EmployemntManagementSystem source codes

Controller

```
package com.Employee.employee.Controller;

import com.Employee.employee.Exception2.ResourceNotFoundException;
import com.Employee.employee.Model.Employee;
import com.Employee.employee.Repository.EmployeeRepository;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.*;

import java.util.HashMap;
import java.util.List;
import java.util.Map;

@RestController
@CrossOrigin
@RequestMapping ("api/v2/")
public class EmployeeController {
    @Autowired
    private EmployeeRepository employeeRepository;
    @GetMapping("/Employee")

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

    @PostMapping("/Employee")
    public Employee addEmployee(@RequestBody Employee employee) {
        return employeeRepository.save(employee);

    }

    @GetMapping("/Employee/{id}")
    public ResponseEntity<Employee> getEmployeeByID(@PathVariable int id) {
        Employee employee = employeeRepository.findById(id)
                .orElseThrow(() -> new ResourceNotFoundException("Invalid
Id: " + id));
        return ResponseEntity.ok(employee);
    }

    @DeleteMapping("/Employee/{id}")
    public ResponseEntity<Map<String,Boolean>> deleteEmployee(@PathVariable
int id)
    {
        Employee employee = employeeRepository.findById(id)

                .orElseThrow(() -> new ResourceNotFoundException("Invalid
Id: "+id));

        employeeRepository.delete(employee);

        Map<String,Boolean> response = new HashMap<>();
        response.put("course was Deleted : ",Boolean.TRUE);
        return ResponseEntity.ok(response);
    }
    @PutMapping("/employee/{id}")
    public ResponseEntity<Employee> updateEmployee(@PathVariable int id,
```

```
@RequestBody Employee employee)
    {

        Employee employee1 = employeeRepository.findById(id)
                .orElseThrow(() -> new ResourceNotFoundException("Invalid
Id: "+id));

        employee1.setEmployee_Name(employee1.getEmployee_Name());
        employee1.setAddress(employee1.getAddress());
        employee1.setId(employee1.getId());

        // saving the new values

        return ResponseEntity.ok(employeeRepository.save(employee1));
    }

}
```

exception handling

```
package com.Employee.employee.Exception2;

import org.springframework.http.HttpStatus;
import org.springframework.web.bind.annotation.ResponseStatus;

import java.io.Serial;

@ResponseStatus(HttpStatus.NOT_FOUND)
public class ResourceNotFoundException extends RuntimeException{

    @Serial
    private static final long serialVersionUID=1L;

    public ResourceNotFoundException(String msg) {super(msg);}
}
```

employee model

```
package com.Employee.employee.Model;

import lombok.Data;

import javax.persistence.Entity;
import javax.persistence.Table;

@Entity
@Table
@Data

public class Employee {

    private int Id;
    private String Employee_Name;
    private String Address;
```

```
    public Employee(int id, String employee_Name, String address) {
        Id = id;
        Employee_Name = employee_Name;
        Address = address;
    }

    public int getId() {
        return Id;
    }

    public String getEmployee_Name() {
        return Employee_Name;
    }

    public String getAddress() {
        return Address;
    }

    public void setId(int id) {
        Id = id;
    }

    public void setEmployee_Name(String employee_Name) {
        Employee_Name = employee_Name;
    }

    public void setAddress(String address) {
        Address = address;
    }
}
```

employee Repository

```
package com.Employee.employee.Repository;

import com.Employee.employee.Model.Employee;
import org.springframework.data.jpa.repository.JpaRepository;

public interface  EmployeeRepository extends JpaRepository
<Employee,Integer> {
}
```

employee Services

```
package com.Employee.employee.Repository;



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

@Service
public class EmployeeServiceImpl implements EmployeeService {
```

```
    @Autowired
    private EmployeeRepository employeeRepository;

    @Override
    public Employee createEmployee(Employee employee) {
        return employeeRepository.save(employee);
    }

    @Override
    public Employee getEmployeeById(Long id) {
        return employeeRepository.findById(id).orElse(null);
    }
```