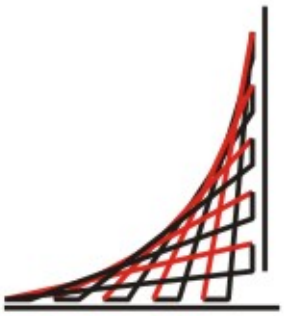

ESCUELA COLOMBIANA DE INGENIERÍA JULIO GARAVITO

AREP

Arquitectura Empresarial

Documentación Arquitectura: Laboratorio spark Frameork server

Authors:
Andrés Felipe Marcelo Rubiano

Escuela Colombiana de Ingeniería Julio Garavito
Septiembre 2020

Índice

1. Introduction

La transformación digital en el mundo ha impulsado el crecimiento del uso de servicios cloud en las organizaciones, esto convierte las necesidades de computación sobre la nube en una oportunidad vital para publicar los servicios que se necesiten de manera rápida, práctica y segura. El objetivo de este informe es el de definir arquitecturas y presentar resultados acerca de una implementación sobre la nube de un servidor web que emplee las carácteristicas funcionales de un framework que muchas empresas y desarrolladores estan usando actualmente en sus puestas a producción conocido cuyo nombre es Spark.Se van a presentar los diseños arquitecturales del servidor y se va a explicar detalladamente el funcionamiento de cada uno.

2. Arquitectura del servidor

La arquitectura del servidor se presenta a cotinuación:

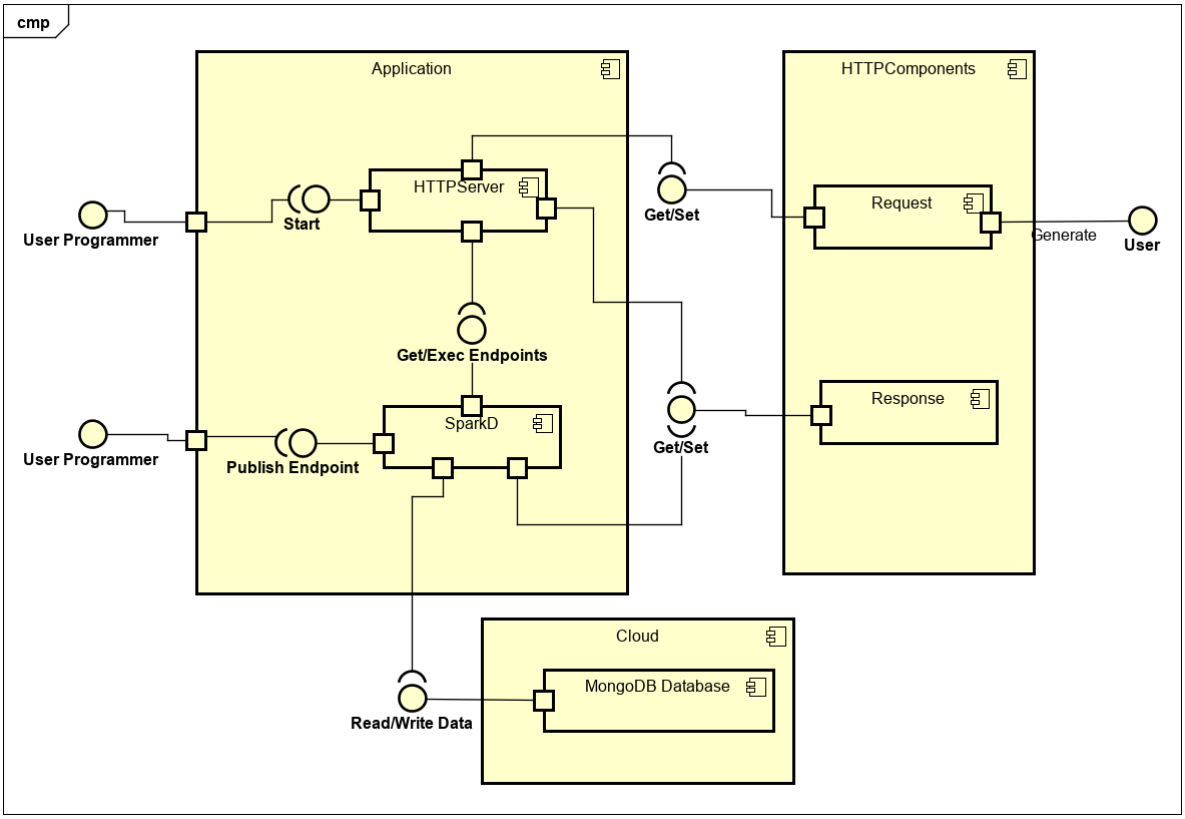

Figura 1: Arquitectura implementación Servidor

Para entender el diagrama es necesario explicar cuales son las funciones de cada componente y que

servicios se ofrecen y se consumen.

- **HTTPServer:** Este componente se encargará de crear y manipular la infraestructura del servidor, es decir, creará y escuchara los sockets de comunicacion entre clientes y servidor, también recibirá las peticiones del cliente: Pueden ser archivos estáticos (archivos con extensiones html,js,png,jpg,mp4,etc) y se encargará de responder con el archivo o con la implementación del endpoint solicitado.
- **SparkD:** Este componente es el que se encarga de definir los metodos disponibles (GET y POST) tomar las implementaciones del usuario (Por medio de lambda Functions) almacenarlas y mantenerlas disponibles para que cuando se soliciten, se ejecuten y se genere una respuesta con el resultado de la ejecución.Cabe resaltar que este componente esta conectado con la base de datos almacenada en el cloud debido a que si se implementa una conexión con la base de datos, la ejecución de esta misma estará implementada dentro de SparkD.
- **Request y Response:** Su función es la de contener la información de entrada y salida de cada una de las peticiones.
- **MongoDB Databases:** Este componente es el encargado de mantener la informacion persistente en la nube por medio de un cluster de bases de datos MongoDB llamado ATLAS.

3. Flujo De la aplicación

Antes que nada hay que definir los 2 tipos de usuarios que van a interactuar con el servidor: el **usuario programador**, aquel que define los endpoints y el **usuario normal** que es aquel que hace requests al servidor.
El flujo inicia cuando el usuario programador usa **sparkD** para almacenar los endpoints (GET y POST) y las definiciones de lo que se va a ejecutar usando este se consulte, a su vez el servidor inicia su ejecución y establece la comunicación con el cliente por merdio de sockets, este se queda esperando hasta que reciba una request para procesar. Cuando la request es recibida,el servidor busca si es un **endpoint** o si es un **archivo estático.**
Si es un endpoint, el servidor hace un llamado a **sparkD** para generar una respuesta(Response) con las funciones definidas por el usuario programador.En cambio, si es un archivo estático, el servidor procede a buscar el archivo en memoria y si lo encuentra, lo transforma en un arreglo de bytes y le envía el resultado al usuario por medio de un **PrintStream**.Cabe resaltar que antes de enviar la respuesta se generan los respectivos headers de acuerdo al tipo de respuesta (MimeType).

4. Pruebas

A continuación se presenta evidencia del funcionamiento del servidor desplegado en Heroku:

4.1. petición GET a un archivo estático

Primero que todo haremos una petición de tipo GET al archivo estático index.html que contiene varios recursos mp4, js, html, css, etc.(Figura 2)

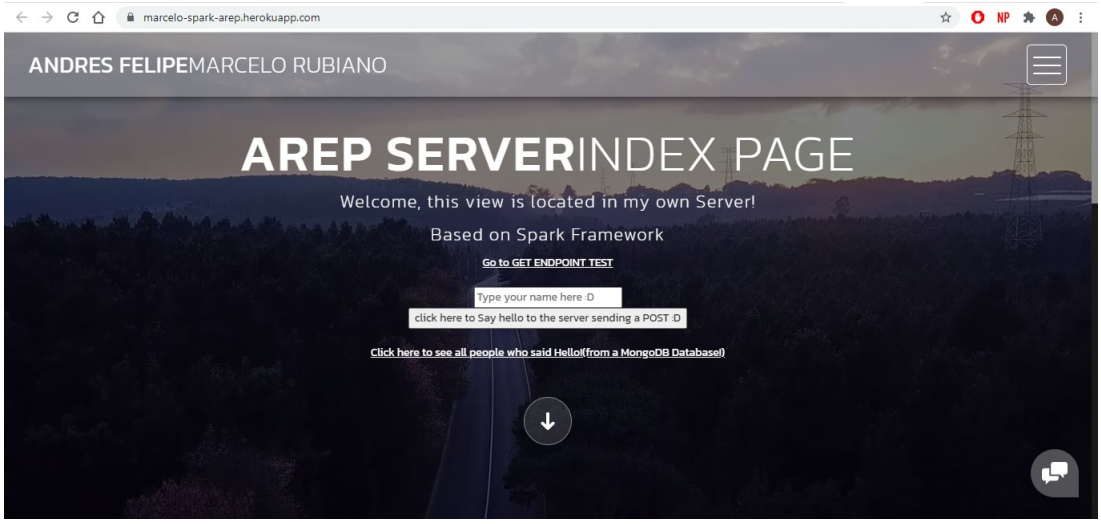

Figura 2: GET A /index.html

4.2. Peticion POST al endpoint de base de datos

Otra funcionalidad (Endpoint) disponible en el servidor es una POST request para insertar una entrada (Nombre) a la base de datos MongoDB. Si la petición es exitosa, se hará un alert al usuario que informe que se agrego a la base de datos y dará un número aleatorio de 0 a 10. (Figura 3)

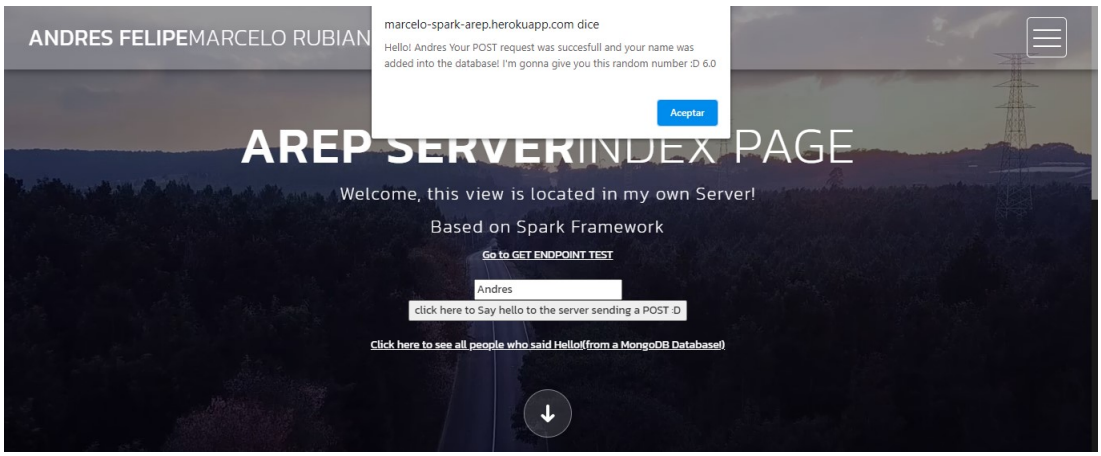

Figura 3: POST request al endpoint /testPost

4.3. Petición GET al endpoint de base de datos

Y finalmente haremos un GET request a la base de datos MongoDB para mostrar los datos almacenados anteriormente con los nombres ingresados y la fecha de ingreso.(Figura 4)

Figura 4: GET request al endpoint /testDB (Comunicación con Base de datos)

5. Conclusión

Esta actividad permitió conocer la arquitectura básica del framework de spark, su funcionamiento, comportamiento y su metodología.Además del comportamiento de los encabezados HTTP, su estructura, envío y recepción.