

Product Table:

```
CREATE TABLE product (  
    id int NOT NULL AUTO_INCREMENT,  
    name varchar(255) NOT NULL,  
    price decimal(10,2) NOT NULL,  
    PRIMARY KEY (id)  
);
```

Customer Table:

```
CREATE TABLE customer (  
    id int NOT NULL AUTO_INCREMENT,  
    name varchar(255) NOT NULL,  
    address varchar(255) NOT NULL,  
    PRIMARY KEY (id)  
);
```

Manufacturer Table:

```
CREATE TABLE manufacturer (  
    id int NOT NULL AUTO_INCREMENT,  
    name varchar(255) NOT NULL,  
    PRIMARY KEY (id)  
);
```

class table:

```
CREATE TABLE `class` (  
  `class_id` int(11) NOT NULL AUTO_INCREMENT,  
  `class_name` varchar(255) NOT NULL,  
  `class_code` varchar(255) NOT NULL,  
  `class_description` text NOT NULL,  
  `class_size` int(11) NOT NULL,  
  `class_room` varchar(255) NOT NULL,  
  `class_time` time NOT NULL,  
  `class_date` date NOT NULL,  
  PRIMARY KEY (`class_id`)  
)
```

1) insert a new record inside product table

```
Product product = new Product();  
  
product.setName("");  
  
product.setManufacturer("");  
  
productRepository.save(product);
```

2) update an existing record in product table

```
Product product = productRepository.findOne(1L);  
product.setName("");  
product.setManufacturer("");  
  
productRepository.save(product);
```

3) delete an existing record from product table

```
Product product = productRepository.findOne(1L);  
productRepository.delete(product);
```

4) execute an query and return the manufacturer list for a given product name

```
List<Product> products = productRepository.findByName("");  
  
for (Product product : products) {  
    System.out.println(product.getManufacturer());  
}
```

service

```
package jp.mitsuruog.springboot.sample.Customer.service;  
  
import jp.mitsuruog.springboot.sample.Customer.domain.Customer;
```

```
import jp.mitsuruog.springboot.sample.Customer.repository.CustomerRepository;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.data.domain.Page;

import org.springframework.data.domain.Pageable;

import org.springframework.stereotype.Service;


import java.util.List;


@Service

public class CustomerService {


    @Autowired

    CustomerRepository customerRepository;


    public Page<Customer> findAll(Pageable pageable) {

        return customerRepository.findAllOrderByName(pageable);

    }


    public Customer findById(Integer id) {

        return customerRepository.findOne(id);

    }

}
```

```
public Customer create(Customer customer) {  
    return customerRepository.save(customer);  
}
```

```
public Customer update(Customer customer) {  
    return customerRepository.save(customer);  
}
```

```
public void delete(Integer id) {  
    customerRepository.delete(id);  
}  
}
```

model

```
import org.springframework.boot.autoconfigure.SpringBootApplication;
```

```
@SpringBootApplication
```

```
public class DemoApplication {  
    public static void main(String[] args) {  
        SpringApplication.run(DemoApplication.class, args);  
    }  
}
```

Dao

```
public class UserDao implements Dao<User> {

    private List<User> users = new ArrayList<>();

    public UserDao() {
        users.add(new User("John", "john@domain.com"));
        users.add(new User("Susan", "susan@domain.com"));
    }

    @Override
    public Optional<User> get(long id) {
        return Optional.ofNullable(users.get((int) id));
    }

    @Override
    public List<User> getAll() {
        return users;
    }
}
```

@Override

RestController

```
package es.macero.dev;
```

```
import org.springframework.boot.SpringApplication;
```

```
import org.springframework.boot.autoconfigure.SpringBootApplication;
```

```
import org.springframework.context.annotation.Bean;
```

```
import springfox.documentation.builders.PathSelectors;
```

```
import springfox.documentation.builders.RequestHandlerSelectors;
```

```
import springfox.documentation.spi.DocumentationType;
```

```
import springfox.documentation.spring.web.plugins.Docket;
```

```
import springfox.documentation.swagger2.annotations.EnableSwagger2;
```

```
@SpringBootApplication
```

```
@EnableSwagger2
```

```
public class SpringBootRestControllerExampleApplication {
```

```
    public static void main(String[] args) {
```

```
        SpringApplication.run(SpringBootRestControllerExampleApplication.class, args);
```

```
    }
```

```

@Bean

public Docket api() {

    return new Docket(DocumentationType.SWAGGER_2)

        .select()

        .apis(RequestHandlerSelectors.any())

        .paths(PathSelectors.regex("/getCustomersDetails"))

        .build();

}

}

```

Pom.xml

```

<project xmlns="https://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="https://maven.apache.org/POM/4.0.0
http://maven.apache.org/maven-v4_0_0.xsd">

    <modelVersion>4.0.0</modelVersion>

    <groupId>com.customer</groupId>

    <artifactId>customer-ws-spring-boot</artifactId>

    <version>2.3.0</version>

    <name>Spring Boot</name>

    <description>web services using Spring Boot.</description>

```



```
<parent>

  <groupId>org.springframework.boot</groupId>

  <artifactId>spring-boot-starter-parent</artifactId>

  <version>2.1.1.RELEASE</version>

</parent>
```

```
<properties>

  <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

  <java.version>11</java.version>

</properties>
```

```
<dependencies>

  <dependency>

    <groupId>org.springframework.boot</groupId>

    <artifactId>spring-boot-starter-web</artifactId>

  </dependency>
```

```
<dependency>

  <groupId>org.springframework.boot</groupId>

  <artifactId>spring-boot-starter-actuator</artifactId>

</dependency>
```

```
<dependency>  
  <groupId>org.springframework.boot</groupId>  
  <artifactId>spring-boot-starter-data-jpa</artifactId>  
</dependency>
```

```
<!-- Dependencies for Unit Testing -->
```

```
<dependency>  
  <groupId>org.springframework.boot</groupId>  
  <artifactId>spring-boot-starter-test</artifactId>  
  <scope>test</scope>  
</dependency>
```

```
<dependency>  
  <groupId>org.springframework.security</groupId>  
  <artifactId>spring-security-test</artifactId>  
  <scope>test</scope>  
</dependency>
```

```
<dependency>  
  <groupId>com.google.guava</groupId>  
  <artifactId>guava</artifactId>  
  <version>27.0.1-jre</version>  
  <scope>test</scope>  
</dependency>
```

```
<build>

<plugins>

<plugin>

  <groupId>org.springframework.boot</groupId>

  <artifactId>spring-boot-maven-plugin</artifactId>

  <configuration>

    <executable>true</executable>

  </configuration>

  <executions>

    <execution>

      <goals>

        <goal>build-info</goal>

      </goals>

    </execution>

  </executions>

</plugin>

</plugins>

</build>

</project>
```