Points possible: 70

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

Coding Steps:

- \checkmark 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.
- √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.

I finished the coding for all of the videos, I am too far past this point for a screen shot

- √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.
- 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. ■

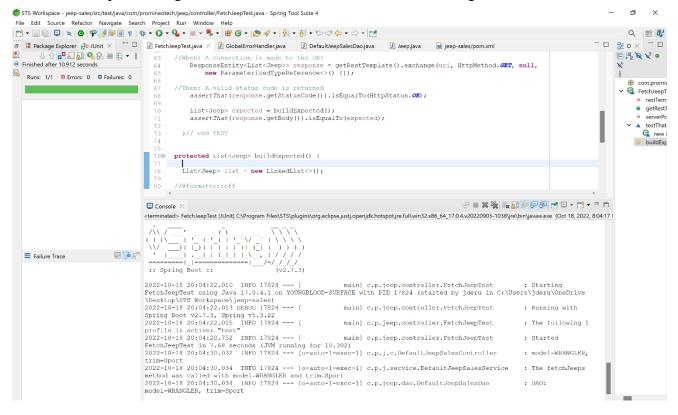
```
STS Workspace - jeep-sales/src/main/java/com/promineotech/jeep/dao/DefaultJeepSalesDao.java - Spring Tool Suite 4
                                                                                                                                                                                                                0
                                                                                                                                                                                                                       ×
Q 图 學
### 1 package com.promineotech.jeep.dao;
                                                                                                                                                                                                                           8
      3⊕ import java.math.BigDecimal;
    19 public class DefaultJeepSalesDao implements JeepSalesDao {
           private NamedParameterJdbcTemplate jdbcTemplate;
           public List<Jeep> fetchJeeps [JeepModel model, String trim) {
   log.info("DAO: model={}, trim={}", model, trim);
                           "WHERE model id = :model id AND trim level = :trim level";
              // @formatter:on
              Map<String, Object> params = new HashMap<>();
params.put("model_id", model.toString());
params.put("trim_level", trim);
              return jdbcTemplate.query(sql, params, new RowMapper<Jeep>() {
                public Jeep mapRow(ResultSet rs, int rowNum) throws SQLException {
                   // #formatterroff
return Jeep.builder()
.basePrice(new Biglecimal(rs.getString("base_price"))
.model1d(JeepModel.valueOf(rs.getString("model_id")))
.modelPK(rs.getLong("model_pk"))
.numDoors(rs.getLong("model_pk"))
.trimLevel(rs.getString("trim_level"))
                          .wheelSize(rs.getInt("wheel size"))
              .build();

// @formatter:on

}//end MAPROW

});//end QUERY
            )// end LIST fetchJeeps
```

- √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 & status bar. ■



URL to GitHub Repository: