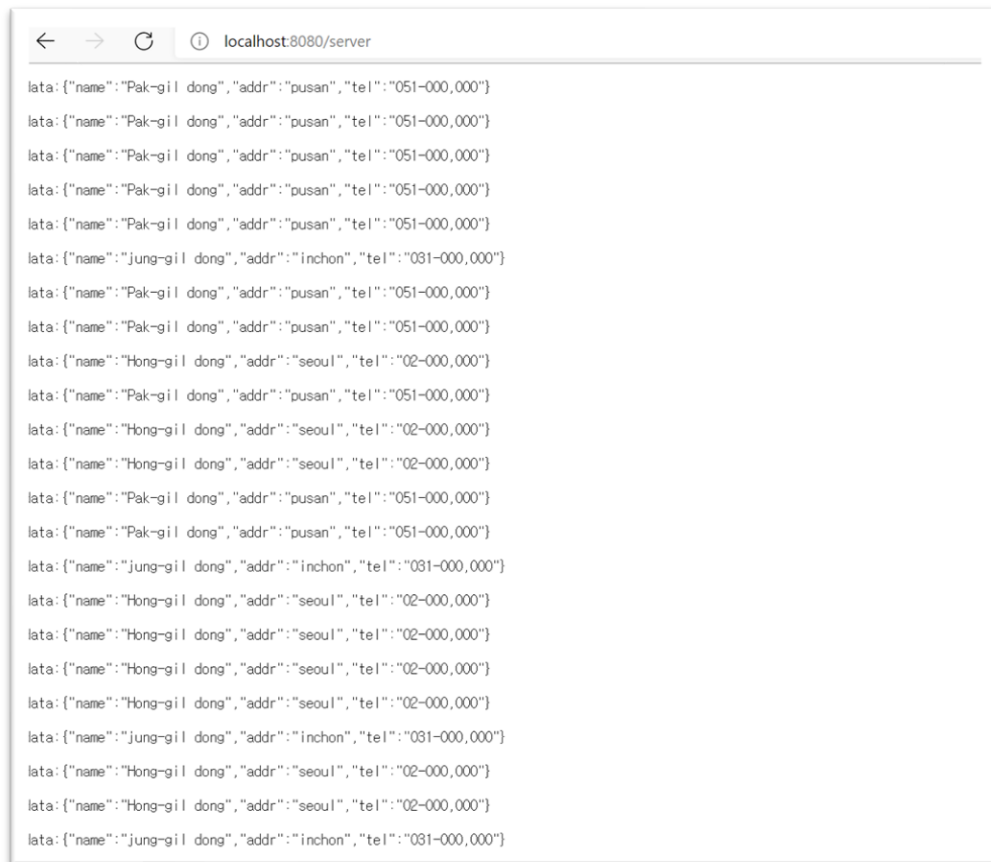
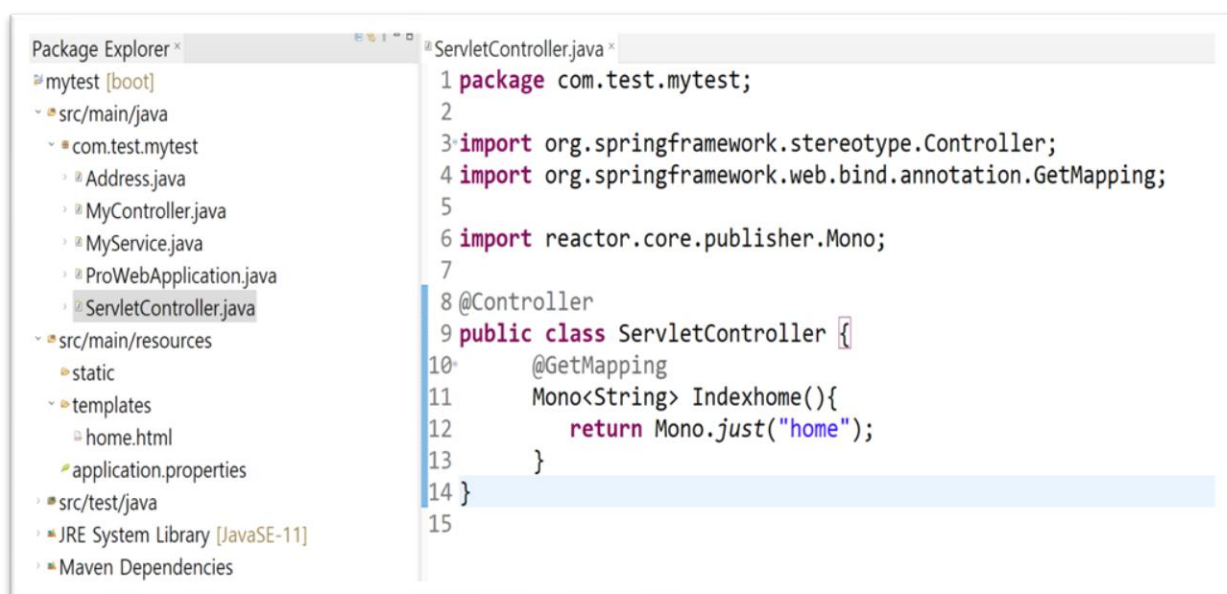## 3-4. 만들어진 ProWebApplication.java -> 오.마 -> Run As -> SpringBoot APP 실행

이때 Console에 port 확인

3-5 . 브라우저 확인

[STEP 04] 클래스를 추가한다.


```java
package com.test.mytest;

import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.GetMapping;

import reactor.core.publisher.Mono;

@Controller
public class ServletController {
    @GetMapping
    Mono<String> Indexhome(){
        return Mono.just("home");
    }
}
```

[STEP 05] 웹페이지를 추가한다. (오타조심)

5-1. /src/main/resources/templates/home.html 로 페이지 추가 후 내용 작성
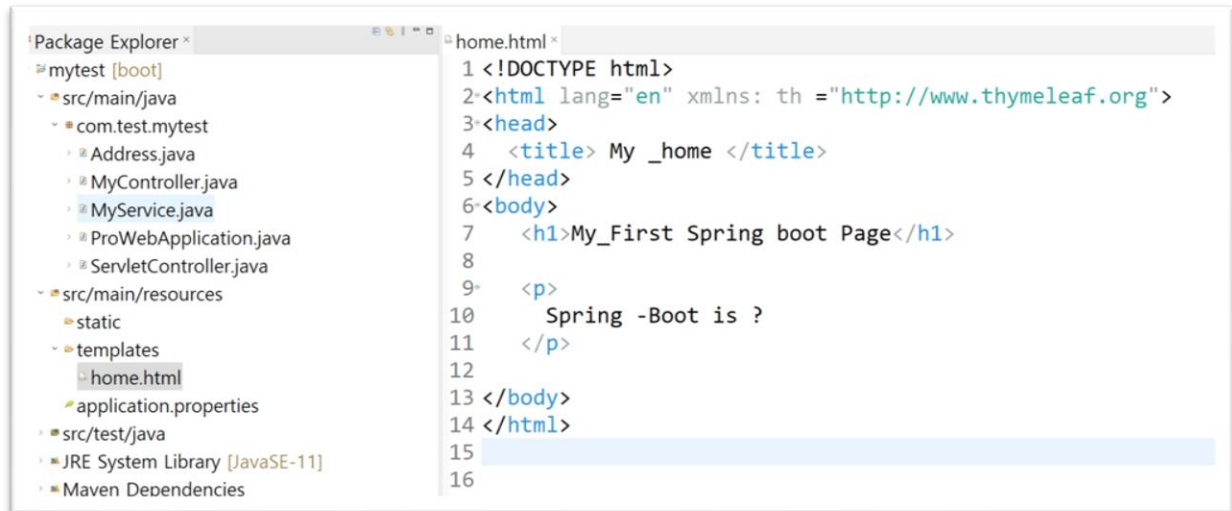
[STEP 06] 실행 후 브라우저 확인

6-1. 만들어진 ProWebApplication.java -> 오.마 -> Run As -> SpringBoot APP 실행