# Creating Microservices for Account and Loan

In this hands-on exercise, we will build two independent Spring Boot microservices for a banking application: one to manage account details and another to manage loan details. These services will be simple REST APIs with hardcoded responses and no backend/database connectivity.

## Account Microservice

* Steps to Create:

1. 1. Create a folder with your employee ID in the D: drive.
2. 2. Inside that, create a folder named 'microservices'.
3. 3. Open https://start.spring.io in your browser.
4. 4. Fill in the form as below:

* - Group: com.cognizant  
   - Artifact: account

1. 5. Select the following dependencies:

* - Developer Tools > Spring Boot DevTools  
   - Web > Spring Web

1. 6. Click Generate, download the ZIP file, and extract the 'account' folder into your microservices folder.
2. 7. Open Command Prompt in the 'account' folder and run:

mvn clean package

1. 8. Import the project into Eclipse.
2. 9. Implement a controller as below:

AccountController.java

package com.cognizant.account.controller;  
  
import org.springframework.web.bind.annotation.\*;  
  
@RestController  
@RequestMapping("/accounts")  
public class AccountController {  
  
 @GetMapping("/{number}")  
 public Account getAccount(@PathVariable String number) {  
 return new Account(number, "savings", 234343);  
 }  
  
 record Account(String number, String type, double balance) {}  
}

application.properties

server.port=8080  
spring.application.name=account

Test URL: http://localhost:8080/accounts/00987987973432

* Sample Output:

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

## Loan Microservice

Repeat the same steps as Account Microservice, changing the artifact to 'loan'.

1. Implement the following controller:

LoanController.java

package com.cognizant.loan.controller;  
  
import org.springframework.web.bind.annotation.\*;  
  
@RestController  
@RequestMapping("/loans")  
public class LoanController {  
  
 @GetMapping("/{number}")  
 public Loan getLoan(@PathVariable String number) {  
 return new Loan(number, "car", 400000, 3258, 18);  
 }  
  
 record Loan(String number, String type, double loan, double emi, int tenure) {}  
}

application.properties

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

Test URL: http://localhost:8081/loans/H00987987972342

* Sample Output:

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

## Final Result

• Account Service runs on port 8080

• Loan Service runs on port 8081

• Both microservices can be tested independently via browser or Postman

To switch between running consoles in Eclipse, use the monitor icon in the console view.