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.

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.

Solution:

Microservices Hands-On Exercise

===============================

Objective:

----------

Create two independent Spring Boot REST microservices for a bank:

- One for handling account details

- One for handling loan details

Both services will be simple REST APIs without any backend database.

------------------------------------------------------------

Step-by-Step Instructions:

------------------------------------------------------------

1. Create Folder Structure

--------------------------

- Go to D: drive

- Create a folder with your Employee ID

Example: D:\123456

- Inside that, create another folder:

D:\123456\microservices

------------------------------------------------------------

2. Account Microservice

-----------------------

A. Generate Spring Boot Project

-------------------------------

- Open https://start.spring.io

- Fill in:

Group: com.cognizant

Artifact: account

- Add Dependencies:

- Spring Boot DevTools

- Spring Web

- Click "Generate", download ZIP

B. Setup Project

----------------

- Extract the ZIP

- Move the 'account' folder into:

D:\123456\microservices

- Open Command Prompt in 'account' folder:

Run:

mvn clean package

C. Import in Eclipse and Add Code

---------------------------------

- Import as a Maven Project

- Create file: src/main/java/com/cognizant/account/controller/AccountController.java

Content:

--------

package com.cognizant.account.controller;

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

@RestController

public class AccountController {

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

public Account getAccount(@PathVariable String number) {

return new Account("00987987973432", "savings", 234343);

}

static class Account {

public String number;

public String type;

public double balance;

public Account(String number, String type, double balance) {

this.number = number;

this.type = type;

this.balance = balance;

}

}

}

D. Run and Test

---------------

- Run the application

- Open browser:

http://localhost:8080/accounts/00987987973432

Expected Output:

----------------

{

"number": "00987987973432",

"type": "savings",

"balance": 234343

}

------------------------------------------------------------

3. Loan Microservice

--------------------

A. Generate Spring Boot Project

-------------------------------

- Open https://start.spring.io

- Fill in:

Group: com.cognizant

Artifact: loan

- Add Dependencies:

- Spring Boot DevTools

- Spring Web

- Click "Generate", download ZIP

B. Setup Project

----------------

- Extract the ZIP

- Move 'loan' folder into:

D:\123456\microservices

- Open Command Prompt in 'loan' folder:

Run:

mvn clean package

C. Set Port in application.properties

-------------------------------------

- Open src/main/resources/application.properties

- Add:

server.port=8081

D. Import in Eclipse and Add Code

---------------------------------

- Import as Maven Project

- Create file: src/main/java/com/cognizant/loan/controller/LoanController.java

Content:

--------

package com.cognizant.loan.controller;

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

@RestController

public class LoanController {

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

public Loan getLoan(@PathVariable String number) {

return new Loan("H00987987972342", "car", 400000, 3258, 18);

}

static class Loan {

public String number;

public String type;

public double loan;

public double emi;

public int tenure;

public Loan(String number, String type, double loan, double emi, int tenure) {

this.number = number;

this.type = type;

this.loan = loan;

this.emi = emi;

this.tenure = tenure;

}

}

}

E. Run and Test

---------------

- Run the loan microservice (after ensuring account service is running)

- Open browser:

http://localhost:8081/loans/H00987987972342

Expected Output:

----------------

{

"number": "H00987987972342",

"type": "car",

"loan": 400000,

"emi": 3258,

"tenure": 18

}

------------------------------------------------------------

4. Final Notes

--------------

- Now both microservices are running independently:

- Account service on port 8080

- Loan service on port 8081

- Use the monitor icon in Eclipse console to switch between logs of the two services