

Digital Nurture 4.0 Deep Skilling

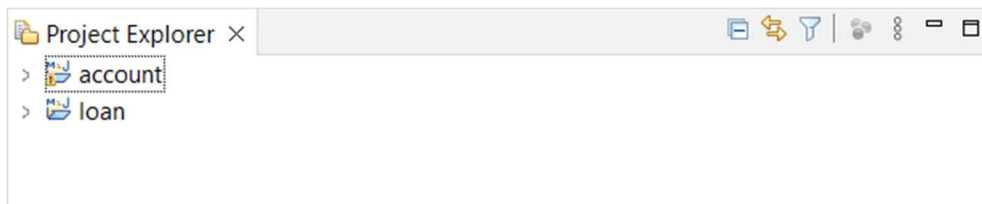
Week 5

Creating Microservices for account and loan In this hands on exercises, we will create two microservices for a bank. One microservice for handing accounts and one for handling loans.

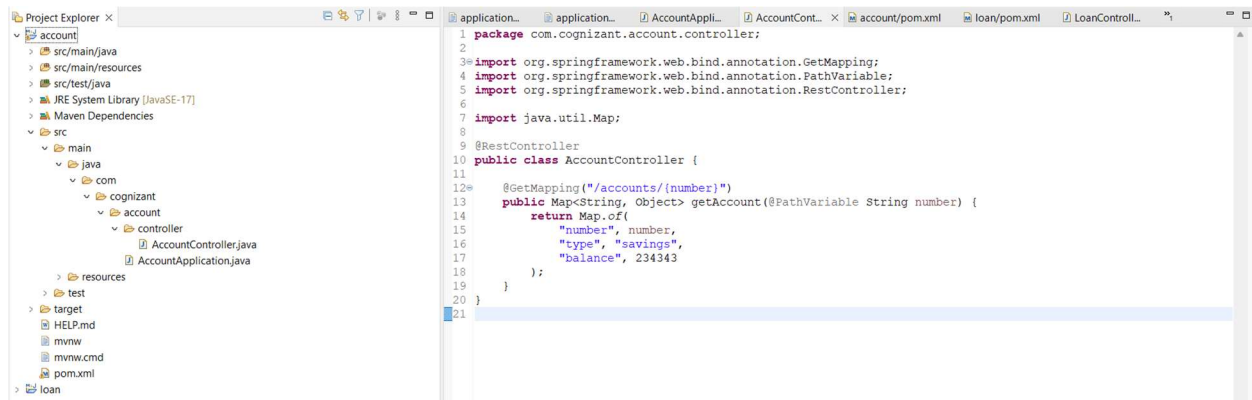
Each microservice will be a specific independent Spring RESTful Webservice maven project having it's own pom.xml. The only difference is that, instead of having both account and loan as a single application, it is split into two different applications. These webservices will be a simple service without any backend connectivity.

OUTCOMES

- Created **two separate Spring Boot microservices** — account and loan
- Assigned them **different ports** (8300 and 8087)
- Implemented simple **GET APIs** with **dummy responses**
- Confirmed that **both services are running independently**

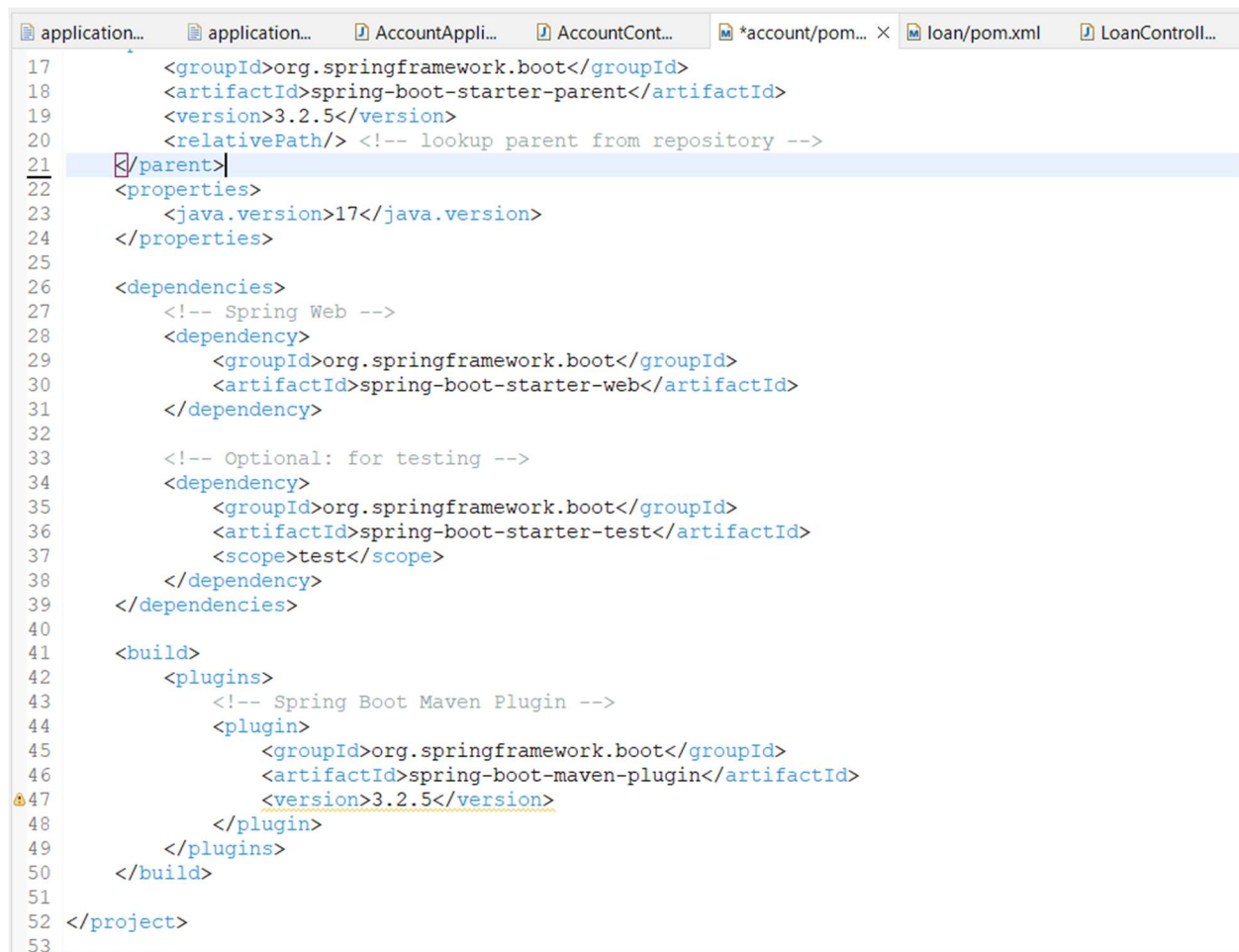


Account



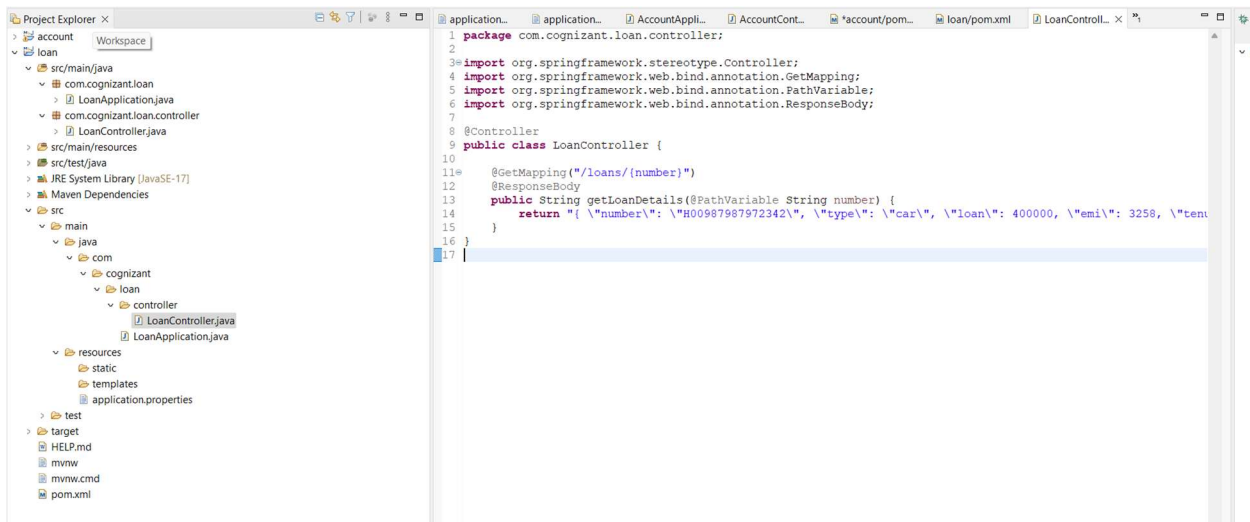
POM.XML

Dependencies



```
1 package com.cognizant.account;
2
3 import org.springframework.boot.SpringApplication;
4 import org.springframework.boot.autoconfigure.SpringBootApplication;
5
6 @SpringBootApplication
7 public class AccountApplication {
8     public static void main(String[] args) {
9         SpringApplication.run(AccountApplication.class, args);
10    }
11 }
12
```

Loan



```
1 package com.cognizant.loan.controller;
2
3 import org.springframework.stereotype.Controller;
4 import org.springframework.web.bind.annotation.GetMapping;
5 import org.springframework.web.bind.annotation.PathVariable;
6 import org.springframework.web.bind.annotation.ResponseBody;
7
8 @Controller
9 public class LoanController {
10
11     @GetMapping("/loans/{number}")
12     @ResponseBody
13     public String getLoanDetails(@PathVariable String number) {
14         return "{ \"number\": \"H00987987972342\", \"type\": \"car\", \"loan\": 400000, \"emi\": 3258, \"ten\"";
15     }
16 }
17
```

```
1 package com.cognizant.loan;
2
3 import org.springframework.boot.SpringApplication;
4
5
6 @SpringBootApplication
7 public class LoanApplication {
8
9     public static void main(String[] args) {
10         SpringApplication.run(LoanApplication.class, args);
11     }
12 }
13 }
14
```

Pom.xml

Dependencies

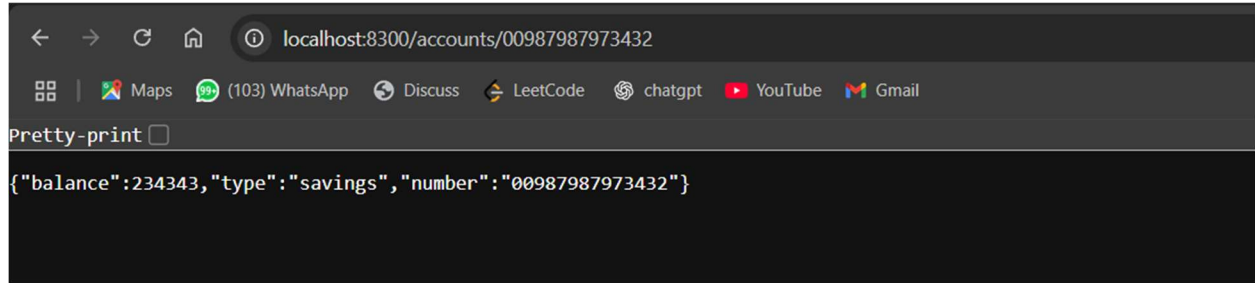
```
</properties>
<dependencies>
  <!-- Spring Boot Starter Web -->
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-web</artifactId>
  </dependency>

  <!-- Spring Boot DevTools (optional for development) -->
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-devtools</artifactId>
    <scope>runtime</scope>
    <optional>true</optional>
  </dependency>

  <!-- Spring Boot Test (for unit tests, optional) -->
  <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>
```

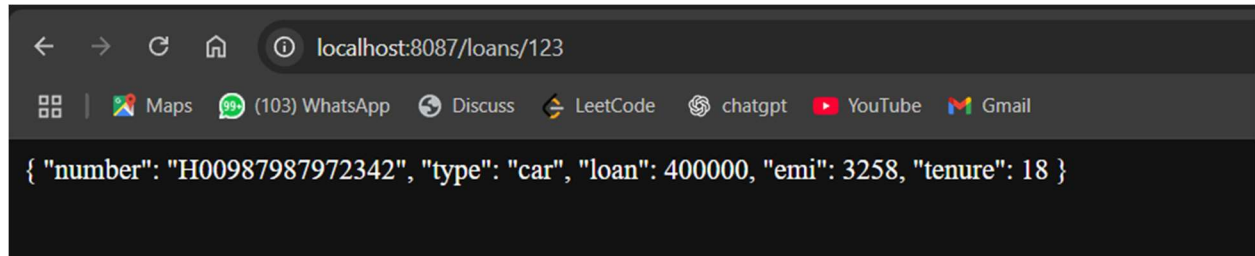
Output :



```
localhost:8300/accounts/00987987973432
```

Pretty-print ☐

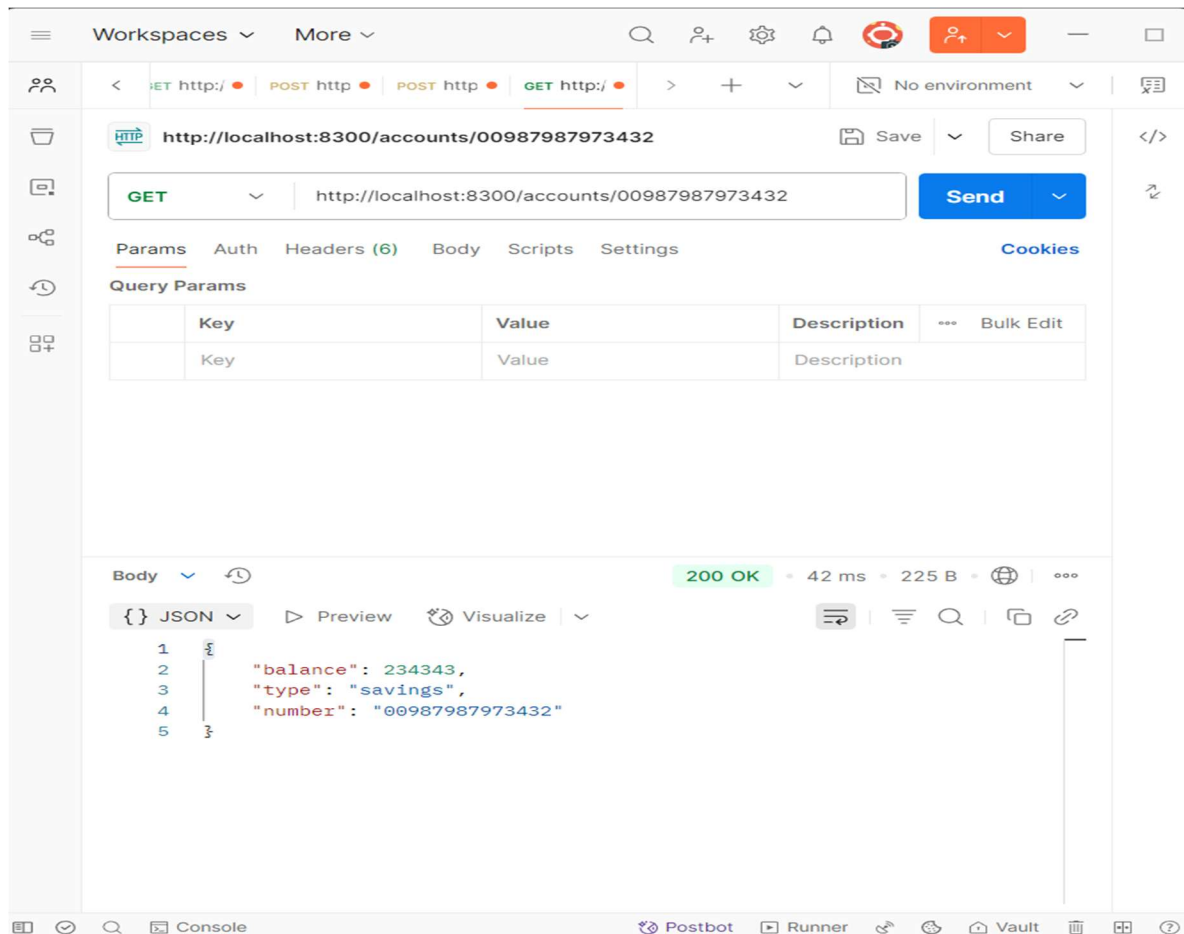
```
{\"balance\":234343,\"type\":\"savings\",\"number\":\"00987987973432\"}
```



```
localhost:8087/loans/123
```

```
{ \"number\": \"H00987987972342\", \"type\": \"car\", \"loan\": 400000, \"emi\": 3258, \"tenure\": 18 }
```

Postman :



Workspaces ▾ More ▾

GET http://

POST http

POST http

GET http://

>

+

▾

No environment ▾

HTTP

http://localhost:8087/loans/123

Save ▾

Share

</>

GET ▾

http://localhost:8087/loans/123

Send ▾

↕

Params

Auth

Headers (6)

Body

Scripts

Settings

Cookies

Query Params

	Key	Value	Description	...	Bulk Edit
	Key	Value	Description		

Body ▾

🕒

200 OK

• 106 ms • 253 B • 🌐

...

Raw ▾

▶ Preview

🔗 Visualize ▾

🔍

🔍

🔗

1

```
{ "number": "H00987987972342", "type": "car", "loan": 400000, "emi": 3258, "tenure": 18 }
```

🔍

🕒

🔍

🔍

Console

Postbot

Runner

🔗

🔗

🔗

Vault

🔍

🔍

🔍

?