

# Web API Design with Spring Boot Week 15 Coding Assignment

**Points possible: 75**

**URL to GitHub Repository:** <https://github.com/oscarc257/SpringBoot-Assignments.git>


**URL to Public Link of your Video:**

<https://www.dropbox.com/s/5ptbf0wkc3gaeef/Week%2015%20Assignment.mp4?dl=0>


---

## Instructions :

1. Follow the **Coding Steps** below to complete this assignment.

- In Spring Tool Suite (STS), or an IDE of your choice, write the code that accomplishes the objectives listed below. Ensure that the code compiles and runs as directed.
- Use your existing repo or create a new repository on GitHub for this week's assignment and push your completed code to the repo, including your entire Maven Project Directory (e.g., jeep-sales) and any additional files (e.g. .sql files) that you create. In addition, screenshot your ERD and push the screenshot to your GitHub repo.
- Include the screenshots into this Assignment Document indicated by: 
- Create a video showcasing your work:
  - In this video: record and present your project verbally while showing the results of the working project.
  - Easy way to Create a video: Start a meeting in Zoom, share your screen, open Eclipse with the code and your Console window, start recording & record yourself describing and running the program showing the results.
  - Your video should be a maximum of 5 minutes.
  - Upload your video with a public link.
  - Easy way to Create a Public Video Link: Upload your video recording to YouTube with a public link.

2. In addition, please include the following in your Coding Assignment Document:


- The requested screenshots, indicated by: 
- The URL for this week's GitHub repository.
- The URL of the public link of your video.

3. Save the Coding Assignment Document as a .pdf and do the following:

- Push the .pdf to the GitHub repo for this week.
- Upload the .pdf to the LMS in your Coding Assignment Submission.

# Web API Design with Spring Boot Week 15 Coding Assignment

---

**Here's a friendly tip:** as you watch the videos, code along with the videos. This will help you with the homework. When a screenshot is required, look for the icon:  You will keep adding to this project throughout this part of the course. When it comes time for the final project, use this project as a starter.



**Project Resources:** <https://github.com/promineotech/Spring-Boot-Course-Student-Resources>

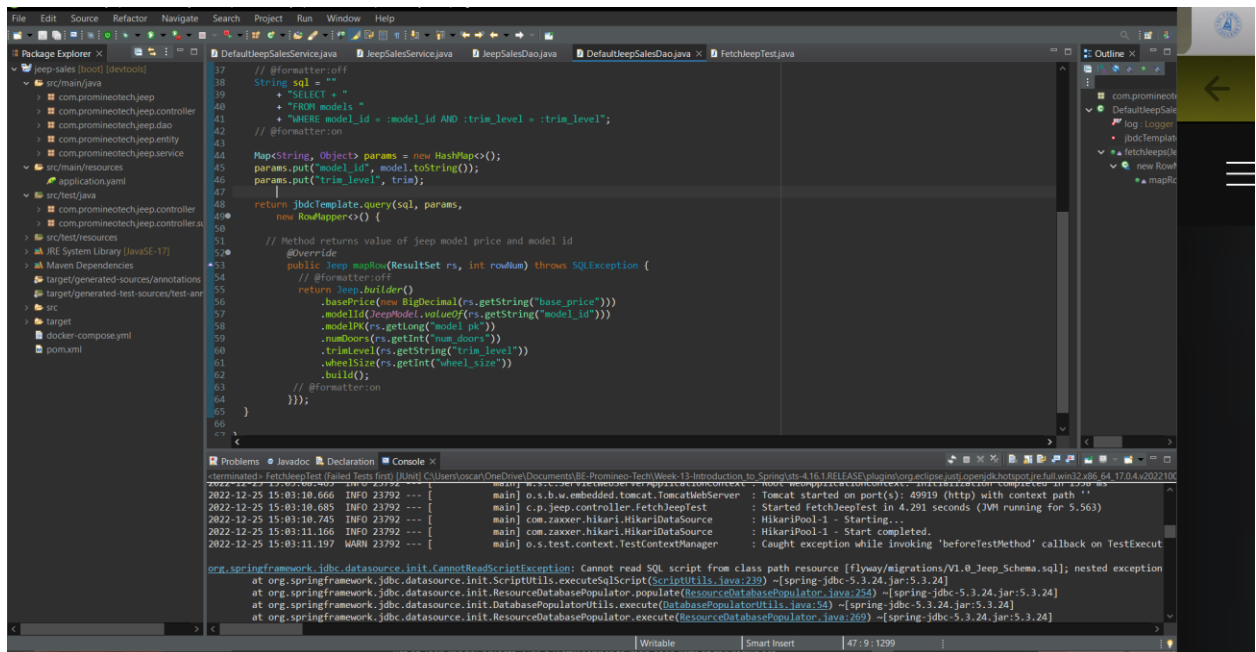
## Coding Steps:

- 1) In the application you've been building add a DAO layer:
  - a) Add the package, `com.promineotech.jeepp.dao`.
  - b) In the new package, create an interface named `JeepSalesDao`.
  - c) In the same package, create a class named `DefaultJeepSalesDao` that implements `JeepSalesDao`.
  - d) Add a method in the DAO interface and implementation that returns a list of Jeep models (class `Jeep`) and takes the model and trim parameters. Here is the method signature:

```
List<Jeep> fetchJeeps(JeepModel model, String trim);
```
- 2) In the Jeep sales service implementation class, inject the DAO interface as an instance variable. The instance variable should be private and should be named `jeepSalesDao`. Call the DAO method from the service method and store the returned value in a local variable named `jeeps`. Return the value in the `jeeps` variable (we will add to this later).

# Web API Design with Spring Boot Week 15 Coding Assignment


- 3) In the DAO implementation class (DefaultJeepSalesDao):
  - a) Add the class-level annotation: `@Service`.
  - b) Add a log statement in `DefaultJeepSalesDao.fetchJeeps()` that logs the model and trim level. Run the integration test. Produce a screenshot showing the DAO implementation class and the log line in the IDE's console. 
  - c) In `DefaultJeepSalesDao`, inject an instance variable of type `NamedParameterJdbcTemplate`.
  - d) Write SQL to return a list of Jeep models based on the parameters: model and trim. Be sure to utilize the SQL Injection prevention mechanism of the `NamedParameterJdbcTemplate` using `:model_id` and `:trim_level` in the query.
  - e) Add the parameters to a parameter map as shown in the video. Don't forget to convert the `JeepModel` enum value to a String (i.e., `params.put("model_id", model.toString());`)
  - f) Call the query method on the `NamedParameterJdbcTemplate` instance variable to return a list of Jeep model objects. Use a `RowMapper` to map each row of the result set. Remember to convert `modelId` to a `JeepModel`. See the video for details. Produce a screenshot to show the complete method in the implementation class. 

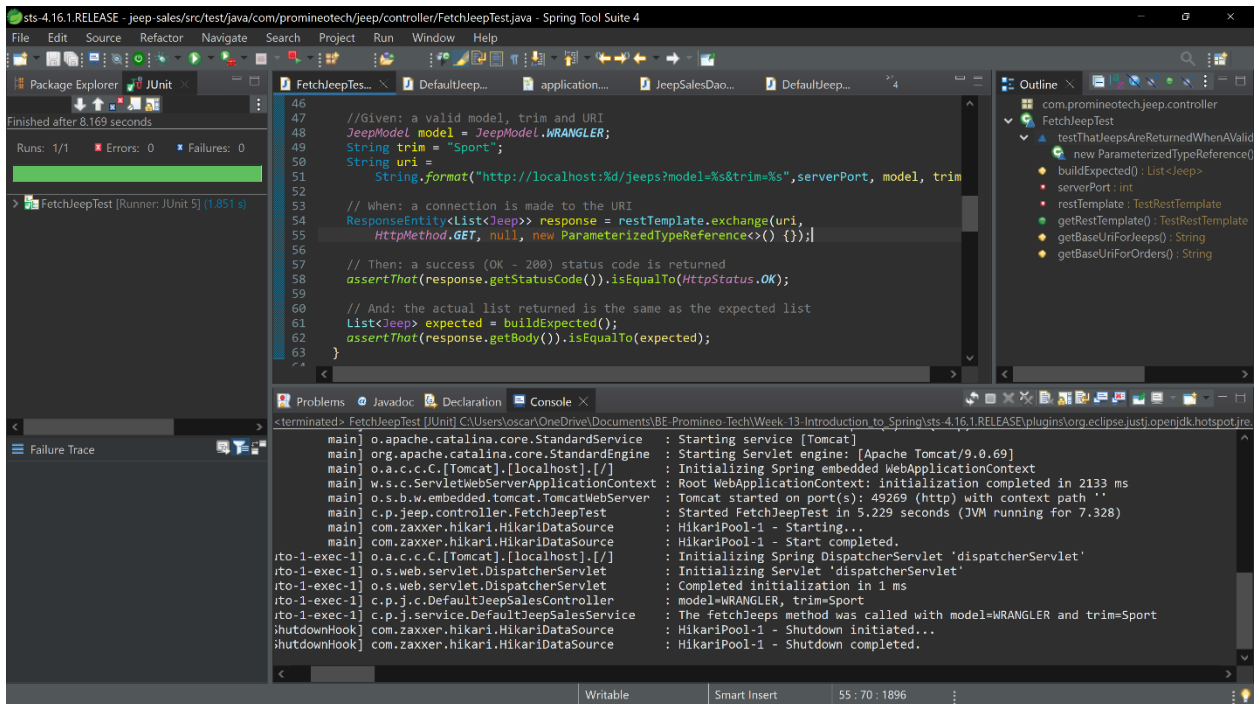


The screenshot shows an IDE with the `DefaultJeepSalesDao.java` file open. The code implements the `fetchJeeps()` method, which uses `NamedParameterJdbcTemplate` to execute a SQL query. The query is: `SELECT * FROM models WHERE model_id = :model_id AND :trim_level = :trim_level;`. The parameters are added to a `Map` and passed to the `query()` method. The result is mapped using a `RowMapper` that creates `Jeep` objects. The console output shows the following log lines:

```
2022-12-25 15:03:10.666 INFO 23792 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port(s): 49919 (http) with context path ''
2022-12-25 15:03:10.665 INFO 23792 --- [main] c.p.jee.controller.FetchJeepTest : Started FetchJeepTest in 4.291 seconds (JVM running for 5.563)
2022-12-25 15:03:10.745 INFO 23792 --- [main] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Starting...
2022-12-25 15:03:11.106 INFO 23792 --- [main] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Start completed.
2022-12-25 15:03:11.197 WARN 23792 --- [main] o.s.test.context.TestContextManager : Caught exception while invoking 'beforeTestMethod' callback on TestExecut...
```

# Web API Design with Spring Boot Week 15 Coding Assignment

- 4) Add a getter in the Jeep class for modelPK. Add the @JsonIgnore annotation to the getter to exclude the modelPK value from the returned object.
- 5) Run the test to produce a green status bar. Produce a screenshot showing the test and the green status bar. 



```
sts-4.16.1.RELEASE - jeep-sales/src/test/java/com/promineotech/jeep/controller/JeepControllerTest.java - Spring Tool Suite 4
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer JUnit
Finished after 8.169 seconds
Runs: 1/1 Errors: 0 Failures: 0
FetchJeepTest (Runner: JUnit 5) (1.851 s)
46 //Given: a valid model, trim and URI
47 JeepModel model = JeepModel.WRANGLER;
48 String trim = "Sport";
49 String uri =
50 String.format("http://localhost:%d/jeeps?model=%s&trim=%s",serverPort, model, trim
51
52 // When: a connection is made to the URI
53 ResponseEntity<List<Jeep>> response = restTemplate.exchange(uri,
54 HttpMethod.GET, null, new ParameterizedTypeReference<>() {});
55
56 // Then: a success (OK - 200) status code is returned
57 assertEquals(response.getStatusCode(), HttpStatus.OK);
58
59 // And: the actual list returned is the same as the expected list
60 List<Jeep> expected = buildExpected();
61 assertEquals(response.getBody(), expected);
62
63
Problems Javadoc Declaration Console
terminated> FetchJeepTest [JUnit] C:\Users\oscar\OneDrive\Documents\BE-Promineo-Tech\Week-13-Introduction to Spring\sts-4.16.1.RELEASE\plugins\org.eclipse.justi.openjdk.hotspot.jre
main] o.apache.catalina.core.StandardService : Starting service [Tomcat]
main] org.apache.catalina.core.StandardEngine : Starting Servlet engine: [Apache Tomcat/9.0.69]
main] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring embedded WebApplicationContext
main] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization completed in 2133 ms
main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port(s): 49269 (http) with context path ''
main] c.p.jee.controller.FetchJeepTest : Started FetchJeepTest in 5.229 seconds (JVM running for 7.328)
main] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Starting...
main] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Start completed.
ito-1-exec-1] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring DispatcherServlet 'dispatcherServlet'
ito-1-exec-1] o.s.web.servlet.DispatcherServlet : Initializing Servlet 'dispatcherServlet'
ito-1-exec-1] o.s.web.servlet.DispatcherServlet : Completed initialization in 1 ms
ito-1-exec-1] c.p.j.c.DefaultJeepSalesController : model=WRANGLER, trim=Sport
ito-1-exec-1] c.p.j.service.DefaultJeepSalesService : The fetchJeeps method was called with model=WRANGLER and trim=Sport
shutdownHook] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Shutdown initiated...
shutdownHook] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Shutdown completed.
```