**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.

**Program;**

**LoanController.java:**

package com.cognizant.loan;

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

@RestController

public class LoanController {

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

public Loan getLoan(@PathVariable String number) {

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

}

}

class Loan {

private String number;

private String type;

private double loan;

private int emi;

private int tenure;

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

this.number = number;

this.type = type;

this.loan = loan;

this.emi = emi;

this.tenure = tenure;

}

// Getters and setters

public String getNumber() { return number; }

public void setNumber(String number) { this.number = number; }

public String getType() { return type; }

public void setType(String type) { this.type = type; }

public double getLoan() { return loan; }

public void setLoan(double loan) { this.loan = loan; }

public int getEmi() { return emi; }

public void setEmi(int emi) { this.emi = emi; }

public int getTenure() { return tenure; }

public void setTenure(int tenure) { this.tenure = tenure; }

}

**Application.properties:**

server.port=8081

**AccountController.java:**

package com.cognizant.account;

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

@RestController

public class AccountController {

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

public Account getAccount(@PathVariable String number) {

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

}

}

class Account {

private String number;

private String type;

private double balance;

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

this.number = number;

this.type = type;

this.balance = balance;

}

// Getters and setters (required for JSON conversion)

public String getNumber() { return number; }

public void setNumber(String number) { this.number = number; }

public String getType() { return type; }

public void setType(String type) { this.type = type; }

public double getBalance() { return balance; }

public void setBalance(double balance) { this.balance = balance; }

}

**Output:**



