

## API REST

-----  
  
package com.example.demo;  
  
-----

```
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
com.example.demo
@SpringBootApplication
public class ProjectApiApplication {

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

}
```

com.example.demo.controller  
  
-----

Employee controller.java

```
package com.example.demo.controller;
import java.util.List;
```

```
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.DeleteMapping;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.PutMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;
```

```
import com.example.demo.model.Employee;
import com.example.demo.service.EmployeeService;
```

```
@RestController
```

```
@RequestMapping("/api/employees")
```

```
public class EmployeeController {
```

```
    private EmployeeService service;
```

```
    public EmployeeController(EmployeeService service) {
```

```
        super();
```

```
        this.service = service;
```

```
    }
```

```
//build create employee REST API
```

```

    @PostMapping()

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

        return new
ResponseEntity<Employee>(service.saveEmployee(employee),HttpStatus.CREAT
ED);

    }

```

```

//Get All Employees

@GetMapping

public List<Employee> getAllEmployees(){

    return service.getAllEmployees();

}

//build getEmployeeById REST API
//http://localhost:8080/api/employees/1

@GetMapping("{id}")

public ResponseEntity<Employee> getEmployeeById(@PathVariable("id")
long employeeId){

    return new
ResponseEntity<Employee>(service.getEmployeeById(employeeId),HttpStatus.O
K);

}

//build update

```

```

//http://localhost:8080/api/employees/1

@PutMapping("{id}")

public ResponseEntity<Employee> updateEmployee(@PathVariable("id") long
employeeId,@RequestBody Employee employee){

```

```
        return new  
        ResponseEntity<Employee>(service.updateEmployee(employee,employeeId),Http  
        Status.OK);  
    }  
}
```

```
//delete  
  
@DeleteMapping("{id}")  
//http://localhost:8080/api/employees/1  
  
public ResponseEntity<String> deleteEmployee(@PathVariable("id") long  
employeeId){  
    service.deleteEmployee(employeeId);  
    return new ResponseEntity<String>("Employee deleted  
successfully",HttpStatus.OK);  
}  
}
```

com.example.demo.exception

-----

Exception.java

```
package com.example.demo.exception;
```

```
import org.springframework.http.HttpStatus;
```

```
import org.springframework.web.bind.annotation.ResponseStatus;
```

```
@ResponseStatus(value=HttpStatus.NOT_FOUND)
```

```
public class Exception extends RuntimeException {

    private static final long serialVersionUID = 1L;
    private String resourceNmae;
    private String fieldname;
    private Object fieldValue;

    public Exception(String resourceNmae, String fieldname, Object fieldValue) {
        super();
        this.resourceNmae = resourceNmae;
        this.fieldname = fieldname;
        this.fieldValue = fieldValue;
    }

    public String getResourceNmae() {
        return resourceNmae;
    }

    public String getFieldname() {
        return fieldname;
    }

    public Object getFieldValue() {
        return fieldValue;
    }
}
```

com.example.demo.model

-----

Employee.java

```
package com.example.demo.model;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;

import lombok.Data;
@SuppressWarnings("unused")

@Data
@Entity
@Table(name="devika")
public class Employee {
    @Id
    @GeneratedValue(strategy=GenerationType.IDENTITY)
    private long id;
    public long getId() {
        return id;
    }
}
```

```
public void setId(long id) {
    this.id = id;
}
public String getFirstName() {
    return firstName;
}
public void setFirstName(String firstName) {
    this.firstName = firstName;
}
public String getLastname() {
    return lastname;
}
public void setLastname(String lastname) {
    this.lastname = lastname;
}
public String getEmail() {
    return email;
}
public void setEmail(String email) {
    this.email = email;
}
@Column(name="first_name")
private String firstName;
@Column(name="last_name")
private String lastname;
```

```
        @Column(name="email")
private String email;

}
```

com.example.demo.repository

-----

EmployeeRepository.java

```
package com.example.demo.repository;
```

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

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

```
public interface EmployeeRepository extends JpaRepository<Employee,Long>{
```

```
}
```

com.example.demo.service

-----

EmployeeService.java



```
package com.example.demo.service;
```

```
import java.util.List;
```

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

```
public interface EmployeeService {
```

```
    Employee saveEmployee(Employee employee);
```

```
    List<Employee> getAllEmployees();
```

```
    Employee getEmployeeByID(long id);
```

```
    Employee updateEmployee(Employee employee,long id);
```

```
    void deleteEmployee(long id);
```

```
}
```

```
com.example.demo.service.impl
```

```
-----
```

```
EmployeeImpl.java
```

```
package com.example.demo.service.impl;
```

```
import java.util.List;
```

```
import java.util.Optional;
```

```
import org.springframework.stereotype.Service;
```

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

```
import com.example.demo.repository.EmployeeRepository;
```

```
import com.example.demo.service.EmployeeService;
```

```
@Service
```

```
public class EmployeeImpl implements EmployeeService {
```

```
    private EmployeeRepository employeeRepository;
```

```
    public EmployeeImpl(EmployeeRepository employeeRepository) {
```

```
        super();
```

```
        this.employeeRepository=employeeRepository;
```

```
    }
```

```
    @Override
```

```
    public Employee saveEmployee(Employee employee) {
```

```
        return employeeRepository.save(employee);
```

```
    }
```

```
    @Override
```

```
public List<Employee> getAllEmployees() {  
  
    return employeeRepository.findAll();  
}
```

@Override

```
public Employee getEmployeeByID(long id) {  
    Optional<Employee> employee=employeeRepository.findById(id);  
    if(employee.isPresent()) {  
        return employee.get();  
    }  
    else {  
        return null;  
        //throw new Exception("Employee","Id",id);  
    }  
  
}
```

@Override

```
public Employee updateEmployee(Employee employee, long id) {  
    //first check whether it is present
```

```
Employee existing=employeeRepository.findById(id).orElseThrow(()  
-> null);  
  
existing.setFirstName(employee.getFirstName());  
existing.setLastname(employee.getLastname());  
existing.setEmail(employee.getEmail());  
  
employeeRepository.save(existing);  
return null;  
}
```

@Override

```
public void deleteEmployee(long id) {  
    //check existing  
    employeeRepository.findById(id).orElseThrow(() -> null);  
    employeeRepository.deleteById(id);  
  
}  
  
}
```

application.properties

-----

spring.datasource.url=jdbc:mysql://localhost:3306/API?useSSL=false

spring.datasource.username=root

spring.datasource.password=1521

spring.jpa.open-in-view=false

#Hibernate

spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL5InnoDBDialect

#create,update

spring.jpa.hibernate.ddl-auto=update

pom.xml

-----

<?xml version="1.0" encoding="UTF-8"?>

<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  
http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<parent>

```
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-parent</artifactId>
<version>2.7.5</version>
<relativePath/> <!-- lookup parent from repository -->
</parent>
<groupId>com.example</groupId>
<artifactId>ProjectAPI</artifactId>
<version>0.0.1-SNAPSHOT</version>
<name>ProjectAPI</name>
<description>Demo project for Spring Boot</description>
<properties>
    <java.version>1.8</java.version>
</properties>
<dependencies>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-data-jpa</artifactId>
    </dependency>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-web</artifactId>
    </dependency>

    <dependency>
        <groupId>com.h2database</groupId>
```

```
        <artifactId>h2</artifactId>
        <scope>runtime</scope>
    </dependency>
    <dependency>
        <groupId>com.mysql</groupId>
        <artifactId>mysql-connector-j</artifactId>
        <scope>runtime</scope>
    </dependency>
    <dependency>
        <groupId>org.projectlombok</groupId>
        <artifactId>lombok</artifactId>
        <optional>true</optional>
    </dependency>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-test</artifactId>
        <scope>test</scope>
    </dependency>
</dependencies>

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

```
<configuration>
  <excludes>
    <exclude>

<groupId>org.projectlombok</groupId>
      <artifactId>lombok</artifactId>
    </exclude>
  </excludes>
</configuration>
</plugin>
</plugins>
</build>

</project>
```