

Creating Microservices for account and loan

In this hands on exercises, we will create two microservices for a bank. One microservice for handling 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.

Follow steps below to implement the two microservices:

Account Microservice

- Create folder with employee id in D: drive
 - Create folder named 'microservices' in the new folder created in previous step. This folder will contain all the sample projects that we will create for learning microservices.
 - Open <https://start.spring.io/> in browser
 - Enter form field values as specified below:
 - o Group: com.cognizant
 - o Artifact: account
 - Select the following modules
 - o Developer Tools > Spring Boot DevTools
 - o Web > Spring Web
 - Click generate and download the zip file
 - Extract 'account' folder from the zip and place this folder in the 'microservices' folder created earlier
 - Open command prompt in account folder and build using mvn clean package command
 - Import this project in Eclipse and implement a controller method for getting account details based on account number. Refer specification below:
 - o Method: GET
 - o Endpoint: /accounts/{number}
 - o Sample Response. Just a dummy response without any backend connectivity.
- ```
{ number: "00987987973432", type: "savings", balance: 234343 }
```
- Launch by running the application class and test the service in browser

## Loan Microservice

- Follow similar steps specified for Account Microservice and implement a service API to get loan account details
  - o Method: GET
  - o Endpoint: /loans/{number}

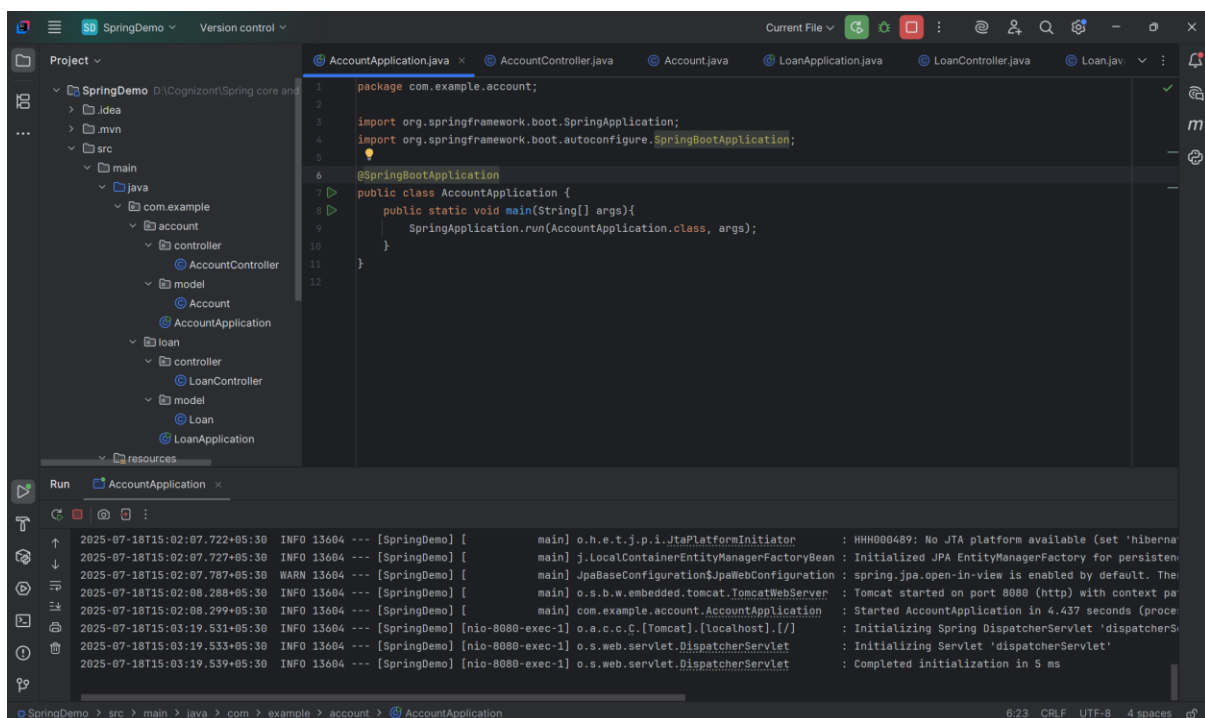
o Sample Response. Just a dummy response without any backend connectivity.

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

- Launching this application by having account service already running
- This launch will fail with error that the bind address is already in use
- The reason is that each one of the service is launched with default port number as 8080. Account service is already using this port and it is not available for loan service.
- Include "server.port" property with value 8081 and try launching the application
- Test the service with 8081 port

Now we have two microservices running on different ports.

NOTE: The console window of Eclipse will have both the service console running. To switch between different consoles use the monitor icon within the console view.



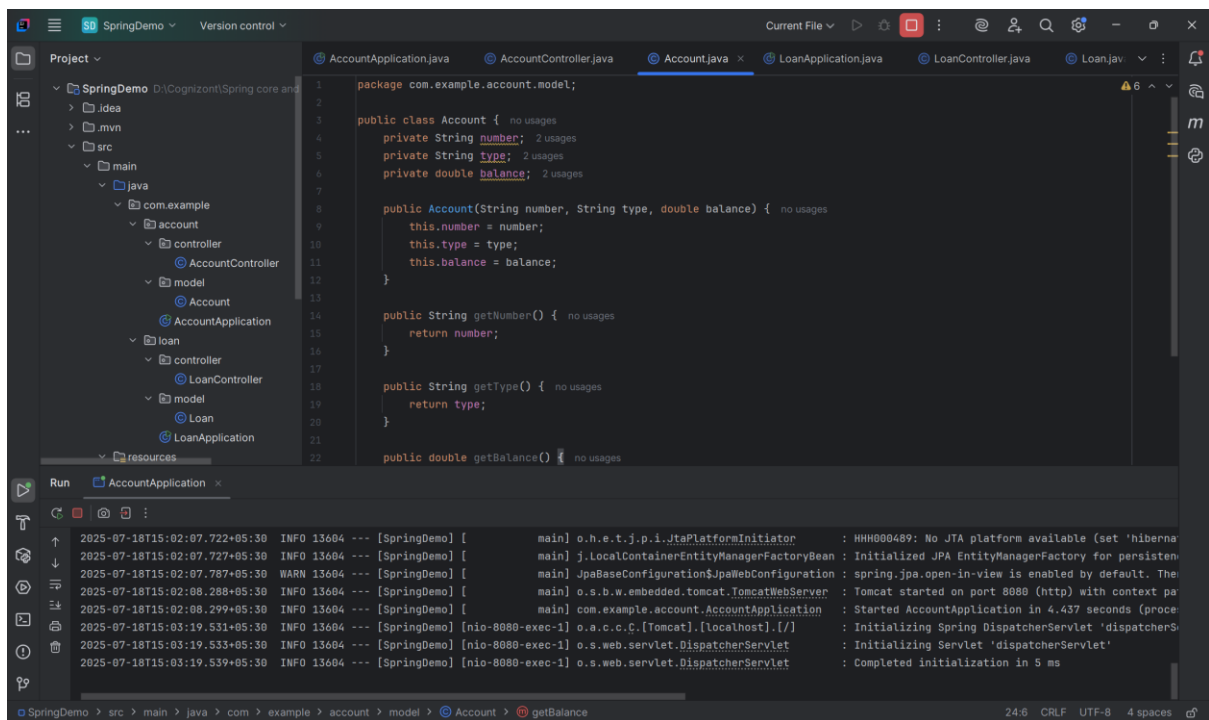
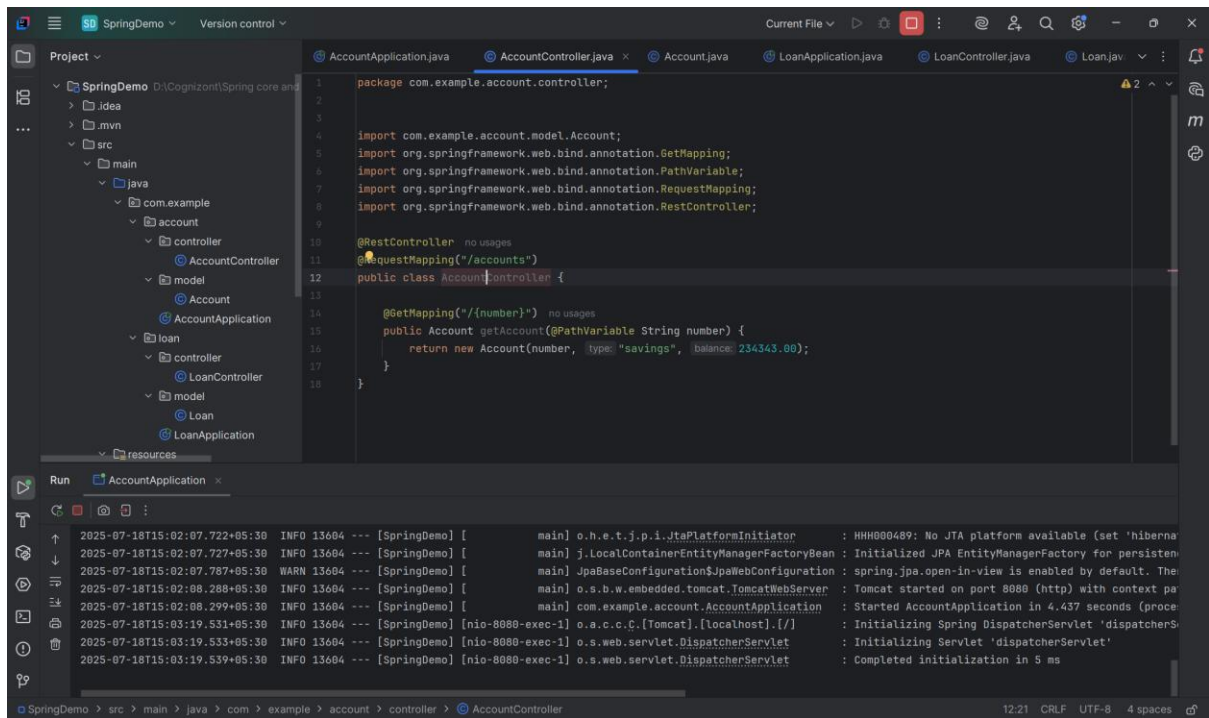
The screenshot shows the Eclipse IDE interface. The top part displays the 'Project' view on the left, showing the project structure for 'SpringDemo'. The main editor area shows the 'AccountApplication.java' file, which is a Spring Boot application. The bottom part shows the 'Run' console, which displays the output of the application. The console output indicates that the application is running successfully on port 8080.

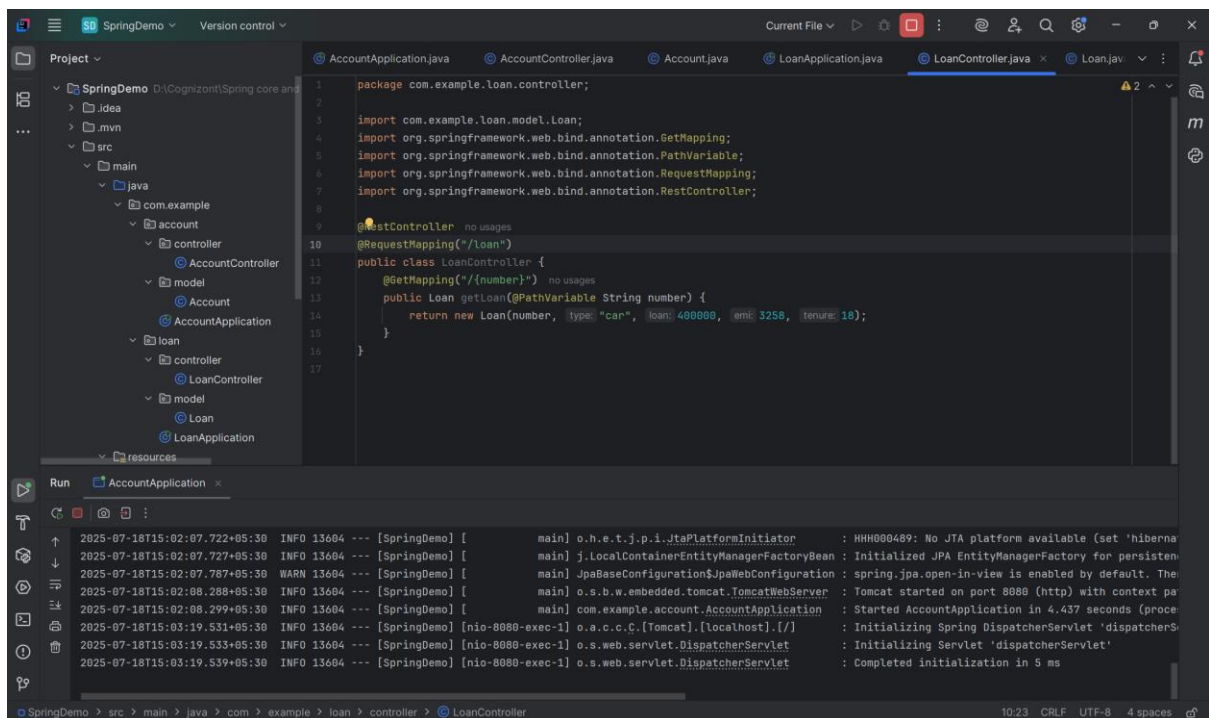
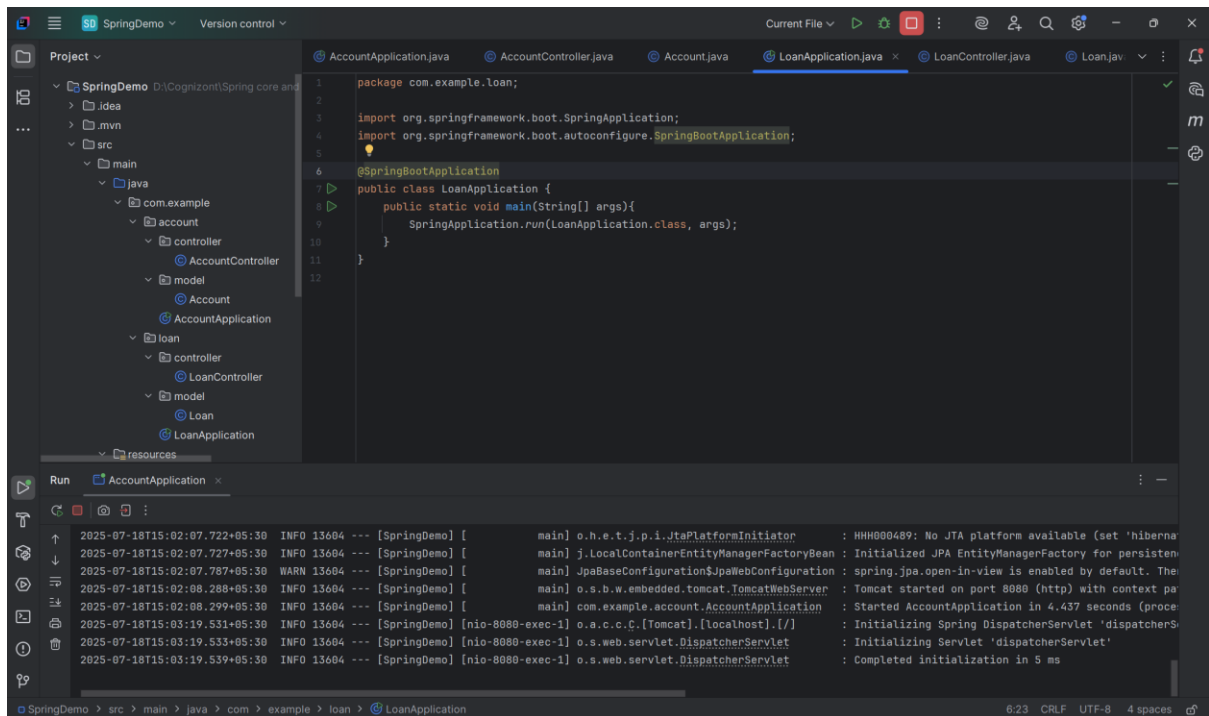
```
package com.example.account;

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

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

```
2025-07-18T15:02:07.722+05:30 INFO 13604 --- [SpringDemo] [main] o.h.e.t.j.p.i.JtaPlatformInitiator : HHH000489: No JTA platform available (set 'hibern
2025-07-18T15:02:07.727+05:30 INFO 13604 --- [SpringDemo] [main] j.LocalContainerEntityManagerFactoryBean : Initialized JPA EntityManagerFactory for persisten
2025-07-18T15:02:07.787+05:30 WARN 13604 --- [SpringDemo] [main] JpaBaseConfiguration$JpaWebConfiguration : spring.jpa.open-in-view is enabled by default. The
2025-07-18T15:02:08.288+05:30 INFO 13604 --- [SpringDemo] [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port 8080 (http) with context pa
2025-07-18T15:02:08.299+05:30 INFO 13604 --- [SpringDemo] [main] com.example.account.AccountApplication : Started AccountApplication in 4.437 seconds (proce
2025-07-18T15:03:19.531+05:30 INFO 13604 --- [SpringDemo] [nio-8080-exec-1] o.a.c.c.c.[Tomcat].[localhost].[/] : Initializing Spring DispatcherServlet 'dispatcherS
2025-07-18T15:03:19.533+05:30 INFO 13604 --- [SpringDemo] [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet : Initializing Servlet 'dispatcherServlet'
2025-07-18T15:03:19.539+05:30 INFO 13604 --- [SpringDemo] [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet : Completed initialization in 5 ms
```





The screenshot shows an IDE with the following components:

- Project Explorer:** Shows the project structure for 'SpringDemo'. The 'loan' package is expanded, showing 'LoanController' and 'Loan'.
- Code Editor:** Displays the 'Loan.java' file with the following code:

```
1 package com.example.loan.model;
2
3 public class Loan {
4 private String number;
5 private String type;
6 private int loan;
7 private int emi;
8 private int tenure;
9
10 public Loan(String number, String type, int loan, int emi, int tenure) {
11 this.number = number;
12 this.type = type;
13 this.loan = loan;
14 this.emi = emi;
15 this.tenure = tenure;
16 }
17
18 public String getNumber() {
19 return number;
20 }
21
22 public String getType() {
```

- Run Console:** Shows the output of the application. The logs indicate that the application started successfully on port 8080. The logs are as follows:

```
2025-07-18T15:02:07.722+05:30 INFO 13604 --- [SpringDemo] [main] o.h.e.t.j.p.i.JtaPlatformInitiator : HHH000489: No JTA platform available (set 'hiberna
2025-07-18T15:02:07.727+05:30 INFO 13604 --- [SpringDemo] [main] j.LocalContainerEntityManagerFactoryBean : Initialized JPA EntityManagerFactory for persisten
2025-07-18T15:02:07.787+05:30 WARN 13604 --- [SpringDemo] [main] JpaBaseConfiguration$JpaWebConfiguration : spring.jpa.open-in-view is enabled by default. The
2025-07-18T15:02:08.288+05:30 INFO 13604 --- [SpringDemo] [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port 8080 (http) with context pa
2025-07-18T15:02:08.299+05:30 INFO 13604 --- [SpringDemo] [main] com.example.account.AccountApplication : Started AccountApplication in 4.437 seconds (proce
2025-07-18T15:03:19.531+05:30 INFO 13604 --- [SpringDemo] [nio-8080-exec-1] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring DispatcherServlet 'dispatcherS
2025-07-18T15:03:19.533+05:30 INFO 13604 --- [SpringDemo] [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet : Initializing Servlet 'dispatcherServlet'
2025-07-18T15:03:19.539+05:30 INFO 13604 --- [SpringDemo] [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet : Completed initialization in 5 ms
```

