**Spring Boot- Rest API- Get or Select Operation**

**Step-1- Create Spring Boot Project (As per previous session)**

**Step-2- Add Maven dependency into pom.xml file as per below**

<dependency>

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

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

</dependency>

<dependency>

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

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

<scope>test</scope>

</dependency>

<dependency>

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

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

</dependency>

<dependency>

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

<artifactId>spring-boot-starter-data-jpa</artifactId>

</dependency>

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

</dependency>

**Step-3- Go to application.properties file and make the below changes into it.**

#MYSQL Connection code

server.port=8082

spring.datasource.driver-class-name=com.mysql.jdbc.Driver

spring.datasource.url=jdbc:mysql://localhost:3306/restapiaprilbatch

spring.datasource.username=root

spring.datasource.password=root

spring.jpa.hibernate.ddl-auto=update

spring.jpa.show-sql=true

spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL8Dialect

**Step-4- Create Model Class as Employee**

package com.example.demo.model;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.Table;

@Entity

@Table(name = "Employee")

public class Employee {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private int id; // Integer , long -Long

private String name;

private String city;

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getCity() {

return city;

}

public void setCity(String city) {

this.city = city;

}

@Override

public String toString() {

return "Employee [id=" + id + ", name=" + name + ", city=" + city + "]";

}

}

**Step-5- Create Service and its implementation**

**import** com.example.demo.model.Employee;

**public** **interface** EmployeeService {

**public** Employee getEmployeeById(Integer Id);

}

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

import org.springframework.stereotype.Service;

import com.example.demo.model.Employee;

import com.example.demo.repository.EmployeeRepository;

@Service

public class EmployeeServiceImpl implements EmployeeService {

@Autowired

private EmployeeRepository employeeRepository;

@Override

public Employee getEmployeeById(Integer Id) {

Employee employee=employeeRepository.findById(Id);// null

return employee;

}

}

**Step-6- Create Respository for Employee**

**import** java.io.Serializable;

**import** org.springframework.data.repository.CrudRepository;

**import** org.springframework.stereotype.Repository;

**import** com.example.demo.model.Employee;

@Repository

**public** **interface** EmployeeRepository **extends** CrudRepository<Employee, Serializable> {

**public** Employee findById(Integer Id);

}

**Step-7- Design Employee Rest Controller**

package com.example.demo.controller;

import java.util.ArrayList;

import java.util.List;

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

import org.springframework.http.ResponseEntity;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.RestController;

import com.example.demo.model.Employee;

import com.example.demo.service.EmployeeService;

@RestController

public class EmployeeController {

// inject the EmployeeService here

@Autowired

private EmployeeService employeeService;

@GetMapping("/get/{id}")

**public** ResponseEntity<Employee> getEmployeeById(@PathVariable("id") Integer id) {

Employee employee = employeeService.getEmployeeById(id);

**return** ResponseEntity.*ok*().body(employee);

}

}

**Spring Boot- Rest API- Delete Operation**

**Step-1- Create Spring Boot Project (As per previous session)**

**Step-2- Add Maven dependency into pom.xml file as per below**

<dependency>

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

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

</dependency>

<dependency>

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

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

<scope>test</scope>

</dependency>

<dependency>

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

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

</dependency>

<dependency>

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

<artifactId>spring-boot-starter-data-jpa</artifactId>

</dependency>

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

</dependency>

**Step-3- Go to application.properties file and make the below changes into it.**

#MYSQL Connection code

server.port=8082

spring.datasource.driver-class-name=com.mysql.jdbc.Driver

spring.datasource.url=jdbc:mysql://localhost:3306/restapiaprilbatch

spring.datasource.username=root

spring.datasource.password=root

spring.jpa.hibernate.ddl-auto=update

spring.jpa.show-sql=true

spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL8Dialect

**Step-4- Create Model Class as Employee**

package com.example.demo.model;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.Table;

@Entity

@Table(name = "Employee")

public class Employee {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private int id; // Integer , long -Long

private String name;

private String city;

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getCity() {

return city;

}

public void setCity(String city) {

this.city = city;

}

@Override

public String toString() {

return "Employee [id=" + id + ", name=" + name + ", city=" + city + "]";

}

}

**Step-5- Create Service and its implementation**

**import** com.example.demo.model.Employee;

**public** **interface** EmployeeService {

**public void** deleteEmployee(Integer id);

}

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

import org.springframework.stereotype.Service;

import com.example.demo.model.Employee;

import com.example.demo.repository.EmployeeRepository;

@Service

public class EmployeeServiceImpl implements EmployeeService {

@Autowired

private EmployeeRepository employeeRepository;

@Override

**public** **void** deleteEmployee(Integer id) {

employeeRepository.deleteById(id);

}

}

**Step-6- Create Respository for Employee**

**import** java.io.Serializable;

**import** org.springframework.data.repository.CrudRepository;

**import** org.springframework.stereotype.Repository;

**import** com.example.demo.model.Employee;

@Repository

**public** **interface** EmployeeRepository **extends** CrudRepository<Employee, Serializable> {

}

**Step-7- Design Employee Rest Controller**

package com.example.demo.controller;

import java.util.ArrayList;

import java.util.List;

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

import org.springframework.http.ResponseEntity;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.RestController;

import com.example.demo.model.Employee;

import com.example.demo.service.EmployeeService;

@RestController

public class EmployeeController {

// inject the EmployeeService here

@Autowired

private EmployeeService employeeService;

@DeleteMapping("/delete/{id}")

**public** **void** deleteEmployeeById(@PathVariable("id") Integer id) **throws** Exception {

employeeService.deleteEmployee(id);

}

}