

THE STATE UNIVERSITY OF ZANZIBAR (SUZA)



SCHOOL OF BUSINESS (SoB)

DEPARTMENT OF COMPUTER SCIENCE AND INFORMATION

TECHNOLOGY

BACHELOR DEGREE IN INFORMATION TECHNOLOGY WITH

ACCOUNTING


STUDENT NAME: IBTISAM ABDULLA RASHID

REGISTRATION: BITA/5/21/013/TZ

TASK: MARKUP TEST

QUESTION 1:

a. Project



Project
☐ Gradle - Groovy ☐ Gradle - Kotlin ☒ **Java** ☐ Kotlin ☐ Groovy
☒ **Maven**

Language
☒ **Java** ☐ Kotlin ☐ Groovy

Spring Boot
☐ 3.2.0 (SNAPSHOT) ☐ 3.2.0 (M1) ☐ 3.1.3 (SNAPSHOT) ☐ 3.1.2
☐ 3.0.10 (SNAPSHOT) ☐ 3.0.9 ☐ 2.7.15 (SNAPSHOT) ☒ **2.7.14**

Project Metadata

Group

Artifact

Name

Description

Package name

Packaging ☒ **Jar** ☐ War

Java ☐ 20 ☒ **17** ☐ 11 ☐ 8

b. Dependencies

Dependencies

ADD DEPENDENCIES... CTRL + B

Spring Data JPA **SQL**

Persist data in SQL stores with Java Persistence API using Spring Data and Hibernate.

Lombok **DEVELOPER TOOLS**

Java annotation library which helps to reduce boilerplate code.

MySQL Driver **SQL**

MySQL JDBC driver.

Spring Web **WEB**

Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.

c. **Application.properties**

```
spring.datasource.url=jdbc:mysql://localhost:3306/ems_db
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
spring.datasource.username=root
spring.datasource.password=
spring.jpa.show-sql=true

spring.jpa.hibernate.ddl-auto=update
```

d. **Model**

e. **EmployeeModel**

```
package EmployeeManagementSystem.ems.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;

@Entity
@Table(name = "employees")

public class Employee {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private long id;

    @Column(name = "first_name")
    private String first_name;

    @Column(name = "last_name")
    private String last_name;

    @Column(name = "email_id")
    private String email_id;

    public Employee() {
```

```
}

public Employee(String first_name, String last_name, String email_id) {
    super();
    this.first_name = first_name;
    this.last_name = last_name;
    this.email_id = email_id;
}

public long getId() {
    return id;
}

public void setId(long id) {
    this.id = id;
}

public String getFirst_name() {
    return first_name;
}

public void setFirst_name(String first_name) {
    this.first_name = first_name;
}

public String getLast_name() {
    return last_name;
}

public void setLast_name(String last_name) {
    this.last_name = last_name;
}

public String getEmail_id() {
    return email_id;
}

public void setEmail_id(String email_id) {
    this.email_id = email_id;
}

}
```

f. EmployeeRepository

```
package EmployeeManagementSystem.ems.repositoty;

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

import EmployeeManagementSystem.ems.model.Employee;

public interface EmployeeRepository extends JpaRepository<Employee, Long > {

}
```

g. Service

```
// Get All Employees

@GetMapping("/employees")
public List<Employee> getAllEmployee(){
    return employeeRepository.findAll();
}

// Create Employees rest api

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

// Get Employee by Id rest api

@GetMapping("/employees/{id}")
public ResponseEntity<Employee> getEmployeeById(@PathVariable Long id) {
    Employee employee = employeeRepository.findById(id)
        .orElseThrow(() -> new ResourceNotFoundException("Employee Not
Exist with Id :" + id));
    return ResponseEntity.ok(employee);
}

// Delete Employee rest api

@DeleteMapping("/employee/{id}")
```

```

    public ResponseEntity<Map<String, Boolean>> deleteCustomer(@PathVariable Long
id){
        Employee employee = employeeRepository.findById(id)
            .orElseThrow(() -> new ResourceNotFoundException("Employee Not
Exist With Id :" + id));

        employeeRepository.delete(employee);
        Map<String, Boolean> response = new HashMap<>();
        response.put("deleted", Boolean.TRUE);
        return ResponseEntity.ok(response);
    }

```

h. Controller

```

package EmployeeManagementSystem.ems.controller;

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

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.CrossOrigin;
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.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

import EmployeeManagementSystem.ems.exception.ResourceNotFoundException;
import EmployeeManagementSystem.ems.model.Employee;
import EmployeeManagementSystem.ems.repositoty.EmployeeRepository;

@CrossOrigin
@RestController
@RequestMapping("/api/v1/")

public class EmployeeController {

    @Autowired
    private EmployeeRepository employeeRepository;

```

```

// Get All Employees

@GetMapping("/employees")
public List<Employee> getAllEmployee(){
    return employeeRepository.findAll();
}

// Create Employees rest api

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

// Get Employee by Id rest api

@GetMapping("/employees/{id}")
public ResponseEntity<Employee> getEmployeeById(@PathVariable Long id) {
    Employee employee = employeeRepository.findById(id)
        .orElseThrow(() -> new ResourceNotFoundException("Employee Not
Exist with Id :" + id));
    return ResponseEntity.ok(employee);
}

// Delete Employee rest api

@DeleteMapping("/employee/{id}")
public ResponseEntity<Map<String, Boolean>> deleteCustomer(@PathVariable Long
id){
    Employee employee = employeeRepository.findById(id)
        .orElseThrow(() -> new ResourceNotFoundException("Employee Not
Exist With Id :" + id));

    employeeRepository.delete(employee);
    Map<String, Boolean> response = new HashMap<>();
    response.put("deleted", Boolean.TRUE);
    return ResponseEntity.ok(response);
}
}

```

i. API

POST localhost:8080/api/v1/employees

localhost:8080/api/v1/employees

POST localhost:8080/api/v1/employees

Params Authorization Headers (8) Body Pre-request Script Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary JSON Beautify

```
1 {
2   "first_name": "Ibtisam",
3   "last_name": "Rashid",
4   "email_id": "ibty@gmail.com"
5 }
```

Body Cookies Headers (8) Test Results 200 OK 455 ms 333 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": 1,
3   "first_name": "Ibtisam",
4   "last_name": "Rashid",
5   "email_id": "ibty@gmail.com"
6 }
```

phpMyAdmin

Recent Favorites

- New
- ems_db
- form
- information_schema
- mysql
- performance_schema
- phpmyadmin
- tbs_db
- test
- tgms_db

Server: 127.0.0.1 » Database: ems_db » Table: employees

Browse Structure SQL Search Insert Export Import Privileges Operations Triggers

Showing rows 0 - 0 (1 total, Query took 0.0003 seconds.)

SELECT * FROM `employees`

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all Number of rows: 25 Filter rows: Search this table

Extra options

	id	email_id	first_name	last_name
<input type="checkbox"/>	1	ibty@gmail.com	Ibtisam	Rashid

Check all With selected: Edit Copy Delete Export

Show all Number of rows: 25 Filter rows: Search this table