Taller Heroku

Calderón Ortega Andrés Mateo

Arquitecturas Empresariales
Ingeniería de sistemas, Escuela Colombiana de Ingeniería Julio Garavito, Bogota, Colombia
4 de febrero de 2021

Resumen: Se realizará la explicación del funcionamiento de una aplicación web que realiza 2 cálculos estadísticos como es la media y la desviación estándar; el desarrollo guiado a una programación Orientada a Objetos, su arquitectura básica y la estructura de datos utilizada para el almacenamiento de los datos durante el procesamiento de los resultados, ademas se explicara la forma en la que se manejo el framework Spark y la libreria axios para el manejo de peticiones REST.

1. Introducción

El problema tratado en este proyecto se basa en un problema estadístico; donde se debe calcular la media y la desviación estándar de un conjunto n de números que serán ingresados por el usuario vía web; para solucionar este problema en primera instancia se realizara una Arquitectura Orientada a Objetos para definir de manera ordenada cada una de las secciones que dividen el proyecto; inicialmente para esto se utiliza una implementación de una lista enlazada que es compatible con la API de Java como forma de almacenamiento de datos, para la lectura de los datos utilizaremos un formulario creado en ***HTML*** el cual se comunicara con un JavaScript llamado ***app.js***, dentro de este JavaScript se encuentra implementado el uso de la librería axios de JavaScript para el manejo de peticiones REST, estas peticiones se realizan desde el aplicativo funcional diseñado en Java que hace uso del framework Spark. El usuario deberá ingresar los datos de la siguiente manera:

```
15.0 69.9 6.5 22.4 28.4 65.9 ... n
```

La forma de lectura web para los datos decimales se realiza con un '.' y la diferenciación de datos se entiende por un espacio en blanco entre ellos, el usuario deberá hacer click en un botón que dice 'Enviar' para que sus datos sean enviados y calculados, y finalmente por pantalla se muestra la respuesta solicitada.

2. Arquitectura

2.1. Diseño

2.1.1. Implementacion Lista Enlazada

Para la realización de esta clase de tipo $List$, se decidió utilizar un nodo el cual contendrá el valor representativo de sí mismo y además un apuntador en memoria al siguiente nodo en la lista, esto para poder realizar un recorrido completo de la lista a partir de su cabeza, que será almacenada en la Clase $LinkedListImpl$.

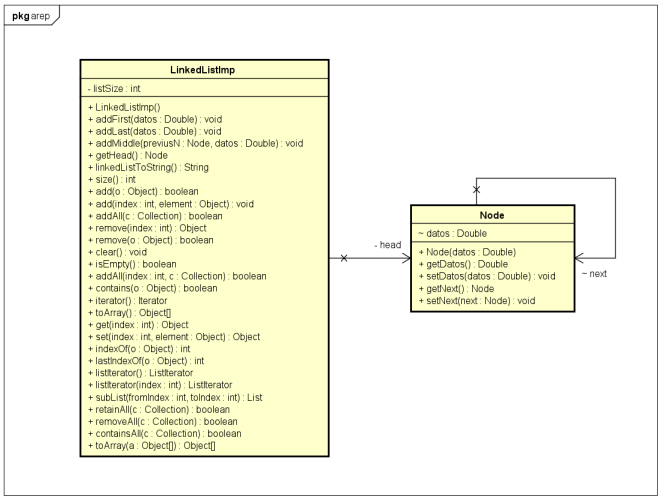

Figura 1: Modelo de clases 1

Podremos observar que la clase Node, contiene un atributo de tipo Double para almacenar en sí mismo el valor significativo que este tiene, adicionalmente cuenta con una referencia a un nodo siguiente que inicialmente es marcado como

”Null”significando esto que no hay más datos en la lista, se tienen también los métodos para acceso a los mismos. En la clase LinkedListImp tendremos disponibles todos y cada uno de los métodos que posee una lista exceptuando algunos que no fueron considerados necesarios para la implementación, la lógica detrás de esta implementación se basa en el uso de nodos como referencias en la lista, para recorrer la lista para realizar cualquier tipo de acción (Búsqueda, Eliminación, Modificación, Adicionar) se realiza un ciclo de tipo while, que tiene fin cuando se llega a un nodo que no tiene otro como referencia, y esto querrá decir que es el final de la lista, puesto que todos y cada uno de los nodos deben tener una referencia empezando por la cabeza de la lista.

2.2. Diagrama General

Para la solución del problema estadístico utilizamos la clase llamada CalculadoraEstadistica la cual recibirá desde la clase SparkWebApp un string que contendrá los datos correspondientes a ser evaluados, esta lista será convertida a Nodos y estos serán agregados a la lista enlazada en espera de la solicitud de evaluación de los datos, posteriormente desde la clase SparkWebApp se hara la solicitud de evaluación de los datos.

En la siguiente imagen podremos ver la arquitectura del diseño utilizado para la implementacion:

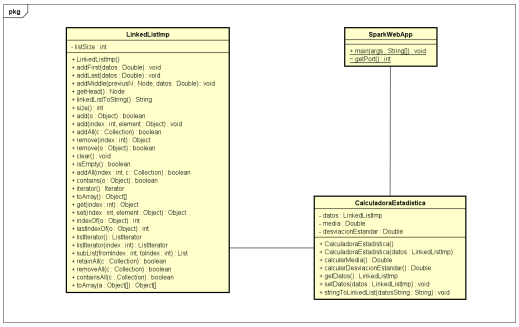

Figura 2: Modelo de clases 2

Para los dos problemas estadisticos se consideraron las siguientes ecuaciones:

Media (Promedio) :

$$x_{avg} = \frac{\sum_{i=1}^{n} x_i}{n} \tag{1}$$

Desviación Estándar:

$$\sigma = \sqrt{\frac{\sum_{i=1}^{n} (x_i - x_{avg})^2}{n-1}} \tag{2}$$

Luego de realizar la lectura correspondiente y pertinente se utiliza la clase CalculadoraEstadistica que por medio de operación y haciendo uso de la lista enlazada anteriormente mostrada resuelve los problemas de la media y la desviación estándar.

Para la conexión Web utilizamos el framework Spark que realizara las solicitudes pertinentes para solicitar y devolver los datos, las peticiones REST son manejad haciendo uso de la librería axios de JavaScript; la cual creara un Objeto Json con los datos y los enviara para ser evaluados, luego de ser evaluados recibe otro objeto JSON que es retornado desde la clase SparkWebApp que para la creación de este objeto utiliza la dependencia de google.json de maven, y finalmente lee este objeto y escribe los resultados en el HTML correspondiente.

3. Pruebas

Para asegurar un buen funcionamiento se decidió hacer pruebas de unidad por medio del uso de JUnit, se realizaron un total de 6 pruebas, con las cuales se cubre el correcto funcionamiento de los cálculos estadísticos de la clase ’CalculadoraEstadistica’, los datos fueron sacados de un documento de tipo ’estudiantil’ donde se realizaban estos cálculos.

Para correr las pruebas se realiza una ejecución por medio de consola, haciendo uso de maven, para esto utilizamos el siguiente comando:

```
$ mvn test
```

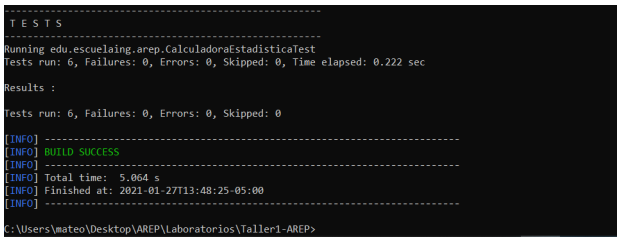

Figura 3: Resultados de pruebas Unitarias

Como podemos observar las pruebas pasan de manera satisfactoria, dando así un resultado positivo diciéndonos así que el funcionamiento del programa es correcto.

4. Conclusiones

Se adquirieron conocimientos en el uso de JavaScript para los manejos de peticiones REST y creación de objetos de tipo JSON, además se evidenció el gran beneficio de tecnologías como el framework Spark para hacer la integración de las capas de FrontEnd y BackEnd, además de la gran adaptabilidad que tienen las tecnologías HTML y JavaScript.
Se pudo observar el gran beneficio que trae usar herramientas como maven al momento de la creacion de arquitecturas nuevas; gracias a su gran facilidad de creacion de ambientes de trabajo y en la realizacion de pruebas unitarias, ademas de su gran manejo de dependencias, adicionalmente se adquirio un mayor conocimiento en el uso de git, HerokuCli y en la construccion de arquitecuturas.

Referencias

[1] Sparkjava.com. 2021. Spark Framework: An expressive web framework for Kotlin and Java. [online] Available at: https://sparkjava.com/ [Accessed 4 Febrero 2021].

[2] Palacios, D., 2021. Solicitudes HTTP con Axios. [online] Styde.net. Available at: https://styde.net/solicitudes-http-con-axios/ [Accessed 4 Febrero 2021].

[3] Developer.mozilla.org. 2021. Trabajando con JSON - Aprende sobre desarrollo web MDN. [online] Available at: https://developer.mozilla.org/es/docs/Learn/JavaScript/Objects/JSON [Accessed 4 Febrero 2021].

[4] java?, H. and Myasnychenko, D., 2021. How to get data from POST-form with spark java?. [online] Stack Overflow. Available at: https://stackoverflow.com/questions/33779033/how-to-get-data-from-post-form-with-spark-java [Accessed 4 February 2021].

[5] Daniel Benavides. 2021. Clase de laboratorio. [29 Enero 2021].

[6] David Vargas. 2021. Explicación de JavaScript y axios. [2 Enero 2021]