

## WEEK – 5

- 1) Create Microservices for account and loan

### AccountController.java

```
package com.cognizant.account.controller;

import org.springframework.web.bind.annotation.*;

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

@RestController
@RequestMapping("/accounts")

public class AccountController {

    @GetMapping("/{number}")

    public Map<String, Object> getAccountDetails(@PathVariable String number) {

        Map<String, Object> response = new HashMap<>();

        response.put("number", number);

        response.put("type", "savings");

        response.put("balance", 234343);

        return response;

    }
}
```

### LoanController.java

```
package com.cognizant.loan.controller;

import org.springframework.web.bind.annotation.*;

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

@RestController
@RequestMapping("/loans")

public class LoanController {

    @GetMapping("/{number}")
```

```

public Map<String, Object> getLoanDetails(@PathVariable String number) {

    Map<String, Object> response = new HashMap<>();

    response.put("number", number);

    response.put("type", "car");

    response.put("loan", 400000);

    response.put("emi", 3258);

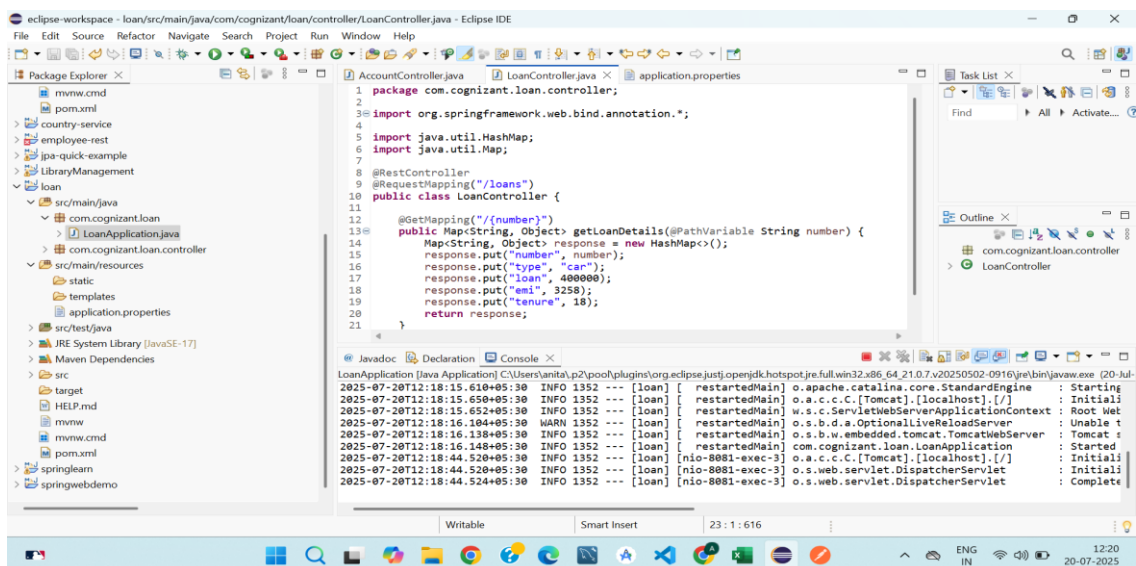
    response.put("tenure", 18);

    return response;

}

```

## OUTPUT :



The screenshot shows the Eclipse IDE with the `LoanController.java` file open. The code defines a `LoanController` class with a `getLoanDetails` method that returns a `Map<String, Object>`. The console output shows the application starting successfully, with the `LoanController` class loaded.

```

package com.cognizant.loan.controller;

import org.springframework.web.bind.annotation.*;
import java.util.HashMap;
import java.util.Map;

@RestController
@RequestMapping("/loans")
public class LoanController {

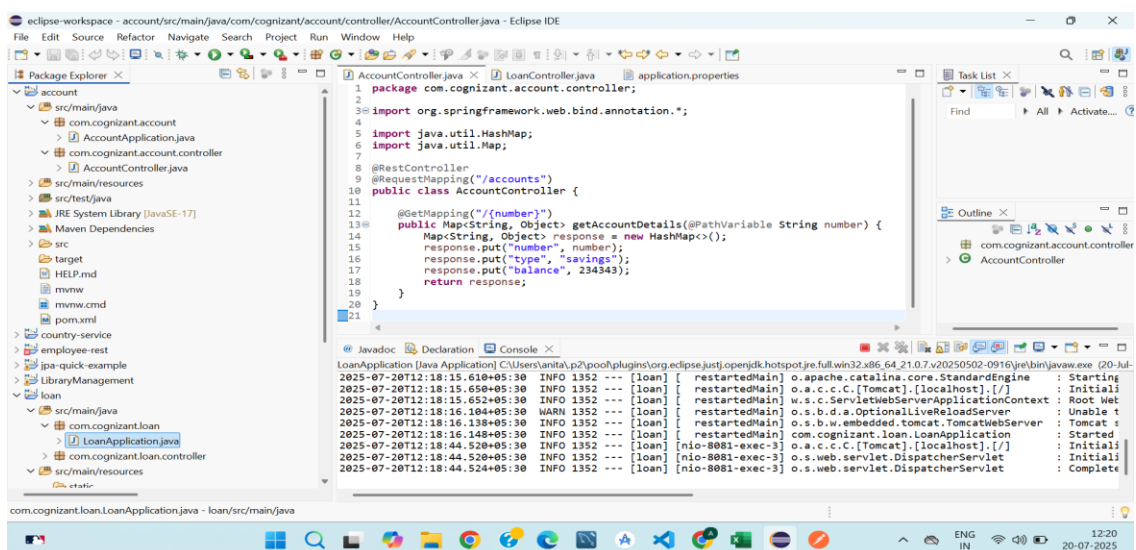
    @GetMapping("/{number}")
    public Map<String, Object> getLoanDetails(@PathVariable String number) {
        Map<String, Object> response = new HashMap<>();
        response.put("number", number);
        response.put("type", "car");
        response.put("loan", 400000);
        response.put("emi", 3258);
        response.put("tenure", 18);
        return response;
    }
}

```

```

2025-07-20T12:18:15.610+05:30 INFO 1352 --- [loan] [ restartedMain] o.apache.catalina.core.StandardEngine : Starting
2025-07-20T12:18:15.650+05:30 INFO 1352 --- [loan] [ restartedMain] o.a.c.c.c.[Tomcat].[localhost].[/] : Initiali
2025-07-20T12:18:15.652+05:30 INFO 1352 --- [loan] [ restartedMain] w.s.c.ServletWebServerApplicationContext : Root Wet
2025-07-20T12:18:16.104+05:30 WARN 1352 --- [loan] [ restartedMain] o.s.b.d.a.OptionalLiveReloadServer : Unable t
2025-07-20T12:18:16.138+05:30 INFO 1352 --- [loan] [ restartedMain] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat s
2025-07-20T12:18:16.148+05:30 INFO 1352 --- [loan] [ restartedMain] com.cognizant.loan.LoanApplication : Started
2025-07-20T12:18:44.520+05:30 INFO 1352 --- [loan] [nio-8081-exec-3] o.a.c.c.c.[Tomcat].[localhost].[/] : Initiali
2025-07-20T12:18:44.520+05:30 INFO 1352 --- [loan] [nio-8081-exec-3] o.s.web.servlet.DispatcherServlet : Initiali
2025-07-20T12:18:44.524+05:30 INFO 1352 --- [loan] [nio-8081-exec-3] o.s.web.servlet.DispatcherServlet : Complete

```



The screenshot shows the Eclipse IDE with the `AccountController.java` file open. The code defines an `AccountController` class with a `getAccountDetails` method that returns a `Map<String, Object>`. The console output shows the application starting successfully, with the `AccountController` class loaded.

```

package com.cognizant.account.controller;

import org.springframework.web.bind.annotation.*;
import java.util.HashMap;
import java.util.Map;

@RestController
@RequestMapping("/accounts")
public class AccountController {

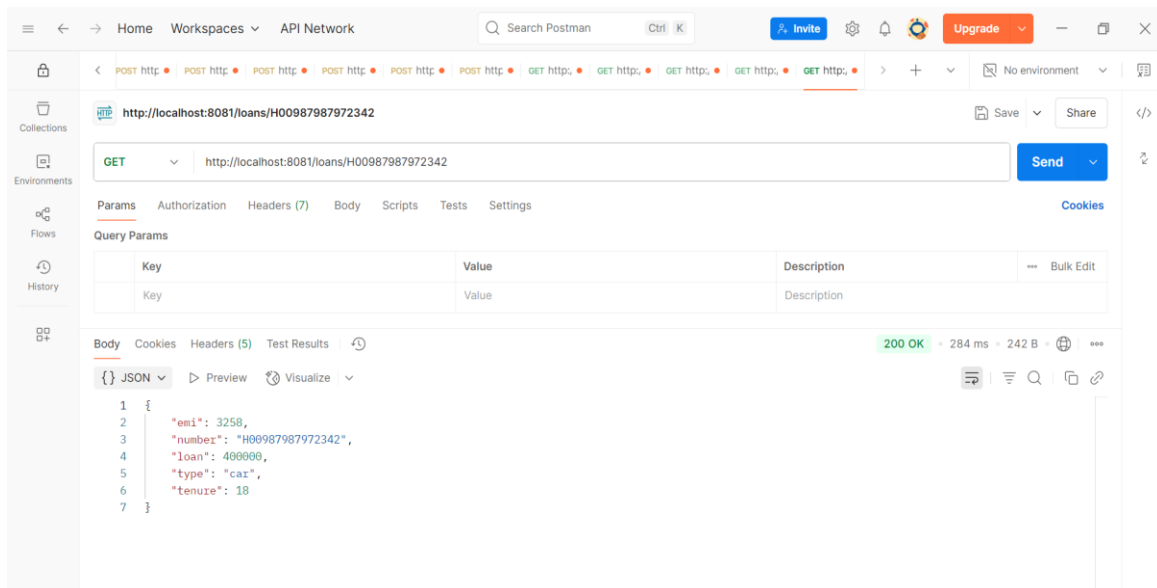
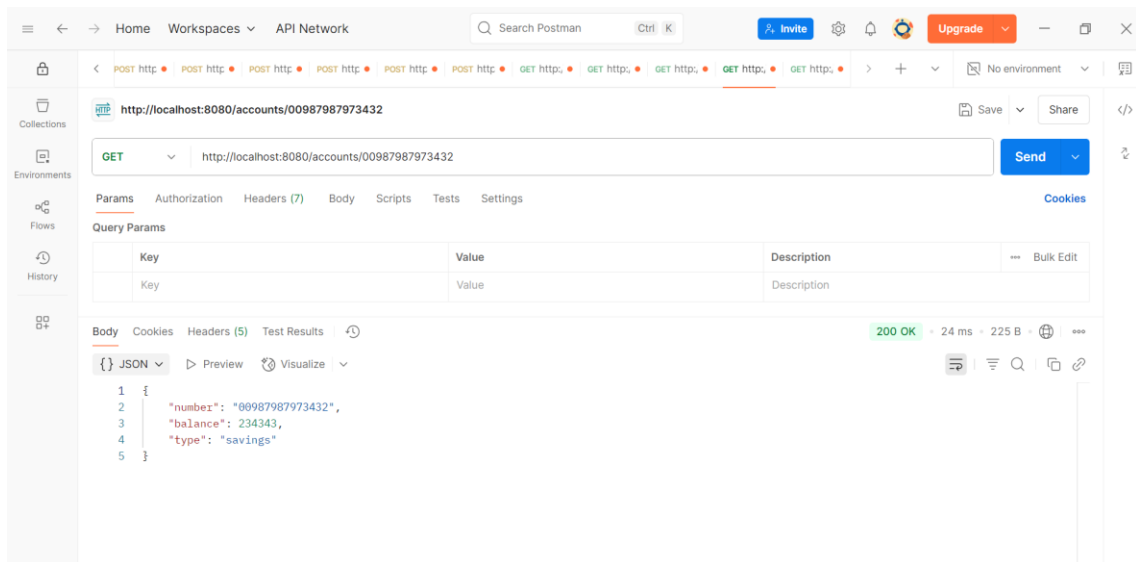
    @GetMapping("/{number}")
    public Map<String, Object> getAccountDetails(@PathVariable String number) {
        Map<String, Object> response = new HashMap<>();
        response.put("number", number);
        response.put("type", "savings");
        response.put("balance", 234343);
        return response;
    }
}

```

```

2025-07-20T12:18:15.610+05:30 INFO 1352 --- [loan] [ restartedMain] o.apache.catalina.core.StandardEngine : Starting
2025-07-20T12:18:15.650+05:30 INFO 1352 --- [loan] [ restartedMain] o.a.c.c.c.[Tomcat].[localhost].[/] : Initiali
2025-07-20T12:18:15.652+05:30 INFO 1352 --- [loan] [ restartedMain] w.s.c.ServletWebServerApplicationContext : Root Wet
2025-07-20T12:18:16.104+05:30 WARN 1352 --- [loan] [ restartedMain] o.s.b.d.a.OptionalLiveReloadServer : Unable t
2025-07-20T12:18:16.138+05:30 INFO 1352 --- [loan] [ restartedMain] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat s
2025-07-20T12:18:16.148+05:30 INFO 1352 --- [loan] [ restartedMain] com.cognizant.loan.LoanApplication : Started
2025-07-20T12:18:44.520+05:30 INFO 1352 --- [loan] [nio-8081-exec-3] o.a.c.c.c.[Tomcat].[localhost].[/] : Initiali
2025-07-20T12:18:44.520+05:30 INFO 1352 --- [loan] [nio-8081-exec-3] o.s.web.servlet.DispatcherServlet : Initiali
2025-07-20T12:18:44.524+05:30 INFO 1352 --- [loan] [nio-8081-exec-3] o.s.web.servlet.DispatcherServlet : Complete

```



## HANDS-ON EXERCISES

- 1) Create Eureka Discovery Server and register Microservices

### EurekaDiscoveryServerApplication.java

```
package com.cognizant.eurekadiscoveryserver;
```

```
import org.springframework.boot.SpringApplication;
```

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

```
import org.springframework.cloud.netflix.eureka.server.EnableEurekaServer;
```

@SpringBootApplication

@EnableEurekaServer

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

### **AccountApplication.java**

```
package com.cognizant.account;  
  
import org.springframework.boot.SpringApplication;  
import org.springframework.boot.autoconfigure.SpringBootApplication;  
import org.springframework.cloud.client.discovery.EnableDiscoveryClient;  
  
@SpringBootApplication  
@EnableDiscoveryClient  
  
public class AccountApplication {  
    public static void main(String[] args) {  
        SpringApplication.run(AccountApplication.class, args);  
    }  
}
```

### **LoanApplication.java**

```
package com.cognizant.loan;  
  
import org.springframework.boot.SpringApplication;  
import org.springframework.boot.autoconfigure.SpringBootApplication;  
import org.springframework.cloud.client.discovery.EnableDiscoveryClient;  
  
@SpringBootApplication  
@EnableDiscoveryClient  
  
public class LoanApplication {  
    public static void main(String[] args) {  
        SpringApplication.run(LoanApplication.class, args);  
    }  
}
```

OUTPUT :

