

Development Approach

Homework 4

Adv. & Distributed Programming Paradigms

CSC 3374 \_ 01

Asmaa Dalil

**Table of Contents:**

1. [Purpose **3**](#_bookmark0)
2. [Development steps **3**](#_bookmark1)
3. [User manual **4**](#_bookmark2)

# Purpose:

The attached application enables the consumer to get screenshots of a remote system (Provider) periodically where the user specifies the period of cycles.

# Development steps:

Service Provider:

* + - On this side, we started by creating a Gradle project and adding the appropriate Spring Boot dependencies to it. This initialized the Spring Boot application. Then, we put our service's techniques and functions into actual, practical use for business purposes. The controller class (RemoteControl) was then marked as a web service by adding the spring annotation "@RestController," which designates it as a request handler. Additionally, we mapped HTTP requests to the REST controller class using the "@RequestMapping" annotation. Similar to this, the "@GetMapping" annotation makes sure that methods are mapped from HTTP GET requests to the appropriate methods. To address this issue, the annotation "@CrossOrigin"

- The service is publicized and ready to accept and manage consumer requests as soon as

the service provider is operational. The specifics of how the request is received and

handled are not disclosed to the programmer.

Service Consumer:

On this code, we first constructed an HTML code that primarily consisted of a text field and a button. Making a JavaScript application with a function that would be invoked whenever a user clicked the button indicated in the html code was the next stage. This function sends a http request to the provider as its initial action, which exposes a web-based RESTful service. The method that was called, RxHR.get('URL'), returns an object of type RxJS.Observable. Then, by subscribing to this observable, the HTTP response is acquired. The answer will also contain a Base64-encoded screenshot of the distant machine. This image will subsequently be shown via the HTML image component. To retrieve screenshots from the remote host on a regular basis after the predetermined number of seconds, we first subscribed to a distinct observable we created using Rx.interval(seconds). We only printed the pictures when the operator filter successfully recovered them (response status code equal to 200).

User Manual:

Service Provider:

 The user only needs to start the service. This is done through the following steps:

* Opening the command line / terminal
* Changing the directory to the gradle project folder
* Building the project using the command: **gradlew build**
* Running the project using the command: **gradlew run**

Service Consumer:

  The user accesses index.html in any browser. After entering the number of seconds he wants to get screenshots of the remote system for, the user must click the start button. When done, the user can click the stop button to stop receiving screenshots.