

# Web API Design with Spring Boot Week 3 Coding Assignment

Points possible: 70

Category	Criteria	% of Grade
<b>Functionality</b>	Does the code work?	25
<b>Organization</b>	Is the code clean and organized? Proper use of white space, syntax, and consistency are utilized. Names and comments are concise and clear.	25
<b>Creativity</b>	Student solved the problems presented in the assignment using creativity and out of the box thinking.	25
<b>Completeness</b>	All requirements of the assignment are complete.	25

**Instructions:** In Eclipse, or an IDE of your choice, write the code that accomplishes the objectives listed below. Ensure that the code compiles and runs as directed. Take screenshots of the code and of the running program (make sure to get screenshots of all required functionality) and paste them in this document where instructed below. Create a new repository on GitHub for this week's assignments and push this document, with your Java project code, to the repository. Add the URL for this week's repository to this document where instructed and submit this document to your instructor when complete.

**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.jeep.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).
- 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.

```
vtet      . Completed initialization in 2 ms
oller     : Model = WRANGLER, Trim = Sport
sservice   : The fetchJeeps method was called with arguments: (model= WRANGLER, trim= Sport)
ource    : HikariPool-1 - Shutdown initiated
```

c)

```
1 package com.promineotech.jeep.dao;
2
3 import java.math.BigDecimal;
4 import java.sql.ResultSet;
5 import java.sql.SQLException;
6 import java.util.HashMap;
7 import java.util.List;
8 import java.util.Map;
9
10 import org.springframework.beans.factory.annotation.Autowired;
11 import org.springframework.jdbc.core.RowMapper;
12 import org.springframework.jdbc.core.namedparam.NamedParameterJdbcTemplate;
13 import org.springframework.stereotype.Component;
14 import org.springframework.stereotype.Service;
15
16 import com.mysql.cj.log.Log;
17 import com.promineotech.jeep.entity.Jeep;
18 import com.promineotech.jeep.entity.JeepModel;
19
20 import lombok.extern.slf4j.Slf4j;
21 @Service
22 @Component
23 @Slf4j
24
25 public class DefaultJeepSalesDao implements JeepSalesDao {
26
27     @Autowired
28     private NamedParameterJdbcTemplate jdbcTemplate;
29
30     @Override
31     public List<Jeep> fetchJeeps(JeepModel model, String trim) {
32         log.debug("DAO: model={}, trim={}", model, trim);
33
34         return null;
35     }
36 }
37
```

d)

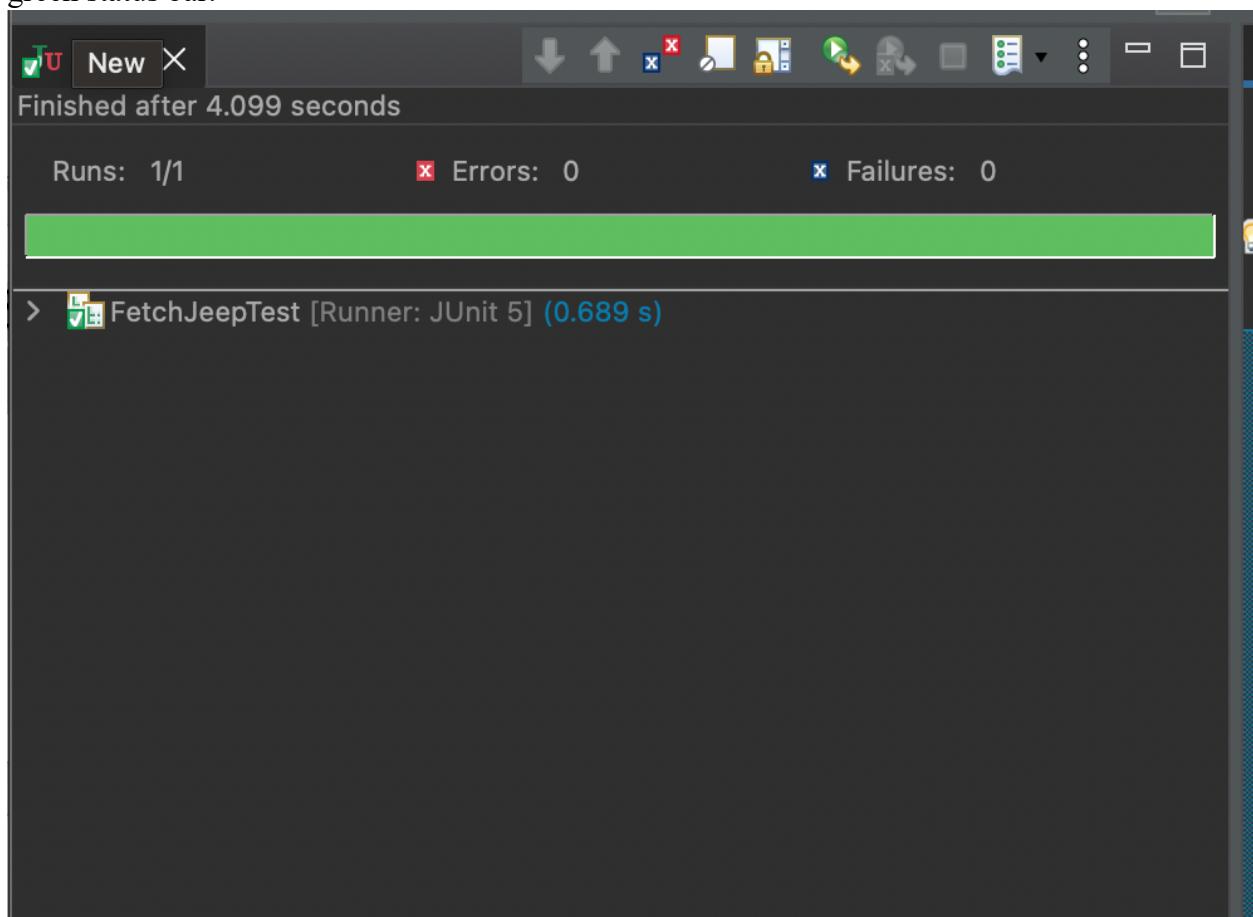
- e) In DefaultJeepSalesDao, inject an instance variable of type NamedParameterJdbcTemplate.
- f) Write SQL to return a list of Jeep models based on the parameters: model and trim. Be sure to utilize the SQL Injection prevention mechanism of the NamedParameterJdbcTemplate using :model\_id and :trim\_level in the query.
- g) 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());)
- h) 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.

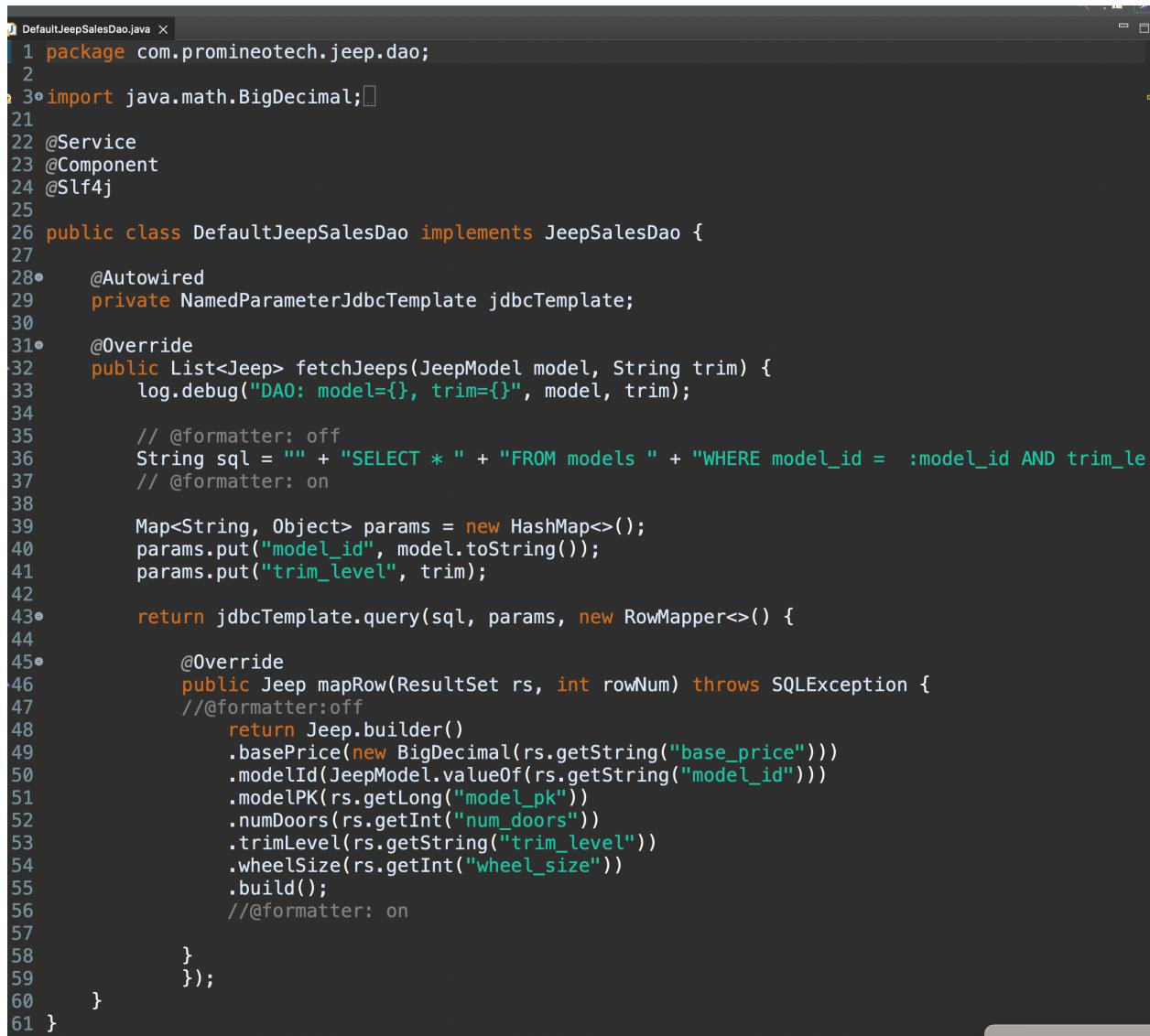
```
 8 import java.util.Map;
9
10 import org.springframework.beans.factory.annotation.Autowired;
11 import org.springframework.jdbc.core.RowMapper;
12 import org.springframework.jdbc.core.namedparam.NamedParameterJdbcTemplate;
13 import org.springframework.stereotype.Component;
14 import org.springframework.stereotype.Service;
15
16 import com.mysql.cj.log.Log;
17 import com.promineotech.jeep.entity.Jeep;
18 import com.promineotech.jeep.entity.JeepModel;
19
20 import lombok.extern.slf4j.Slf4j;
21
22 @Service
23 @Component
24 @Slf4j
25
26 public class DefaultJeepSalesDao implements JeepSalesDao {
27
28     @Autowired
29     private NamedParameterJdbcTemplate jdbcTemplate;
30
31     @Override
32     public List<Jeep> fetchJeeps(JeepModel model, String trim) {
33         log.debug("DAO: model={}, trim={}", model, trim);
34
35         // @formatter: off
36         String sql = "" + "SELECT * " + "FROM models " + "WHERE model_id = :model_id AND trim_level"
37         // @formatter: on
38
39         Map<String, Object> params = new HashMap<>();
40         params.put("model_id", model.toString());
41         params.put("trim_level", trim);
42
43         return jdbcTemplate.query(sql, params, new RowMapper<>() {
44
45             @Override
46             public Jeep mapRow(ResultSet rs, int rowNum) throws SQLException {
47                 // @formatter:off
48                 return Jeep.builder()
49                     .basePrice(new BigDecimal(rs.getString("base_price")))
50                     .modelId(JeepModel.valueOf(rs.getString("model_id")))
51                     .modelPK(rs.getLong("model_pk"))
52                     .numDoors(rs.getInt("num_doors"))
53                     .trimLevel(rs.getString("trim_level"))
54                     .wheelSize(rs.getInt("wheel_size"))
55                     .build();
56                 // @formatter: on
57             }
58         });
59     }
60 }
61 }
```

- 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.



## Screenshots of Code:



```
1 package com.promineotech.jeep.dao;
2
3 import java.math.BigDecimal;
4
5 @Service
6 @Component
7 @Slf4j
8
9 public class DefaultJeepSalesDao implements JeepSalesDao {
10
11     @Autowired
12     private NamedParameterJdbcTemplate jdbcTemplate;
13
14     @Override
15     public List<Jeep> fetchJeeps(JeepModel model, String trim) {
16         log.debug("DAO: model={}, trim={}", model, trim);
17
18         // @formatter: off
19         String sql = "" + "SELECT * " + "FROM models " + "WHERE model_id = :model_id AND trim_le
20         // @formatter: on
21
22         Map<String, Object> params = new HashMap<>();
23         params.put("model_id", model.toString());
24         params.put("trim_level", trim);
25
26         return jdbcTemplate.query(sql, params, new RowMapper<>() {
27
28             @Override
29             public Jeep mapRow(ResultSet rs, int rowNum) throws SQLException {
30                 // @formatter: off
31                 return Jeep.builder()
32                     .basePrice(new BigDecimal(rs.getString("base_price")))
33                     .modelId(JeepModel.valueOf(rs.getString("model_id")))
34                     .modelPK(rs.getLong("model_pk"))
35                     .numDoors(rs.getInt("num_doors"))
36                     .trimLevel(rs.getString("trim_level"))
37                     .wheelSize(rs.getInt("wheel_size"))
38                     .build();
39                 // @formatter: on
40             }
41         });
42     }
43 }
```

```
1 package com.promineotech.jeep.dao;
2
3 import java.util.List;
4
5 public interface JeepSalesDao {
6
7     /**
8      * @param model
9      * @param trim
10     * @return
11     */
12     List<Jeep> fetchJeeps(JeepModel model, String trim);
13
14 }
```

#### Screenshots of Running Application:

localhost:8080/swagger-ui/index.html#/default-jeep-sales-controller/jeeps

Swagger API DOCUMENTATION /v3/api-docs Explore

## Jeep Sales Service

/v3/api-docs

Servers

http://localhost:8080 - Local server.

### default-jeep-sales-controller

GET /jeeps Returns a list of Jeeps

Returns a list of Jeeps given an optional model and/or trim.

Parameters

Name Description

model	The model name (i.e., 'WRANGLER')
string (query)	WRANGLER
trim	The trim level (i.e., 'Sport')
string (query)	Sport

Cancel

Execute Clear

Responses

Curl

```
curl -X 'GET' \
'http://localhost:8080/jeeps?model=WRANGLER&trim=Sport' \
-H 'accept: application/json'
```

Request URL

http://localhost:8080/jeeps?model=WRANGLER&trim=Sport

Server response

**Code** **Details**

200 Response body

```
[  
  {  
    "modelId": "WRANGLER",  
    "trimLevel": "Sport",  
    "numDoors": 2,  
    "wheelSize": 17,  
    "basePrice": 28475  
  },  
  {  
    "modelId": "WRANGLER",  
    "trimLevel": "Sport",  
    "numDoors": 4,  
    "wheelSize": 17,  
    "basePrice": 31975  
  }  
]
```

**Response headers**

```
connection: keep-alive  
content-type: application/json  
date: Wed, 05 Oct 2022 22:10:05 GMT  
keep-alive: timeout=60  
transfer-encoding: chunked
```

**Responses**

Code	Description	Links
200	A list of Jeeps is returned	No links
400	The request parameters are invalid	No links
404	No Jeeps were found with the input criteria	No links
500	An unplanned error occurred	No links

Media type  Controls Accept header.

Example Value | Schema

```
{  
  "modelId": "GRAND_CHEROKEE",  
  "trimLevel": "string",  
  "numDoors": 0,  
  "wheelSize": 0,  
  "basePrice": 0  
}
```

**Schemas**

```
Jeep ▼ {  
  modelId: string  
  trimLevel: string  
  numDoors: integer($int32)  
  wheelSize: integer($int32)  
  basePrice: number  
}
```

JUnit X

Finished after 4.551 seconds

Runs: 1/1 Errors: 0 Failures: 0

FetchJeepTest [JUnit 6] (0.881 s)

Console

```

<terminated> FetchJeepTest () [JUnit] /Library/Java/JavaVirtualMachines/jdk-11.0.15.jdk/Contents/Home/bin/java [Oct 6, 2022, 9:06:28 AM - 9:06:34 AM] [pid: 16657]
09:06:29.575 [main] DEBUG org.springframework.test.context.BootstrapUtils - Instantia
09:06:29.582 [main] DEBUG org.springframework.test.context.BootstrapUtils - Instantia
09:06:29.611 [main] DEBUG org.springframework.test.context.BootstrapUtils - Instantia
09:06:29.619 [main] INFO org.springframework.boot.test.context.SpringBootTestB
09:06:29.622 [main] DEBUG org.springframework.test.context.support.AbstractContextLoa
09:06:29.622 [main] INFO org.springframework.test.context.support.AbstractContextLoa
09:06:29.622 [main] INFO org.springframework.test.context.support.AbstractContextLoad
09:06:29.623 [main] INFO org.springframework.test.context.support.AnnotationConfigCon
09:06:29.736 [main] DEBUG org.springframework.context.annotation.ClassPathScanningCan
09:06:29.737 [main] INFO org.springframework.boot.test.context.SpringBootTestTestConte
09:06:29.815 [main] DEBUG org.springframework.boot.test.context.SpringBootTestConte
09:06:29.815 [main] INFO org.springframework.boot.test.context.SpringBootTestConte
09:06:29.831 [main] INFO org.springframework.boot.test.context.SpringBootTestConte
09:06:29.834 [main] DEBUG org.springframework.test.context.support.AbstractDirtiesCon
09:06:29.844 [main] DEBUG org.springframework.test.context.support.DependencyInjectio

  _/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_
 ( ( )\_\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_
 \\\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_
   \_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_
=====\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_/\_
:: Spring Boot ::          (v2.7.4)

2022-10-06 09:06:30.163 INFO 16657 --- [           main] c.p.jeep.controller.FetchJe
2022-10-06 09:06:30.164 INFO 16657 --- [           main] c.p.jeep.controller.FetchJe
2022-10-06 09:06:30.659 INFO 16657 --- [           main] .s.d.r.c.RepositoryConfigur
2022-10-06 09:06:30.672 INFO 16657 --- [           main] .s.d.r.c.RepositoryConfigur
2022-10-06 09:06:31.151 INFO 16657 --- [           main] o.s.b.w.embedded.tomcat.Tom
2022-10-06 09:06:31.161 INFO 16657 --- [           main] o.apache.catalina.core.Stan
2022-10-06 09:06:31.161 INFO 16657 --- [           main] org.apache.catalina.core.St
2022-10-06 09:06:31.251 INFO 16657 --- [           main] o.a.c.c.C.[Tomcat].[localho
2022-10-06 09:06:31.251 INFO 16657 --- [           main] w.s.c.ServletWebServerAppli
2022-10-06 09:06:32.560 INFO 16657 --- [           main] com.zaxxer.hikari.HikariDat
2022-10-06 09:06:32.832 INFO 16657 --- [           main] com.zaxxer.hikari.HikariDat
2022-10-06 09:06:33.087 INFO 16657 --- [           main] o.s.b.w.embedded.tomcat.Tom
2022-10-06 09:06:33.104 INFO 16657 --- [           main] c.p.jeep.controller.FetchJe
2022-10-06 09:06:33.767 INFO 16657 --- [o-auto-1-exec-1] o.a.c.c.C.[Tomcat].[localho
2022-10-06 09:06:33.767 INFO 16657 --- [o-auto-1-exec-1] o.s.web.servlet.DispatcherS
2022-10-06 09:06:33.769 INFO 16657 --- [o-auto-1-exec-1] o.s.web.servlet.DispatcherS
2022-10-06 09:06:33.816 INFO 16657 --- [o-auto-1-exec-1] c.p.j.c.DefaultJeepSalesCon
2022-10-06 09:06:33.832 INFO 16657 --- [o-auto-1-exec-1] c.p.j.service.DefaultJeepSa
2022-10-06 09:06:34.026 INFO 16657 --- [ionShutdownHook] com.zaxxer.hikari.HikariDat
2022-10-06 09:06:34.030 INFO 16657 --- [ionShutdownHook] com.zaxxer.hikari.HikariDat

```

Failure Trace

## URL to GitHub Repository:

<https://github.com/geraldinedepaul17/BackEndClass/tree/main/Week%2015>