

Spring-boot lab04

[STEP 01] 주어진 정보를 이용해서 프로젝트를 생성한다.

1-1 : STS 툴 -> File -> New Spring Starter Project 에서 아래와 같은 정보로 추가한 다음 프로젝트를 생성한다.

```
<groupId>com.myjpa</groupId>
<artifactId>Test_JPA</artifactId>
<version>0.0.1-SNAPSHOT</version>
<name>Test_JPA</name>
<description>Test project for Spring Boot-JPA</description>
<properties>
  <java.version>11</java.version>
</properties>
```

1-2 : pom.xml에서 추가되는 정보는 다음과 같다. [h2 , validation, jpa 추가 다시 확인]

```
Dependencies
spring-boot-starter-data-jdbc (managed:2.6.5)
spring-boot-starter-thymeleaf (managed:2.6.5)
spring-boot-starter-web (managed:2.6.5)
h2 [runtime] (managed:1.4.200)
mysql-connector-java [runtime] (managed:8.0.28)
lombok (managed:1.18.22)
spring-boot-starter-test [test] (managed:2.6.5)
spring-boot-starter-data-jpa (managed:2.6.5)
spring-boot-starter-validation (managed:2.6.5)
```

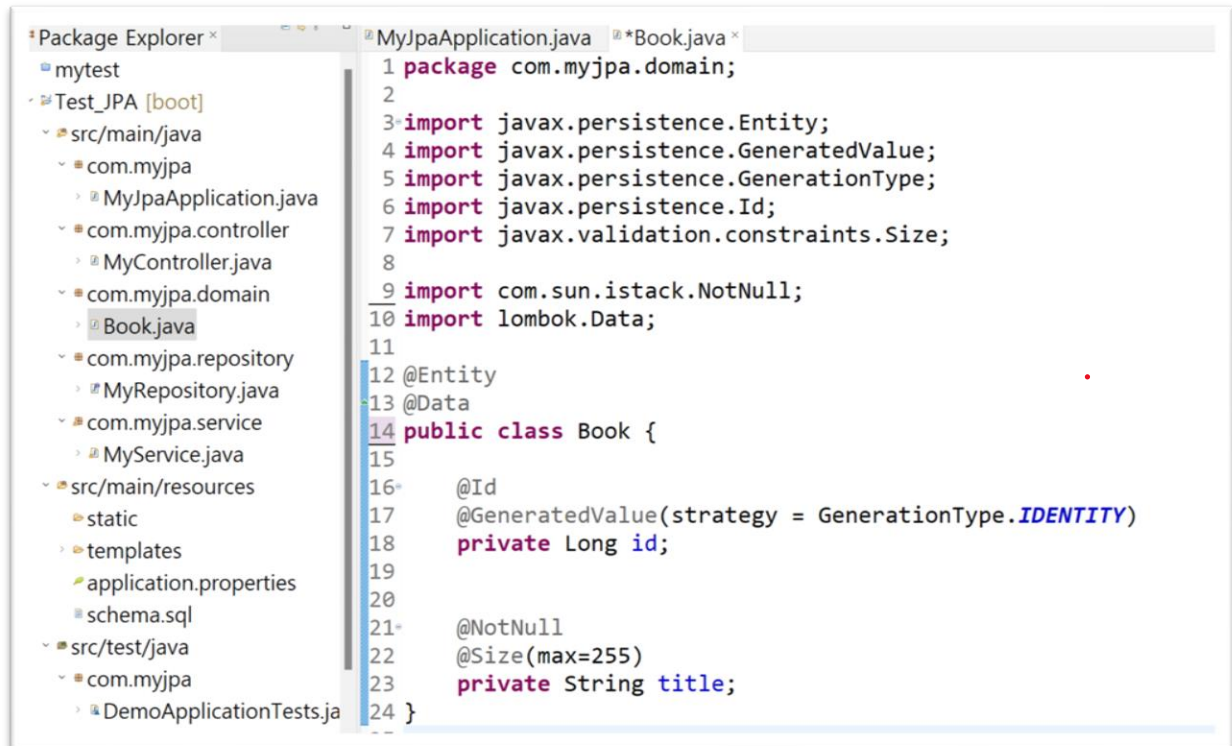
[STEP 02] 완성형 구조 폴더를 확인한다. _패키지와 추가 클래스 및 파일을 생성한다.

```
Test_JPA [boot]
├── src/main/java
│   ├── com.myjpa
│   │   ├── MyJpaApplication.java
│   │   ├── com.myjpa.controller
│   │   │   ├── MyController.java
│   │   ├── com.myjpa.domain
│   │   │   ├── Book.java
│   │   ├── com.myjpa.repository
│   │   │   ├── MyRepository.java
│   │   ├── com.myjpa.service
│   │   │   ├── MyService.java
│   └── src/main/resources
│       ├── static
│       ├── templates
│       ├── application.properties
│       └── schema.sql
```

[STEP 03] 클래스를 추가한다.

3-1. 클래스를 추가한다. /src/main/java/com/myjpa/MyJpaApplication.java는 main있는 클래스로 생성하지 않는다. 이름이 다르게 생겨도 상관없다.

도메인 Book 클래스에 다음과 같이 코드를 추가한다. [임포트 주의한다]



```
1 package com.myjpa.domain;
2
3 import javax.persistence.Entity;
4 import javax.persistence.GeneratedValue;
5 import javax.persistence.GenerationType;
6 import javax.persistence.Id;
7 import javax.validation.constraints.Size;
8
9 import com.sun.istack.NotNull;
10 import lombok.Data;
11
12 @Entity
13 @Data
14 public class Book {
15
16     @Id
17     @GeneratedValue(strategy = GenerationType.IDENTITY)
18     private Long id;
19
20
21     @NotNull
22     @Size(max=255)
23     private String title;
24 }
```

3-2. MyRepository.java 작성한다.



```
1 package com.myjpa.repository;
2
3 import org.springframework.data.jpa.repository.JpaRepository;
4 import org.springframework.stereotype.Repository;
5
6 import com.myjpa.domain.Book;
7
8
9
10 @Repository
11 public interface MyRepository extends JpaRepository<Book, Long>{
12
13 }
14
15
16
```

3-3. MyController.java 작성한다.



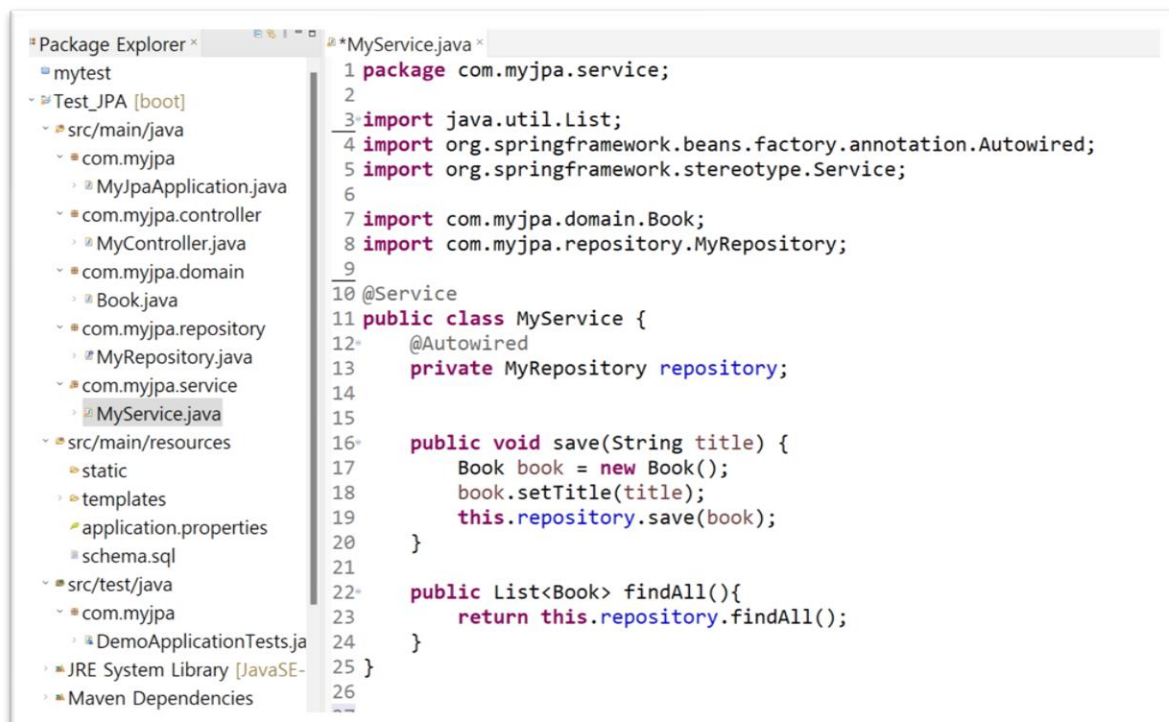
The screenshot shows an IDE with the Package Explorer on the left and the code editor on the right. The Package Explorer shows the project structure with the following packages and files:

- mytest
 - Test_JPA [boot]
 - src/main/java
 - com.myjpa
 - MyJpaApplication.java
 - MyController.java
 - MyRepository.java
 - MyService.java
 - src/main/resources
 - static
 - templates
 - application.properties
 - schema.sql
 - src/test/java
 - com.myjpa
 - DemoApplicationTests.java
 - JRE System Library [JavaSE-8]

The code editor shows the code for MyController.java:

```
1 package com.myjpa.controller;
2
3 import java.util.Objects;
4 import org.springframework.beans.factory.annotation.Autowired;
5 import org.springframework.stereotype.Controller;
6 import org.springframework.ui.Model;
7 import org.springframework.web.bind.annotation.RequestMapping;
8 import org.springframework.web.bind.annotation.RequestParam;
9 import com.myjpa.service.MyService;
10
11 @Controller
12 public class MyController {
13     @Autowired
14     private MyService service;
15
16     @RequestMapping
17     public String index(@RequestParam(required = false) String title, Model model) {
18
19         if (!Objects.isNull(title) && !title.isBlank()) {
20             this.service.save(title);
21         }
22         model.addAttribute("books", this.service.findAll());
23         return "index";
24     }
25 }
```

3-4. MyService.java 작성한다.



The screenshot shows an IDE with the Package Explorer on the left and the code editor on the right. The Package Explorer shows the project structure with the following packages and files:

- mytest
 - Test_JPA [boot]
 - src/main/java
 - com.myjpa
 - MyJpaApplication.java
 - MyController.java
 - MyRepository.java
 - MyService.java
 - src/main/resources
 - static
 - templates
 - application.properties
 - schema.sql
 - src/test/java
 - com.myjpa
 - DemoApplicationTests.java
 - JRE System Library [JavaSE-8]
 - Maven Dependencies

The code editor shows the code for MyService.java:

```
1 package com.myjpa.service;
2
3 import java.util.List;
4 import org.springframework.beans.factory.annotation.Autowired;
5 import org.springframework.stereotype.Service;
6
7 import com.myjpa.domain.Book;
8 import com.myjpa.repository.MyRepository;
9
10 @Service
11 public class MyService {
12     @Autowired
13     private MyRepository repository;
14
15     public void save(String title) {
16         Book book = new Book();
17         book.setTitle(title);
18         this.repository.save(book);
19     }
20
21     public List<Book> findAll(){
22         return this.repository.findAll();
23     }
24 }
25
26
```

3-5. schema.sql를 작성해서 실행한다. application.properties는 이전 속성파일과 같이 작성한다.

```
CREATE TABLE BOOK (  
    ID int auto_increment PRIMARY KEY,  
    TITLE VARCHAR(255) NOT NULL  
);  
INSERT INTO BOOK(TITLE) VALUES ('Getting Start Java');  
INSERT INTO BOOK(TITLE) VALUES ('Getting String Spring');  
INSERT INTO BOOK(TITLE) VALUES ('Getting String Web');
```

3-6. index.html 은 다음과 같이 코딩한다.

```
index.html <  
1 <!DOCTYPE html>  
2 <html xmlns:th="http://www.thymeleaf.org">  
3 <head>  
4 <meta charset="UTF-8">  
5 <title></title>  
6 <link  
7     href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.1/dist/css/bootstrap.min.css"  
8     rel="stylesheet">  
9 </head>  
10 <body>  
11     <section class="border p-1 m-3">  
12         <form method="post" th:action="@{/}">  
13             <div class="mb-3">  
14                 <label for="title" class="form-label">title</label> <input  
15                     type="text" class="form-control" name="title" id="title">  
16             </div>  
17             <div>  
18                 <button type="submit" class="btn btn-primary">확인</button>  
19             </div>  
20         </form>  
21     </section>  
22     <main class="border p-1 m-3">  
23         <h1> 목록</h1>  
24         <ul class="list-group">  
25             <li class="list-group-item" th:each="book : ${books}"  
26                 th:text="${book.title}">  
27         </ul>  
28     </main>  
29 </body>  
30 </html>
```

[STEP 04] 브라우저 확인을 한다.

