

Phase – 3 Practice Project: Assisted Practice

1. Create a project to demonstrate microservices with Spring Boot.

- Code

- ✓ RestApiOneApplication.java

```
package com.example.test;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class RestApiOneApplication {

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

}
```

- ✓ PersonEntity.java

```
package com.example.test;

import jakarta.persistence.Column;
import jakarta.persistence.Entity;
import jakarta.persistence.GeneratedValue;
import jakarta.persistence.GenerationType;
import jakarta.persistence.Id;

@Entity
public class PersonEntity {

    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    @Column(name = "id", updatable = false, nullable = false)
    private Integer personId;
    @Column
    private String name;
    @Column
    private Integer age;

    public PersonEntity() {
        super();
    }

    public PersonEntity(Integer personId, String name, Integer age) {
        super();
        this.personId = personId;
        this.name = name;
        this.age = age;
    }

    public Integer getPersonId() {
        return personId;
    }

    public void setPersonId(Integer personId) {
```

```

        this.personId = personId;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public Integer getAge() {
        return age;
    }
    public void setAge(Integer age) {
        this.age = age;
    }
}

```

✓ PersonRepository.java

```

package com.example.test;

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

@Repository
public interface PersonRepository extends JpaRepository<PersonEntity,
Integer> {

}

```

✓ PersonService.java

```

package com.example.test;

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

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import org.springframework.web.client.RestTemplate;

@Service
public class PersonService {

    @Autowired
    PersonRepository personRepository;

    RestTemplate restTemplate = new RestTemplate();

    public PersonResonse getPerson(int personId) {
        final String uri =
"http://localhost:8082/webapitwo/hobby/{personId}";

        Map<String, Integer> params = new HashMap<String, Integer>();
        params.put("personId", personId);
    }
}

```

```

        String result = restTemplate.getForObject(uri, String.class,
params);
        PersonEntity pe=personRepository.findById(personId).get();
        PersonResonse pr=new PersonResonse();
        pr.setPersonId(pe.getPersonId());
        pr.setName(pe.getName());
        pr.setAge(pe.getAge());
        pr.setHobby(result);

        return pr;
    }
    public void addPerson(PersonEntity pe){
        personRepository.save(pe);
    }
}

```

✓ PersonController.java

```

package com.example.test;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.RestController;

@RestController
@RequestMapping(path = "/webapione")
public class PersonController {

    @Autowired
    PersonService personService;

    @RequestMapping("/person/{personId}")
    public PersonResonse getPerson(@PathVariable int personId){
        return personService.getPerson(personId);
    }

    @RequestMapping(method=RequestMethod.POST, value="/person")
    public void addPerson(@RequestBody PersonEntity pe ) {
        personService.addPerson(pe);
    }
}

```

✓ PersonResonse.java

```

package com.example.test;

public class PersonResonse {

    private Integer personId;
    private String name;
    private Integer age;
    private String hobby;
    public Integer getPersonId() {

```

```

        return personId;
    }
    public void setPersonId(Integer personId) {
        this.personId = personId;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public Integer getAge() {
        return age;
    }
    public void setAge(Integer age) {
        this.age = age;
    }
    public String getHobby() {
        return hobby;
    }
    public void setHobby(String result) {
        this.hobby = result;
    }
}

```

✓ application.properties

```

server.port=8081
spring.application.name=RestApiOne

```

✓ 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
https://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>3.1.0</version>
        <relativePath/> <!-- lookup parent from repository -->
    </parent>
    <groupId>com.example</groupId>
    <artifactId>RestApiOne</artifactId>
    <version>0.0.1-SNAPSHOT</version>
    <name>RestApiOne</name>
    <description>Demo project for Spring Boot </description>
    <properties>
        <java.version>20</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>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>
        </plugin>
    </plugins>
</build>

</project>

```

• Output

The screenshot shows a REST client interface with the following components:

- Top Bar:** Includes a "New Collection / New Request" button, a "Save" button, and a "Send" button.
- Request Section:**
 - Method:** POST
 - URL:** http://localhost:8081/webapione/person
 - Buttons:** Params, Auth, Headers (8), Body (selected), Pre-req., Tests, Settings, Cookies, Beautify.
- Response Section:**
 - Format:** JSON
 - Content:**

```

1 {
2   "name": "John",
3   "age": 23
4 }

```

- **Code**

- ✓ **RestApiTwoApplication.java**

```
package com.example.test;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class RestApiTwoApplication {

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

}
```

- ✓ **HobbyEntity.java**

```
package com.example.test;

import jakarta.persistence.Column;
import jakarta.persistence.Entity;
import jakarta.persistence.GeneratedValue;
import jakarta.persistence.GenerationType;
import jakarta.persistence.Id;

@Entity
public class HobbyEntity {

    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    @Column(name = "id", updatable = false, nullable = false)
    private Integer id;
    @Column
    private Integer personId;
    @Column
    private String name;

    public HobbyEntity() {
        super();
    }
    public HobbyEntity(Integer personId, String name) {
        super();
        this.personId = personId;
        this.name = name;
    }
    public Integer getPersonId() {
        return personId;
    }
    public void setPersonId(Integer personId) {
        this.personId = personId;
    }
    public String getName() {
        return name;
    }
}
```

```

        public void setName(String name) {
            this.name = name;
        }
    }
}

```

✓ HobbyRepository.java

```

package com.example.test;

import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.data.jpa.repository.Query;
import org.springframework.stereotype.Repository;

@Repository
public interface HobbyRepository extends JpaRepository<HobbyEntity,
Integer> {

    @Query("SELECT h.name FROM HobbyEntity h WHERE
h.personId=:personId")
    public String findByPersonId(Integer personId);

}

```

✓ HobbyService.java

```

package com.example.test;

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

@Service
public class HobbyService {

    @Autowired
    HobbyRepository hobbyRepository;

    public String findByPersonId(int personid){
        return hobbyRepository.findByPersonId(personid);
    }

    public void addHobby(HobbyEntity he){
        hobbyRepository.save(he);
    }

}

```

✓ HobbyController.java

```

package com.example.test;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.RestController;

@RestController
@RequestMapping(path = "/webapitwo")

```

```

public class HobbyController {

    @Autowired
    HobbyService hobbyService;

    @RequestMapping("/hobby/{personid}")
    public String findByPersonId(@PathVariable int personid) {
        return hobbyService.findByPersonId(personid);
    }

    @RequestMapping(method=RequestMethod.POST, value="/hobby")
    public void addHobby(@RequestBody HobbyEntity he ) {
        hobbyService.addHobby(he);
    }

}

```

✓ application.properties

```

server.port=8082
spring.application.name=RestApiTwo

```

• Output

The screenshot shows a REST client interface with the following details:

- Overview Tab:** Shows the request method as **POST** and the URL as **http://localhost:8082/**.
- Request Details:** The URL is **http://localhost:8082/webapitwo/hobby**. The method is **POST**.
- Body Tab:** The body is formatted as **JSON** and contains the following content:


```

{
  "personId": 1,
  "name": "badminton"
}

```
- Buttons:** There are buttons for **Save**, **Send**, **Beautify**, and **Cookies**.

POST http://localhost:8080/webapi/...

GET http://localhost:8081/webapion...

+

...

No Environment

GET

http://localhost:8081/webapione/person/1

Send

Save

1

Body

Cookies (1)

Headers (3)

Test Results

Status: 200 OK

Time: 66ms

Size: 187 B

Save Response

Pretty

Raw

Preview

JSON

1

{

2

"personId": 1,

3

"name": "John",

4

"age": 23,

5

"hobby": "badminton"

6

}