**MICROSERVICES WITH SPRING BOOT 3 AND SPRING CLOUD**

**2. Microservices with API gateway**

**Creating Microservices for account and loan**

In this handson 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.

**Account Microservice**

1. Create folder with employee id in D: drive
2. 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.
3. Open https://start.spring.io/ in browser
4. Enter form field values as specified below:

* Group: com.cognizant
* Artifact: account

1. Select the following modules

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

1. Click generate and download the zip file
2. Extract 'account' folder from the zip and place this folder in the 'microservices' folder created earlier
3. Open command prompt in account folder and build using:

* mvn clean package

1. Import this project in Eclipse

* New -> import -> existing maven projects -> paste path -> browse -> Finish

1. Account data model

* Src/main/java -> -> com.cognizant.account.model (package) -> Account (class)
* Account.java

package com.cognizant.account.model;

public class Account {

private String number;

private String type;

private int balance;

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

this.number = number;

this.type = type;

this.balance = balance;

}

public String getNumber() {

return number;

}

public String getType() {

return type;

}

public int getBalance() {

return balance;

}

}

1. Controller method

* src/main/java -> com.cognizant.account.controller (package) ->

AccountController (class)

* AccountController.java

package com.cognizant.account.controller;

import com.cognizant.account.model.Account;

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

@RestController

@RequestMapping("/accounts")

public class AccountController {

@GetMapping("/{number}")

public Account getAccountDetails(@PathVariable String number) {

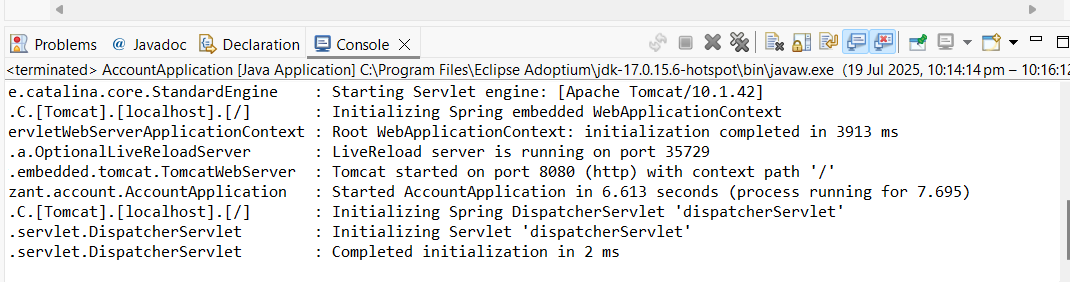
return new Account(number, "savings", 234343);

}

}

1. Run application

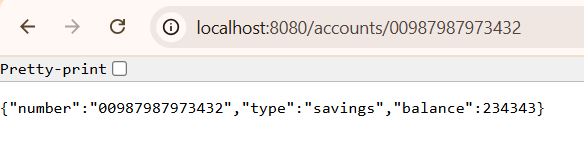
* AccountApplication.java -> Run As -> Java Application



* Running in default port no: 8080.

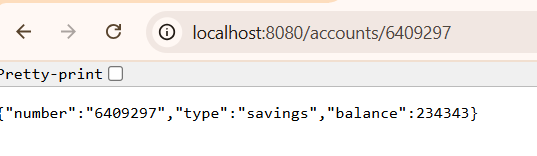
1. Open chrome/browser for testing
   * <http://localhost:8080/accounts/00987987973432>

OUTPUT:



* We can give our own “number”, for example:
* <http://localhost:8080/accounts/6409297>

OUTPUT:



**Loan Microservice**

1. Create folder with employee id in D: drive
2. 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.
3. Open https://start.spring.io/ in browser
4. Enter form field values as specified below:

* Group: com.cognizant
* Artifact: loan

1. Select the following modules

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

1. Click generate and download the zip file
2. Extract 'loan' folder from the zip and place this folder in the 'microservices' folder created earlier
3. Open command prompt in account folder and build using:

* mvn clean package

1. Import this project in Eclipse

* New -> import -> existing maven projects -> paste path -> browse -> Finish

1. Loan data model

* Src/main/java -> com.cognizant.loan.model (package) -> Loan (class)
* Loan.java

package com.cognizant.loan.model;

public class Loan {

private String number;

private String type;

private int loan;

private int emi;

private int tenure;

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

this.number = number;

this.type = type;

this.loan = loan;

this.emi = emi;

this.tenure = tenure;

}

public String getNumber() {

return number;

}

public String getType() {

return type;

}

public int getLoan() {

return loan;

}

public int getEmi() {

return emi;

}

public int getTenure() {

return tenure;

}

}

1. Loan account details

* Src/main/java -> com.cognizant.loan.controller (package) -> LoanController
* LoanController.java

package com.cognizant.loan.controller;

import com.cognizant.loan.model.Loan;

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

@RestController

@RequestMapping("/loans")

public class LoanController {

@GetMapping("/{number}")

public Loan getLoanDetails(@PathVariable String number) {

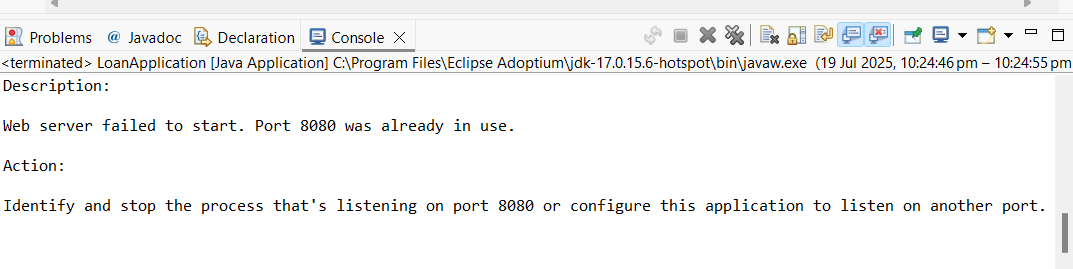
return new Loan(number, "car", 400000, 3258, 18);

}

}

1. Run Application

* LoanApplication.java -> Run As -> Java Application.



This launch will fail with error that the bind address is already in use.

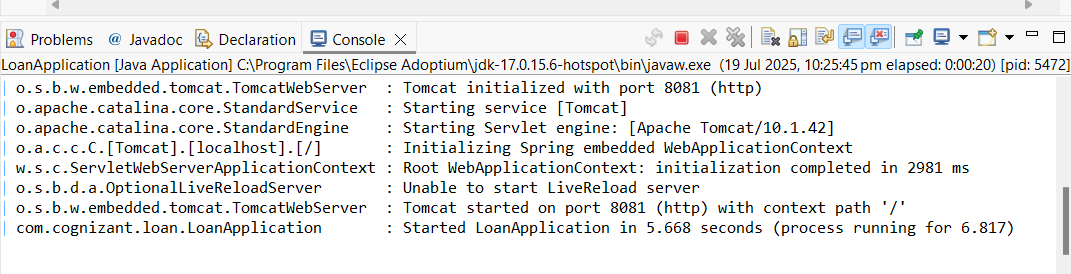
The reason is that each one of the service is launched with default port number as 8080. Account service is already using this port and it is not available for loan service.

1. application.properties

spring.application.name=loan

server.port=8081

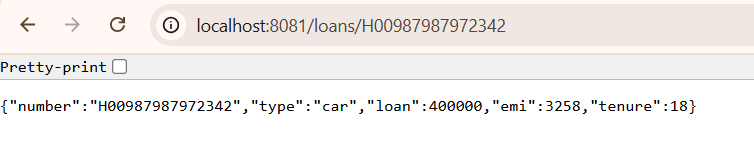
* again launch the application (it will run with 8081 port)



1. Open chrome/browser for testing

* <http://localhost:8081/loans/H00987987972342>

OUTPUT:



* We can give our own “number”, for example:
* <http://localhost:8081/loans/JAVA6409297>

OUTPUT:

