**Microservices with API Gateway**

**Creating Microservices for account and loan**

In this practical exercise, we will build two Spring Boot microservices — one for managing account details and another for handling loan-related data. Each microservice will be developed as a standalone Spring RESTful web service using Maven, with its own pom.xml. Unlike a monolithic setup, the services will be separated into two independent applications. Both services will return static data without any backend integration.

### Account Microservice

* Create a folder named after your employee ID in the **D:** drive.
* Inside this folder, create another directory named **microservices**.
* This directory will hold all sample microservice applications created during the learning sessions.
* Open <https://start.spring.io/> in your browser.
* Fill out the form fields with the following values:
  + **Group**: com.cognizant
  + **Artifact**: account
* Choose the following dependencies:
  + **Developer Tools** → Spring Boot DevTools
  + **Web** → Spring Web
* Click **Generate** and download the resulting ZIP archive.
* Extract the account folder from the ZIP file and move it into the microservices directory created earlier.
* Open a command prompt inside the account directory and execute the following command to build the project:

go

CopyEdit

mvn clean package

* Import the project into **Eclipse IDE** and create a controller method to fetch account information based on the account number. Refer to the details below:
  + **HTTP Method**: GET
  + **Endpoint**: /accounts/{number}
  + **Example Response**:

json

CopyEdit

{

"number": "00987987973432",

"type": "savings",

"balance": 234343

}

* Run the main application class and verify the endpoint using a browser or Postman.

### Loan Microservice

* Repeat the previous steps used for creating the Account Microservice.
* Access <https://start.spring.io/> again.
* Enter the following values:
  + **Group**: com.cognizant
  + **Artifact**: loan
* Select these dependencies:
  + **Spring Boot DevTools**
  + **Spring Web**
* Click **Generate**, download the ZIP file, and extract the loan folder into the same microservices directory.
* Open the command prompt in the loan directory and run:

go

CopyEdit

mvn clean package

* Import the project into Eclipse IDE and implement a controller method to return loan account details using the loan number.
  + **HTTP Method**: GET
  + **Endpoint**: /loans/{number}
  + **Example Response**:

json

CopyEdit

{

"number": "H00987987972342",

"type": "car",

"loan": 400000,

"emi": 3258,

"tenure": 18

}

* Start the Loan Microservice while keeping the Account service running in the background.
* You will encounter a **port binding error** since both applications try to use the default port **8080**.
* To resolve this, open the application.properties file in the loan project and add the following configuration:

ini

CopyEdit

server.port=8081

* Save the changes and re-run the Loan application. This time, it should start successfully on port **8081**.
* Test the endpoint using port **8081** in your browser or API testing tool.

Now, both microservices are running independently on different ports and can be tested concurrently.  
To switch between the running services' logs in Eclipse, use the **monitor icon** in the Console view to toggle between console outputs.

**Account Microservice**

**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.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

import com.cognizant.account.model.Account;

@RestController

@RequestMapping("/accounts")

public class AccountController {

    @GetMapping("/{number}")

    public Account getAccount(@PathVariable String number) {

        // Return dummy response without backend connectivity

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

    }

}

**Account.java**

package com.cognizant.account.model;

public class Account {

private String number;

private String type;

private double balance;

public Account() {

}

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

this.number = number;

this.type = type;

this.balance = balance;

}

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;

}

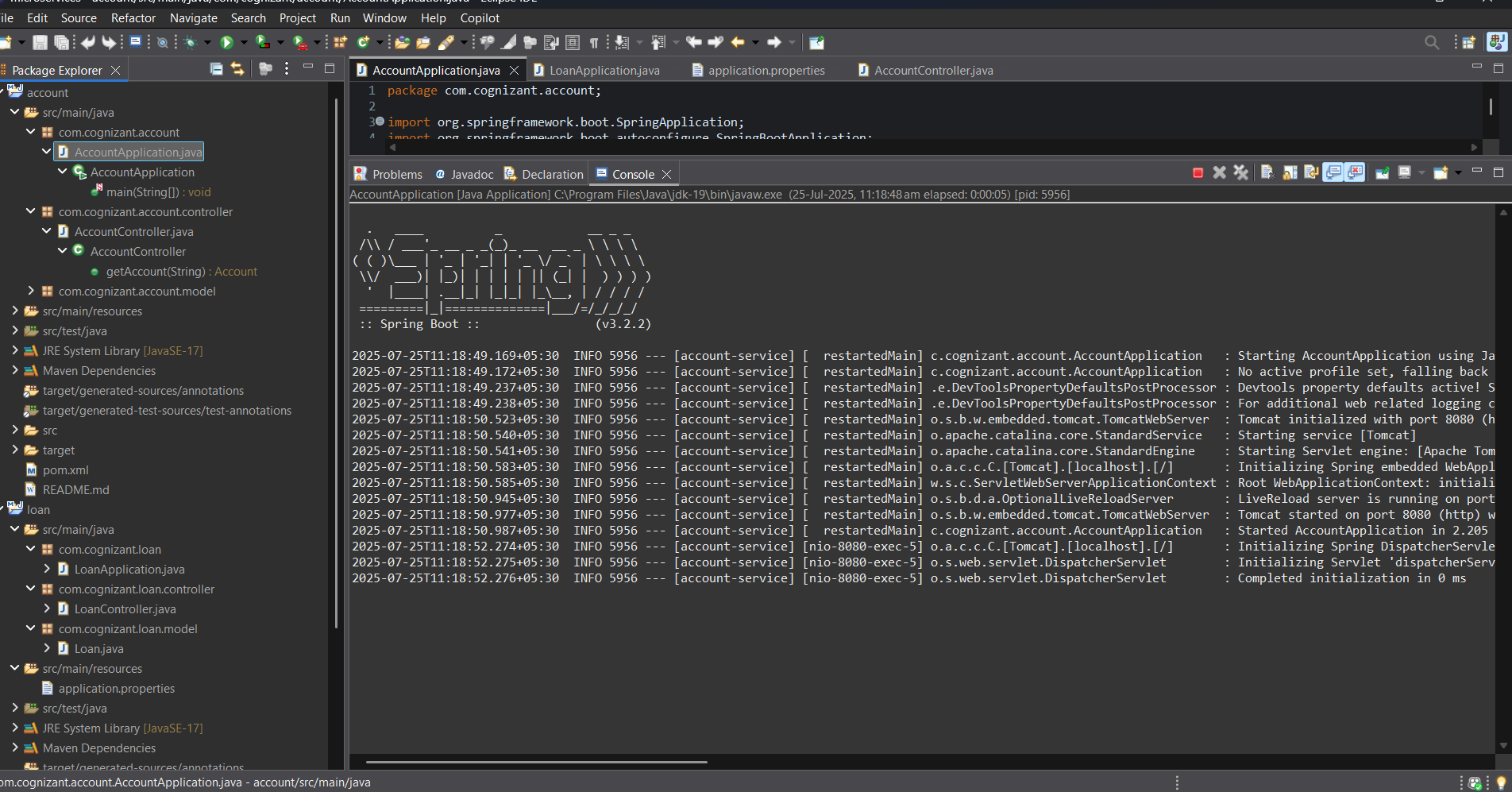
}

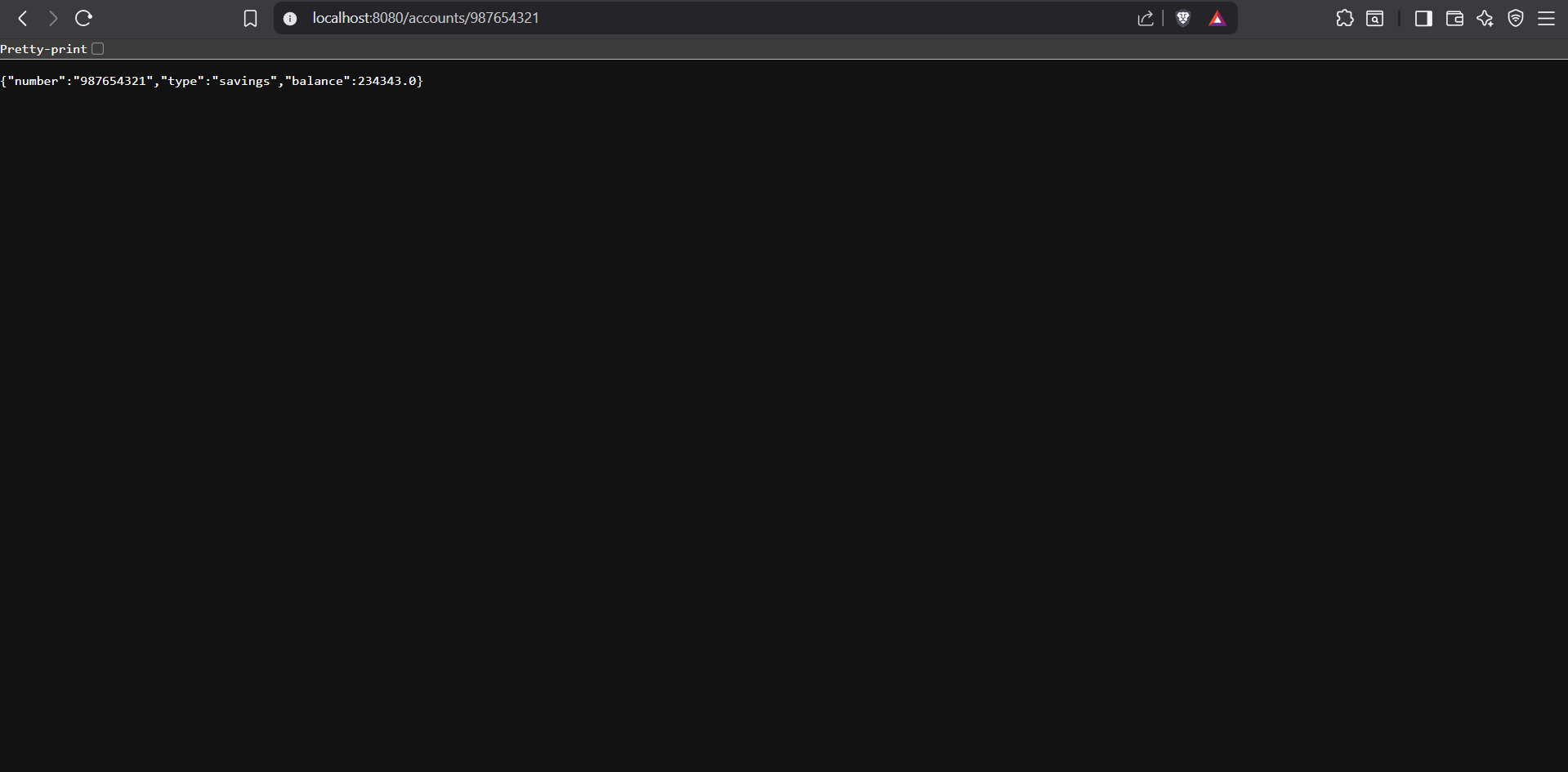
**application.properties**

# Account Microservice Configuration

server.port=8080

spring.application.name=account-service

**OUTPUT**

****

**Loan Microservice**

**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.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

import com.cognizant.loan.model.Loan;

@RestController

@RequestMapping("/loans")

public class LoanController {

@GetMapping("/{number}")

public Loan getLoan(@PathVariable String number) {

// Return dummy response without backend connectivity

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

}

}

**Loan.java**

package com.cognizant.loan.model;

public class Loan {

private String number;

private String type;

private double loan;

private double emi;

private int tenure;

public Loan() {

}

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;

}

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 double getEmi() {

return emi;

}

public void setEmi(double emi) {

this.emi = emi;

}

public int getTenure() {

return tenure;

}

public void setTenure(int tenure) {

this.tenure = tenure;

}

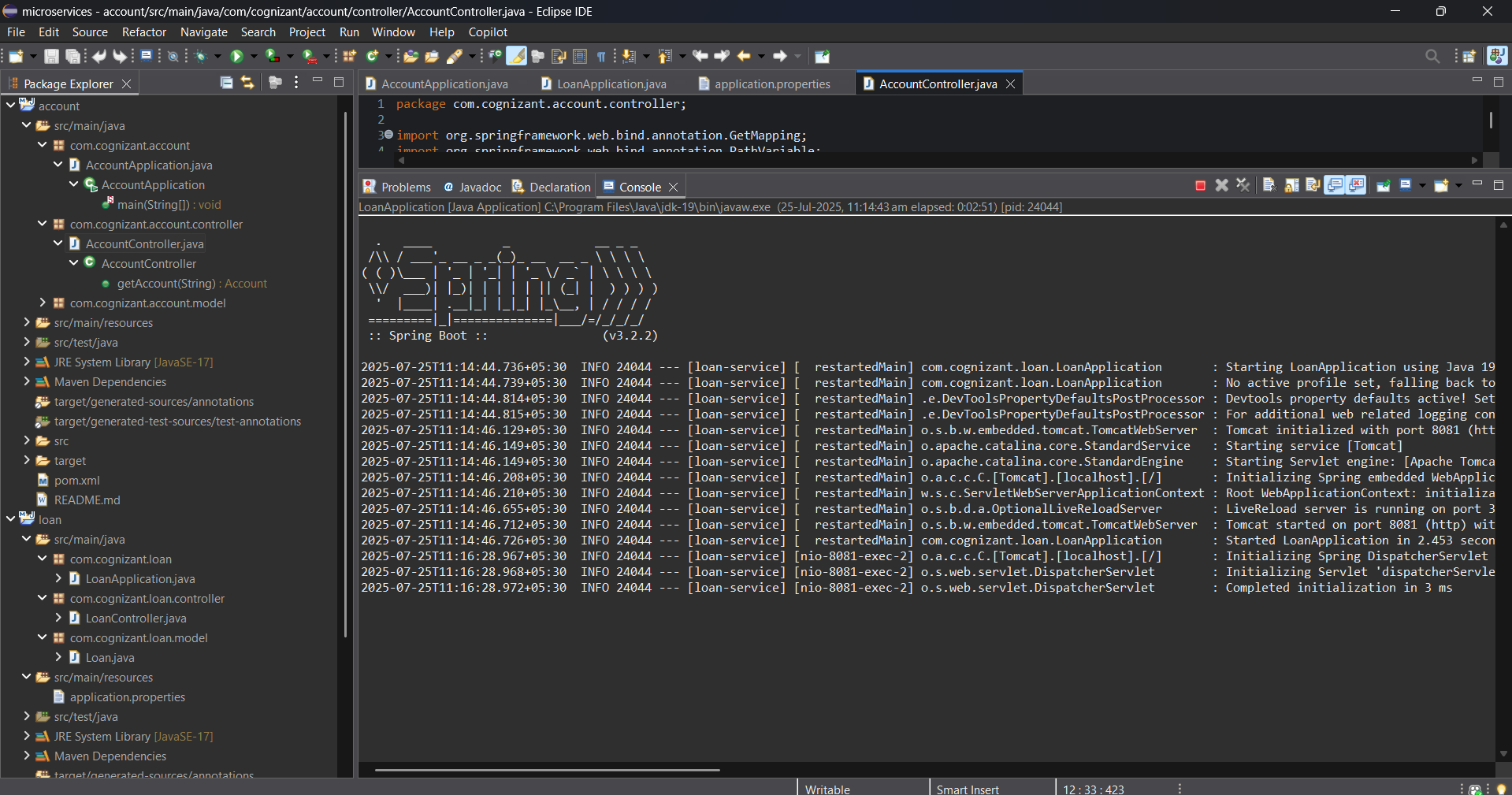
}

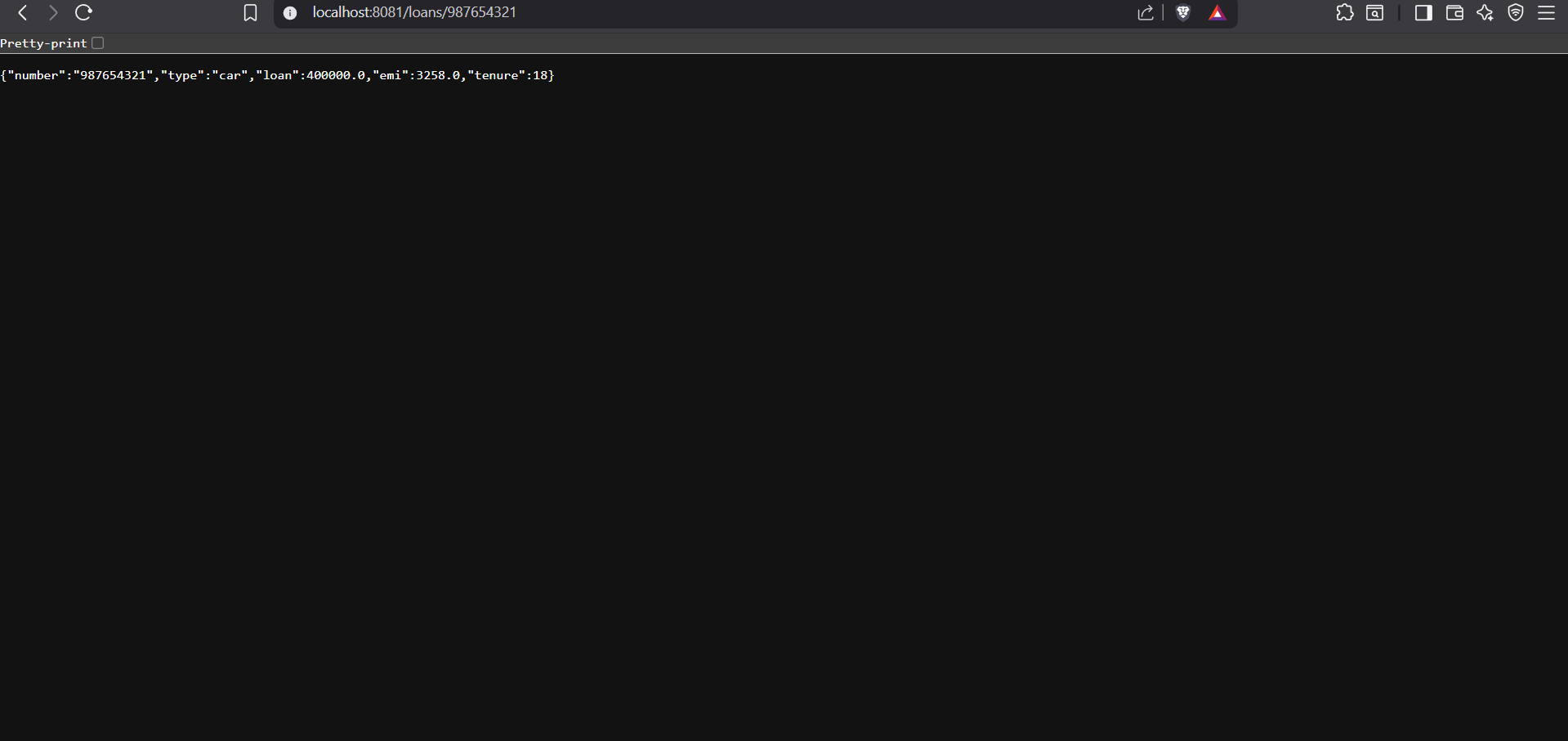
**application.properties**

spring.application.name=loan

server.port=8081

**OUTPUT**

****

****