

3 Service class UserService

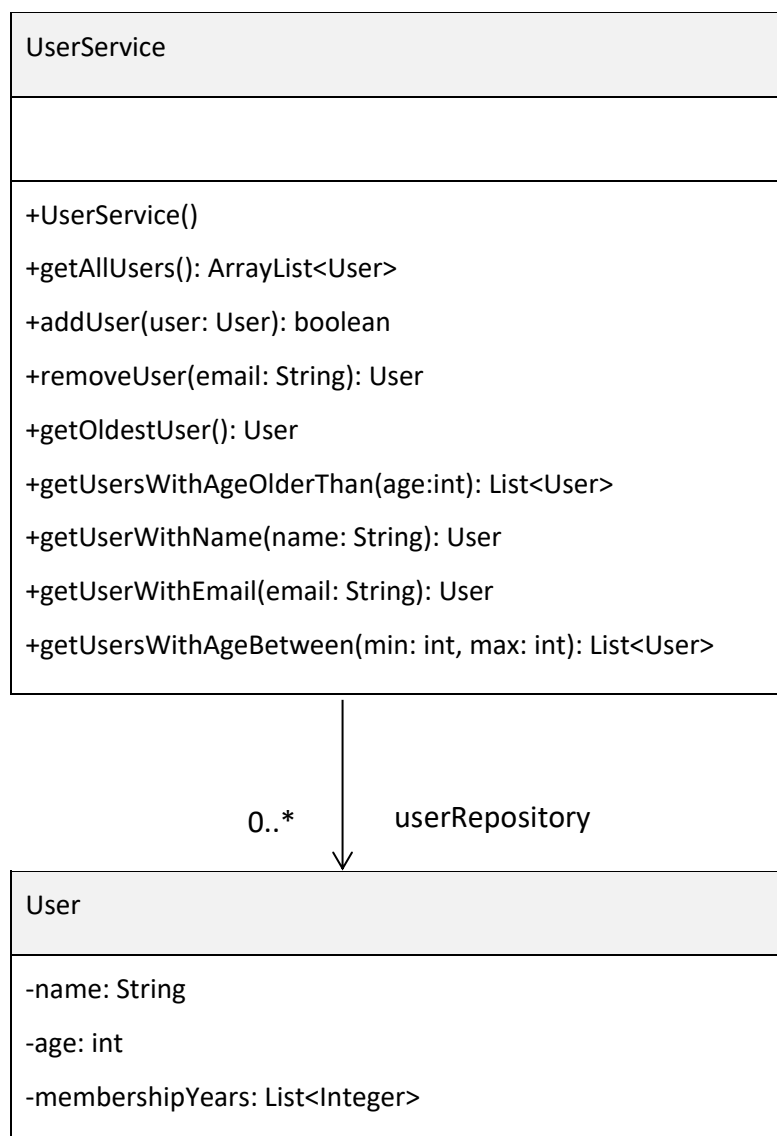
3.1 Copy-Paste demo code


Create a class UserService in src/main/java/demo and copy the [democode](#) in it.

Create a class UserServiceTest in src/test/java/demo and copy the [democode](#) in it.

3.2 Class diagram UserService

Given the class diagram for the User class, look at it and try to understand it as good as you can. Keep the class diagram open while you will do paragraph 2.4.





-email: String -password: String
+User(name: String, age: int, email: String, password: String) +getName(): String +getAge(): int +getEmail(): String +getPassword(): String +countYearsOfMembership(): int +addMembershipYear(year: int) +countMembershipYearsAfter1999(): int +getFirstMembershipYear(): int +getNumberOfMembershipYearsIn2000(): int +isPasswordCorrect(password: String): boolean +toString(): String

3.3 Run the tests and implement the UserService class until all tests pass

Follow the same steps as described in paragraph 2.4.

3.4 Add new functionality

We want to add the following functionality: get the list of all users who were membership in a given year. Add the right method(s) in the User and UserService classes and implement them. You don't need to write test methods but test it in de DemoApplication class by creating a happy case and an unhappy case.

3.5 Push your code to your GitHub repository

Only when all test methods are colored green, commit and push your code to your repo. Don't forget to give a descriptive comment each time you commit and push!

4 REST API – GET

4.1 Copy-Paste demo code

Create a class `UserRestController` in `src/main/java/demo` and copy the [democode](#) in it.

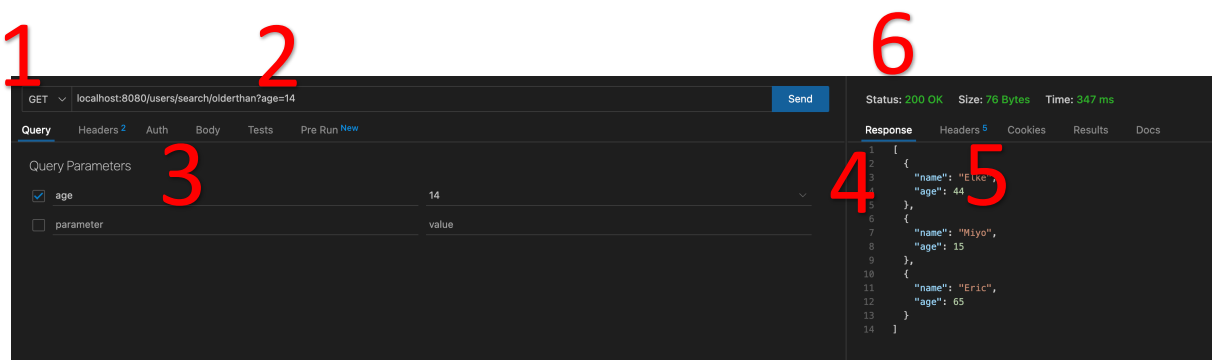
4.2 Thunder Client

Install the Thunder Client extension for Visual Studio code if you don't already have it.

Thunder Client is a lightweight Rest API Client Extension for Visual Studio Code for Testing APIs. Use it to send requests to your application server and see the response and check/test the answers.

Create a collection and call it “users”, we will save all the URIs that we will test in this collection.

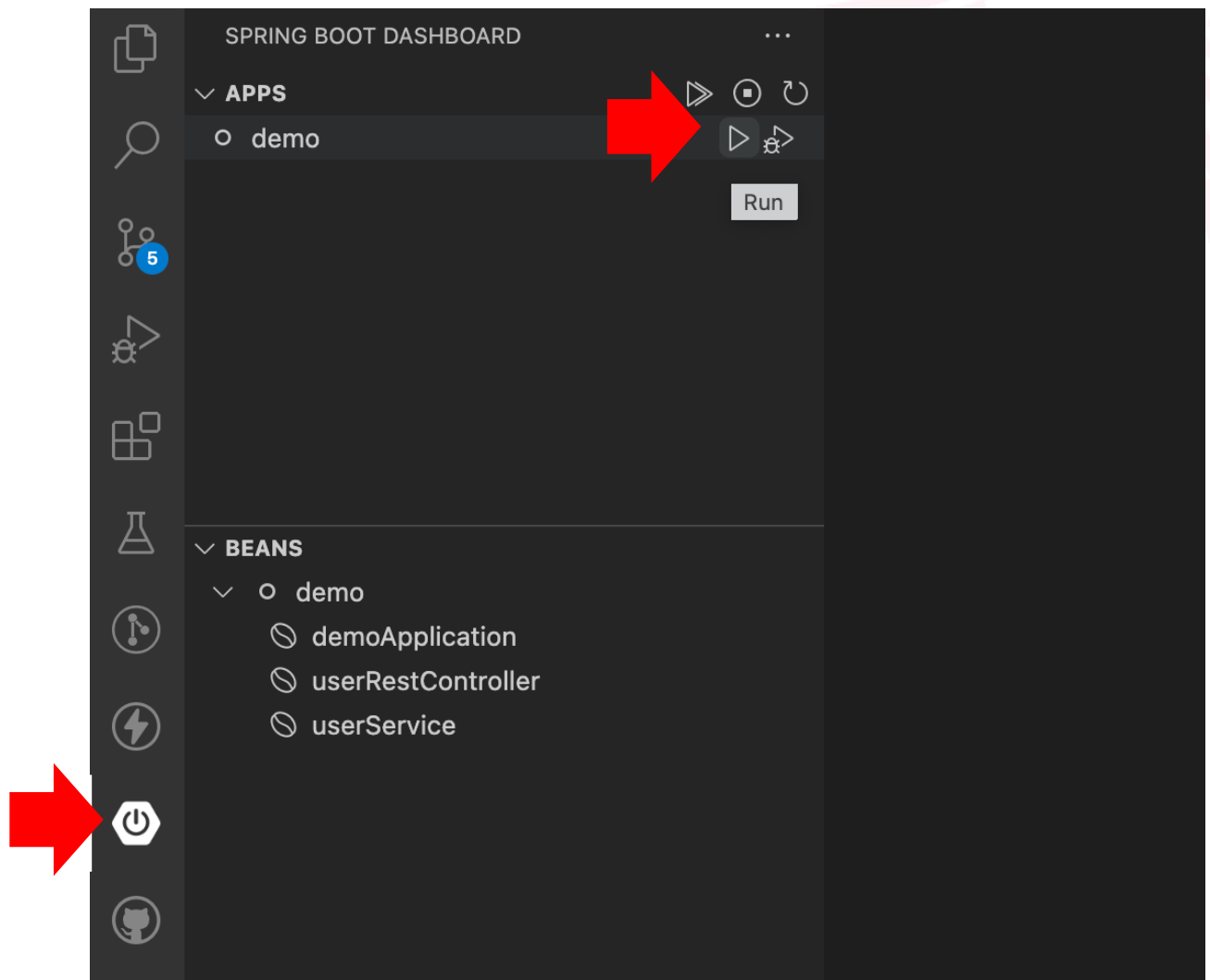
In what follows you find a screenshot of a request in Thunder Client. Answer the questions below the figure.



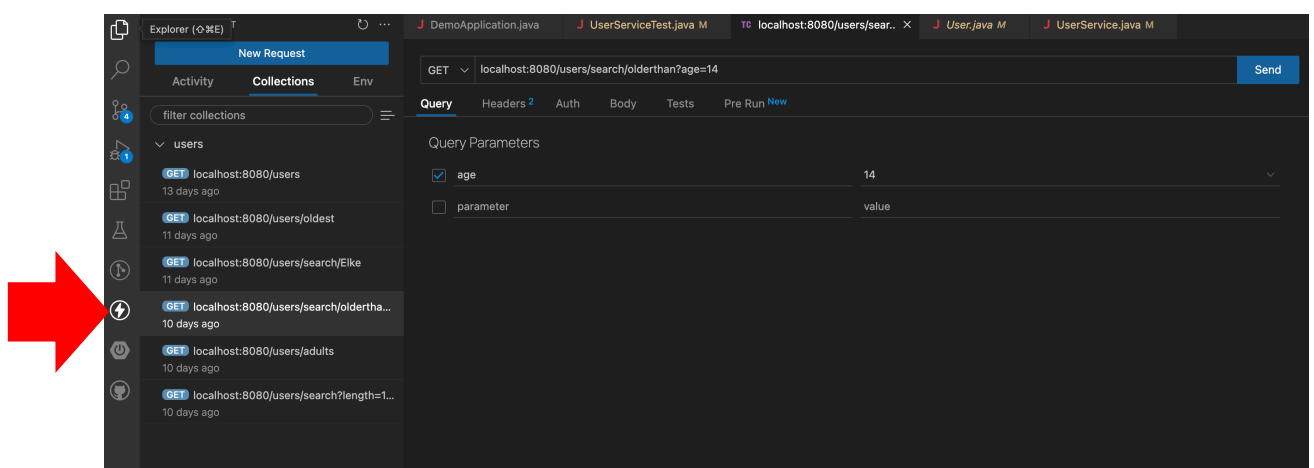
1. Here you see a GET-request. Give other types of request (click on arrow ∨).
2. Mark different components of the request: host, path, parameters, query string.
3. Change the parameter in the query string. What happens?
4. Analyze the body of the response.
5. Look at the headers of the response. There should be at least one that you need to be able to explain.
6. What does status 200 OK mean? Given an example of another status code.

4.3 Understanding demo code

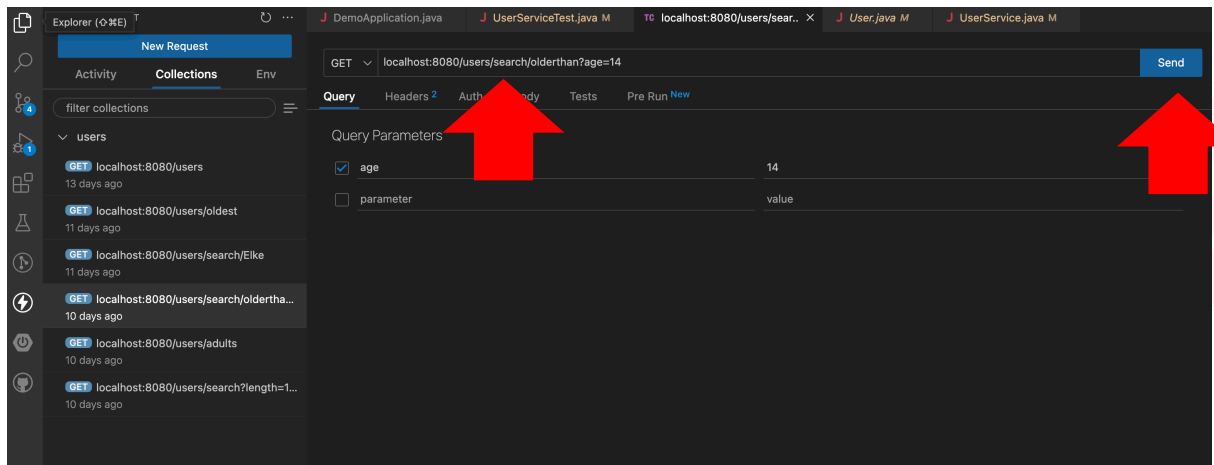
1. Start your Spring Boot demo application via the Spring Boot Dashboard tab.



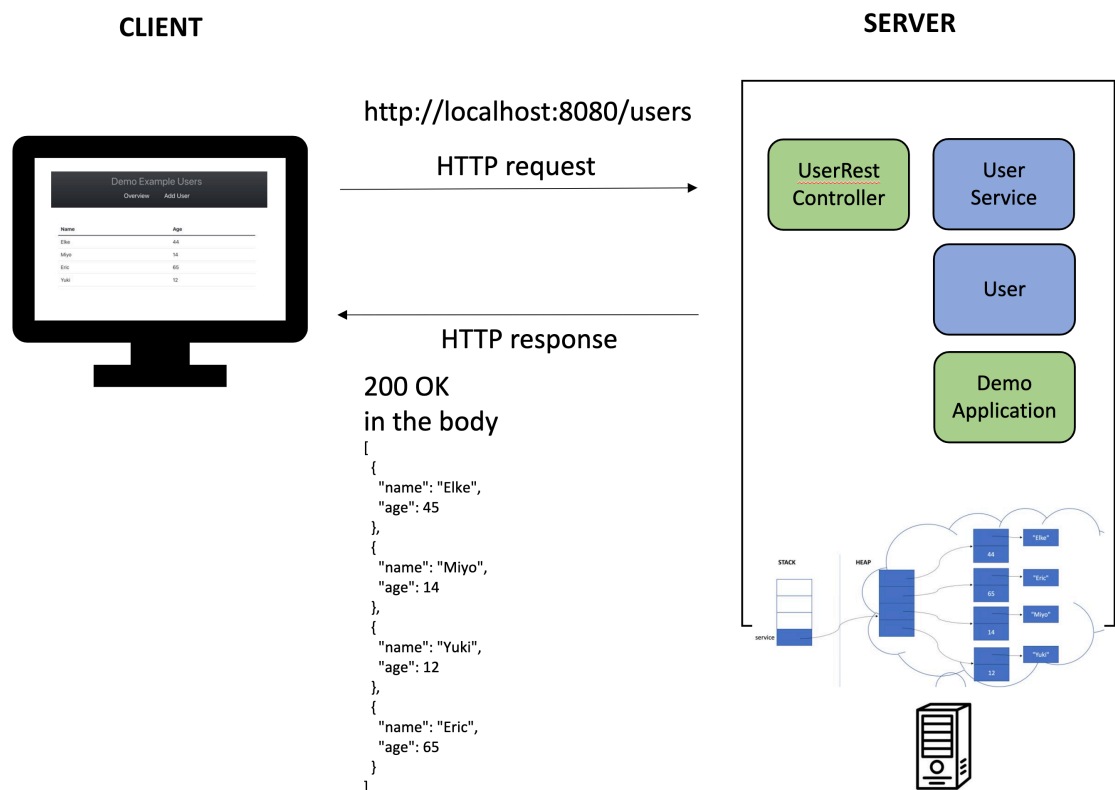
2. Go to the Thunder Client tab in Visual Studio Code.



3. Send the GET request <http://localhost:8080/users/search/olderthan?age=14>.



- Analyze the interaction between client and server and draw in it a schema. Use the schema below to draw the interaction between client and server below. The schema below is the client-server interaction for the <http://localhost:8080/users> GET request.



- Try the same URI for the age 50 and analyze this request-response and draw what happened.

4.4 Rest API

Write the necessary methods in the UserRestController so that given URI's work correctly.

Take care of parameters. Be aware of the difference between

- Request parameter
- Path variable

Method	URI	Return
GET	http://localhost8080/users/adults	<ul style="list-style-type: none">- In JSON format the list of all data of all adult (older than 17) users if any exist, if no adult users exist [] is returned.- [] is returned when no users are registered.
GET	http://localhost8080/users/search/email/{email}	<ul style="list-style-type: none">- In JSON format all the data of the user with the given email, if no user is found with this email empty body response is returned.
GET	http://localhost8080/users/search?email=email&age=age	<ul style="list-style-type: none">- In JSON format the list of all data of all users with the given email and age, if there are no found [] is returned.
GET	http://localhost:8080/users/search/age/{min}/{max}	<ul style="list-style-type: none">- In JSON format the list of all data of the users which ages are between the given min and max values (min and max value not included), if no users match [] is returned.

4.5 Test the URIs

Write in Thunder Client in the collection “users” for each URI a request and save it.

Test all requests with all possible paths and save them.

Remark you can add several query parameters in Thunder Client in one request and you can select which one you want to send in your request that you are testing.

4.6 Interaction client server

Make a schema of the client server interaction for at least 3 requests:

1. one without parameter
2. one with parameter as path variable
3. one with parameter as request parameter/query string

4.7 Add URI for new functionality of 3.4

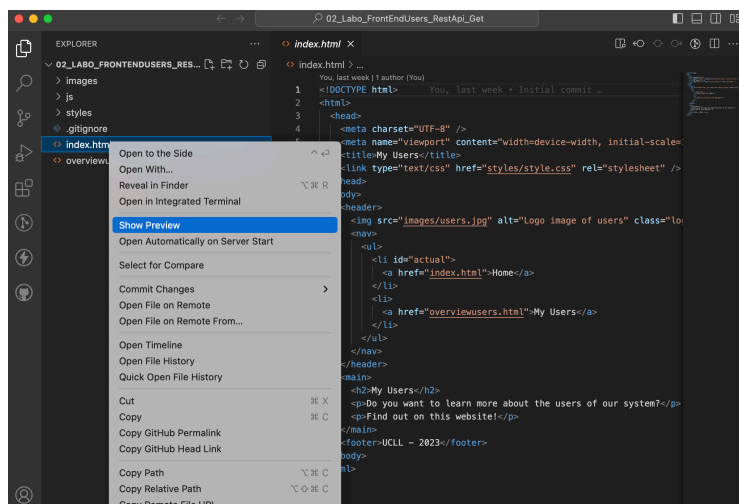
Add a GET request with a correct URI for the new functionality that you implemented in paragraph 3.4 and test it in Thunder Client.

4.8 Front-End

Download the given Front-End that we made for you and enjoy your project.

https://github.com/UCLLBackEndDevelopment/02_Labo_FrontEndUsers_RestApi_Get

To start the Front-End



Make sure your Back-End is still running 😊