Contents

[Customer Web Service Project 2](#_Toc183721428)

[1. Project Metadata: 2](#_Toc183721429)

[2. Setup Environment: 2](#_Toc183721430)

[2.1. Backend: 2](#_Toc183721431)

[2.2. Database: 2](#_Toc183721432)

[2.3. Build Tool: 2](#_Toc183721433)

[2.4. Testing: 3](#_Toc183721434)

[3. Structure: 3](#_Toc183721435)

[3.1. Project Layers: 3](#_Toc183721436)

[3.2. Project Details: 4](#_Toc183721437)

[3.3. POM (Project Object Model) Details: 4](#_Toc183721438)

[4. Related Maven Commands: 7](#_Toc183721439)

[4.1. Run the App: 7](#_Toc183721440)

[4.2. Build the App: 7](#_Toc183721441)

[4.3. Update the Dependencies: 7](#_Toc183721442)

[5.Web Service Call Examples: 7](#_Toc183721443)

[5.1. Fetch Customers: 7](#_Toc183721444)

[5.2. Add a New Customer: 7](#_Toc183721445)

[5.3. Update a Customer: 8](#_Toc183721446)

[5.4. Delete Customer/All Customers: 8](#_Toc183721447)

[6. Spring deployments, Rules, and Configurations: 9](#_Toc183721448)

[6.1. IoC (Inversion of Controls): 9](#_Toc183721449)

[6.2. DI (Dependency Injection): 10](#_Toc183721450)

[7. Workflow: 10](#_Toc183721451)

[8. Run-Debug Configuration: 10](#_Toc183721452)

[8.1. Possible Errors During Running the Code: 13](#_Toc183721453)

# Customer Web Service Project

## 1. Project Metadata:

**Group:** com.example

**Artifact:** springboot-oracle

**Dependencies:**

* Spring Web
* Spring Data JPA
* Oracle Driver
* Spring Boot DevTools (optional)
* Spring Boot Validation => Spring Boot Validation seçilebilir değil, çünkü zaten Spring Initializar içinde var. Ancak sonrasında pom'a eklenecek:

(böylece annotations like @NotEmpty, @Email, and @Size in your entity classes to enforce validation rules.)

*<dependency>*

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

*<artifactId>spring-boot-starter-validation</artifactId>*

*</dependency>*

## 2. Setup Environment:

**Scope:** A java based web service project in idea that using Spring boot, Oracle SQL, Maven, Spring Web, Spring Initalizer. Idea is configured to use javac 23.0.1

**IDE:** IntelliJ IDEA

**JDK:** Ensure that JDK 23.0.1 is installed and configured.

**Database:** Oracle SQL

**Tools:** Spring Initializr, Maven

### 2.1. Backend:

**- Spring Boot:** For building the web application.

**- Spring Data JPA:** For database operations.

**- Spring Security:** For authentication and authorization.

### 2.2. Database:

**- Oracle Database:** Stores customer and related information.

### 2.3. Build Tool:

**- Maven:** For dependency management and building the project.

### 2.4. Testing:

**- Postman or SOAPUI:** Used to test RESTful APIs.

## 3. Structure:

|  |  |
| --- | --- |
|  |  |

### 3.1. Project Layers:

The project uses a layered architecture to separate concerns and make the application more modular and maintainable.

#### 3.1.1. Controller Layer:

This is the entry point for all client interactions with the web service, handling HTTP requests and returning appropriate responses. It handles API endpoints, map them to service methods, and return responses to the client.

#### 3.1.2. Service Layer:

This contains the business logic of the application. It interacts with the repository layer to fetch or manipulate data and applies any application-specific rules. It implements business logic and manage transaction boundaries.

#### 3.1.3. Repository Layer:

This layer is responsible for interacting with the database. It uses Spring Data JPA to simplify data access and allows custom query methods. It provides an abstraction for database operations.

#### 3.1.4. Entity Layer:

This layer is responsible for interacting with the database. It uses Spring Data JPA to simplify data access and allows custom query methods. It represents database tables in the application

#### 3.1.5. Configuration Layer:

This layer contains configuration classes for security, database connections, or other system-level configurations. It centralizes system and framework configuration.

### 3.2. Project Details:

* **Project Location:** D:\IdeaProjects\Webservice1-OrMaSpr
* **Database Type:** Oracle
* **Database Tool:** Oracle SQL Developer Version 24.3.0

|  |  |
| --- | --- |
| Localhost Connection  **Username:**  *system*  **Password:**  *password* |  |

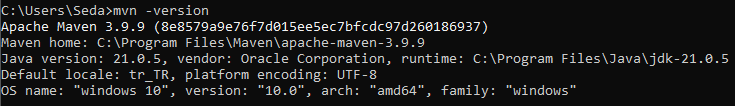
* **Web Service Test Tools:** Postman or SOAP UI

### 3.3. POM (Project Object Model) Details:

#### 3.3.1. Basic Information:

|  |
| --- |
| <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">  <**modelVersion**>4.0.0</modelVersion>  <**groupId**>com.demo</groupId>  <**artifactId**>springboot-oracle</artifactId>  <**version**>0.0.1-SNAPSHOT</version>  <**name**>Webservice1-OrMaSpr</name>  <**description**>Webservice1-OrMaSpr</description>  <**packaging**>jar</packaging>  </project> |

* **modelVersion:** Specifies the version of POM model. Always 4.0.0 for Maven 3+



* **groupId:** Represents the project’s organization or group: *com.demo*
* **artifactId:** The name of the project or application: *springboot-oracle*
* **version:** The version of the app: *0.0.1-SNAPSHOT*
* **name:** *Webservice1-OrMaSpr*
* **description:** *Webservice1-OrMaSpr*
* **packaging**: Defines the output type of the build: *jar*

#### 3.3.2. Parent Configuration:

|  |
| --- |
| <parent>  <groupId>org.springframework.boot</groupId>  <artifactId>spring-boot-starter-parent</artifactId>  <version>3.4.0</version>  <relativePath/> <!-- lookup parent from repository -->  </parent> |

* **<parent>:** Inherits configurations from Spring Boot’s parent POM. This simplifies dependency management and provides default settings for plugins, and other project settings. It’s nor related to the Maven version installed in the system (*3.9.9*). They are completely different concepts.
* **<version>:** Specifies the Spring Voot version used in the project. It defines the framework version and configuration in the project. (No need to match the Maven version and Spring Boot version).

#### 3.3.2. Dependencies:

The libraries and/or frameworks in the project. Maven downloads the related libraries/frameworks from repositories and includes them in the build:

##### 3.3.2.1. Spring Boot Dependencies:

*spring-boot-starter-web* artifact provides core Spring MVC features, such as RESTful APIs:

|  |
| --- |
| <dependency>  <groupId>org.springframework.boot</groupId>  <artifactId>***spring-boot-starter-web***</artifactId>  </dependency> |

spring-boot-starter-data-jpa add Spring Data JPA for database interaction:

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

spring-boot-starter-security enables Spring Security for authentication and autherization:

|  |
| --- |
| <dependency>  <groupId>org.springframework.boot</groupId>  <artifactId>***spring-boot-starter-security***</artifactId>  </dependency> |

##### 3.3.2.2. Validation:

spring-boot-starter-validation provides javax.validation for validating input (@NotNull, @Email):

|  |
| --- |
| <dependency>  <groupId>org.springframework.boot</groupId>  <artifactId>***spring-boot-starter-validation***</artifactId>  </dependency> |

##### 3.3.2.3. Database Connectivity:

odbc11 provides Oracle JDBC driver for connecting to the Oracle database:

|  |
| --- |
| <dependency>  <groupId>com.oracle.database.jdbc</groupId>  <artifactId>***ojdbc11***</artifactId>  <version>21.1.0.0</version>  </dependency> |

##### 3.3.2.4. Testing Dependencies:

spring-boot-starter-test provides testing tools like Junit and Mockito:

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

##### 3.3.2.5. Build Pluggins:

Spring-boot-maven-plugin allows to package and run the Spring Boot application using Maven. It support creating a runnable JAR with an embedded Tomcat server.

|  |
| --- |
| <build>  <plugins>  <plugin>  <groupId>org.springframework.boot</groupId>  <artifactId>***spring-boot-maven-plugin***</artifactId>  </plugin>  </plugins>  </build> |

##### 3.3.2.6. Hibernate Dependency:

In this project, Hibernate is not directly included. But Hibernate dependency is due to transitive dependencies provided by Spring Boot starters. Spring Boot automatically configures Hibernate as the JPA. spring-boot-starter-jpa brings in Hibernate as a transitive dependency because Hibernate is the default JPA implementation used by Spring Boot. spring-boot-starter-jpa includes the following key libraries:

- Spring Data JPA: to provide abstractions for DB operations.

- Hibernate ORM: As the default JPA implementation.

- HikariCP: For DB connection pooling.

Check the dependency with:

|  |
| --- |
| mvn dependency:tree |

## 4. Related Maven Commands:

### 4.1. Run the App:

mvn spring-boot:run

### 4.2. Build the App:

mvn clean package

### 4.3. Update the Dependencies:

mvn dependency:resolve

## 5.Web Service Call Examples:

### 5.1. Get Customers:

**Request URL:** GET <http://localhost:8080/api/customers>

|  |  |
| --- | --- |
| Request | Response |
| GET http://localhost:8080/api/customers |  |

### 5.2. Add a New Customer:

**Request URL:** POST <http://localhost:8080/api/customers>

**Headers:**

* **Key:** Content-Type
* **Value:** application/json

|  |  |
| --- | --- |
| Request | Response |
| {  "name": "Jane Smith",  "email": "jane.smith@example.com"  } | {  "id": 1952,  "name": "Vedat Kelle",  "email": "vedatKelle@example.com"  } |
| {  "name": "Vedat Kelle",  "email": "vedatKelle@example.com"  } | 409 - Conflict  Customer with email vedatKelle@example.com already exists. |
| {  "name": "",  "email": "invalid-email"  } | {  "timestamp": "2024-11-22T12:34:56.789",  "status": 400,  "error": "Bad Request",  "message": "Name is required; Email is invalid",  "path": "/api/customers"  } |

|  |  |
| --- | --- |
| Request | Response |
| {          "name": "Cahit Kelle",          "email": "cahitKelle@den.com",          "phoneNumber": "05111111111",          "addresses": [              {                  "addressName": "Home",                  "street": "Cahit Sokagi",                  "city": "Istanbul",                  "state": "Turkey",                  "postalCode": "34000"              },              {                  "addressName": "Is",                  "street": "Cahit Is Sokagi",                  "city": "Istanbul",                  "state": "Turkey",                  "postalCode": "34000"              }          ]      } | {      "customerId": 12,      "name": "Cahit Kelle",      "email": "cahitKelle@den.com",      "phoneNumber": "05111111111",      "addresses": [          {              "addressId": 5,              "addressName": "Home",              "street": "Cahit Sokagi",              "city": "Istanbul",              "state": "Turkey",              "postalCode": "34000"          },          {              "addressId": 6,              "addressName": "Is",              "street": "Cahit Is Sokagi",              "city": "Istanbul",              "state": "Turkey",              "postalCode": "34000"          }      ]  } |

### 5.3. Update a Customer:

**Request URL:** PUT [http://localhost:8080/api/customers/*{id}*](http://localhost:8080/api/customers/%7bid%7d)

### 5.4. Delete Customer/All Customers:

#### 5.4.1. Delete A Specific Customer By Customer Id:

**Request URL:** DELETE [http://localhost:8080/api/customers/*{id}*](http://localhost:8080/api/customers/%7bid%7d)

Change *{id}* with related customer id

|  |  |
| --- | --- |
| Request | Response |
| DELETE http://localhost:8080/api/customers/952 | 204 - No Content: The resource was successfully deleted.  404 - Not Found: The resource with the given ID doesn’t exist. |

#### 5.4.2. Delete A Specific Customer By Whole Customer Information:

**Request URL:** DELETE <http://localhost:8080/api/customers>

|  |  |
| --- | --- |
| Request | Response |
| {  "id": 1,  "name": "John Doe",  "email": "john.doe@example.com"  } | 404 - Not Found:  Customer not found: 1 |
| {  "id": 4,  "name": "Ahmet Mithat",  "email": "ahmetmithat@hop.com"  } | 404 - Not Found  Customer e-mail do not match the database record |
| {  "id": 4,  "name": "Ahmet Mithat",  "email": "ahmetmithat@example.com"  } | 204 - No Content : Deleted successfully. |

#### 5.4.3. Delete All Customers:

##### 5.4.3.1. Delete All Customers Without Confirmation:

|  |  |
| --- | --- |
| Request | Response |
| DELETE http://localhost:808/api/customers/all | RESPONSE: 204 - No Content |

##### 5.4.3.2. Delete All Customers With Confirmation:

|  |  |
| --- | --- |
| Request | Response |
| DELETE http://localhost:808/api/customers/all | RESPONSE: 400 - Bad REQUEST |
| DELETE http://localhost:808/api/customers/all?confirm=true | RESPONSE: 204 - No Content |

6. REST Call Examples:

The System have customers that have a One-to-Many relation with Address models.

6.1. General Errors:

1. Trying to add a customer without Running the app:

URL: POST <http://localhost:8080/api/customers>

|  |  |
| --- | --- |
| {      "name": "Alice Smith",      "email": "alice.smith@example.com",      "addresses": [          {              "addressName": "Home",              "street": "123 Maple Street",              "city": "Springfield",              "state": "IL",              "zipcode": "62701"          }      ]  } | POST http://localhost:8080/api/customers  Error: connect ECONNREFUSED 127.0.0.1:8080 |

6.2. Add a New Customer:

URL: POST <http://localhost:8080/api/customers>

|  |  |
| --- | --- |
| {  "name": "Alice Smith",  "email": "alice.smith@example.com",  "addresses": [  {  "addressName": "Home",  "street": "123 Maple Street",  "city": "Springfield",  "state": "IL",  "zipcode": "62701"  }  ]  } |  |

## 6. Spring deployments, Rules, and Configurations:

### 6.1. IoC (Inversion of Controls):

IoC means control of object creation and dependency management invertedly. The programmer does not creat any objects manually or manage dependencies, the Spring IoC container does this automatically. IoC is achived through DI (Dependency Injection).

When the app starts, Spring scans the project for classes annotated with @Component, @Service, @Controller, @Repository. Spring creates beans for those components and manages their lifecycle. It injects dependencies into components where needed. All beans are stored in the IoC container (also called the application context) and are accessible for use.

In the project, the IoC components are:

* **Controllers:** Handling the HTTP requests:
  + ***CustomerController****:* The CustomerController is annotated with @RestController, marking it as a Spring-managed bean. Spring automatically injects an instance of CustomerService into the controller.
* **Services:** Contains business logic:
  + ***CustomerService:*** The CustomerService is annotated with @Service, making it a Spring-managed components. Spring injects the CustomerRepository into CustomerService.
* **Repositories:** Interaction with the DB:
  + ***CustomerRepository:*** The CustomerRepository is annotated with @Repository (Optional for JPA repositories: Spring detects them automatically). The IoC container provides an implementation of the JpaRepository Interface.
* **Configurations:** Management of settings like security and data source connections:
  + ***SecurityConfig*:** This configuration layer sets up application-level settings. The SecurityConfig class is annotated with @Configuration, signaling that it provides Spring configuration. Spring manages the configuration and applies it to the app.

### 6.2. DI (Dependency Injection):

Dependency Injection is a central concept used to implement IoC. It ensures that the dependencies are provided to the class rather than the class creating them itself. DI is a design pattern where an object receives its dependencies from an external soruce rather than creating them itself. It the project:

* Controller Layer: The controller receives dependencies via constructor-based injection.

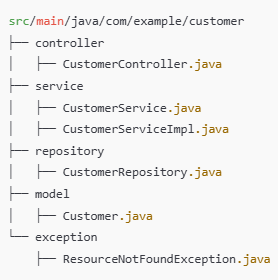
### 6.3. Spring Beans:

Controller like “CustomerController” are Spring Beans because they’re annotated with @RestController. Spring auotamatically detects and registers them during component scanning.

Services like “CustomerService are annotated with @Service, making them Spring Beans. This ensures they are available for injection into controllers or other beans.

Repositories like CustomerRepository are Spring Beans by default because they extend JpaRepository and are annotated with @Repository.

### 6.4. Spring MVC:



* Model => Customer.java
* Repository => CustomerRepository.java: process on DB with Spring Data JPA
* Service => CustomerService. Java and CustomerServiceImpl.java: Business logic
* Controller => CustomerController.java: Process HTTP request from the clients
* Error Management => ResourceNotFoundException.java: istisna sınıfı

## 7. Spring REST Mechanism in the Project:

### 7.1. Controller Layer: Entry Point:

The CustomerController class handles incoming HTTP requests. It is annotated with @RestController, which is specialization of @Controller and @ResponseBody. @Controller maps requests to methods. @ResponseBody automatically converts method responses into JSON or XML.

### 7.2. Service Layer: Business Logic:

The CustomerService class processes requests and enforces business rules. It is annotated with @Service to be managed as a Spring Bean.

### 7.3. Repository Layer: Database Access:

The CustomerRepository class interacts with the database using Spring Data JPA. It is annotated with @Repository and extends JpaRepository, which provides built-in CRUD methods.

### 7.4. RESTful Endpoints in the Project

The project defines endpoints to handle customer and address operations: GET, POST, PUT(Update), DELETE

## 7. Related Maven commands:

7.1. Checking any version dependency:

mvn dependency:tree:

|  |
| --- |
| ...  [INFO] | +- org.springframework:spring-aop:jar:6.2.0:compile  [INFO] | +- org.springframework.security:spring-security-config:jar:6.4.1:compile  [INFO] | | \- org.springframework.security:spring-security-core:jar:6.4.1:compile  [INFO] | \- org.springframework.security:spring-security-web:jar:6.4.1:compile  [INFO] +- org.springframework.boot:spring-boot-starter-validation:jar:3.4.0:compile  [INFO] | +- org.apache.tomcat.embed:tomcat-embed-el:jar:10.1.33:compile  ... |

## 8. Workflow:

TODO: Fill this part later with related schemas, graphics, UMLs, etc.

## 9. Run-Debug Configuration:

To run the program on IDEA:

1. Choose (click) Webservice1OrMaSprApplication and Run (Shift + F10)
2. Access the APIs using a tool like Postman or SOAP UI, or via browser.

|  |
| --- |
| Success Run:  "C:\Program Files\Java\jdk-23\bin\java.exe" -XX:TieredStopAtLevel=1 -Dspring.output.ansi.enabled=always -Dcom.sun.management.jmxremote -Dspring.jmx.enabled=true -Dspring.liveBeansView.mbeanDomain -Dspring.application.admin.enabled=true "-Dmanagement.endpoints.jmx.exposure.include=\*" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2023.1.7\lib\idea\_rt.jar=57849:C:\Program Files\JetBrains\IntelliJ IDEA 2023.1.7\bin" -Dfile.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8 -Dsun.stderr.encoding=UTF-8 -classpath D:\IdeaProjects\Webservice1-OrMaSpr\target\classes;C:\Users\Seda\.m2\repository\org\springframework\boot\spring-boot-starter-data-jpa\3.4.0\spring-boot-starter-data-jpa-3.4.0.jar;C:\Users\Seda\.m2\repository\org\springframework\boot\spring-boot-starter\3.4.0\spring-boot-starter-3.4.0.jar;C:\Users\Seda\.m2\repository\org\springframework\boot\spring-boot-starter-logging\3.4.0\spring-boot-starter-logging-3.4.0.jar;C:\Users\Seda\.m2\repository\ch\qos\logback\logback-classic\1.5.12\logback-classic-1.5.12.jar;C:\Users\Seda\.m2\repository\ch\qos\logback\logback-core\1.5.12\logback-core-1.5.12.jar;C:\Users\Seda\.m2\repository\org\apache\logging\log4j\log4j-to-slf4j\2.24.1\log4j-to-slf4j-2.24.1.jar;C:\Users\Seda\.m2\repository\org\apache\logging\log4j\log4j-api\2.24.1\log4j-api-2.24.1.jar;C:\Users\Seda\.m2\repository\org\slf4j\jul-to-slf4j\2.0.16\jul-to-slf4j-2.0.16.jar;C:\Users\Seda\.m2\repository\jakarta\annotation\jakarta.annotation-api\2.1.1\jakarta.annotation-api-2.1.1.jar;C:\Users\Seda\.m2\repository\org\yaml\snakeyaml\2.3\snakeyaml-2.3.jar;C:\Users\Seda\.m2\repository\org\springframework\boot\spring-boot-starter-jdbc\3.4.0\spring-boot-starter-jdbc-3.4.0.jar;C:\Users\Seda\.m2\repository\com\zaxxer\HikariCP\5.1.0\HikariCP-5.1.0.jar;C:\Users\Seda\.m2\repository\org\springframework\spring-jdbc\6.2.0\spring-jdbc-6.2.0.jar;C:\Users\Seda\.m2\repository\org\hibernate\orm\hibernate-core\6.6.2.Final\hibernate-core-6.6.2.Final.jar;C:\Users\Seda\.m2\repository\jakarta\persistence\jakarta.persistence-api\3.1.0\jakarta.persistence-api-3.1.0.jar;C:\Users\Seda\.m2\repository\jakarta\transaction\jakarta.transaction-api\2.0.1\jakarta.transaction-api-2.0.1.jar;C:\Users\Seda\.m2\repository\org\jboss\logging\jboss-logging\3.6.1.Final\jboss-logging-3.6.1.Final.jar;C:\Users\Seda\.m2\repository\org\hibernate\common\hibernate-commons-annotations\7.0.3.Final\hibernate-commons-annotations-7.0.3.Final.jar;C:\Users\Seda\.m2\repository\io\smallrye\jandex\3.2.0\jandex-3.2.0.jar;C:\Users\Seda\.m2\repository\com\fasterxml\classmate\1.7.0\classmate-1.7.0.jar;C:\Users\Seda\.m2\repository\net\bytebuddy\byte-buddy\1.15.10\byte-buddy-1.15.10.jar;C:\Users\Seda\.m2\repository\org\glassfish\jaxb\jaxb-runtime\4.0.5\jaxb-runtime-4.0.5.jar;C:\Users\Seda\.m2\repository\org\glassfish\jaxb\jaxb-core\4.0.5\jaxb-core-4.0.5.jar;C:\Users\Seda\.m2\repository\org\eclipse\angus\angus-activation\2.0.2\angus-activation-2.0.2.jar;C:\Users\Seda\.m2\repository\org\glassfish\jaxb\txw2\4.0.5\txw2-4.0.5.jar;C:\Users\Seda\.m2\repository\com\sun\istack\istack-commons-runtime\4.1.2\istack-commons-runtime-4.1.2.jar;C:\Users\Seda\.m2\repository\jakarta\inject\jakarta.inject-api\2.0.1\jakarta.inject-api-2.0.1.jar;C:\Users\Seda\.m2\repository\org\antlr\antlr4-runtime\4.13.0\antlr4-runtime-4.13.0.jar;C:\Users\Seda\.m2\repository\org\springframework\data\spring-data-jpa\3.4.0\spring-data-jpa-3.4.0.jar;C:\Users\Seda\.m2\repository\org\springframework\data\spring-data-commons\3.4.0\spring-data-commons-3.4.0.jar;C:\Users\Seda\.m2\repository\org\springframework\spring-orm\6.2.0\spring-orm-6.2.0.jar;C:\Users\Seda\.m2\repository\org\springframework\spring-context\6.2.0\spring-context-6.2.0.jar;C:\Users\Seda\.m2\repository\org\springframework\spring-aop\6.2.0\spring-aop-6.2.0.jar;C:\Users\Seda\.m2\repository\org\springframework\spring-tx\6.2.0\spring-tx-6.2.0.jar;C:\Users\Seda\.m2\repository\org\springframework\spring-beans\6.2.0\spring-beans-6.2.0.jar;C:\Users\Seda\.m2\repository\org\slf4j\slf4j-api\2.0.16\slf4j-api-2.0.16.jar;C:\Users\Seda\.m2\repository\org\springframework\spring-aspects\6.2.0\spring-aspects-6.2.0.jar;C:\Users\Seda\.m2\repository\org\aspectj\aspectjweaver\1.9.22.1\aspectjweaver-1.9.22.1.jar;C:\Users\Seda\.m2\repository\org\springframework\boot\spring-boot-starter-web\3.4.0\spring-boot-starter-web-3.4.0.jar;C:\Users\Seda\.m2\repository\org\springframework\boot\spring-boot-starter-json\3.4.0\spring-boot-starter-json-3.4.0.jar;C:\Users\Seda\.m2\repository\com\fasterxml\jackson\core\jackson-databind\2.18.1\jackson-databind-2.18.1.jar;C:\Users\Seda\.m2\repository\com\fasterxml\jackson\core\jackson-annotations\2.18.1\jackson-annotations-2.18.1.jar;C:\Users\Seda\.m2\repository\com\fasterxml\jackson\core\jackson-core\2.18.1\jackson-core-2.18.1.jar;C:\Users\Seda\.m2\repository\com\fasterxml\jackson\datatype\jackson-datatype-jdk8\2.18.1\jackson-datatype-jdk8-2.18.1.jar;C:\Users\Seda\.m2\repository\com\fasterxml\jackson\datatype\jackson-datatype-jsr310\2.18.1\jackson-datatype-jsr310-2.18.1.jar;C:\Users\Seda\.m2\repository\com\fasterxml\jackson\module\jackson-module-parameter-names\2.18.1\jackson-module-parameter-names-2.18.1.jar;C:\Users\Seda\.m2\repository\org\springframework\boot\spring-boot-starter-tomcat\3.4.0\spring-boot-starter-tomcat-3.4.0.jar;C:\Users\Seda\.m2\repository\org\apache\tomcat\embed\tomcat-embed-core\10.1.33\tomcat-embed-core-10.1.33.jar;C:\Users\Seda\.m2\repository\org\apache\tomcat\embed\tomcat-embed-websocket\10.1.33\tomcat-embed-websocket-10.1.33.jar;C:\Users\Seda\.m2\repository\org\springframework\spring-web\6.2.0\spring-web-6.2.0.jar;C:\Users\Seda\.m2\repository\io\micrometer\micrometer-observation\1.14.1\micrometer-observation-1.14.1.jar;C:\Users\Seda\.m2\repository\io\micrometer\micrometer-commons\1.14.1\micrometer-commons-1.14.1.jar;C:\Users\Seda\.m2\repository\org\springframework\spring-webmvc\6.2.0\spring-webmvc-6.2.0.jar;C:\Users\Seda\.m2\repository\org\springframework\spring-expression\6.2.0\spring-expression-6.2.0.jar;C:\Users\Seda\.m2\repository\org\springframework\boot\spring-boot-devtools\3.4.0\spring-boot-devtools-3.4.0.jar;C:\Users\Seda\.m2\repository\org\springframework\boot\spring-boot\3.4.0\spring-boot-3.4.0.jar;C:\Users\Seda\.m2\repository\org\springframework\boot\spring-boot-autoconfigure\3.4.0\spring-boot-autoconfigure-3.4.0.jar;C:\Users\Seda\.m2\repository\org\springframework\boot\spring-boot-starter-validation\3.4.0\spring-boot-starter-validation-3.4.0.jar;C:\Users\Seda\.m2\repository\org\apache\tomcat\embed\tomcat-embed-el\10.1.33\tomcat-embed-el-10.1.33.jar;C:\Users\Seda\.m2\repository\org\hibernate\validator\hibernate-validator\8.0.1.Final\hibernate-validator-8.0.1.Final.jar;C:\Users\Seda\.m2\repository\jakarta\validation\jakarta.validation-api\3.0.2\jakarta.validation-api-3.0.2.jar;C:\Users\Seda\.m2\repository\com\oracle\database\jdbc\ojdbc11\21.1.0.0\ojdbc11-21.1.0.0.jar;C:\Users\Seda\.m2\repository\jakarta\xml\bind\jakarta.xml.bind-api\4.0.2\jakarta.xml.bind-api-4.0.2.jar;C:\Users\Seda\.m2\repository\jakarta\activation\jakarta.activation-api\2.1.3\jakarta.activation-api-2.1.3.jar;C:\Users\Seda\.m2\repository\org\springframework\spring-core\6.2.0\spring-core-6.2.0.jar;C:\Users\Seda\.m2\repository\org\springframework\spring-jcl\6.2.0\spring-jcl-6.2.0.jar com.demo.springbootoracle.Webservice1OrMaSprApplication  . \_\_\_\_ \_ \_\_ \_ \_  /\\ / \_\_\_'\_ \_\_ \_ \_(\_)\_ \_\_ \_\_ \_ \ \ \ \  ( ( )\\_\_\_ | '\_ | '\_| | '\_ \/ \_` | \ \ \ \  \\/ \_\_\_)| |\_)| | | | | || (\_| | ) ) ) )  ' |\_\_\_\_| .\_\_|\_| |\_|\_| |\_\\_\_, | / / / /  =========|\_|==============|\_\_\_/=/\_/\_/\_/  :: Spring Boot :: (v3.4.0)  2024-11-22T20:20:19.253+03:00 INFO 15092 --- [Webservice1-OrMaSpr] [ restartedMain] c.d.s.Webservice1OrMaSprApplication : Starting Webservice1OrMaSprApplication using Java 23.0.1 with PID 15092 (D:\IdeaProjects\Webservice1-OrMaSpr\target\classes started by Seda in D:\IdeaProjects\Webservice1-OrMaSpr)  2024-11-22T20:20:19.257+03:00 INFO 15092 --- [Webservice1-OrMaSpr] [ restartedMain] c.d.s.Webservice1OrMaSprApplication : No active profile set, falling back to 1 default profile: "default"  2024-11-22T20:20:19.334+03:00 INFO 15092 --- [Webservice1-OrMaSpr] [ restartedMain] o.s.b.devtools.restart.ChangeableUrls : The Class-Path manifest attribute in C:\Users\Seda\.m2\repository\com\oracle\database\jdbc\ojdbc11\21.1.0.0\ojdbc11-21.1.0.0.jar referenced one or more files that do not exist: file:/C:/Users/Seda/.m2/repository/com/oracle/database/jdbc/ojdbc11/21.1.0.0/oraclepki.jar  2024-11-22T20:20:19.334+03:00 INFO 15092 --- [Webservice1-OrMaSpr] [ restartedMain] .e.DevToolsPropertyDefaultsPostProcessor : Devtools property defaults active! Set 'spring.devtools.add-properties' to 'false' to disable  2024-11-22T20:20:19.334+03:00 INFO 15092 --- [Webservice1-OrMaSpr] [ restartedMain] .e.DevToolsPropertyDefaultsPostProcessor : For additional web related logging consider setting the 'logging.level.web' property to 'DEBUG'  2024-11-22T20:20:20.107+03:00 INFO 15092 --- [Webservice1-OrMaSpr] [ restartedMain] .s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data JPA repositories in DEFAULT mode.  2024-11-22T20:20:20.164+03:00 INFO 15092 --- [Webservice1-OrMaSpr] [ restartedMain] .s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 46 ms. Found 1 JPA repository interface.  2024-11-22T20:20:20.759+03:00 INFO 15092 --- [Webservice1-OrMaSpr] [ restartedMain] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat initialized with port 8080 (http)  2024-11-22T20:20:20.774+03:00 INFO 15092 --- [Webservice1-OrMaSpr] [ restartedMain] o.apache.catalina.core.StandardService : Starting service [Tomcat]  2024-11-22T20:20:20.774+03:00 INFO 15092 --- [Webservice1-OrMaSpr] [ restartedMain] o.apache.catalina.core.StandardEngine : Starting Servlet engine: [Apache Tomcat/10.1.33]  2024-11-22T20:20:20.834+03:00 INFO 15092 --- [Webservice1-OrMaSpr] [ restartedMain] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring embedded WebApplicationContext  2024-11-22T20:20:20.835+03:00 INFO 15092 --- [Webservice1-OrMaSpr] [ restartedMain] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization completed in 1500 ms  2024-11-22T20:20:21.036+03:00 DEBUG 15092 --- [Webservice1-OrMaSpr] [ restartedMain] j.LocalContainerEntityManagerFactoryBean : Building JPA container EntityManagerFactory for persistence unit 'default'  2024-11-22T20:20:21.059+03:00 INFO 15092 --- [Webservice1-OrMaSpr] [ restartedMain] o.hibernate.jpa.internal.util.LogHelper : HHH000204: Processing PersistenceUnitInfo [name: default]  2024-11-22T20:20:21.118+03:00 INFO 15092 --- [Webservice1-OrMaSpr] [ restartedMain] org.hibernate.Version : HHH000412: Hibernate ORM core version 6.6.2.Final  2024-11-22T20:20:21.158+03:00 INFO 15092 --- [Webservice1-OrMaSpr] [ restartedMain] o.h.c.internal.RegionFactoryInitiator : HHH000026: Second-level cache disabled  2024-11-22T20:20:21.471+03:00 INFO 15092 --- [Webservice1-OrMaSpr] [ restartedMain] o.s.o.j.p.SpringPersistenceUnitInfo : No LoadTimeWeaver setup: ignoring JPA class transformer  2024-11-22T20:20:21.503+03:00 INFO 15092 --- [Webservice1-OrMaSpr] [ restartedMain] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Starting...  2024-11-22T20:20:21.821+03:00 INFO 15092 --- [Webservice1-OrMaSpr] [ restartedMain] com.zaxxer.hikari.pool.HikariPool : HikariPool-1 - Added connection oracle.jdbc.driver.T4CConnection@f86210e  2024-11-22T20:20:21.823+03:00 INFO 15092 --- [Webservice1-OrMaSpr] [ restartedMain] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Start completed.  2024-11-22T20:20:21.989+03:00 WARN 15092 --- [Webservice1-OrMaSpr] [ restartedMain] org.hibernate.orm.deprecation : HHH90000025: OracleDialect does not need to be specified explicitly using 'hibernate.dialect' (remove the property setting and it will be selected by default)  2024-11-22T20:20:22.032+03:00 INFO 15092 --- [Webservice1-OrMaSpr] [ restartedMain] org.hibernate.orm.connections.pooling : HHH10001005: Database info:  Database JDBC URL [Connecting through datasource 'HikariDataSource (HikariPool-1)']  Database driver: undefined/unknown  Database version: 21.0  Autocommit mode: undefined/unknown  Isolation level: undefined/unknown  Minimum pool size: undefined/unknown  Maximum pool size: undefined/unknown  2024-11-22T20:20:23.028+03:00 INFO 15092 --- [Webservice1-OrMaSpr] [ restartedMain] o.h.e.t.j.p.i.JtaPlatformInitiator : HHH000489: No JTA platform available (set 'hibernate.transaction.jta.platform' to enable JTA platform integration)  2024-11-22T20:20:23.293+03:00 INFO 15092 --- [Webservice1-OrMaSpr] [ restartedMain] j.LocalContainerEntityManagerFactoryBean : Initialized JPA EntityManagerFactory for persistence unit 'default'  2024-11-22T20:20:23.392+03:00 DEBUG 15092 --- [Webservice1-OrMaSpr] [ restartedMain] tor$SharedEntityManagerInvocationHandler : Creating new EntityManager for shared EntityManager invocation  2024-11-22T20:20:23.436+03:00 DEBUG 15092 --- [Webservice1-OrMaSpr] [ restartedMain] tor$SharedEntityManagerInvocationHandler : Creating new EntityManager for shared EntityManager invocation  2024-11-22T20:20:23.491+03:00 DEBUG 15092 --- [Webservice1-OrMaSpr] [ restartedMain] tor$SharedEntityManagerInvocationHandler : Creating new EntityManager for shared EntityManager invocation  2024-11-22T20:20:23.589+03:00 WARN 15092 --- [Webservice1-OrMaSpr] [ restartedMain] JpaBaseConfiguration$JpaWebConfiguration : spring.jpa.open-in-view is enabled by default. Therefore, database queries may be performed during view rendering. Explicitly configure spring.jpa.open-in-view to disable this warning  2024-11-22T20:20:23.948+03:00 INFO 15092 --- [Webservice1-OrMaSpr] [ restartedMain] o.s.b.d.a.OptionalLiveReloadServer : LiveReload server is running on port 35729  2024-11-22T20:20:23.988+03:00 INFO 15092 --- [Webservice1-OrMaSpr] [ restartedMain] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port 8080 (http) with context path '/'  2024-11-22T20:20:23.997+03:00 INFO 15092 --- [Webservice1-OrMaSpr] [ restartedMain] c.d.s.Webservice1OrMaSprApplication : Started Webservice1OrMaSprApplication in 5.276 seconds (process running for 6.14)  2024-11-22T20:20:32.299+03:00 INFO 15092 --- [Webservice1-OrMaSpr] [nio-8080-exec-1] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring DispatcherServlet 'dispatcherServlet'  2024-11-22T20:20:32.300+03:00 INFO 15092 --- [Webservice1-OrMaSpr] [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet : Initializing Servlet 'dispatcherServlet'  2024-11-22T20:20:32.301+03:00 INFO 15092 --- [Webservice1-OrMaSpr] [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet : Completed initialization in 1 ms  2024-11-22T20:20:32.324+03:00 DEBUG 15092 --- [Webservice1-OrMaSpr] [nio-8080-exec-1] o.j.s.OpenEntityManagerInViewInterceptor : Opening JPA EntityManager in OpenEntityManagerInViewInterceptor  2024-11-22T20:20:32.347+03:00 DEBUG 15092 --- [Webservice1-OrMaSpr] [nio-8080-exec-1] o.s.orm.jpa.JpaTransactionManager : Found thread-bound EntityManager [SessionImpl(731672050<open>)] for JPA transaction  2024-11-22T20:20:32.348+03:00 DEBUG 15092 --- [Webservice1-OrMaSpr] [nio-8080-exec-1] o.s.orm.jpa.JpaTransactionManager : Creating new transaction with name [org.springframework.data.jpa.repository.support.SimpleJpaRepository.findAll]: PROPAGATION\_REQUIRED,ISOLATION\_DEFAULT,readOnly  2024-11-22T20:20:32.353+03:00 DEBUG 15092 --- [Webservice1-OrMaSpr] [nio-8080-exec-1] o.s.orm.jpa.JpaTransactionManager : Exposing JPA transaction as JDBC [org.springframework.orm.jpa.vendor.HibernateJpaDialect$HibernateConnectionHandle@12f7e048]  2024-11-22T20:20:32.480+03:00 DEBUG 15092 --- [Webservice1-OrMaSpr] [nio-8080-exec-1] org.hibernate.SQL : select c1\_0.id,c1\_0.email,c1\_0.name from customer c1\_0  Hibernate: select c1\_0.id,c1\_0.email,c1\_0.name from customer c1\_0  2024-11-22T20:20:32.504+03:00 DEBUG 15092 --- [Webservice1-OrMaSpr] [nio-8080-exec-1] o.s.orm.jpa.JpaTransactionManager : Initiating transaction commit  2024-11-22T20:20:32.505+03:00 DEBUG 15092 --- [Webservice1-OrMaSpr] [nio-8080-exec-1] o.s.orm.jpa.JpaTransactionManager : Committing JPA transaction on EntityManager [SessionImpl(731672050<open>)]  2024-11-22T20:20:32.507+03:00 DEBUG 15092 --- [Webservice1-OrMaSpr] [nio-8080-exec-1] o.s.orm.jpa.JpaTransactionManager : Not closing pre-bound JPA EntityManager after transaction  2024-11-22T20:20:32.556+03:00 DEBUG 15092 --- [Webservice1-OrMaSpr] [nio-8080-exec-1] o.j.s.OpenEntityManagerInViewInterceptor : Closing JPA EntityManager in OpenEntityManagerInViewInterceptor |

### 9.1. Possible Errors During Running the Code:

Check some errors below:

1. No error on Idea side, but getting 403 Forbidden over Postman:

|  |
| --- |
| Request: POST http://localhost:8080/api/customers |
| Authorization: Basic Auth  User: admin  Pass: admin123 |
| {    "name": "John Doe",    "email": "johndoe@example.com",    "addresses": [      {        "addressName": "Home",        "street": "123 Elm Street",        "city": "Springfield",        "zipCode": "62701"      },      {        "addressName": "Work",        "street": "456 Oak Avenue",        "city": "Springfield",        "zipCode": "62702"      }    ]  } |
| {      "timestamp": "2024-12-01T08:15:26.983+00:00",      "status": 403,      "error": "Forbidden",      "message": "Access Denied",      "path": "/api/customers"  } |

Security Debug logs on Idea side

|  |
| --- |
| Initializing Spring DispatcherServlet 'dispatcherServlet'  Initializing Servlet 'dispatcherServlet'  Completed initialization in 1 ms  Securing POST /api/customers  Set SecurityContextHolder to anonymous SecurityContext  Saved request http://localhost:8080/api/customers?continue to session  Pre-authenticated entry point called. Rejecting access  Securing POST /error  Secured POST /error  Opening JPA EntityManager in OpenEntityManagerInViewInterceptor  Closing JPA EntityManager in OpenEntityManagerInViewInterceptor  Set SecurityContextHolder to anonymous SecurityContext |

Temporary Solution: Permit all request for now by disabling admin related lines in SecurityConfig, check it later:

|  |
| --- |
| http  .csrf(csrf -> csrf.disable()) // Disable CSRF protection  .authorizeHttpRequests(auth -> auth  //.requestMatchers("/api/customers/\*\*").authenticated() // Require authentication  .anyRequest().permitAll() // Permit other requests  )  .httpBasic(customizer -> customizer.disable()); // Disable default httpBasic and use custom setup  return http.build(); |

1. Sending post request with no phone related entry:

|  |
| --- |
| Request: POST http://localhost:8080/api/customers |
| {    "name": "John Doe",    "email": "johndoe@example.com",    "addresses": [      {        "addressName": "Home",        "street": "123 Elm Street",        "city": "Springfield",        "zipCode": "62701"      }    ]  } |
| Response:  org.hibernate.TransientPropertyValueException: Not-null property references a transient value - transient instance must be saved before current operation: com.demo.springbootoracle.entity.Address.customer -> com.demo.springbootoracle.entity.Customer |
| IDEA Warn:  o.h.a.i.UnresolvedEntityInsertActions : HHH000437: Attempting to save one or more entities that have a non-nullable association with an unsaved transient entity. The unsaved transient entity must be saved in an operation prior to saving these dependent entities.  Unsaved transient entity: ([com.demo.springbootoracle.entity.Customer#<null>])  Dependent entities: ([[com.demo.springbootoracle.entity.Address#<null>]])  Non-nullable association(s): ([com.demo.springbootoracle.entity.Address.customer]) |

Solution: Sending the correct request.

1. The response of a GET request is recursive:

|  |
| --- |
| [      {          "customerId": 1,          "name": "Vedat Kelle",          "email": "vedat.kelle@hotmak.com",          "phoneNumber": "05505005050",          "addresses": [              {                  "addressId": 1,                  "addressName": "Home",                  "street": "Palm Sokagi",                  "city": "Istanbul",                  "state": "Turkey",                  "postalCode": "35660",                  "customer": {                      "customerId": 1,                      "name": "Vedat Kelle",                      "email": "vedat.kelle@hotmak.com",                      "phoneNumber": "05505005050",                      "addresses": [                          {                              "addressId": 1,                              "addressName": "Home",                              "street": "Palm Sokagi",                              "city": "Istanbul",                              "state": "Turkey",                              "postalCode": "35660",                              "customer": {                                  "customerId": 1,  ........................ |

Solution: add @JsonIgnore to Customer.java to address\_id related column

And @JsonBackReference to Address.java to customer\_id related column

1. LoyLoy Error:

Dfdsfdsf

## Possible Error:

### 1.Çağrıda Geçersiz Değişkenler Var:

This error returns while trying to add a new customer and a new address that is related with the customer via REST request:

|  |  |
| --- | --- |
| {  "name": "Ayse Fatma ",  "email": "aysefatma@gmail.com",  "phoneNumber": "05005909896",  "addresses": [  {  "addressName": "Home",  "street": "12345 Elm Street",  "city": "Springfielda",  "state": "Amerigaa",  "postalCode": "62701"  }  ]  } | could not prepare statement  [Çağrıda geçersiz bağımsız değişkenler var] [/\* insert for com.demo.springbootoracle.entity.Address  \*/insert into address (address\_name,city,customer\_id, postal\_code,state,street,address\_id) values (?,?,?,?,?,?,default)] |

|  |
| --- |
| o.s.orm.jpa.JpaTransactionManager : Exposing JPA transaction as JDBC [org.springframework.orm.jpa.vendor.HibernateJpaDialect$HibernateConnectionHandle@14052144] Hibernate: select customer\_seq.nextval from dual Hibernate: insert into customer (email, name, phone\_number, customer\_id) values (?, ?, ?, ?) 2024-12-01T16:11:32.756+03:00 WARN 13740 --- [Webservice1-OrMaSpr] [nio-8080-exec-2] o.h.engine.jdbc.spi.SqlExceptionHelper : SQL Error: 1, SQLState: 23000 2024-12-01T16:11:32.757+03:00 ERROR 13740 --- [Webservice1-OrMaSpr] [nio-8080-exec-2] o.h.engine.jdbc.spi.SqlExceptionHelper : ORA-00001: benzersiz kural (SYSTEM.SYS\_C008344) ihlal edildi |

Solutions:

1. Verifying relationships in Entites: @ManyToOne - Address, @OneToMany - Cust.
2. Ensure the Cascade Type
3. Check CustomreService
4. Address Table: has customer\_id column as a foreign key, ref to customer table
   * DESC address
   * DESCRIBE address

Address tablosunun içeriğini verir:

|  |
| --- |
|  |

1. Check sequences in Oracle:
   1. SELECT sequence\_name, min\_value, max\_value, increment\_by, last\_number FROM all\_sequences WHERE sequence\_name IN ('CUSTOMER\_SEQ', 'ADDRESS\_SEQ');



* 1. İf null:

CREATE SEQUENCE customer\_seq START WITH 1 INCREMENT BY 1; CREATE SEQUENCE address\_seq START WITH 1 INCREMENT BY 1;

* 1. If not null: sorguya sequence dönüyor: The related sequence is configured in Oracle.

|  |
| --- |
| CUSTOMER\_SEQ 1 9999999999999999999999999999 1 9 |

Min Value: 1

Max Value: 99999999999999999

Incement by: 1

Last Number: 9 (next value will be 10)

* 1. Address sequence is null:

CREATE SEQUENCE address\_seq

START WITH 1 INCREMENT BY 1 NOCACHE;

//NOCACHE: default cahce 20.

* 1. asdas

2.SYSTEM. SYS\_C008341

Error on Postman while adding new User:

|  |
| --- |
| could not execute statement [ORA-00001: benzersiz kural (SYSTEM.SYS\_C008341) ihlal edildi ] [/\* insert for com.demo.springbootoracle.entity.Address \*/insert into address (address\_name,city,customer\_id,postal\_code,state,street,address\_id) values (?,?,?,?,?,?,?)]; SQL [/\* insert for com.demo.springbootoracle.entity.Address \*/insert into address (address\_name,city,customer\_id,postal\_code,state,street,address\_id) values (?,?,?,?,?,?,?)]; constraint [SYSTEM.SYS\_C008341] |

**SYSTEM.SYS\_C008341**: unique constraint violation on ADDRESS table.

Bu tarz SYS hatalarının detayları Oracle SQL Developer tarafında şu şekilde çekilebilir:

|  |  |
| --- | --- |
| SELECT constraint\_name, column\_name FROM all\_cons\_columns WHERE constraint\_name = 'SYS\_C008341'; |  |
| SELECT constraint\_name, column\_name FROM all\_cons\_columns WHERE constraint\_name = 'SYS\_C008344'; |  |

For SYS\_C008341, it gives ADDRESS\_ID. Check whether the address\_seq is properly generating unique values:

|  |  |
| --- | --- |
| SELECT address\_seq.nextval FROM dual; |  |

In sum, the problem occured because Oracle SQL does not hace