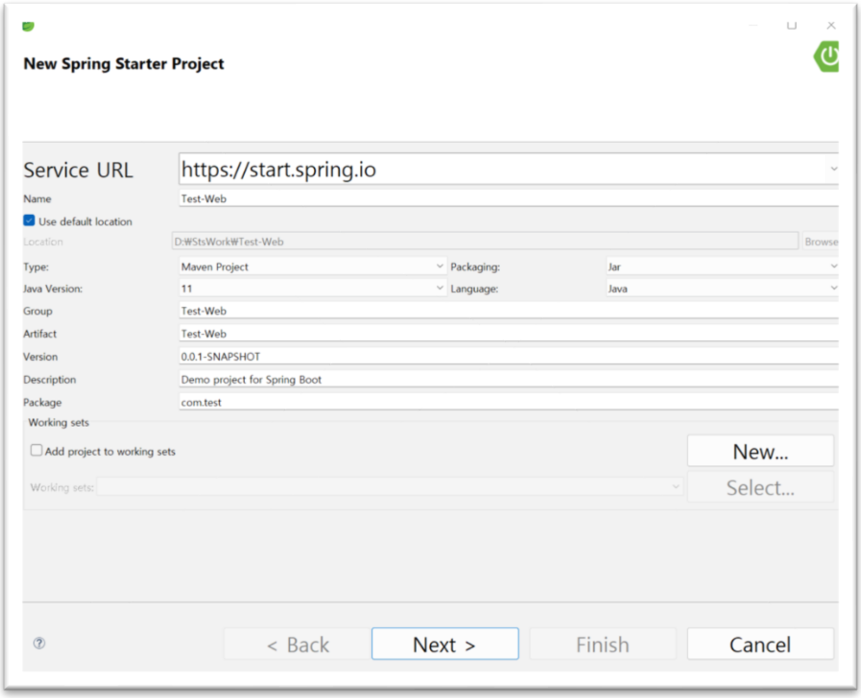
Spring-boot lab02

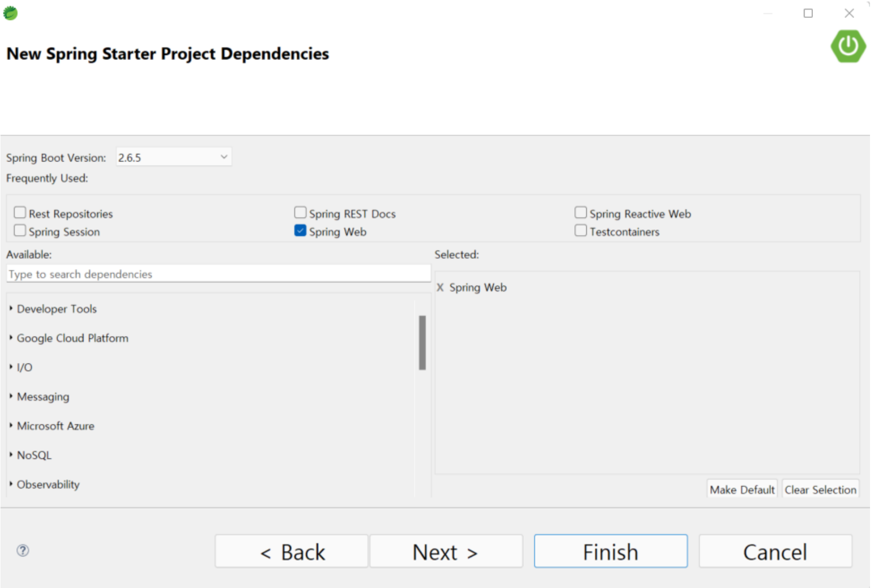
[STEP 01] STS 툴 -> File -> New Spring Starter Project 에서 아래와 같이 추가한 다음

->Next -> Spring Web을 체크 ->Finish



New Spring Starter Project

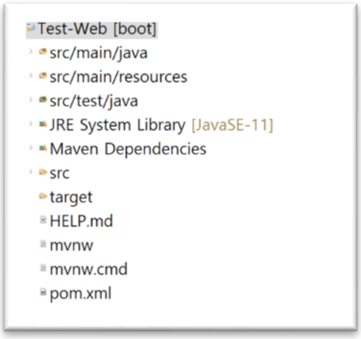
Service URL: https://start.spring.io
Name: Test-Web
☑ Use default location
Location: D:₩StsWork₩Test-Web
Type: Maven Project
Packaging: Jar
Java Version: 11
Language: Java
Group: Test-Web
Artifact: Test-Web
Version: 0.0.1-SNAPSHOT
Description: Demo project for Spring Boot
Package: com.test
Working sets
☐ Add project to working sets
Working sets:



New Spring Starter Project Dependencies

Spring Boot Version: 2.6.5
Frequently Used:
☐ Rest Repositories
☐ Spring Session
☐ Spring REST Docs
☑ Spring Web
☐ Spring Reactive Web
☐ Testcontainers
Available:
Type to search dependencies
‣ Developer Tools
‣ Google Cloud Platform
‣ I/O
‣ Messaging
‣ Microsoft Azure
‣ NoSQL
‣ Observability
Selected:
X Spring Web

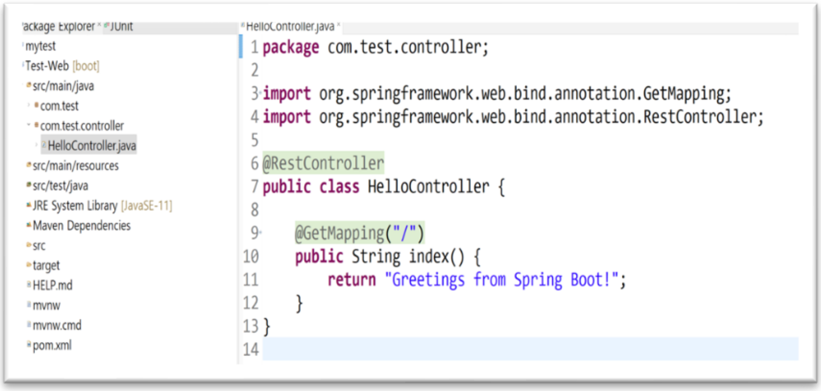
[STEP 02] 완성형 구조 폴더를 확인한다.



```
Test-Web [boot]
  src/main/java
  src/main/resources
  src/test/java
  JRE System Library [JavaSE-11]
  Maven Dependencies
  src
  target
  HELP.md
  mvnw
  mvnw.cmd
  pom.xml
```

[STEP 03] 클래스를 추가한다.

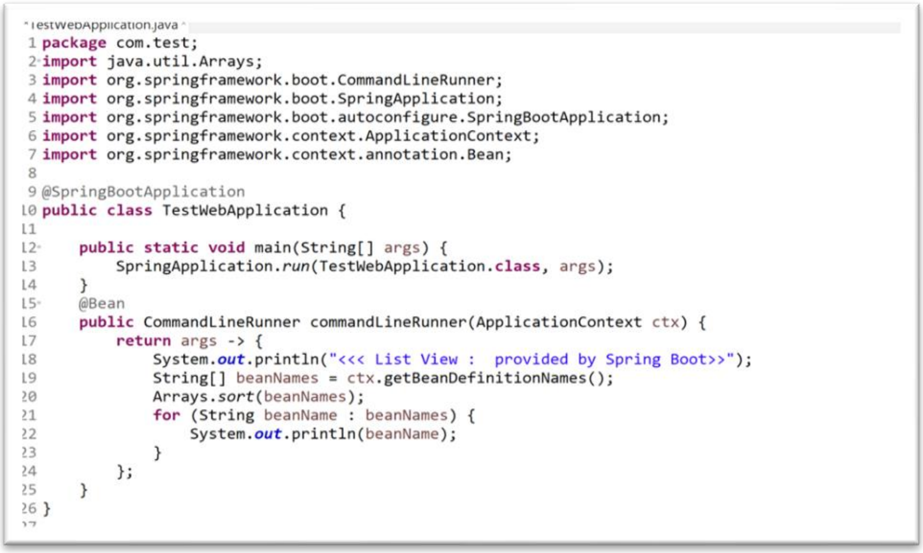
3-1. src/main/java/com/test/controller/HelloController 를 추가한다.



```
mytest
Test-Web [boot]
  src/main/java
    com.test
    com.test.controller
      HelloController.java
  src/main/resources
  src/test/java
  JRE System Library [JavaSE-11]
  Maven Dependencies
  src
  target
  HELP.md
  mvnw
  mvnw.cmd
  pom.xml
```

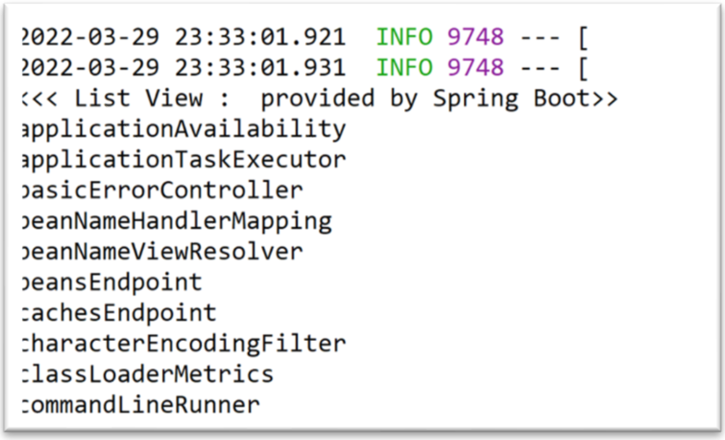
```java
package com.test.controller;

import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RestController;

@RestController
public class HelloController {

    @GetMapping("/")
    public String index() {
        return "Greetings from Spring Boot!";
    }
}
```

3-2 . 이미 만들어진 com.test.TestWebApplication 에 다음과 같이 코드를 추가한다.



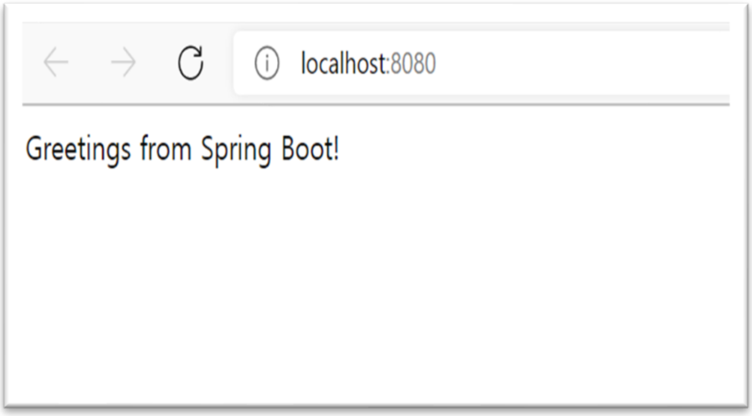
```java
package com.test;
import java.util.Arrays;
import org.springframework.boot.CommandLineRunner;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.context.ApplicationContext;
import org.springframework.context.annotation.Bean;

@SpringBootApplication
public class TestWebApplication {

    public static void main(String[] args) {
        SpringApplication.run(TestWebApplication.class, args);
    }
    @Bean
    public CommandLineRunner commandLineRunner(ApplicationContext ctx) {
        return args -> {
            System.out.println("<<< List View :  provided by Spring Boot>>");
            String[] beanNames = ctx.getBeanDefinitionNames();
            Arrays.sort(beanNames);
            for (String beanName : beanNames) {
                System.out.println(beanName);
            }
        };
    }
}
```

[STEP 04] 프로젝트선택 -> Run As -> Spring Boot app 를 실행한 후 콘솔을 확인 한다.



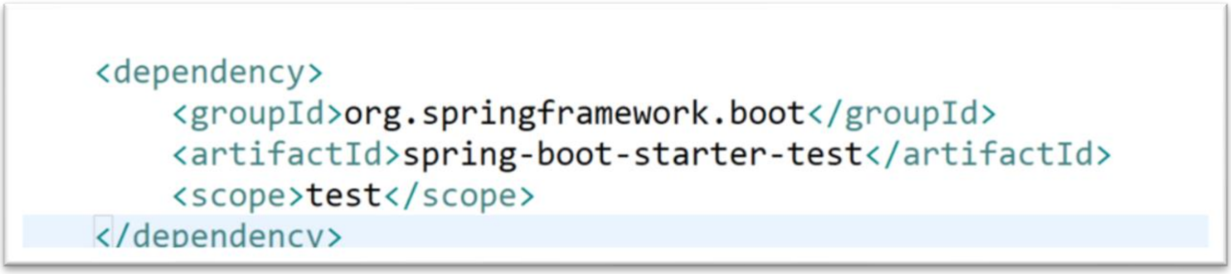
```
2022-03-29 23:33:01.921  INFO 9748 --- [
2022-03-29 23:33:01.931  INFO 9748 --- [
<<< List View :  provided by Spring Boot>>
applicationAvailability
applicationTaskExecutor
basicErrorController
beanNameHandlerMapping
beanNameViewResolver
beansEndpoint
cachesEndpoint
characterEncodingFilter
classLoaderMetrics
commandLineRunner
```

[STEP 05] 브라우저 확인을 한다.



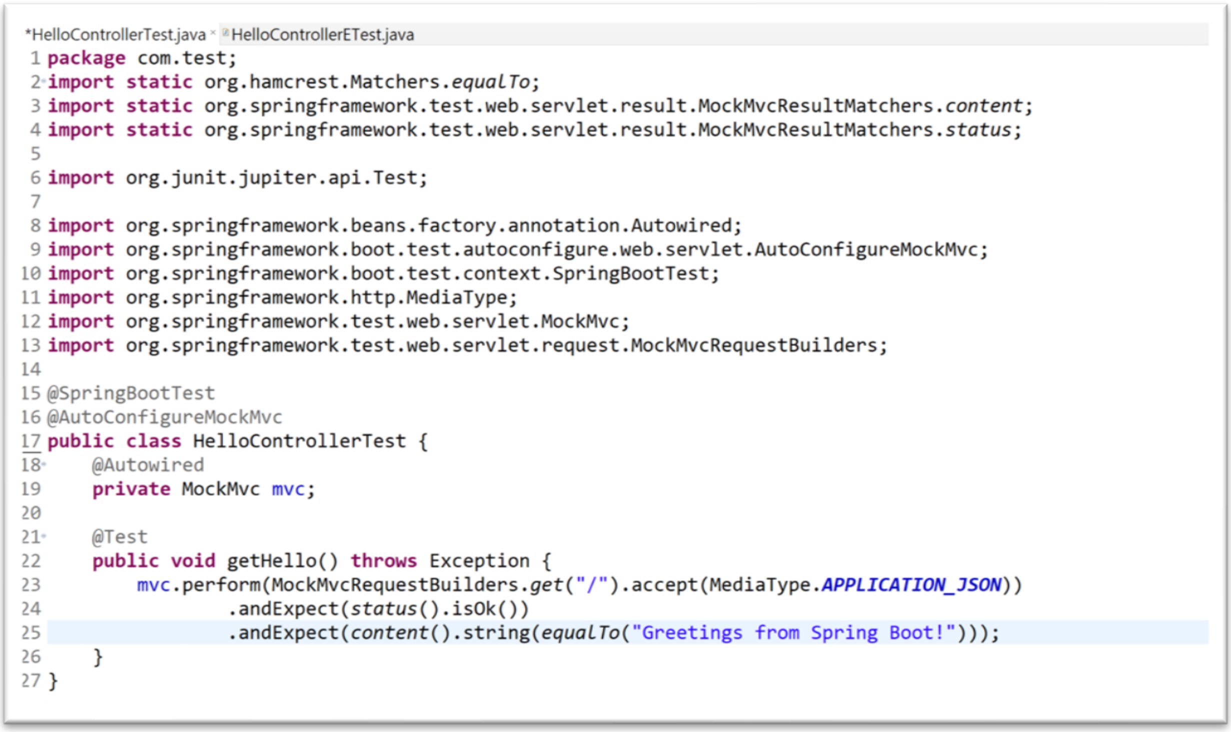
Greetings from Spring Boot!

[STEP 06] 단위 테스트를 추가한다. pom.xml 확인한다.



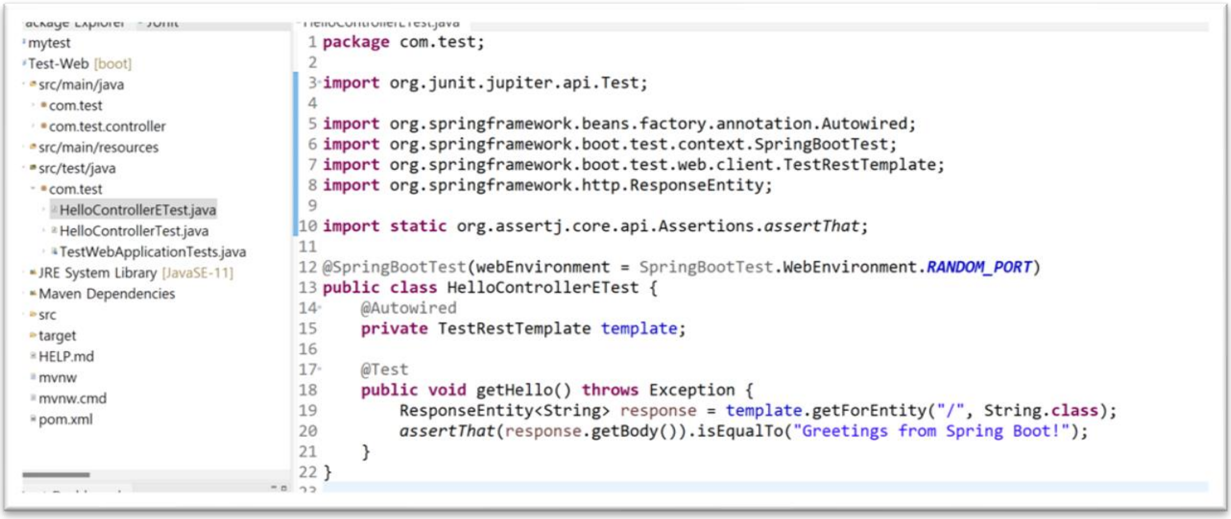
```
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-test</artifactId>
    <scope>test</scope>
</dependency>
```

[STEP 07] 단위 테스트 코드를 src/test/java/com/test/HelloControllerTest로 추가 후 아래와 같이 코딩한다.



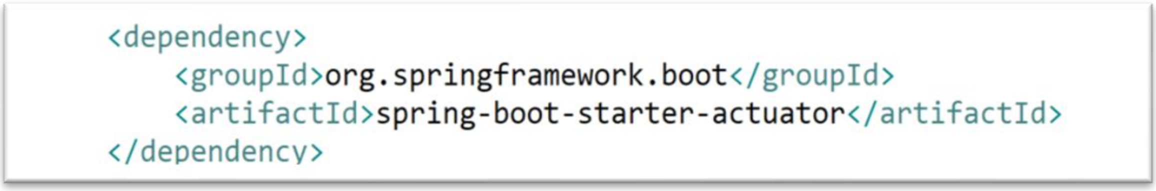
```java
package com.test;
import static org.hamcrest.Matchers.equalTo;
import static org.springframework.test.web.servlet.result.MockMvcResultMatchers.content;
import static org.springframework.test.web.servlet.result.MockMvcResultMatchers.status;

import org.junit.jupiter.api.Test;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.test.autoconfigure.web.servlet.AutoConfigureMockMvc;
import org.springframework.boot.test.context.SpringBootTest;
import org.springframework.http.MediaType;
import org.springframework.test.web.servlet.MockMvc;
import org.springframework.test.web.servlet.request.MockMvcRequestBuilders;

@SpringBootTest
@AutoConfigureMockMvc
public class HelloControllerTest {
    @Autowired
    private MockMvc mvc;

    @Test
    public void getHello() throws Exception {
        mvc.perform(MockMvcRequestBuilders.get("/").accept(MediaType.APPLICATION_JSON))
                .andExpect(status().isOk())
                .andExpect(content().string(equalTo("Greetings from Spring Boot!")));
    }
}
```

[STEP 08] 단위 테스트 코드를 src/test/java/com/test/HelloControllerETest로 추가 후 아래와 같이 코딩한다.



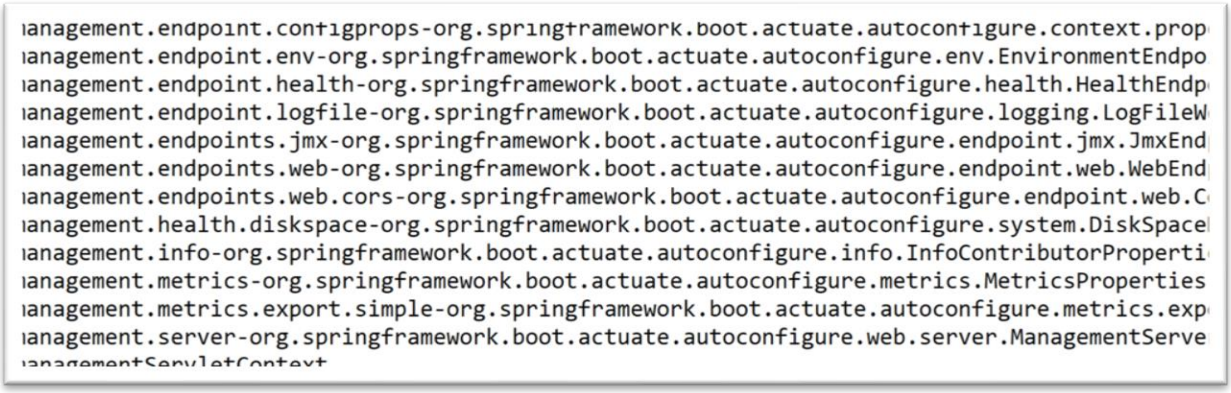
```java
package com.test;

import org.junit.jupiter.api.Test;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.test.context.SpringBootTest;
import org.springframework.boot.test.web.client.TestRestTemplate;
import org.springframework.http.ResponseEntity;

import static org.assertj.core.api.Assertions.assertThat;

@SpringBootTest(webEnvironment = SpringBootTest.WebEnvironment.RANDOM_PORT)
public class HelloControllerETest {
    @Autowired
    private TestRestTemplate template;

    @Test
    public void getHello() throws Exception {
        ResponseEntity<String> response = template.getForEntity("/", String.class);
        assertThat(response.getBody()).isEqualTo("Greetings from Spring Boot!");
    }
}
```

[STEP 09] pom.xml 아래와 같이 추가한다.



```
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-actuator</artifactId>
</dependency>
```

[STEP 10] 프로젝트선택 -> Run As -> Spring Boot app 를 실행한 후 콘솔을 확인 한다.



```
ıanagement.endpoint.configprops-org.springframework.boot.actuate.autoconfigure.context.prop
ıanagement.endpoint.env-org.springframework.boot.actuate.autoconfigure.env.EnvironmentEndpo
ıanagement.endpoint.health-org.springframework.boot.actuate.autoconfigure.health.HealthEndp
ıanagement.endpoint.logfile-org.springframework.boot.actuate.autoconfigure.logging.LogFileW
ıanagement.endpoints.jmx-org.springframework.boot.actuate.autoconfigure.endpoint.jmx.JmxEnd
ıanagement.endpoints.web-org.springframework.boot.actuate.autoconfigure.endpoint.web.WebEnd
ıanagement.endpoints.web.cors-org.springframework.boot.actuate.autoconfigure.endpoint.web.C
ıanagement.health.diskspace-org.springframework.boot.actuate.autoconfigure.system.DiskSpace
ıanagement.info-org.springframework.boot.actuate.autoconfigure.info.InfoContributorProperti
ıanagement.metrics-org.springframework.boot.actuate.autoconfigure.metrics.MetricsProperties
ıanagement.metrics.export.simple-org.springframework.boot.actuate.autoconfigure.metrics.exp
ıanagement.server-org.springframework.boot.actuate.autoconfigure.web.server.ManagementServe
ıanagementServletContext
```

[STEP 11] CMD 창을 열어 응용프로그램 상태를 확인한다.

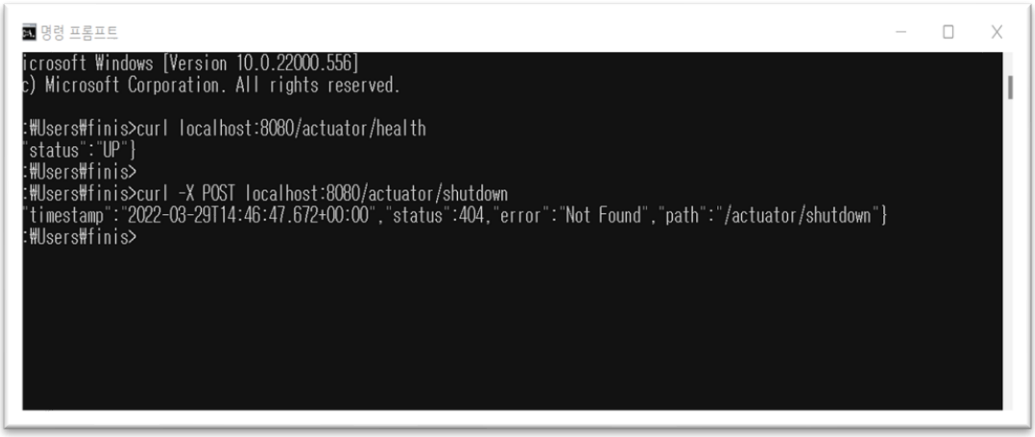
명령과 결과가 다음과 같이 나온다.

C:₩ >curl localhost:8080/actuator/health

{"status":"UP"}

C:₩ >curl -X POST localhost:8080/actuator/shutdown

{"timestamp":"2022-03-29T14:46:47.672+00:00","status":404,"error":"Not Found","path":"/actuator/shutdown"}



```
icrosoft Windows [Version 10.0.22000.556]
c) Microsoft Corporation. All rights reserved.

:₩Users₩finis>curl localhost:8080/actuator/health
"status":"UP"}
:₩Users₩finis>
:₩Users₩finis>curl -X POST localhost:8080/actuator/shutdown
"timestamp":"2022-03-29T14:46:47.672+00:00","status":404,"error":"Not Found","path":"/actuator/shutdown"}
:₩Users₩finis>
```