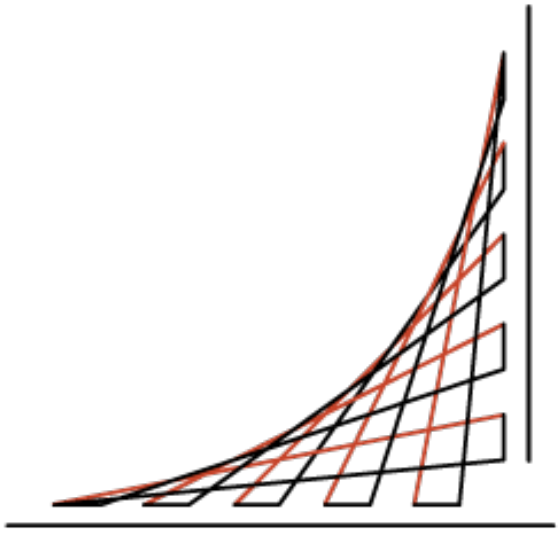


ESCUELA COLOMBIANA DE INGENIERÍA
JULIO GARAVITO

VIGILADA MINEDUCACIÓN

SYSTEMS ENGINEERING

Arquitecturas Empresariales

Laboratory 3

Luis Daniel Benavides Navarro

Author:
Nicolás Aguilera Contreras

Contents

1 Prerequisites

These are the necessary installations to run the application on your computer:

Maven - Dependency Management
Java 8 - Development Environment
Git - Version Control System

2 Introduction

This laboratory consists of the development of two challenges

2.1 Challenge 1

This challenge consists of developing a web server that supports multiple requests in a row (not concurrent). The server should return all requested files, including html pages and images. A website is built with javascript to test the server. The solution is deployed on Heroku.

2.2 Challenge 2

Using the preview server and java. This challenge consists on writing a Spark-like framework that allows you to publish "get" web services with lambda functions and allows you to access static resources such as pages, javascripts, images, and CSSs. In adittion , it includes an application that connects to a database from the server to test the solution. The solution is deployed on Heroku

3 Concepts

3.1 Functional interfaces

A functional interface is an interface that contains only one abstract method. [1]

3.2 Firebase Database

Is a store and sync data with a cloud-hosted NoSQL database. Data is synchronized with all clients in real time and remains available when the application is offline. [2]

4 Architecture

4.1 Class diagram

The program uses the HttpServer class to create a web server. Clients will connect to it via sockets.

The NanoSparkWebDemo class will have the Main method of the application. From

there it will call the NanoSpark class which will take care of initializing the NanoSpark-WebServer class.

The NanoSparkWebServer class will take care of starting the http server. Additionally, it will register the endpoints that the web server saves through a map that contains two parameters: the key corresponding to the path requested by a user and a binary function with two parameters (request and response) that results in a String containing the result of the user action, this last parameter is a functional interface and represents a lambda function.

Finally we have the DataBaseConnector class that will be in charge of connecting to an external Firebase database to develop one of the application's functionalities

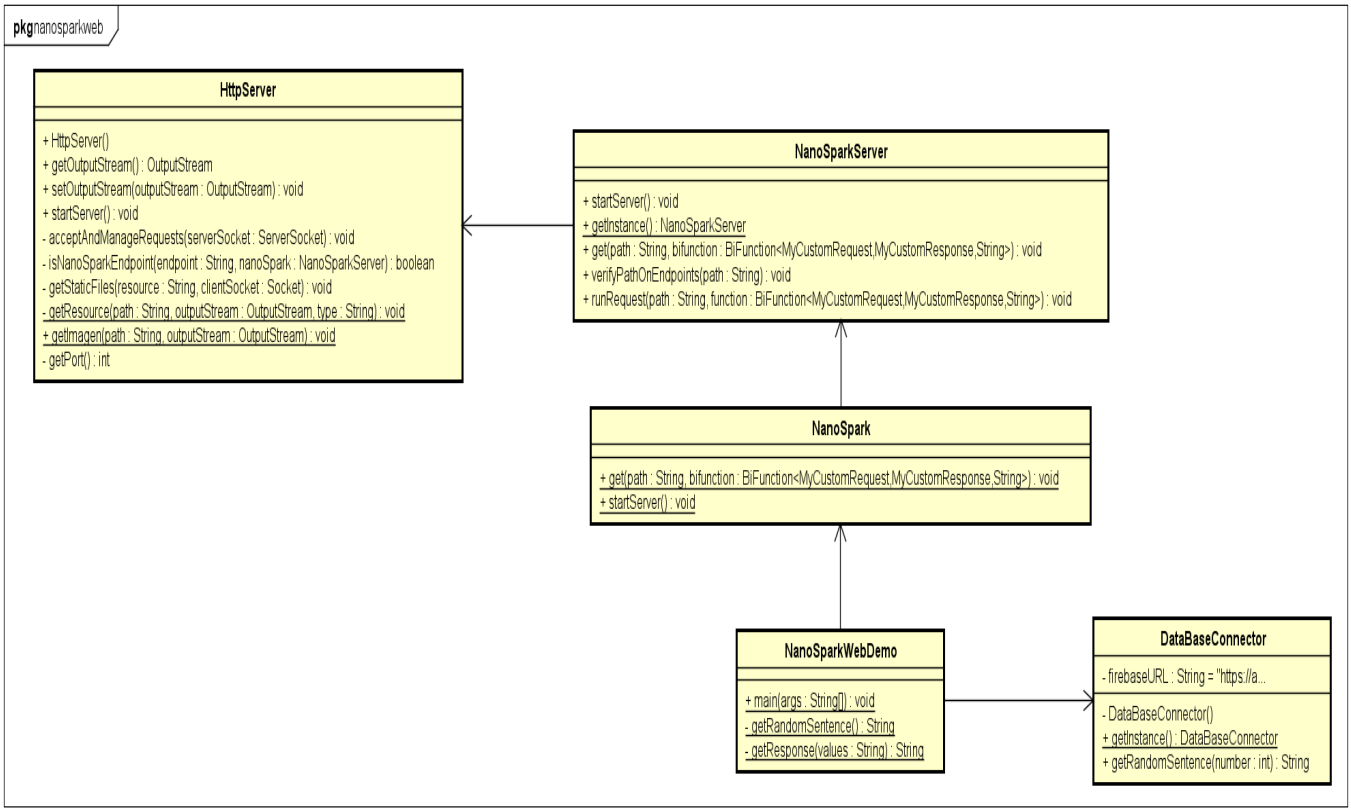


Figure 1: Class diagram

4.2 Components Diagram

We have three main components: FrontEnd, BackEnd, and FirebaseDB.

1. The FrontEnd component will store the static files of our application.

2. The Backend component will be in charge of managing the logic of our web server as well as our connection to the database.

3. Finally in FirebaseDB component we will store the data of our database.

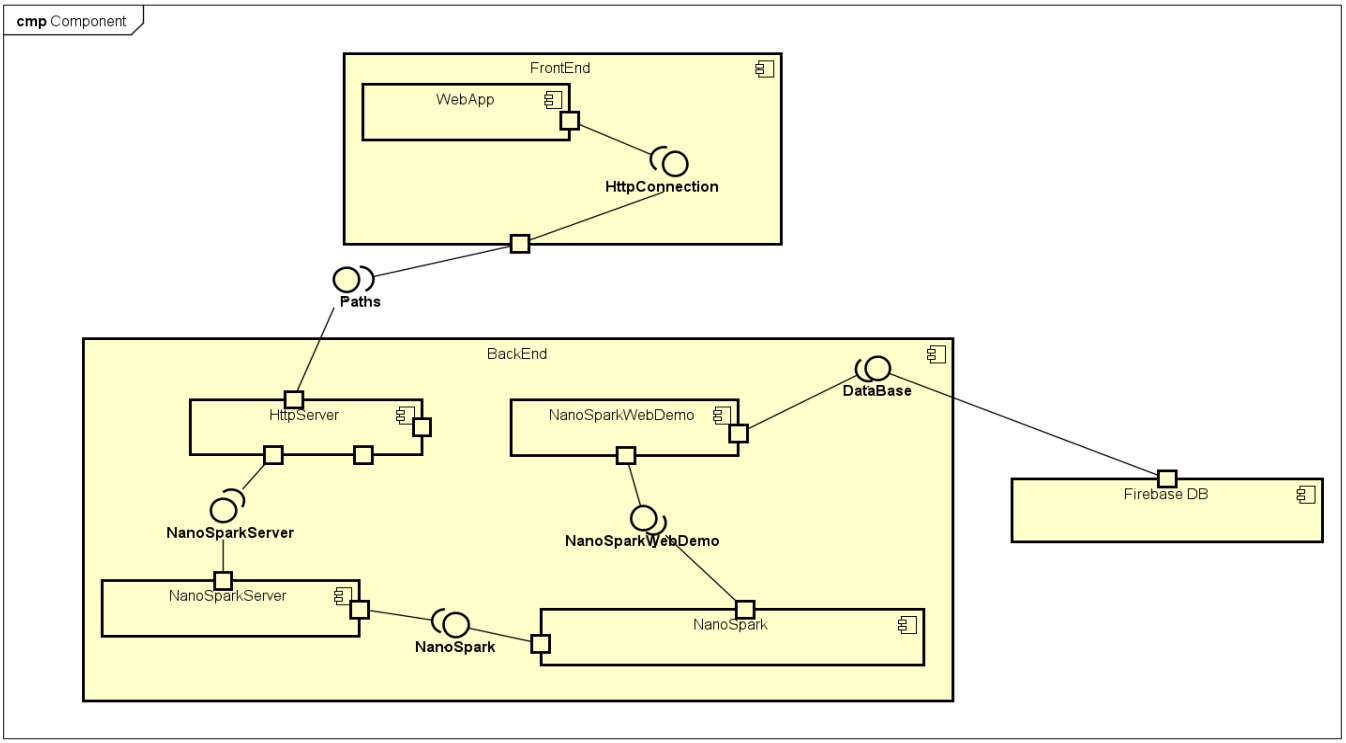


Figure 2: Component Diagram

4.3 Deploy Diagram

The application will be stored on the heroku cloud server. The client will connect to it through the HTTP protocol. On the other hand, our application will connect to the database through the HTTPS protocol

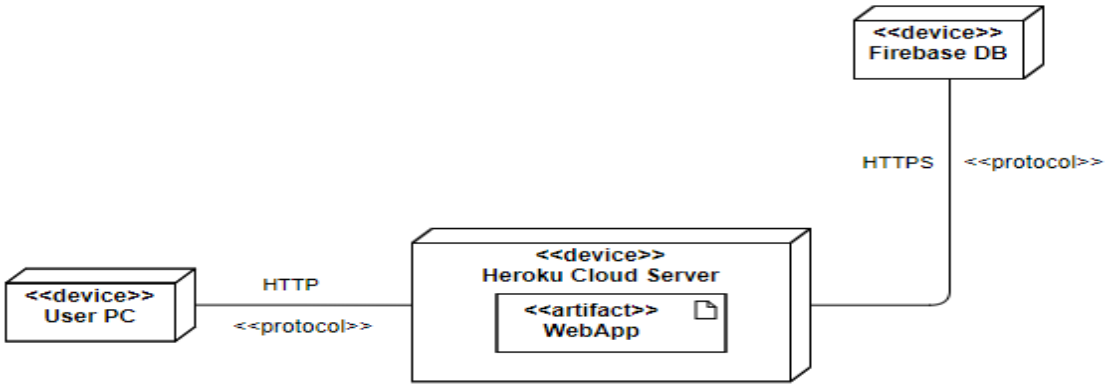


Figure 3: Deploy Diagram

5 Tests

3 tests were made proving that the application will correctly find the static resources

```
-------------------------------------------------------
 T E S T S
-------------------------------------------------------
Running edu.escuelaing.AppTest
Tests run: 3, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 2.296 sec

Results :

Tests run: 3, Failures: 0, Errors: 0, Skipped: 0

[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  5.291 s
[INFO] Finished at: 2021-02-11T21:39:05-05:00
[INFO] ------------------------------------------------------------------------
```

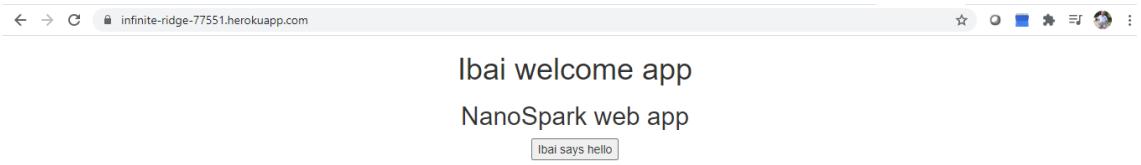
Figure 4: Test results

5.1 Test Challenge 1

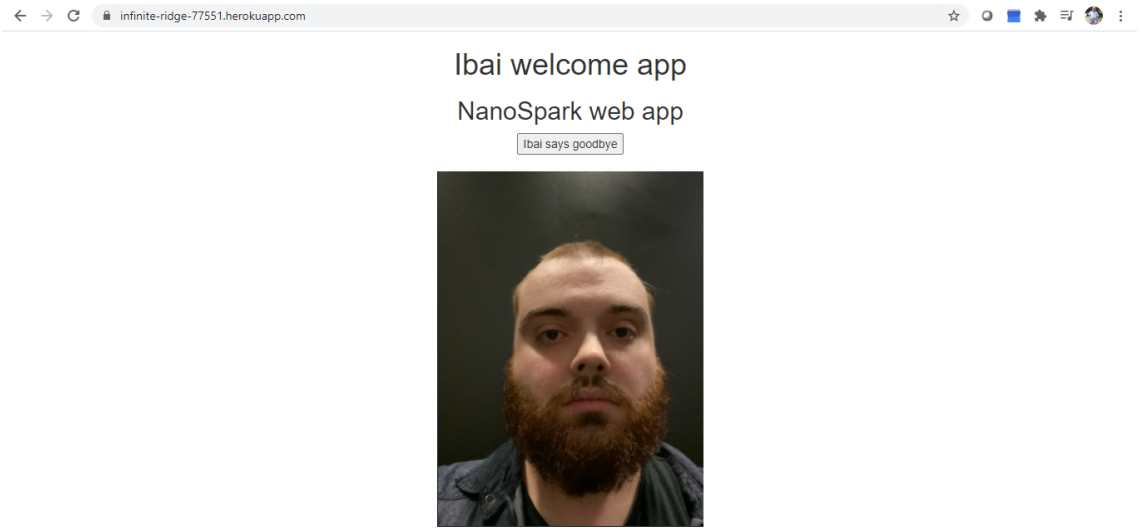
To test Challenge 1 you must enter the following link in your browser

If you use it locally : http://localhost:35000/index.html
If you use it on Heroku : https://infinite-ridge-77551.herokuapp.com/index.html

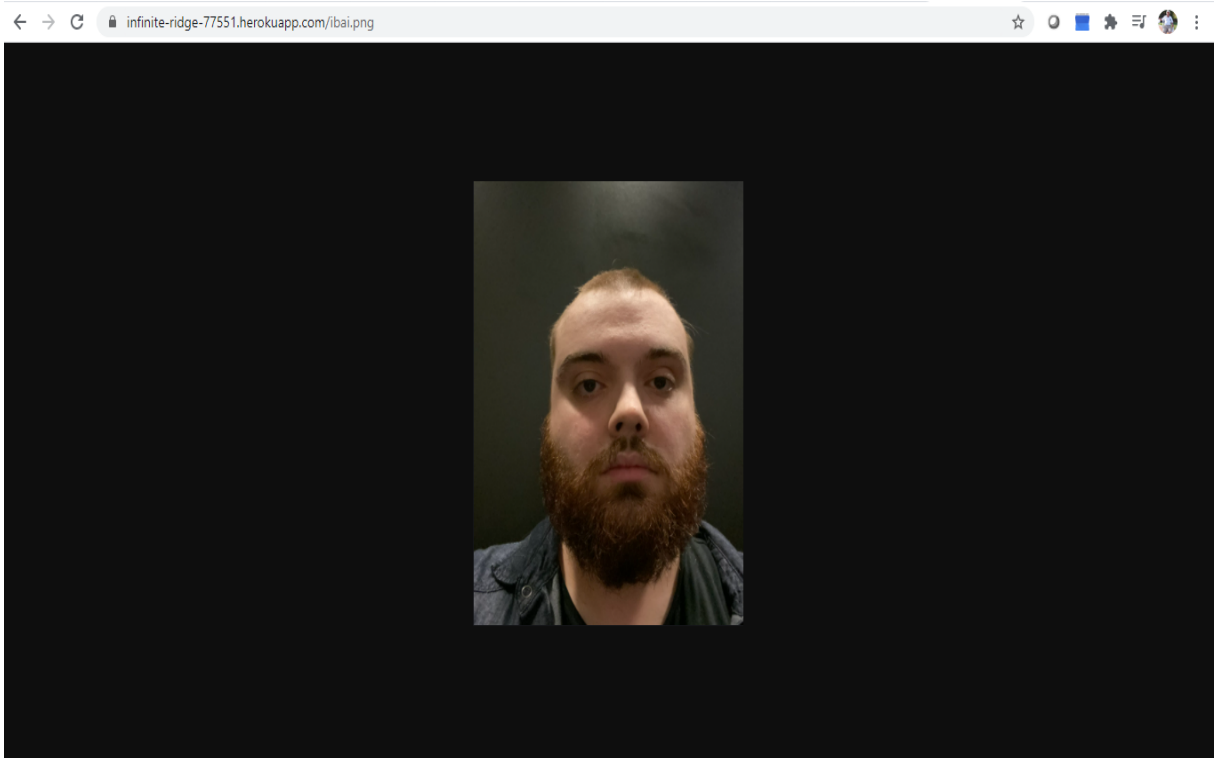
You will get access to the following page



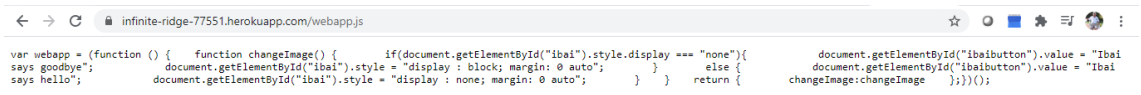
By clicking on the button you can load a static resource, in this case an image



You can also access that image from the url putting /ibai.png at the end of the url



You can also access the javascript on which the page is loaded from the url putting /webapp.js at the end of the url



5.2 Test Challenge 2

To use Challenge 1 you must enter the following link in your browser:

If you use it locally : http://localhost:35000/Apps/hello?value=name
If you use it on Heroku : https://infinite-ridge-77551.herokuapp.com/Apps/hello?value?name

You can replace name with whatever value you want, either with your name or names separated with commas

You will get access to the following page where you will get a greeting with the names you entered.

Every time you reload the page you will get a random phrase from the famous Spanish streamer Ibai Llanos.

These messages are loaded from a Firebase database.


infinite-ridge-77551.herokuapp.com/Apps/hello?value=Nicolas
Welcome to ibai random sentences generator
Hello Nicolas
Every time you regenerate the page you will get a random phrase from Ibai Llanos
Phrase: "Ya me han puesto una mansión, voy con bata de señor"


infinite-ridge-77551.herokuapp.com/Apps/hello?value=Javier,Esteban,Carlos,Pablo
Welcome to ibai random sentences generator
Hello Javier ,Esteban ,Carlos ,Pablo
Every time you regenerate the page you will get a random phrase from Ibai Llanos
Phrase: "Me he llevado una decepción muy seria: estaba en YouTube y he leído un comentario que decía: 'Ibai, casteas muy bien, pero estás gordito"

6 References

[1] geeksforgeeks, *Functional interfaces in java*, `https://www.geeksforgeeks.org/functional-interfaces-java/`, Accessed on 2021-02-11.

[2] Google, *Firebase realtime database*, https://firebase.google.com/docs/database, Accessed on 2021-02-11.