Instructions

The goal of this exercise is to create a simple RESTful service using Spring MVC and Spring Boot.

1) Start out by creating a new project in STS using File -> New -> "Spring Starter Project"

2) In the "Dependencies" screen, select Web -> "Spring Web" as well as SQL -> "Spring Data JPA" and SQL -> "MySQL Driver" and also Lombok (4 dependencies total)

3) Verify that the following dependencies have been added to the generated POM file:

```
</dependency>
<dependency>
   <groupId>org.springframework.boot</groupId>
   <artifactId>spring-boot-starter-data-jpa</artifactId>
</dependency>
<dependency>
   <groupId>mysql</groupId>
   <artifactId>mysql-connector-java</artifactId>
   <scope>runtime</scope>
</dependency>
<dependency>
   <groupId>org.projectlombok</groupId>
   <artifactId>lombok</artifactId>
   <optional>true</optional>
</dependency>
```

4) Write at least two entities for MS SQL AdventureWorks (or Sakila if you are using MySQL).

5) Create the data-access layer for your entity using Spring Data interface JpaRepository. Here is an example:

```
@Repository

public interface MyEntityRepository extends JpaRepository<MyEntity, MyEntityIdType> {

}
```

Hint: This will auto-magically create the data-access layer for you.

6) Create a service layer to expose your repository layer. Autowire your repository bean in your service bean.

7) Create a REST Controller using the @RestController annotation

8) Create a sample controller method annotated with @GetMapping

9) @Autowire the service bean in your controller class

10) Inside the controller method, call one of the methods of your repository type and return the result.

11) Test your application by running the "main". You should be able to view the result of your RESTful service at: http://localhost:8080/ (or any relative URLs from the roots, depending on how you map your @GetMapping)

12) Connect your app to a real database and produce some results!

Hint: You can use the following configuration for configuring the DB connection. Please be sure to rename your configuration file to application.yml since the following is in YML format.

```
spring:
  application:
    name: spring-mvc-simple
  datasource:
    url: jdbc:mysql://localhost:3306/sakila
    username: sakila
    password: sakila
    driverClassName: com.mysql.cj.jdbc.Driver
    hikari:
      maximumPoolSize: 10
  jpa:
    show-sql: true
    hibernate:
      naming:
        physical-strategy: org.hibernate.boot.model.naming.PhysicalNamingStrategyStandardImpl
        implicit-strategy: org.hibernate.boot.model.naming.ImplicitNamingStrategyLegacyJpaImpl
    properties:
      hibernate:
        dialect: org.hibernate.dialect.MySQL8Dialect
logging:
  file:
    name: /Coding/Logging/${spring.application.name}/log
```