**1. Create a Spring Boot Project using Spring Initializr**

Go to [Spring Initializr](https://start.spring.io/) and create a new project with the following dependencies:

* **Spring Web** (For REST API)
* **Spring Data JPA** (For ORM)
* **H2 Database** (For an in-memory database)

Generate the project and extract the ZIP file.

**2. Project Structure**

css

CopyEdit

spring-boot-crud/

│── src/main/java/com/example/demo/

│ ├── DemoApplication.java

│ ├── controller/EmployeeController.java

│ ├── model/Employee.java

│ ├── repository/EmployeeRepository.java

│ ├── service/EmployeeService.java

│── src/main/resources/

│ ├── application.properties

│── pom.xml

**3. Configure application.properties**

Modify src/main/resources/application.properties:

properties

CopyEdit

spring.datasource.url=jdbc:h2:mem:testdb

spring.datasource.driverClassName=org.h2.Driver

spring.datasource.username=sa

spring.datasource.password=

spring.jpa.database-platform=org.hibernate.dialect.H2Dialect

spring.h2.console.enabled=true

spring.h2.console.path=/h2-console

* This enables the H2 database console at http://localhost:8080/h2-console.

**4. Create Employee Model**

Create Employee.java in src/main/java/com/example/demo/model/:

java

CopyEdit

package com.example.demo.model;

import jakarta.persistence.\*;

@Entity

@Table(name = "employees")

public class Employee {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

@Column(nullable = false)

private String name;

@Column(nullable = false)

private String department;

@Column(nullable = false, unique = true)

private String email;

public Employee() {}

public Employee(String name, String department, String email) {

this.name = name;

this.department = department;

this.email = email;

}

public Long getId() {

return id;

}

public void setId(Long id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getDepartment() {

return department;

}

public void setDepartment(String department) {

this.department = department;

}

public String getEmail() {

return email;

}

public void setEmail(String email) {

this.email = email;

}

}

* This entity maps to the employees table.

**5. Create Repository**

Create EmployeeRepository.java in src/main/java/com/example/demo/repository/:

java

CopyEdit

package com.example.demo.repository;

import com.example.demo.model.Employee;

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

import org.springframework.stereotype.Repository;

@Repository

public interface EmployeeRepository extends JpaRepository<Employee, Long> {

}

* Extends JpaRepository to perform database operations.

**6. Create Service Layer**

Create EmployeeService.java in src/main/java/com/example/demo/service/:

java

CopyEdit

package com.example.demo.service;

import com.example.demo.model.Employee;

import com.example.demo.repository.EmployeeRepository;

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

import org.springframework.stereotype.Service;

import java.util.List;

import java.util.Optional;

@Service

public class EmployeeService {

@Autowired

private EmployeeRepository employeeRepository;

public List<Employee> getAllEmployees() {

return employeeRepository.findAll();

}

public Optional<Employee> getEmployeeById(Long id) {

return employeeRepository.findById(id);

}

public Employee createEmployee(Employee employee) {

return employeeRepository.save(employee);

}

public Employee updateEmployee(Long id, Employee employeeDetails) {

Employee employee = employeeRepository.findById(id)

.orElseThrow(() -> new RuntimeException("Employee not found"));

employee.setName(employeeDetails.getName());

employee.setDepartment(employeeDetails.getDepartment());

employee.setEmail(employeeDetails.getEmail());

return employeeRepository.save(employee);

}

public void deleteEmployee(Long id) {

employeeRepository.deleteById(id);

}

}

**7. Create Controller**

Create EmployeeController.java in src/main/java/com/example/demo/controller/:

java

CopyEdit

package com.example.demo.controller;

import com.example.demo.model.Employee;

import com.example.demo.service.EmployeeService;

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

import org.springframework.http.ResponseEntity;

import org.springframework.web.bind.annotation.\*;

import java.util.List;

import java.util.Optional;

@RestController

@RequestMapping("/employees")

public class EmployeeController {

@Autowired

private EmployeeService employeeService;

@GetMapping

public List<Employee> getAllEmployees() {

return employeeService.getAllEmployees();

}

@GetMapping("/{id}")

public ResponseEntity<Employee> getEmployeeById(@PathVariable Long id) {

Optional<Employee> employee = employeeService.getEmployeeById(id);

return employee.map(ResponseEntity::ok)

.orElseGet(() -> ResponseEntity.notFound().build());

}

@PostMapping

public Employee createEmployee(@RequestBody Employee employee) {

return employeeService.createEmployee(employee);

}

@PutMapping("/{id}")

public ResponseEntity<Employee> updateEmployee(@PathVariable Long id, @RequestBody Employee employeeDetails) {

return ResponseEntity.ok(employeeService.updateEmployee(id, employeeDetails));

}

@DeleteMapping("/{id}")

public ResponseEntity<Void> deleteEmployee(@PathVariable Long id) {

employeeService.deleteEmployee(id);

return ResponseEntity.noContent().build();

}

}

**8. Run the Application**

Run DemoApplication.java:

java

CopyEdit

package com.example.demo;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

public class DemoApplication {

public static void main(String[] args) {

SpringApplication.run(DemoApplication.class, args);

}

}

**9. Test Using Postman**

**Endpoints**

1. **Create Employee (POST)**  
   POST http://localhost:8080/employees  
   **Body (JSON)**:

json

CopyEdit

{

"name": "John Doe",

"department": "IT",

"email": "john.doe@example.com"

}

1. **Get All Employees (GET)**  
   GET http://localhost:8080/employees
2. **Get Employee by ID (GET)**  
   GET http://localhost:8080/employees/{id}
3. **Update Employee (PUT)**  
   PUT http://localhost:8080/employees/{id}  
   **Body (JSON)**:

json

CopyEdit

{

"name": "John Updated",

"department": "HR",

"email": "john.updated@example.com"

}

1. **Delete Employee (DELETE)**  
   DELETE http://localhost:8080/employees/{id}

**10. Access H2 Console**

* Open http://localhost:8080/h2-console
* Use JDBC URL: jdbc:h2:mem:testdb
* Login:
  + **User:** sa
  + **Password:** *(leave empty)*
* Run SQL:

sql

CopyEdit

SELECT \* FROM employees;