Creating Microservices for Account and Loan

# Prerequisites

- Eclipse IDE with Spring Tools plugin installed (Install via: Help > Eclipse Marketplace > Search for “Spring Tools” or "Spring Boot")  
- Maven and Java properly set up  
- Working Internet connection to download dependencies

# How to Install Spring Tools Plugin in Eclipse

## Step 1: Open Eclipse Marketplace

Open Eclipse. From the top menu, go to: Help > Eclipse Marketplace...

## Step 2: Search for Spring Plugin

In the Eclipse Marketplace window, use the search bar.  
Type: Spring Tools or Spring Boot.  
Look for a plugin named something like:  
- Spring Tools (aka Spring IDE and Spring Tool Suite)  
- Or Spring Tools 4 for Eclipse

## Step 3: Install Plugin

Click Install next to the plugin. Proceed with all defaults.  
Accept licenses and wait for it to download. After installation, Eclipse will ask to restart — click Restart Now.

## Step 4: Verify Installation

Once Eclipse restarts:  
- Go to File > New > Other…  
- In the dialog that opens, search for: Spring Starter Project  
- If you see Spring Starter Project, you're ready to go!

If You Don't See 'Spring Starter Project' After Installation:  
- Try changing the perspective: Window > Perspective > Open Perspective > Others…  
- Choose Spring or Spring Boot  
- OR restart Eclipse again

# Create Account Microservice

## Step 1: Create the Project

- Open Eclipse  
- Go to File > New > Spring Starter Project  
- In the popup:  
 Name: account  
 Type: Maven  
 Group: com.cognizant  
 Artifact: account  
 Package: com.cognizant.account  
- Click Next

## Step 2: Select Dependencies

In the dependency selection screen, add:  
- Spring Boot DevTools (under Developer Tools)  
- Spring Web (under Web)  
- Click Finish

## Step 3: Add Controller

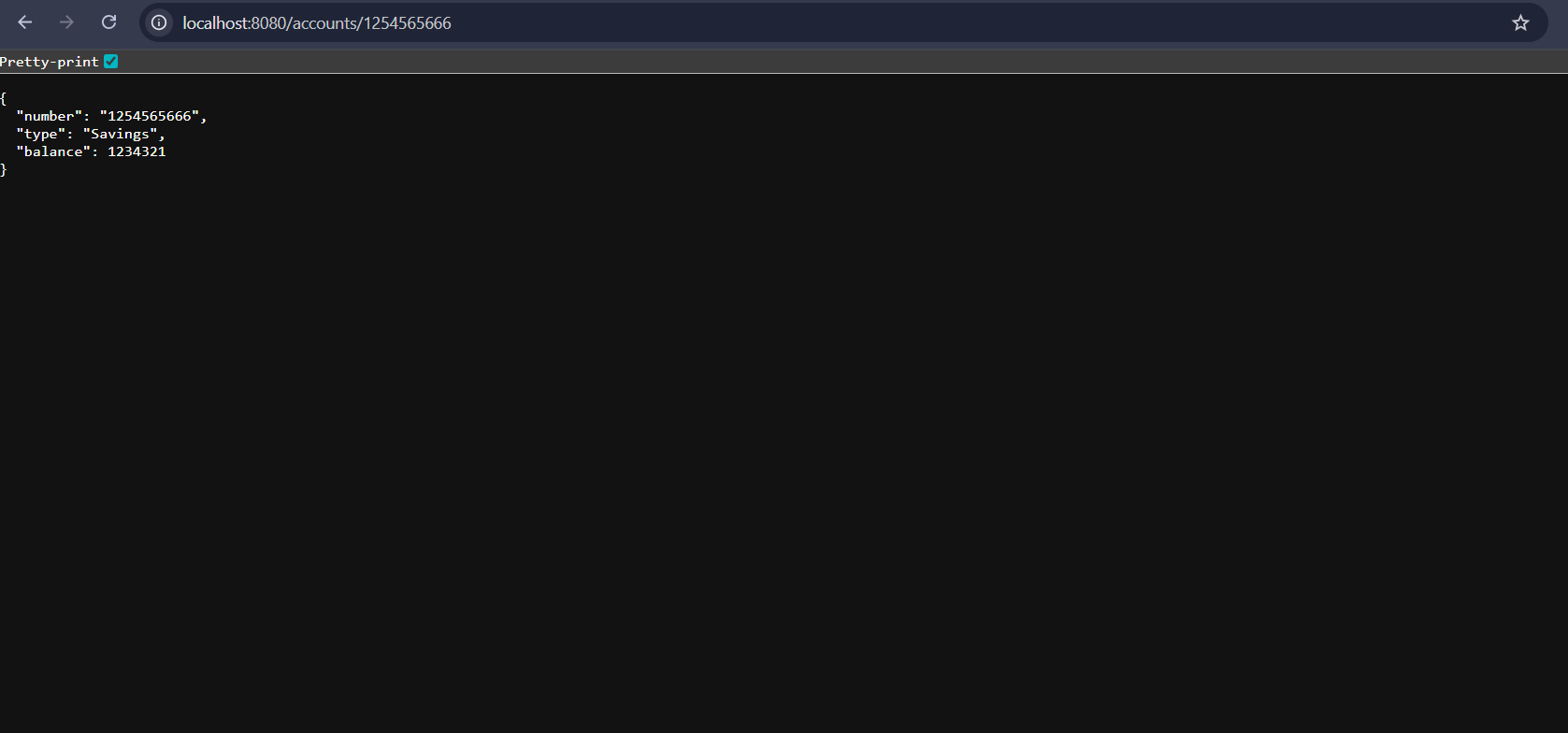
Right-click src/main/java > com.cognizant.account. New > Class → Name it AccountController. Add the following code:

package com.cognizant.account;  
  
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", 1234321);  
 }  
  
 static 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;  
 }  
  
 public String getNumber() { return number; }  
 public String getType() { return type; }  
 public double getBalance() { return balance; }  
 }  
}

## Step 4: Run the Microservice

- Locate the main class: AccountApplication.java  
- Right-click > Run As > Java Application  
- Service will run on port 8080 by default  
- Test in browser or Postman: <http://localhost:8080/accounts/1254565666>

## OUTPUT: Account Microservices



# Create Loan Microservice

## Step 1: Create the Project

Repeat the same process:  
- File > New > Spring Starter Project  
- Name: loan  
- Group: com.cognizant  
- Artifact: loan  
- Package: com.cognizant.loan  
- Add DevTools and Spring Web dependencies  
- Click Finish

## Step 2: Add Controller

package com.cognizant.loan;

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

*@RestController*

*@RequestMapping*("/loans")

public class LoanController {

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

public Loan getLoan(*@PathVariable* String number) {

return new Loan(number, "Bike", 40000, 258, 18,"1 year");

}

static class Loan {

private String number;

private String type;

private double loan;

private int emi;

private int tenure;

private String repayperiod;

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

this.number = number;

this.type = type;

this.loan = loan;

this.emi = emi;

this.tenure = tenure;

this.repayperiod = repayperiod;

}

public String getNumber() { return number; }

public String getType() { return type; }

public double getLoan() { return loan; }

public int getEmi() { return emi; }

public int getTenure() { return tenure; }

public String getrepayperiod() { return repayperiod; }

}

}

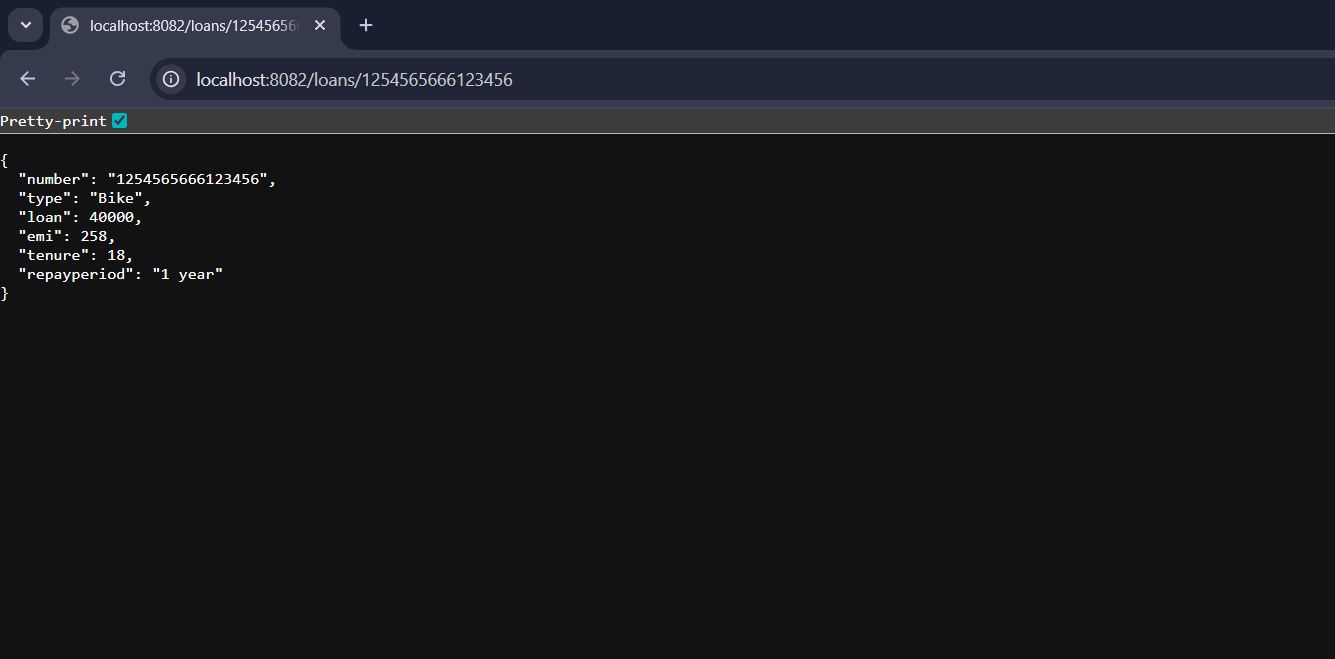
## Step 3: Change Default Port

In src/main/resources, open or create application.properties and add:  
server.port=8082

## Step 4: Run and Test

- Run LoanApplication.java as Java Application  
- Service will start on port 8081  
- Test in browser or Postman: <http://localhost:8082/loans/1254565666123456>

# OUTPUT: Loan Microservices



# Final Setup

- Account microservice is running on http://localhost:8080  
- Loan microservice is running on http://localhost:8082  
- Both microservices are now fully functional and independently running in Eclipse