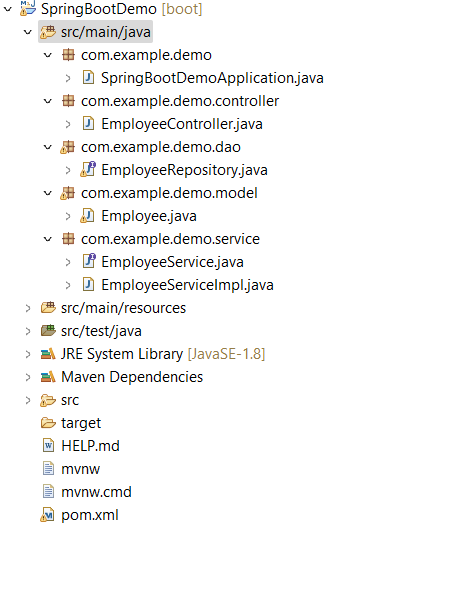
**Spring Boot- Rest API- Save Operation**

Example-1

Design the REST API to to store or save the Employee data into database-



**Step-1 – Create Spring Boot Project(As Per First Session)**

**Step-2- Create Model Classs as Employee**

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-3 – Design Service and Service Implementation**

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

**public** Employee saveEmployee(Employee employee);

}

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

import org.springframework.stereotype.Service;

import com.example.demo.dao.EmployeeRepository;

import com.example.demo.model.Employee;

@Service

public class EmployeeServiceImpl implements EmployeeService {

@Autowired

private EmployeeRepository employeeRepository;

@Override

public Employee saveEmployee(Employee employee) {

return employeeRepository.save(employee);

}

}

**Step-4- Design Repository for Employee**

import java.io.Serializable;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.stereotype.Repository;

import com.example.demo.model.Employee;

@Repository

public interface EmployeeRepository extends JpaRepository<Employee, Integer> {

}

**Step-5- Design the Rest Controller Class as EmployeeController**

@RestController

@RequestMapping("/employee")

**public** **class** EmployeeController {

@Autowired

**private** EmployeeService employeeService;

@PostMapping("/save")

**public** ResponseEntity<Employee> saveEmployee(@RequestBody Employee employee) {

Employee emp = employeeService.saveEmployee(employee);

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

}

**Step-6 – Go to application.properties file and make the below changes**

application.properties

#MysqL

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

spring.datasource.url=jdbc:mysql://localhost:3306/test?autoReconnect=true&useSSL=false

spring.datasource.username=root

spring.datasource.password=root

spring.jpa.hibernate.ddl-auto=create

spring.jpa.show-sql=true

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

**Step-7- Add Maven dependency into pom.xml file**

<dependency>

<groupId>mysql</groupId>

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

<version>8.0.12</version>

</dependency>

<dependency>

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

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

</dependency>