

Web API Design with Spring Boot Week 15 Coding Assignment


Points possible: 75

URL to GitHub Repository: <https://github.com/Anghel-Valdehueza/Jeep-Sales.git>


URL to Public Link of your Video: <https://youtu.be/oTeHMFp8NtQ>

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.

```
32 public List<Jeep> fetchJeeps(JeepModel model, String trim) {
33     Log.debug("DAO: model={}, trim={}", model, trim);
34
35     // @formatter:off
36     String sql = ""
37         + "SELECT * "
38         + "FROM models "
39         + "WHERE model_id = :model_id AND trim_level = :trim_level";
40     // @formatter:on
```

```
07:13:39.380 [main] DEBUG org.springframework.test.context.BootstrapUtils - Instantiating CacheAwareContextLoaderDelegate from class [org.springframework.test.context.cache.DefaultCacheAwareContextLoaderDeleg
07:13:39.453 [main] DEBUG org.springframework.test.context.BootstrapUtils - Instantiating BootstrapContext using constructor [public org.springframework.test.context.support.DefaultBootstrapContext(java.lang.
07:13:39.749 [main] DEBUG org.springframework.test.context.BootstrapUtils - Instantiating TestContextBootstrapper for test class [com.promineotech.jeeptest.controller.FetchJeepTest] from class [org.springframework
07:13:39.792 [main] INFO org.springframework.test.context.SpringBootTestContextBootstrapper - Neither @ContextConfiguration nor @ContextHierarchy found for test class [com.promineotech.jeeptest.controller.FetchJeep
07:13:39.889 [main] DEBUG org.springframework.test.context.support.AbstractContextLoader - Did not detect default resource location for test class [com.promineotech.jeeptest.controller.FetchJeepTest]: class path
07:13:39.812 [main] DEBUG org.springframework.test.context.support.AbstractContextLoader - Did not detect default resource location for test class [com.promineotech.jeeptest.controller.FetchJeepTest]: class path
07:13:39.812 [main] INFO org.springframework.test.context.support.AbstractContextLoader - Could not detect default resource locations for test class [com.promineotech.jeeptest.controller.FetchJeepTest]: no resour
07:13:39.815 [main] INFO org.springframework.test.context.support.AnnotationConfigContextLoaderUtils - Could not detect default configuration classes for test class [com.promineotech.jeeptest.controller.FetchJeep
07:13:40.612 [main] DEBUG org.springframework.test.context.annotation.ClassPathScanningCandidateComponentProvider - Identified candidate component class: file [C:\Users\angva\Documents\workspace-spring-tool-suite-
07:13:40.615 [main] INFO org.springframework.test.context.SpringBootTestContextBootstrapper - Found @SpringBootConfiguration com.promineotech.jeeptest.Jeeptest for test class com.promineotech.jeeptest.controller
07:13:41.283 [main] DEBUG org.springframework.test.context.SpringBootTestContextBootstrapper - @TestExecutionListeners is not present for class [com.promineotech.jeeptest.controller.FetchJeepTest]: using def
07:13:41.284 [main] INFO org.springframework.test.context.SpringBootTestContextBootstrapper - Loaded default TestExecutionListener class names from location [META-INF/spring.factories]: [org.springframework
07:13:41.284 [main] INFO org.springframework.test.context.SpringBootTestContextBootstrapper - Using TestExecutionListeners: [org.springframework.test.context.web.ServletTestExecutionListener@7d343897, or
07:13:41.268 [main] DEBUG org.springframework.test.context.support.AbstractDirtiesContextTestExecutionListener - Before test class: context [DefaultTestContext@6731787b testClass = FetchJeepTest, testInstance
07:13:41.317 [main] DEBUG org.springframework.test.context.support.DependencyInjectionTestExecutionListener - Performing dependency injection for test context [[DefaultTestContext@6731787b testClass = FetchJeepTest, testInstance
```

```
Spring
=====
:: Spring Boot ::
(v2.7.6)

2023-01-06 07:13:43.079 INFO 16472 --- [main] c.p.jeeptest.controller.FetchJeepTest : Starting FetchJeepTest using Java 17.0.4.1 on DESKTOP-3FHBVAB with PID 16472 (started by angva in C:\Users\
2023-01-06 07:13:43.081 INFO 16472 --- [main] c.p.jeeptest.controller.FetchJeepTest : The following 1 profile is active: "test"
2023-01-06 07:13:50.786 INFO 16472 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat initialized with port(s): 0 (http)
2023-01-06 07:13:50.837 INFO 16472 --- [main] o.apache.catalina.core.StandardService : Starting service [Tomcat]
2023-01-06 07:13:50.838 INFO 16472 --- [main] org.apache.catalina.core.StandardEngine : Starting Servlet engine: [Apache Tomcat/9.0.69]
2023-01-06 07:13:51.524 INFO 16472 --- [main] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring embedded WebApplicationContext
2023-01-06 07:13:51.524 INFO 16472 --- [main] s.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization completed in 8351 ms
2023-01-06 07:14:04.497 INFO 16472 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port(s): 62632 (http) with context path ''
2023-01-06 07:14:04.582 INFO 16472 --- [main] c.p.jeeptest.controller.FetchJeepTest : Started FetchJeepTest in 23.092 seconds (JVM running for 29.599)
2023-01-06 07:14:04.774 INFO 16472 --- [main] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Starting...
2023-01-06 07:14:06.083 INFO 16472 --- [main] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Start completed.
2023-01-06 07:14:10.349 INFO 16472 --- [o-auto-1-exec-1] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring DispatcherServlet 'dispatcherServlet'
2023-01-06 07:14:10.349 INFO 16472 --- [o-auto-1-exec-1] o.s.web.servlet.DispatcherServlet : Initializing Servlet 'dispatcherServlet'
2023-01-06 07:14:10.351 INFO 16472 --- [o-auto-1-exec-1] o.s.web.servlet.DispatcherServlet : Completed initialization in 2 ms
2023-01-06 07:14:10.513 INFO 16472 --- [o-auto-1-exec-1] c.p.jeeptest.service.DefaultJeepSalesService : Received a Request in Jeep Sales service with model WRANGLER, trim Sport
2023-01-06 07:14:11.885 INFO 16472 --- [ionShutdownHook] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Shutdown initiated...
2023-01-06 07:14:11.900 INFO 16472 --- [ionShutdownHook] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Shutdown completed.
```

```
Problems | Javadoc | Declaration | Console | Progress
-----
2023-01-11 05:12:29.587 INFO 17656 --- [o-auto-1-exec-1] c.p.jeeptest.service.DefaultJeepSalesService : Received a Request in Jeep Sales service with model WRANGLER, trim Sport
2023-01-11 05:12:30.611 INFO 17656 --- [ionShutdownHook] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Shutdown initiated...
2023-01-11 05:12:30.616 INFO 17656 --- [ionShutdownHook] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Shutdown completed.

[INFO] Results:
[INFO] Tests run: 1, Failures: 0, Errors: 0, Skipped: 0
[INFO] BUILD SUCCESS
[INFO] Total time: 36.718 s
[INFO] Finished at: 2023-01-11T05:12:31-06:00
[INFO]
```

- c) In DefaultJeepSalesDao, inject an instance variable of type NamedParameterJdbcTemplate.

Web API Design with Spring Boot Week 15 Coding Assignment

- 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. 🖥️

```
32 public List<Jeep> fetchJeeps(JeepModel model, String trim) {
33     Log.debug("DAO: model={}, trim={}", model, trim);
34
35     // @formatter:off
36     String sql = ""
37         + "SELECT * "
38         + "FROM models "
39         + "WHERE model_id = :model_id AND trim_level = :trim_level";
40     // @formatter:on
41
42     Map<String, Object> params = new HashMap<>();
43     params.put("model_id", model.toString());
44     params.put("trim_level", trim);
45     return jdbcTemplate.query(sql, params, new RowMapper<>() {
46         @Override
47         public Jeep mapRow(ResultSet rs, int rowNum) throws SQLException {
48             // @formatter:off
49             return Jeep.builder()
50                 .basePrice(new BigDecimal(rs.getString("base_price")))
51                 .modelId(JeepModel.valueOf(rs.getString("model_id")))
52                 .modelPK(rs.getLong("model_pk"))
53                 .numDoors(rs.getInt("num_doors"))
54                 .trimLevel(rs.getString("trim_level"))
55                 .wheelSize(rs.getInt("wheel_size"))
56                 .build();
57             // @formatter:on
58         }
59     });
60 }
```

- 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. 🖥️

Web API Design with Spring Boot Week 15 Coding Assignment

