

Web API Design with Spring Boot Week 15 Coding Assignment


Points possible: 75

URL to GitHub Repository: <https://github.com/hmackowski/SpringBootWeek3JeepSales->


URL to Public Link of your Video: <https://youtu.be/2LDqA42iDzc>

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>

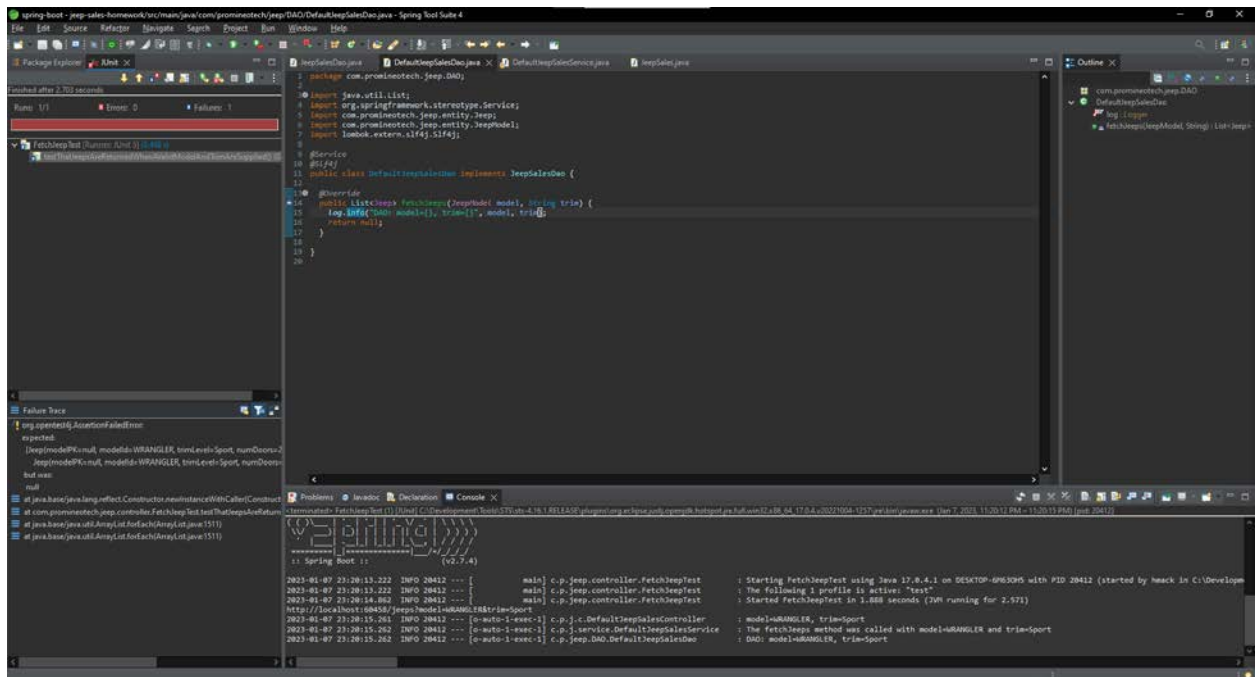
Coding Steps:


- 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. 

Web API Design with Spring Boot Week 15 Coding Assignment

```
15 import lombok.extern.slf4j.Slf4j;
16
17 @Service
18 @Slf4j
19 public class DefaultJeepSalesDao implements JeepSalesDao {
20
21     @Autowired
22     private NamedParameterJdbcTemplate jdbcTemplate;
23
24     @Override
25     public List<Jeep> fetchJeeps(JeepModel model, String trim) {
26         log.info("DAO: model={}, trim={}", model, trim);
27
28         // formatter:off
29         String sql =
30             " " + "SELECT * "
31             + "FROM models "
32             + "WHERE model_id = :model AND trim_level = :trim_level";
33         // formatter:on
34         Map<String, Object> params = new HashMap<>();
35         params.put("model_id", model.toString());
36         params.put("trim_level", trim);
37
38         return jdbcTemplate.query(sql, params, new RowMapper<>() {
39
40             @Override
41             public Jeep mapRow(ResultSet rs, int rowNum) throws SQLException {
42                 //formatter:off
43                 return Jeep.builder()
44                     .basePrice(new BigDecimal(rs.getString("base_price")))
45                     .modelId(JeepModel.valueOf(rs.getString("model_id")))
46                     .modelPK(rs.getLong("model_pk"))
47                     .numDoors(rs.getInt("num_doors"))
48                     .trimLevel(rs.getString("trim_level"))
49                     .wheelSize(rs.getInt("wheel_size"))
50                     .build();
51                 //formatter:on
52             }
53         });
54     }
55 }
```

- 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

The screenshot shows an IDE with a Java test class `FetchJepTest.java` and its console output. The test class is located in the package `com.primmotech.jep.controller` and contains the following code:

```
1: @Test
2: void buildExpectedJep() {
3:     //Given: a valid model, trim, and url
4:     JeepModel model = JeepModel.WRANGLER;
5:     String trim = "Sport";
6:     String url = String.format("http://localhost:8080/jeps/model=%s", serverport, model, trim);
7:
8:     //When: a connection is made to the API
9:     Response response = restTemplate.exchange(url, HttpMethod.GET, null, new ParameterizedTypeReference<>() {});
10:
11:     //Then: a success (200 - 299) status code is returned
12:     assertEquals(response.getStatusCode(), HttpStatus.OK);
13:
14:     //And: the actual list returned matches the expected list
15:     List<Jep> expected = buildExpected();
16:     assertEquals(response.getBody(), expected);
17: }
18:
19: protected List<Jep> buildExpected() {
20:     List<Jep> list = new LinkedList<>();
21:
22:     // @formatter:off
23:     list.add(new Builder()
24:         .modelId(JepModel.WRANGLER)
25:         .trimLevel("Sport")
26:         .year(2017)
27:         .wheelSize(17)
28:         .horsePower(new BigDecimal("23475.00"))
29:         .build());
30:     list.add(new Builder()
31:         .modelId(JepModel.WRANGLER)
32:         .trimLevel("Sport")
33:         .year(2017)
34:         .wheelSize(17)
35:         .horsePower(new BigDecimal("11975.00"))
36:         .build());
37:     // @formatter:on
38:     return list;
39: }
```

The console output shows the following logs:

```
2023-01-08 00:02:02.480 INFO 4140 --- [main] c.p.jep.controller.FetchJepTest : Starting FetchJepTest using Java 17.0.4.1 on DESKTOP-6N63M5 with PID 4140 (started by hmc in C:\Development)
2023-01-08 00:02:02.480 INFO 4140 --- [main] c.p.jep.controller.FetchJepTest : The following 1 profile is active: 'test'
2023-01-08 00:02:04.394 INFO 4140 --- [main] c.p.jep.controller.FetchJepTest : Started FetchJepTest in 2.457 seconds (JVM running for 3.355)
2023-01-08 00:02:05.395 INFO 4140 --- [c-auto-1-exec-1] c.p.jep.controller.FetchJepTest : main: model=WRANGLER, trim=Sport
2023-01-08 00:02:05.396 INFO 4140 --- [c-auto-1-exec-1] c.p.jep.controller.FetchJepTest : The FetchJepTest method was called with model=WRANGLER and trim=Sport
2023-01-08 00:02:05.396 INFO 4140 --- [c-auto-1-exec-1] c.p.jep.controller.FetchJepTest : DAO: model=WRANGLER, trim=Sport
```