Taller Clientes y Servicios

Alejandro Toro Daza

Profesor:
Luis Daniel Benavides Navarro

Arquitecturas Empresariales

Escuela Colombiana de Ingeniería Julio Garavito
4 de Febrero del 2021
Bogotá D.C.

TABLA DE CONTENIDO

1. INTRODUCCIÓN

En el Taller de Clientes y Servicios se realizarán diferentes retos en los que se explorarán los conceptos de esquemas de nombres y de clientes y servicios. Adicionalmente, el taller también explorará la arquitectura de las aplicaciones distribuidas sobre internet, para así ver el funcionamiento detallado de una aplicación web capaz de recibir múltiples solicitudes no concurrentes, creando un framework manualmente muy similar al funcionamiento de Spark que le permite publicar servicios web get y post para poder acceder a recursos estáticos como páginas web, javascripts, imágenes CSSs, entre otras cosas, desplegado usando un servidor web llamado Heroku para poder acceder a ella de manera totalmente remota. Para verificar el funcionamiento de cada uno de los requisitos, se realizó una simulación de la página de la Registraduría Nacional del Estado Civil, en la cual el usuario tiene una interfaz, y al digitar la URL en la que se encuentran las bases de datos, puede observar características como nombres y apellidos de ciudadanos junto con su dirección, cada una de ellas almacenada en una base de datos.

2. OBJETIVOS

- Elaborar un programa creando un framework manualmente que permita publicar los servicios web get y post.
- Implementar un servidor web que soporte múltiples solicitudes seguidas (no concurrentes). El servidor debe retornar todos los archivos solicitados, incluyendo páginas html e imágenes.
- Construir un sitio web con javascript para probar el servidor, desplegando la solución en Heroku sin usar frameworks web como Spark o Spring, usando solo Java y las librerías para manejo de la red.
- Usando un servidor y Java sin usar frameworks web como Spark o Spring, escribir un framework similar a Spark que permita publicar servicios web "get" con funciones lambda y permita acceder a recursos estáticos como páginas, javascripts, imágenes, y CSSs.

3. MARCO TEÓRICO

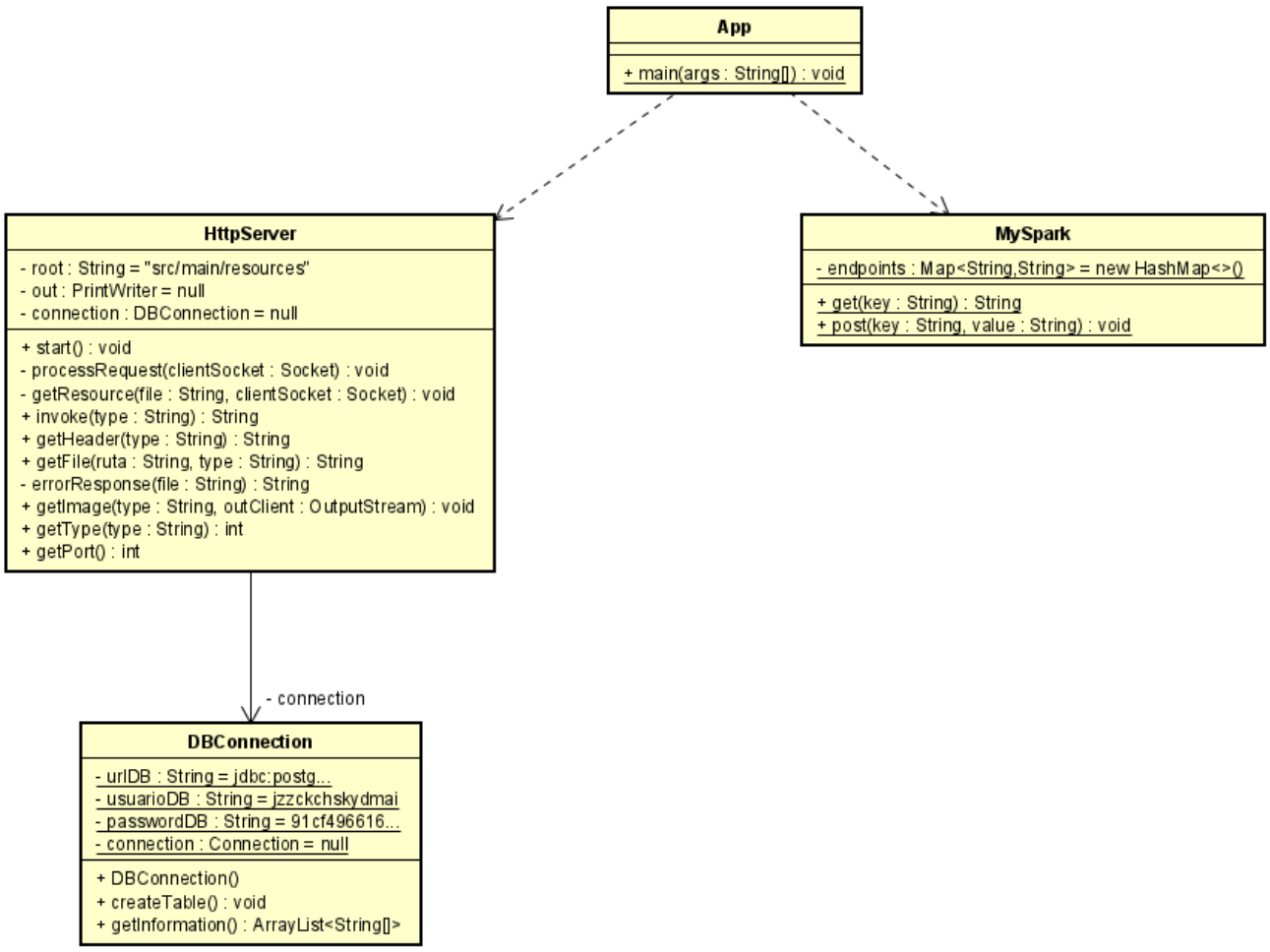
- **JavaScript:** Lenguaje de programación o de secuencias de comandos que te permite implementar funciones complejas en páginas web, cada vez que una página web hace algo más que sentarse allí y mostrar información estática para que la veas, muestra oportunas actualizaciones de contenido, mapas interactivos, animación de Gráficos 2D/3D, desplazamiento de máquinas reproductoras de vídeo, etc., puedes apostar que probablemente JavaScript está involucrado. Es la tercera capa del pastel de las tecnologías web estándar, dos de las cuales (HTML y CSS) hemos cubierto con mucho más detalle en otras partes del Área de aprendizaje.

- **CSS:** Lenguaje que define la apariencia de un documento escrito en un lenguaje de marcado (por ejemplo, HTML).
  Así, a los elementos de la página web creados con HTML se les dará la apariencia que se desee utilizando CSS: colores, espacios entre elementos, tipos de letra, ... separando de esta forma la estructura de la presentación.
  Esta separación entre la estructura y la presentación es muy importante, ya que permite que sólo cambiando los CSS se modifique completamente el aspecto de una página web. Esto posibilita, entre otras cosas, que los usuarios puedan usar hojas de estilo personalizadas (como hojas de estilo de alto contraste o de accesibilidad).
- **HTML:** Lenguaje de marcado de hipertexto, y le permite al usuario crear y estructurar secciones, párrafos, encabezados, enlaces y elementos de cita en bloque (blockquotes) para páginas web y aplicaciones.
  HTML no es un lenguaje de programación, lo que significa que no tiene la capacidad de crear una funcionalidad dinámica. En cambio, hace posible organizar y formatear documentos, de manera similar a Microsoft Word.
- **Framework:** Conjuntos de componentes de software que se utilizan para crear la estructura de un software o una aplicación. Un framework puede verse como una caja de herramientas en la que el desarrollador busca los componentes necesarios. Este marco proporciona una estructura general para facilitar la labor de desarrollo.
  Los frameworks funcionan mediante el lenguaje de programación y desarrollan todo tipo de soporte: sitios web, juegos, aplicaciones móviles, etc. Sin embargo, también puede crear su propio marco de software.

4. DISEÑO Y CONSTRUCCIÓN

Para la creación del proyecto, se dispuso de cuatro clases en total. La Clase **App** que es la clase encargada de crear el servidor Http Server y crear un post con la URL de Heroku junto con el recurso **/acercade** para mostrar en una página web una descripción de la Registraduría Nacional del Estado Civil, mostrando así el correcto funcionamiento del post usando la clase **MySpark**. En la clase **MySpark** se realiza una pequeña implementación muy similar al funcionamiento del framework Spark, el cual se ofrecen métodos get y post de los endpoints que se encuentran disponibles, los cuales se encuentran almacenados en un Hashmap. La clase **DBConnection** se encarga primero de realizar la conexión de la aplicación con la base de datos, en la cual en la misma clase se crea la tabla en donde se van a realizar la consulta de los datos, los cuales han sido manualmente ingresados en una herramienta llamada **DBeaver**, en la cual se gestionó toda la base de datos para su correcto funcionamiento. La clase **HttpServer** es una implementación propia de un servidor el cual es capaz de recibir peticiones HTTP, y asimismo retornar el recurso pedido.

Diagrama de Clases:



4.1. PRUEBAS

Pruebas en Maven

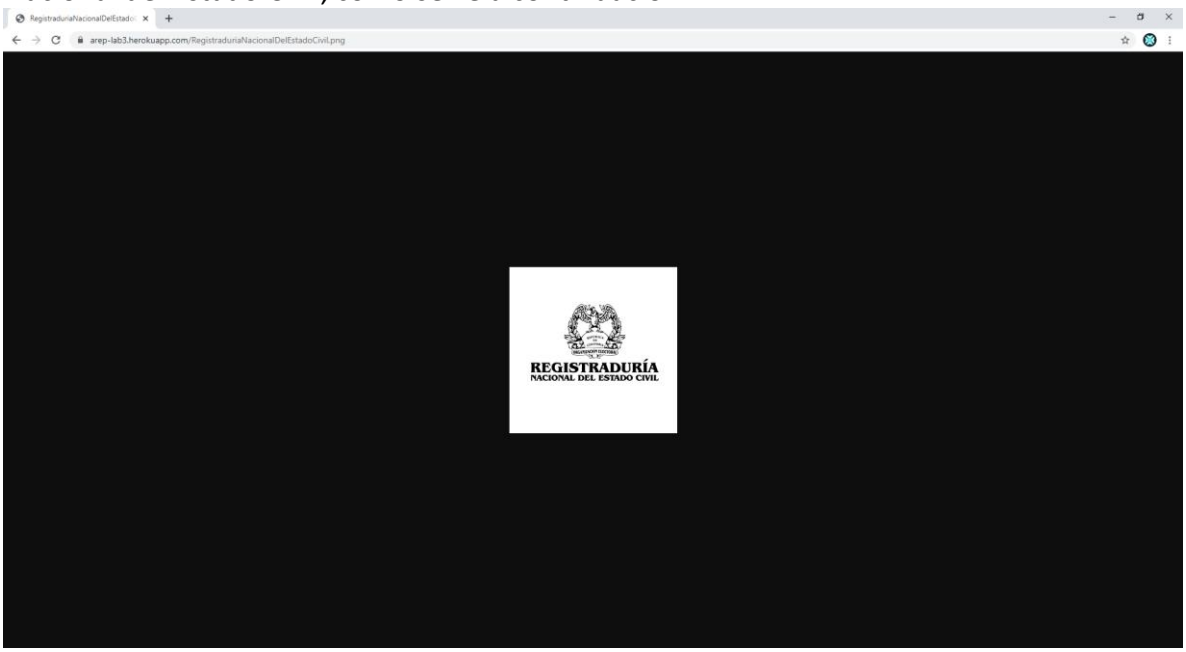
Para verificar que todo el código está funcionando con total normalidad, realizando los cálculos apropiadamente y obteniéndolos de una lista enlazada, se crearon varias pruebas unitarias utilizando **Junit**, herramienta que nos provee el IDE que se utilizó para la realización de todo el código fuente, Eclipse. Para estas pruebas unitarias, también se utilizó Maven, en el cual ejecutando los comandos como **mvn test** también se probó que el código funcionara con total normalidad, demostrando así total funcionamiento del código fuente retornando los resultados esperados.

```
-------------------------------------------------------
 T E S T S
-------------------------------------------------------
Running edu.escuelaing.arep.app.AppTest
Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.029 sec

Results :

Tests run: 1, Failures: 0, Errors: 0, Skipped: 0

[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  1.628 s
[INFO] Finished at: 2021-02-03T15:43:26-05:00
[INFO] ------------------------------------------------------------------------
```

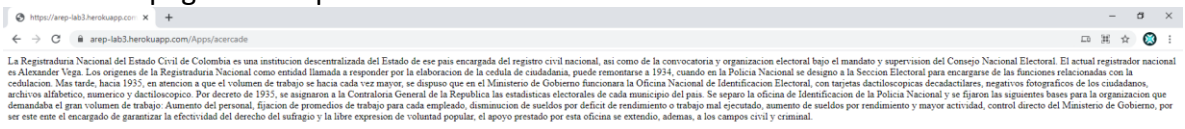
Pruebas en Heroku

Luego de realizar la respectiva configuración del código, con las dependencias añadidas en el archivo **pom.xml** y el **Procfile**, al acceder a la URL de la aplicación desplegada en Heroku (https://arep-lab3.herokuapp.com/) vemos que la aplicación muestra la interfaz de usuario de la siguiente forma, demostrando que la aplicación dispone de una interfaz desplegada en Heroku, mostrando el **index.js** un mensaje de bienvenida, en la que el usuario puede omitirla realizando clic en **Aceptar**, junto con el **index.html** mostrando la página web con la interfaz de la Registraduría Nacional del Estado Civil como se ve a continuación.

Para acceder a los diferentes recursos, como consultar la imagen implementada y desplegada en la página web, falta agregarle a la URL de Heroku, el recurso **/RegistraduriaNacionalDelEstadoCivil.png**, el cual despliega el escudo de la Registraduría Nacional del Estado Civil, como se ve a continuación.



Para acceder al siguiente recurso, el cual es consultar información relacionada de la Registraduría Nacional del Estado Civil, como su fundación, y a qué se dedica en Colombia, se ingresa el recurso **/Apps/acercade** al final de la URL de Heroku. A continuación, se muestra la página web que contiene dicha información.



Por último, para ver el contenido de la página web que tiene las bases de datos de los ciudadanos de Colombia, se añade el último recurso **/Apps/informationDB** al final de la URL de Heroku, mostrando así el contenido de la base de datos a la cual está asociada la aplicación, ordenándola por Nombres y Apellidos, y Dirección de cada uno de ellos, como se ve a continuación.

5. ARQUITECTURA

Al realizar el respectivo despliegue en Heroku de la aplicación, y añadiendo la integración continua con la herramienta CircleCI, vemos que la arquitectura del programa es cliente-servidor, principalmente por lo que la aplicación se encuentra en un servidor de Heroku, la cual puede ser accedida desde cualquier navegador, enviando peticiones HTTP al servidor, el cual las recibe y las procesa para posteriormente por medio de un controlador ofrecido por un framework implementado de forma manual que opera de manera muy similar a Spark, el cual realiza los respectivos **get** y **post** para su respectivo procesamiento. La estructura de la arquitectura luego de realizar el despliegue y la integración continua puede ser descrita de la siguiente forma:

- La aplicación primero utiliza una estructura de datos que funciona como memoria, que es la lista enlazada, que se encarga de guardar los datos ingresados por el cliente, mediante el método **add** implementado en el código fuente, junto con el **read** que se encarga de leer los datos y realizar los cálculos.
- Luego el servidor maneja los request, también implementados en el código para manejar abstractamente la comunicación entre el cliente y el servidor, mediante el protocolo HTTP para el envío y el recibimiento de información.
- Después como el código está implementado en Java, procesa los datos utilizando este lenguaje, para así poder retornar el resultado correcto pedido por el cliente, para así poder realizar los cálculos, y mediante el mismo protocolo HTTP devolverle el resultado al cliente por medio de una interfaz creada en HTML (otro lenguaje).
- Finalmente, la integración continua se maneja junto con Heroku, ya que dentro de la configuración de Heroku se optó por desplegar la aplicación si y solo si el código pasa las pruebas en CircleCI (herramienta utilizada para realizar la integración continua) para así llevar un mejor control de calidad del código, para que se compile apropiadamente para que el cliente o usuario no presente inconvenientes desplegando la aplicación.

6. CONCLUSIONES

- En el laboratorio se utilizó un servidor y java sin utilizar ningún tipo de framework, asegurando que se cumplieran con los requisitos de implementación del código, para así implementar un framework de cero similar a Spark, el cual permitió publicar servicios web "get" con funciones lambda y le permitió acceder a recursos estáticos como la página de la Registraduría Nacional del Estado Civil, en la cual se podían acceder a imágenes como el escudo de la misma, interfaz de usuario e información histórica de la Registraduría Nacional del Estado Civil.
- Posteriormente también se creó una base de datos completamente de cero utilizando la herramienta DBeaver, utilizando SQL para la creación de tablas y para la respectiva población de estas. Desde el servidor fue posible realizar la respectiva conexión a las bases de datos implementado en la clase **DBConnection**, recogiendo los valores almacenados en la tabla **information** de las bases de datos, para su posterior despliegue en la aplicación web desde Heroku, mostrando total funcionalidad entre las bases de datos y la aplicación web.
- Asimismo, el despliegue en Heroku permitió poder ejecutar la aplicación de la Registraduría Nacional del Estado Civil, probando el funcionamiento de la aplicación web en nube y compilando el código en cualquier ordenador, utilizando el protocolo HTTP para la comunicación cliente-servidor para asegurar una total ejecución del código de manera remota.
- Por otro lado, se implementó integración continua en todo el código fuente para llevar un control de calidad del código, para asegurar que el código esté funcionando totalmente sin ningún problema tanto de compilación como de los resultados retornados después de realizar las respectivas operaciones, y así poder desplegar la aplicación sin presentar ningún tipo de errores.

7. REFERENCIAS

- Vargas, Julián. “Aprende Sobre Desarrollo Web.” Aprende Sobre Desarrollo Web | MDN, 23 Junio 2015, developer.mozilla.org/es/docs/Learn/JavaScript/First_steps/Qu%C3%A9_es_JavaScript.
- Delgadillo, Daniel. “CSS, ¿Qué Es?” *Arume*, 1 Abr. 2019, www.arumeinformatica.es/blog/css/.
- Bernal, Gustavo. “¿Qué Es HTML? Explicación De Los Fundamentos.” Tutoriales Hostinger, 10 Dic. 2020, www.hostinger.es/tutoriales/que-es-html/.
- Sánchez, Sebastián. “¿Qué Es Un Framework? - Wild Code School.” Www.wildcodeschool.com, 13 Junio 2019, www.wildcodeschool.com/es-ES/blog/que-es-un-framework.