URL to GitHup Repository: https://github.com/aliustunyer/Web-API-Design-with-Spring-Boot.git

URL to Youtube Video : https://youtu.be/Jj4Twf6rdrM

```
package com.promineotech.jeep.dao;
import java.util.List;
import com.promineotech.jeep.entity.Jeep;
import com.promineotech.jeep.entity.JeepModel;
public interface JeepSalesDao {
  List<Jeep> fetchJeeps(JeepModel model, String trim)
}
package com.promineotech.jeep.dao;
import java.util.List;
import java.util.Map;
import java.math.BigDecimal;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.HashMap;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.jdbc.core.RowMapper;
import org.springframework.jdbc.core.namedparam.NamedParameterJdbcTemplate;
import org.springframework.stereotype.Component;
import org.springframework.stereotype.Service;
import com.promineotech.jeep.entity.Jeep;
import com.promineotech.jeep.entity.JeepModel;
import lombok.extern.slf4j.Slf4j;
@Service
@Component
@Slf4j
public class DefaultJeepSalesDao implements JeepSalesDao {
  @Autowired
```

```
 private NamedParameterJdbcTemplate jdbcTemplate;
 @Override
 public List<Jeep> fetchJeeps(JeepModel model, String trim) {
  log.debug("DAO : model ={}, trim = {},",model, trim);
  //@formatter: off
  String sql = ""
     + "SELECT * "
     + "FROM models "
     + "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);

//The NamedParameterJdbcTemplate processes the SQL query by replacing the placeholders
//(:model_id and :trim_level) with their corresponding values from the params map.
//The query is executed against the database, and the ResultSet containing the rows
//that match the conditions in the WHERE clause is returned.
//The RowMapper implementation is used to iterate over the ResultSet rows
//and convert each row into a Jeep object
//The query() method returns a List of Jeep objects that were created from
//the ResultSet rows.

  return jdbcTemplate.query(sql, params,new RowMapper<> () {

    @Override
    public Jeep mapRow(ResultSet rs, int rowNum) throws SQLException {
     //formatter:on
     return Jeep.builder()
        .basePrice(new BigDecimal(rs.getString("base_price")))
```

```
            .modelId(JeepModel.valueOf(rs.getString("model_id")))
            .modelPK(rs.getLong("model_pk"))
            .numDoors(rs.getInt("num_doors"))
            .trimLevel(rs.getString("trim_level"))
            .wheelSize(rs.getInt("wheel_size"))
            .build();
        //formatter :off
      }
      });
  }
}
```

```
package com.promineotech.jeep.entity;

import java.math.BigDecimal;
import java.util.Comparator;
import com.fasterxml.jackson.annotation.JsonIgnore;
import lombok.AllArgsConstructor;
import lombok.Builder;
import lombok.Data;
import lombok.NoArgsConstructor;

@Data
@Builder
@NoArgsConstructor
@AllArgsConstructor
public class Jeep implements Comparable<Jeep> {
  private Long modelPK;
  private JeepModel modelId;
  private String trimLevel;
  private int numDoors;
  private int wheelSize;
```

```
  private BigDecimal basePrice;

  @JsonIgnore
  public Long getModelPK() {
    return modelPK;
  }

  @Override
public int compareTo(Jeep that) {
  //formatter :off
  return Comparator
      .comparing(Jeep::getModelId)
      .thenComparing(Jeep::getTrimLevel)
      .thenComparing(Jeep::getNumDoors)
      .compare(this, that);
  //formatter :on
}
}
```