**Assessment Parameters**

* *Complete flow of the application with exception handling -70%*
* *Comments/best practice, coding standards- 10%*
* *Execution of the application (Output) – 20%*
* *ScreenShot should be submitted along with the solution*
* *The solution (Project) created by the trainee should have the name like AppName\_Empid Ex:ABCCorp\_675467*
* *Code with compilation errors will not be considered for evaluation*

*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

**Training Management System**

Develop a Spring Boot application with REST API for the Training institute to manage their Training modules in DB.

Requirement will be:

1. Display all Modules
2. Add Module
3. Update Module
4. Delete Module

Develop a Spring Boot Application using Spring REST with JPA Integration.

**SECTION -1**

**CRUD Operations:**

1. Fetch Module based on ID

When a GET request is sent with ID following data should be displayed.

ID, NAME, FEES , DURATION

The details should be displayed in JSON format.

Duration is number of hours.

1. ADD Module

The user should able to ADD Module with details like ID, NAME, FEES and DURATION using POST request.

1. UPDATE Module

The User should able to Update Module details based on ID using

PUT request.

1. DELETE Product

The User should able to delete a particular Module from DB based on ID.

Hint: Use Annotations like @GetMapping,@PostMapping,@PutMapping and @DeleteMapping along with @RestController etc

**SECTION-2**

**Data Validation using Annotations.**

When user performed any CRUD Operations the application should perform the following validations like

1. ID Not Null
2. NAME Not Null, Min Length 3 and Max Length 20
3. FEES Max Value should not exceed 50,000/-
4. DURATION number of hours should be 2 to 3 digits

Hint: Use Annotation for Entity Validations. Like @NotNull etc..and check error messages in Console as Forms/ GUI is not implemented in this application.

**The above given requirement should be implemented on Spring Boot with**

**REST API , JPA Integration in Oracle DB.**

**Use POSTMAN Client/Software tool for testing all CRUD Operations.**

**SECTION -3**

Handle all exceptions by using Global Error Handling concept in Spring Boot.

Note: Follow Layered Architecture and implement layers like Service, DAO , Exception etc. This application is completely GUI / JSP independent all operations should be tested based on POSTMAN only.

**Mark Distribution: [ Marks: 70]**

|  |  |
| --- | --- |
| Display details from DB in POSTMAN client / Browser and Layered Architecture used | 15 |
| Correct Mappings and Annotations like GetMapping, PostMapping etc | 10 |
| Correct Service interface autowiring in Controller. | 10 |
| Correct DAO, Repository interface autowiring in Service. | 10 |
| ADD, UPDATE and DELETE using PostMan | 10 |
| Validations on Entity and using JPA integration | 10 |
| Best Practices and using Spring Boot property file configurations | 5 |

The following Table Script can be used to make data available for fetching on which other CRUD operations can be performed.

**Table Script:**

CREATE TABLE module\_master (ID NUMBER, NAME VARCHAR2(20), FEES (7,2), DURATION NUMBER(4) );

INSERT INTO module \_master VALUES(111,'Core Java', 12000,150);

INSERT INTO module \_master VALUES(112,'Cplus’, 9000, 120);

INSERT INTO module \_master VALUES(113,'Spring',15000,40);

INSERT INTO module \_master VALUES(114,'Angular',18000,50);

COMMIT;

***Note: Participants are kindly requested to follow the same table structure****.*

***Note: Please add ojdbc.jar separately as Maven may not provide by default in Spring Boot and application is completely on REST API and must be tested using Postman without forms or JSPs.***