***Microservices with API gateway***

# *Creating Microservices for account and loan*

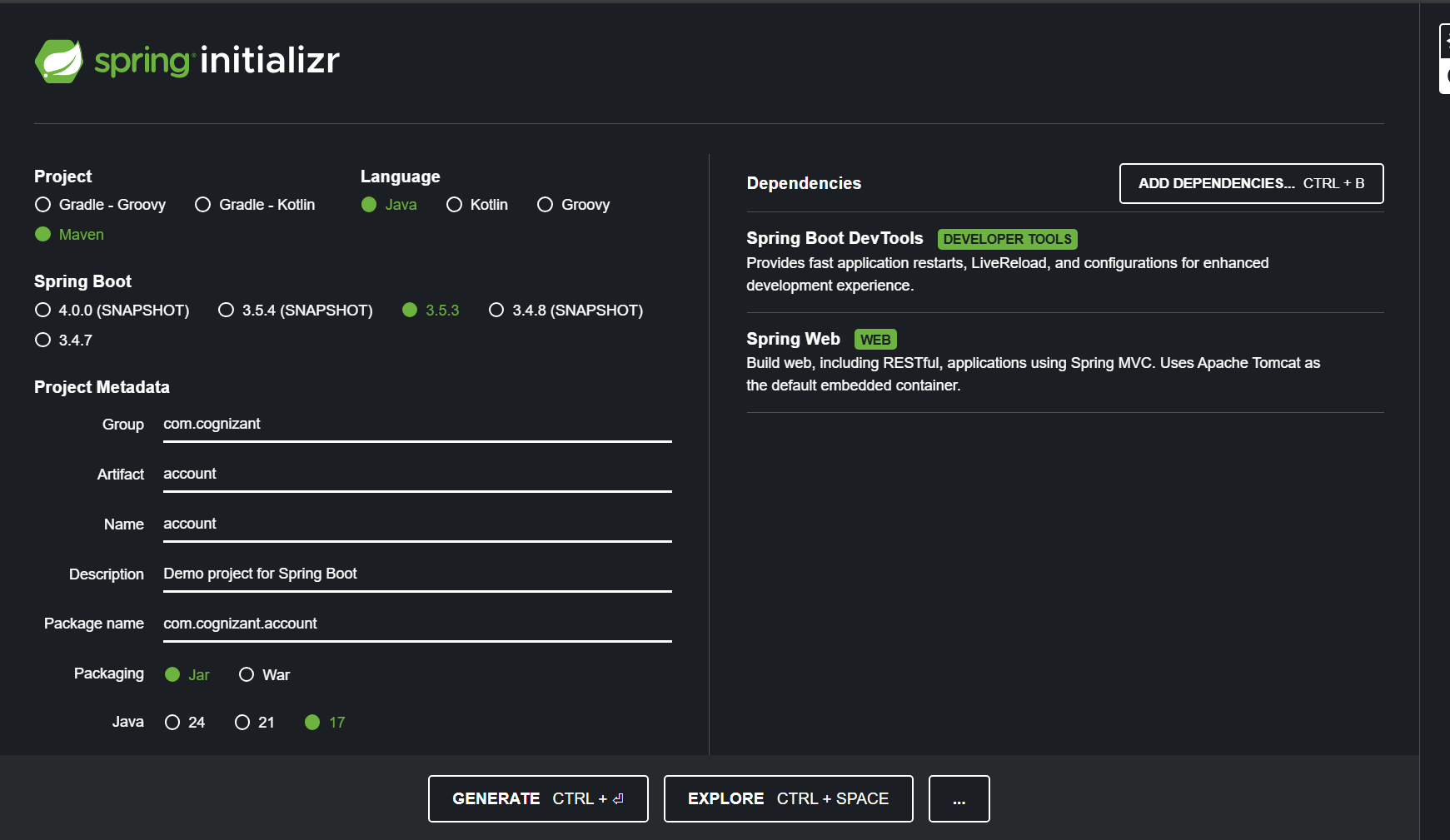
In this hands-on activity, we will build two separate microservices for a banking system. One microservice will manage account-related operations, while the other will handle loan services. Each microservice will be developed as an independent Spring Boot RESTful web application using Maven, with its own pom.xml file. Instead of combining both functionalities into a single application, we are dividing them into standalone services to reflect real-world microservice architecture. These services will be basic implementations without any database connectivity and will return static responses. We will begin by setting up the Account Microservice following the steps below.

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.

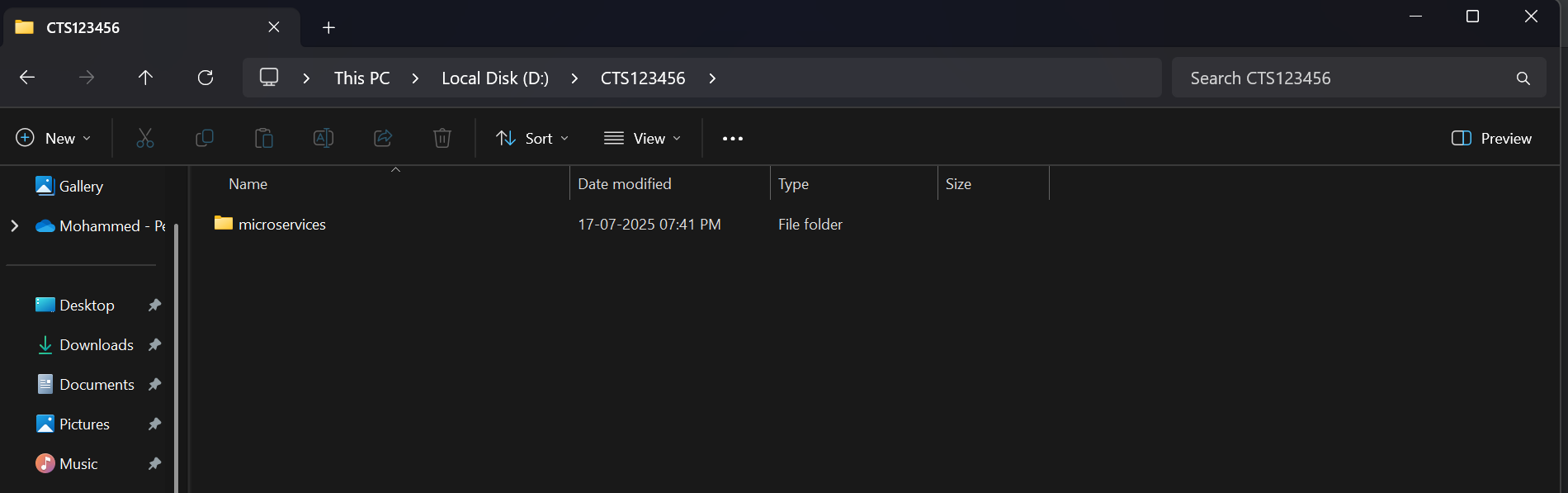
***Follow steps below to implement the two microservices:***

1. Make a new directory on the D: drive using your employee ID as the name.
2. Inside that directory, create another folder named microservices to store your practice projects.
3. Open <https://start.spring.io> in a web browser.
4. Fill in the form fields as follows:
   * Use com.cognizant for the Group.
   * Enter account for the Artifact.
5. Add these dependencies to the project:
   * Enable Spring Boot DevTools under Developer Tools.
   * Choose Spring Web under the Web category.
6. Click the "Generate" button to download the zip archive of the project.
7. Extract the folder named account from the zip file and move it into the microservices folder created earlier.
8. Launch Command Prompt inside the account directory and run the Maven build using mvn clean package.
9. Open Eclipse and import the account project as an existing Maven project.
10. Add a controller with a method to return dummy account details for a given account number, using a GET request to the /accounts/{number} endpoint.

{ number: "00987987973432", type: "savings", balance: 234343 }

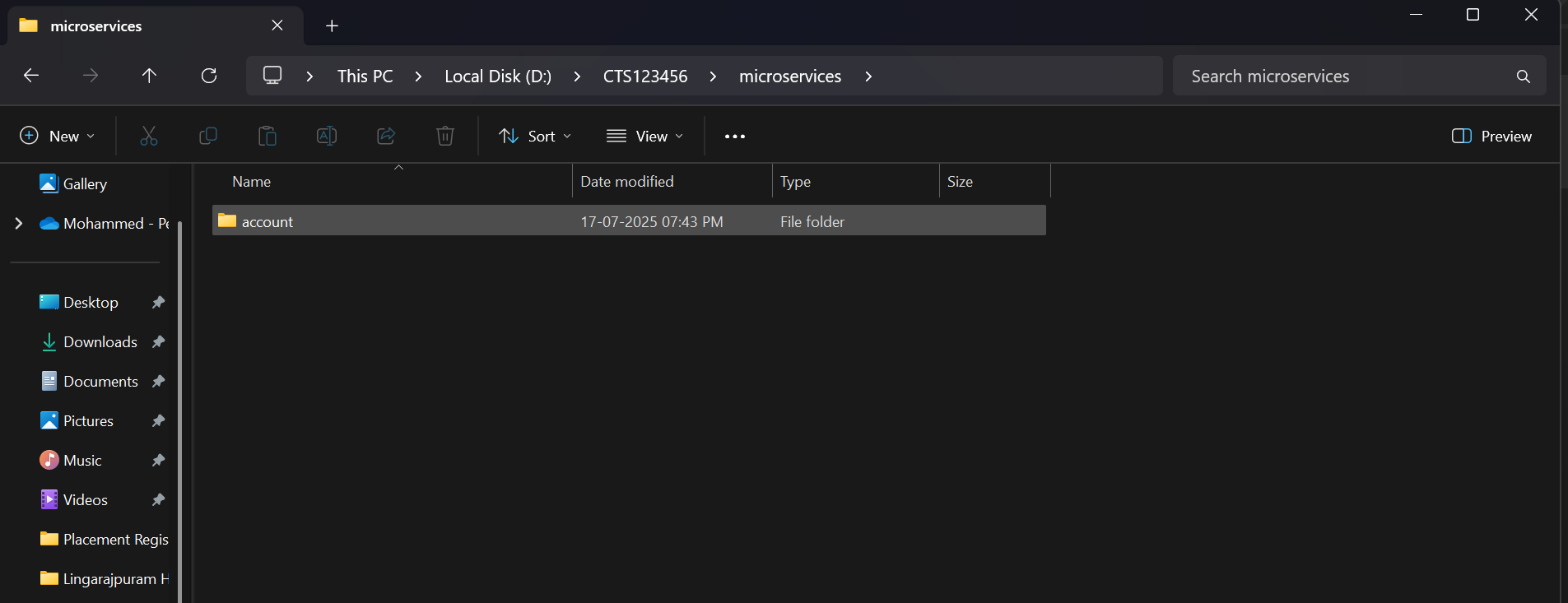
****

***Create a folder in D drive***

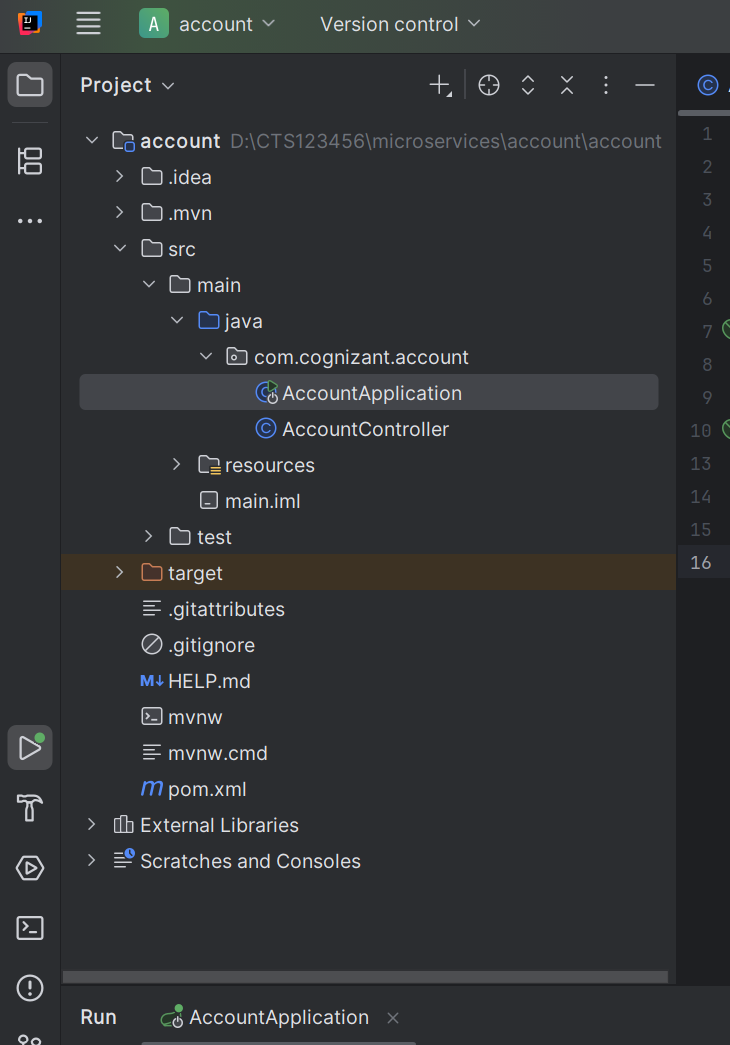
****

**Create a folder named microservices inside the created folder**

**Then, extract the spring initializr file and paste it inside the *“microservices*” folder**

****

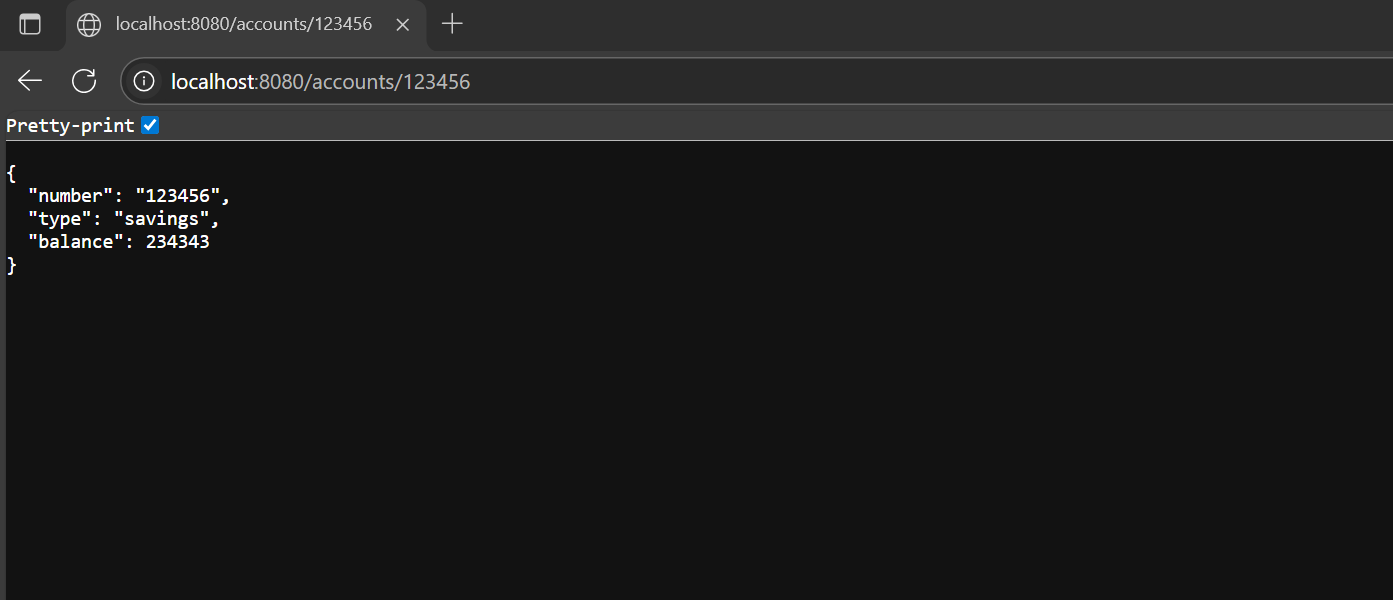
***Folder Structure:***

****

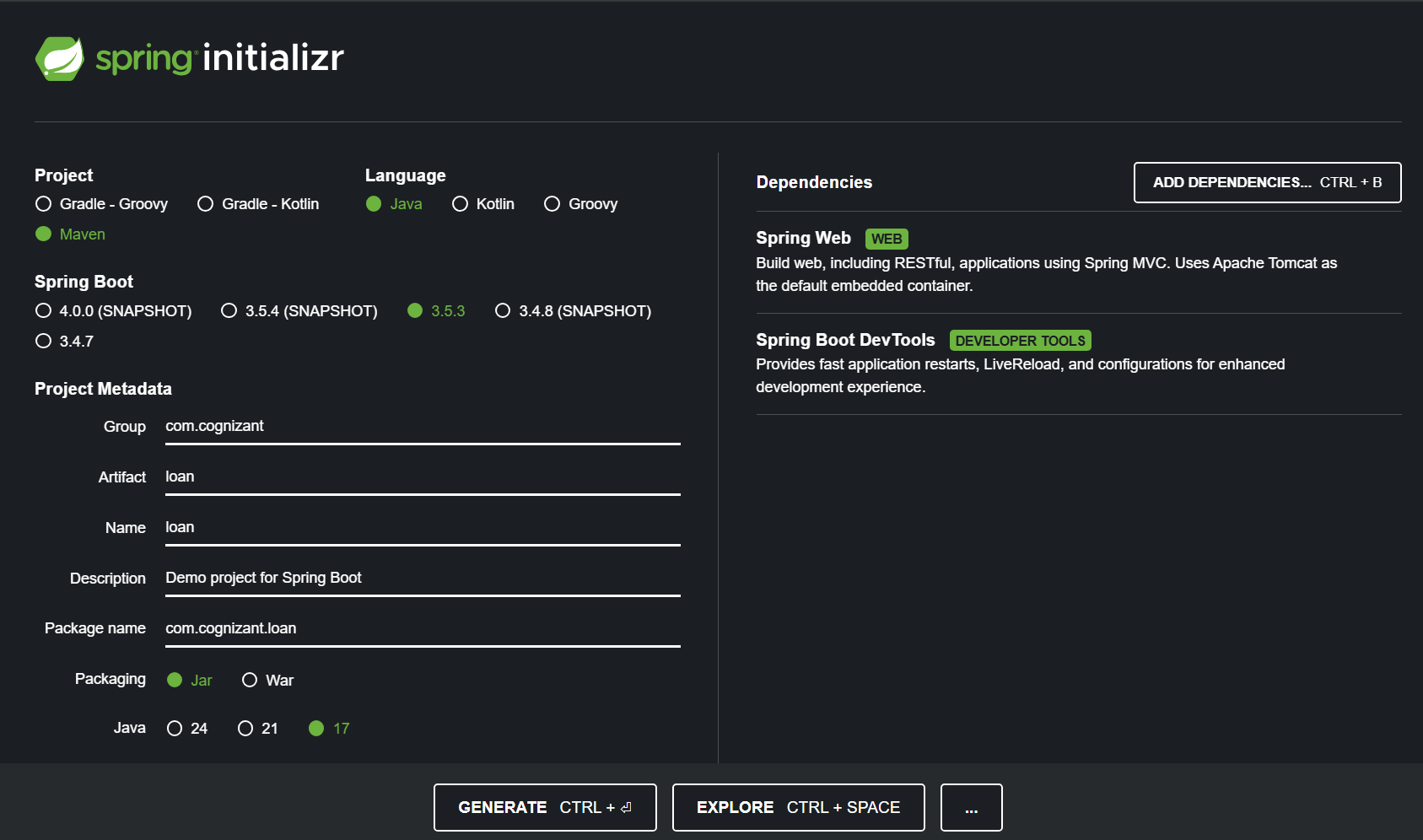
***AccountController.java***

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", 234343);  
 }  
  
 record Account(String number, String type, double balance) {}  
}

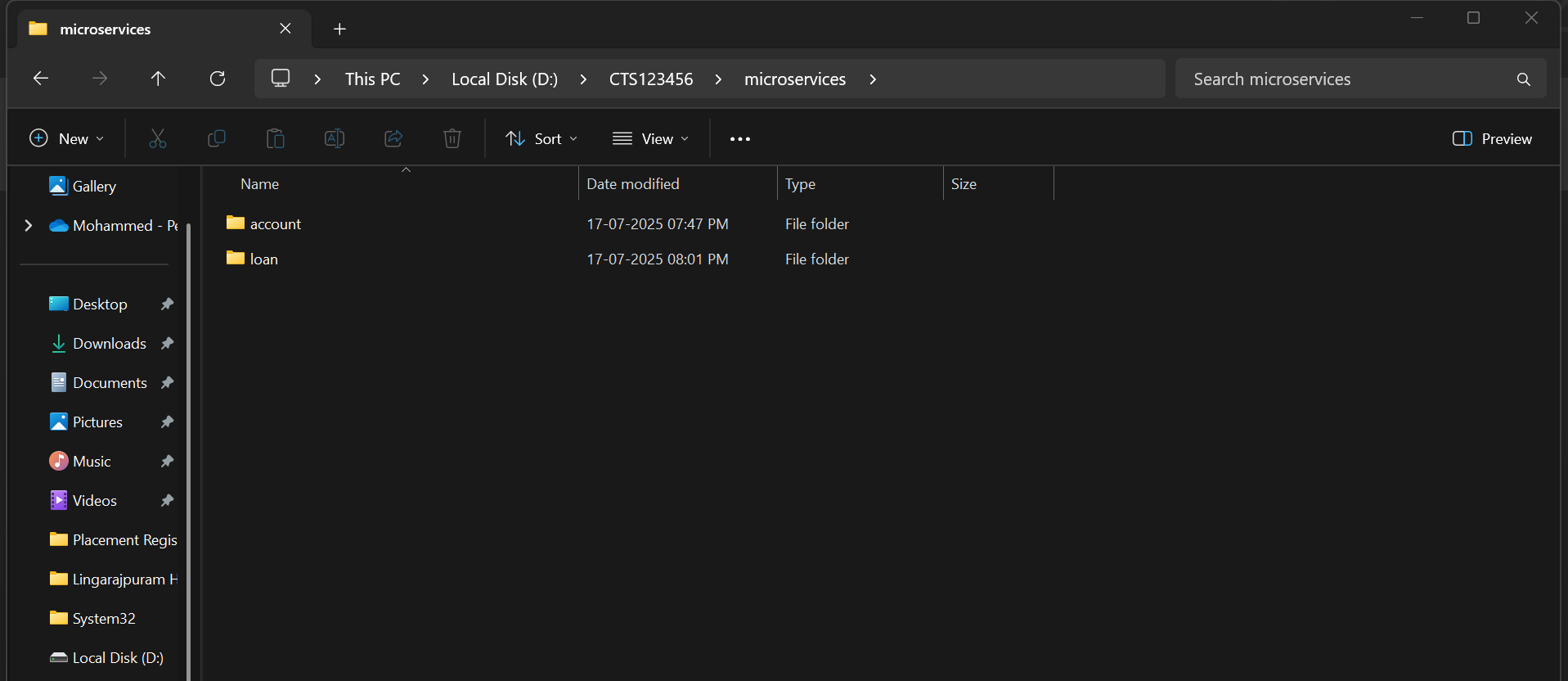
**Then, run AccountApplication.java and go to** [**localhost:8080/accounts/123456**](http://localhost:8080/accounts/123456)

****

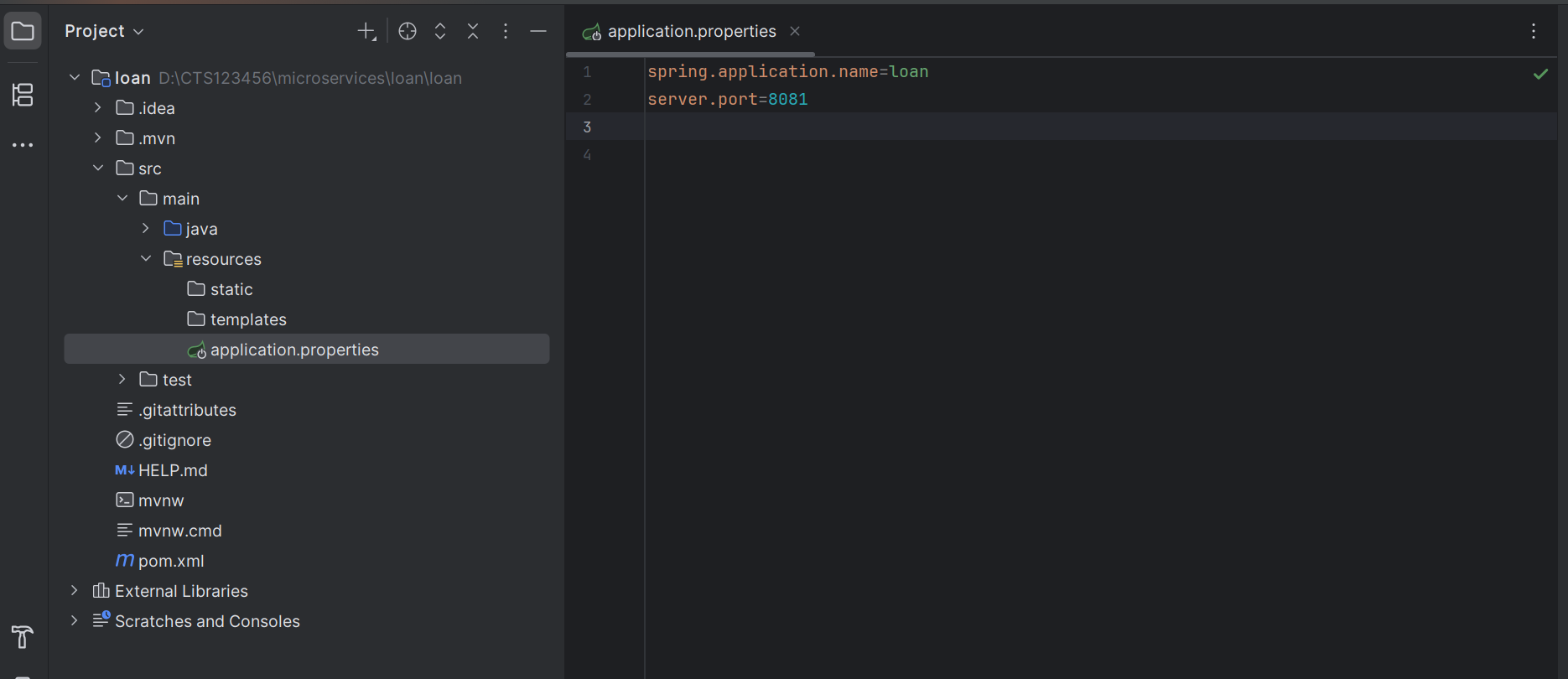
***Creating Loan microservice:***

****

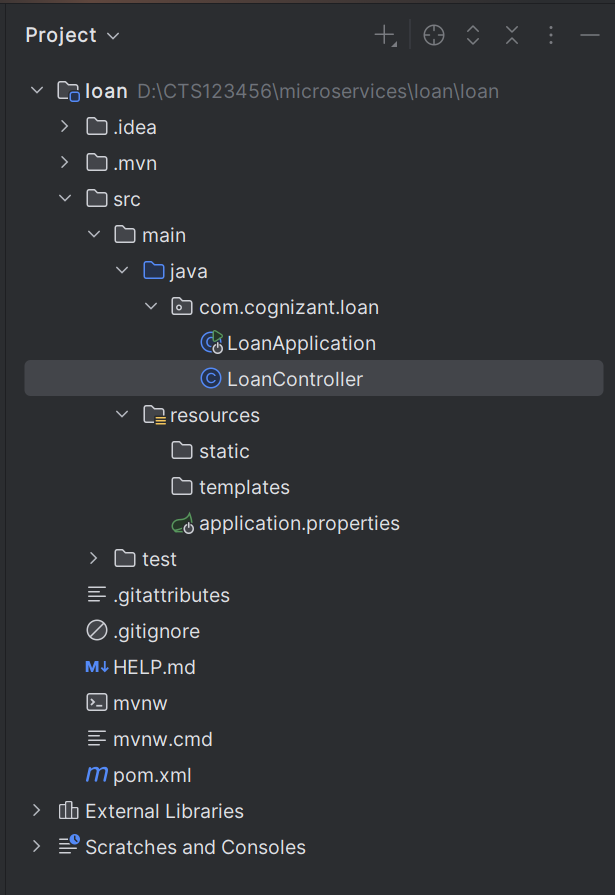
**Then move it to the same old folder,**

****

***Set custom port for the loans project by adding 🡪 server.port=8081***

****

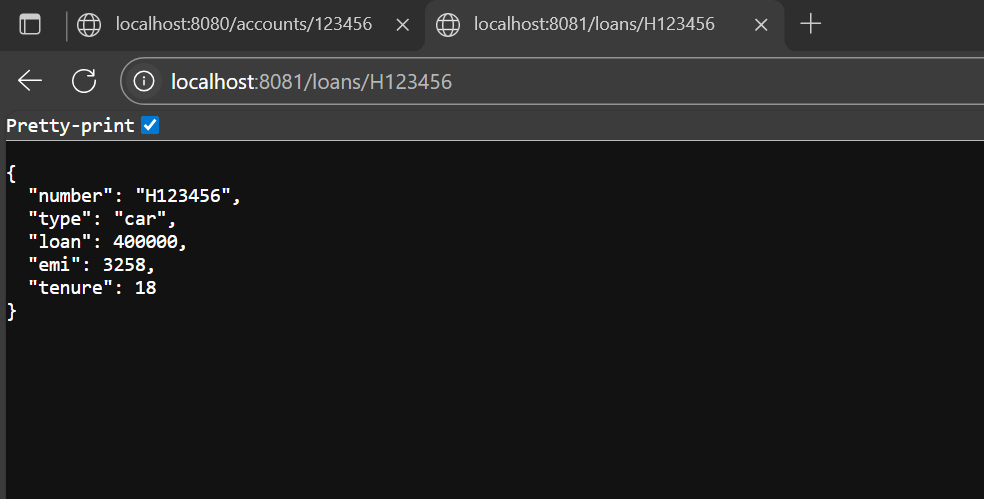
***Folder structure:***

****

***LoanController.java***

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, "car", 400000, 3258, 18);  
 }  
  
 record Loan(String number, String type, double loan, double emi, int tenure) {}  
}

**Run LoanApplication.java and go to** [**http://localhost:8081/loans/H123456**](http://localhost:8081/loans/H123456)

****

*I have successfully created and configured two microservices:*

1. The Account service running on port 8080
2. The Loan service running on port 8081