WEEK5

# Question

Creating Microservices for account and loan  
  
In this hands on exercise, 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 its 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 simple services 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:  
 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  
• 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:  
 { number: "00987987973432", type: "savings", balance: 234343 }  
• Launch by running the application class and test the service in browser  
  
Loan Microservice:  
• Follow similar steps as for Account Microservice and implement a service API to get loan account details:  
 o Method: GET  
 o Endpoint: /loans/{number}  
 o Sample Response:  
 { number: "H00987987972342", type: "car", loan: 400000, emi: 3258, tenure: 18 }  
• Launching this application while account service is already running will fail due to port conflict (both use port 8080 by default).  
• Include "server.port=8081" in application.properties and relaunch.  
• Test the service with 8081 port.  
  
Now we have two microservices running on different ports.

# Answer

## 1. Account Microservice

Controller: AccountController.java

package com.cognizant.account.controller;  
  
import org.springframework.web.bind.annotation.GetMapping;  
import org.springframework.web.bind.annotation.PathVariable;  
import org.springframework.web.bind.annotation.RestController;  
import java.util.HashMap;  
import java.util.Map;  
  
@RestController  
public class AccountController {  
  
 @GetMapping("/accounts/{number}")  
 public Map<String, Object> getAccountDetails(@PathVariable String number) {  
 Map<String, Object> response = new HashMap<>();  
 response.put("number", number);  
 response.put("type", "savings");  
 response.put("balance", 234343);  
 return response;  
 }  
}

## 2. Loan Microservice

Controller: LoanController.java

package com.cognizant.loan.controller;  
  
import org.springframework.web.bind.annotation.GetMapping;  
import org.springframework.web.bind.annotation.PathVariable;  
import org.springframework.web.bind.annotation.RestController;  
import java.util.HashMap;  
import java.util.Map;  
  
@RestController  
public class LoanController {  
  
 @GetMapping("/loans/{number}")  
 public Map<String, Object> getLoanDetails(@PathVariable String number) {  
 Map<String, Object> response = new HashMap<>();  
 response.put("number", number);  
 response.put("type", "car");  
 response.put("loan", 400000);  
 response.put("emi", 3258);  
 response.put("tenure", 18);  
 return response;  
 }  
}

## 3. Loan Microservice application.properties

server.port=8081