Investigación

Grafos, dispersión y otras estructuras de datos afines

Profesor:

Christian Sibaja

Grupo

Conformado por:

Luis Fernando Solano Montoya

Johan Gerardo Araya González

Douglas Daniel Conejo Cascante

Stephanie Quiros Hernandez

Fecha de entrega: 24 de Abril de 2023

Tabla de contenidos

Introducción

El presente documento tiene como propósito explicar la implementación trabajada para el proyecto de “Rutas terrestres internacionales”. Este proyecto lo que trata de resolver es brindar una forma automatizada para encontrar un país en todo el continente americano. Además la aplicación, ayuda a ver los límites terrestres que tienen todos los países de América, por lo que también se tomaron en cuenta las islas y zonas de América que no tienen algún país cercano. Finalmente, la aplicación ayuda a encontrar la distancia mínima y máxima en kilómetros entre 2 países.

Para el desarrollo de esta aplicación se trabajó con conceptos como grafos estructura de datos utilizadas en el mercado actual para crear mapas ya que los grafos permiten estudiar las interrelaciones entre unidades que interactúan unas con otras. En este caso el grafo ayudó a solucionar el problema de la relación entre todos los países del continente americano.

Además se utilizó tablas de dispersión cerradas para guardar cada uno de los países del continente, y de esta manera encontrarlos de la manera más óptima y rápida en la aplicación, mejorando el rendimiento de esta cuando el usuario hace consultas.

Con la parte de algoritmos, se utilizó el algoritmo de dijkstra para resolver el problema de encontrar las distancias máximas y mínimas entre 2 países y el algoritmo de Haversine para encontrar cual es la distancia entre 2 pares de coordenadas.

Detalles de la implementación

Tecnología utilizada

JavaScript: Es un lenguaje de programación de alto nivel, interpretado y orientado a objetos, que se utiliza principalmente en el desarrollo web para crear interacciones dinámicas y animaciones en las páginas web.

Hoy en día es uno de los lenguajes de programación más utilizados en el mundo, tanto en el desarrollo web como en el desarrollo de aplicaciones para dispositivos móviles, servidores y otros ámbitos.

HTML: (HyperText Markup Language) es el lenguaje de marcado estándar utilizado para crear páginas web y otros documentos en la World Wide Web. Es el lenguaje base de la mayoría de los sitios web y proporciona la estructura y el contenido de las páginas web.

CSS: (Cascading Style Sheets) es un lenguaje de estilo utilizado para describir la presentación y el diseño de páginas web escritas en HTML u otros lenguajes de marcado web. CSS permite separar la presentación visual de una página web del contenido estructural, lo que proporciona una mayor flexibilidad y control sobre el diseño y la apariencia de una página web.

Webpack: Es una herramienta de código abierto que se utiliza en el desarrollo web para empacar y optimizar el código fuente de una aplicación web. Webpack permite a los desarrolladores web gestionar de manera eficiente las dependencias de sus proyectos, automatizar tareas de compilación, empaquetar y minificar archivos JavaScript, CSS, imágenes y otros recursos en un único archivo o conjunto de archivos.

Azure: Es una plataforma en la nube de Microsoft que proporciona servicios y herramientas para el desarrollo, implementación y gestión de aplicaciones en la nube. Azure permite a los usuarios alojar aplicaciones, almacenar datos y acceder a una amplia gama de servicios en la nube, como la inteligencia artificial.

REST Countries API: Proporciona información sobre países de todo el mundo. Esta API se basa en el protocolo REST y se puede acceder a ella a través de solicitudes HTTP.

Google Maps API: Permite a los desarrolladores integrar mapas interactivos y servicios de geolocalización en sus aplicaciones web y móviles.

Hash

Clase implementada en el proyecto, para mejorar las búsquedas de países de una manera más eficiente y rápida. Hay 2 tipos de Hashing, el abierto, que introduce los datos en el mismo índice cuando estos coinciden, por medio de una lista enlazada, y el cerrado, que si ocurre una colisión, se agrega en siguiente índice vacío. Hoy en día las funciones hash son comunes en términos cuando hay una gran variedad de datos, un gran ejemplo de ellas son en el Big Data.

En el código del proyecto, se utilizó un hash cerrado, esta clase contiene 2 atributos privados, un #lenght para saber cuántos valores serán los máximos en agregar en el hashtable, y un #table, que su función será almacenar los países.

Además se trabajó en la función #hash, que esta función realiza el hashing de los códigos de los países y para obtener un hash con menos colisiones se utiliza la función #getPrime, que obtiene el número primo más cercano de acuerdo a la cantidad de datos que se pueden insertar, dicha función, será llamada en la función de hash. Además, se trabajó la función get y set, para obtener y guardar los valores. Finalmente se desarrollaron funciones como getIndexByKey, getValueByIndex y getAll para traer los datos del hash table.

Graph

La clase Graph ayuda a relacionar cada país por medio de los arcos. Por ello se trabajó en el método getAdjacencyList este método se usa para obtener la lista de países con el que limita un país en específico**,** el cual recibe como parámetro un string que en este caso será el código del país, esté irá a buscar a la tabla Hash primeramente y verificará si el país está, sino enviará un mensaje de que no se encontró, luego, si el país está dentro de la tabla, se le envía el código recibido, y con esto se obtiene el index de dicho país para después recorrer ese espacio y encontrará todos aquellos paises que estan limitando con el país deseado. Si el país no posee adyacencias indicará un mensaje que no posee adyacencia, de lo contrario dará el nombre de los lugares que limitan con el país seleccionado.

Dijkstra

Para la implementación de este algoritmo se crea los siguientes arreglos:

- vistedNodes: permite almacenar todos los nodos visitados durante el recorrido del algoritmo.
- queue: arreglo que tiene la prioridad de como visitar los nodos partiendo desde el origen y continuando con los vecinos.

Además, se crearon los siguientes objetos:

- distances: contiene todas las distancias de todos los nodos visitados contra el origen.
- previousVertex: contiene todos los nodos cual es el nodo previo visitado de un país a otro, para saber cual es el país con la ruta más cercana.

Este algoritmo lo que ayuda es a sacar la distancia mínima entre 2 vértices de un grafo. Por lo que lo primero que se hace es insertar el nodo actual en los nodos visitados, se obtiene los vecinos de este nodo, y se saca cuanto es la distancia entre cada uno de los nodos para obtener cual es la ruta menor entre todos los vecinos. Ya que se obtiene cual es la ruta más corta entre todos los nodos, se inserta en la cola de prioridad para ser el siguiente en ser analizado.

Si ya se revisaron todos los nodos con el camino más corto puede suceder que el nodo destino no se ha visitado, por lo que si aún no ha sido, se obtiene el último elemento en la lista de prioridad para que se analice de nuevo todos los nodos hasta encontrar el destino. Si se vuelve a analizar el origen y ya no tiene nodos vecinos por visitar, es decir ya todos fueron revisados, significa que no hay ruta entre el origen y destino.

Para obtener la ruta máxima entre 2 vértices también se utilizó el algoritmo de dijkstra, pero con una pequeña modificación, lo que se hace es que el peso de cada nodo se hace negativo para que la distancia más grande ahora sea la menor ya que los números negativos entre más lejos del 0 son menores, y con el mismo algoritmo se logró obtener la distancia máxima entre 2 puntos.

Haversine

Es una fórmula utilizada para calcular la distancia entre dos puntos en una esfera, como la Tierra. Este algoritmo es particularmente útil en aplicaciones que necesitan calcular la distancia entre dos ubicaciones geográficas, como en sistemas de navegación y mapas en línea.

La fórmula Haversine utiliza la latitud y longitud de dos puntos para calcular la distancia entre ellos. La fórmula tiene en cuenta la curvatura de la esfera y se basa en la ley de los cosenos para calcular la distancia entre dos puntos en un círculo máximo de la esfera.

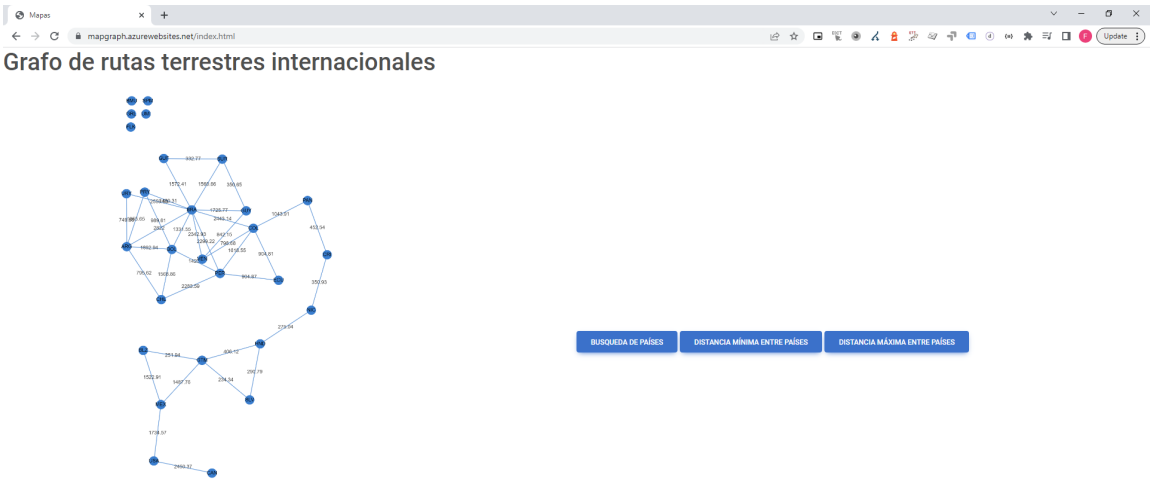
El algoritmo Haversine es ampliamente utilizado en el desarrollo de aplicaciones web y móviles, como en sistemas de seguimiento de flotas, servicios de entrega, servicios de viajes compartidos, y en aplicaciones que requieren cálculos de distancia precisa entre dos puntos geográficos.

Instrucciones de operación

En este apartados pretenden explicar cómo se debe de utilizar la aplicación y cuales son los valores esperados que va tener el usuario al interactuar con el sitio.

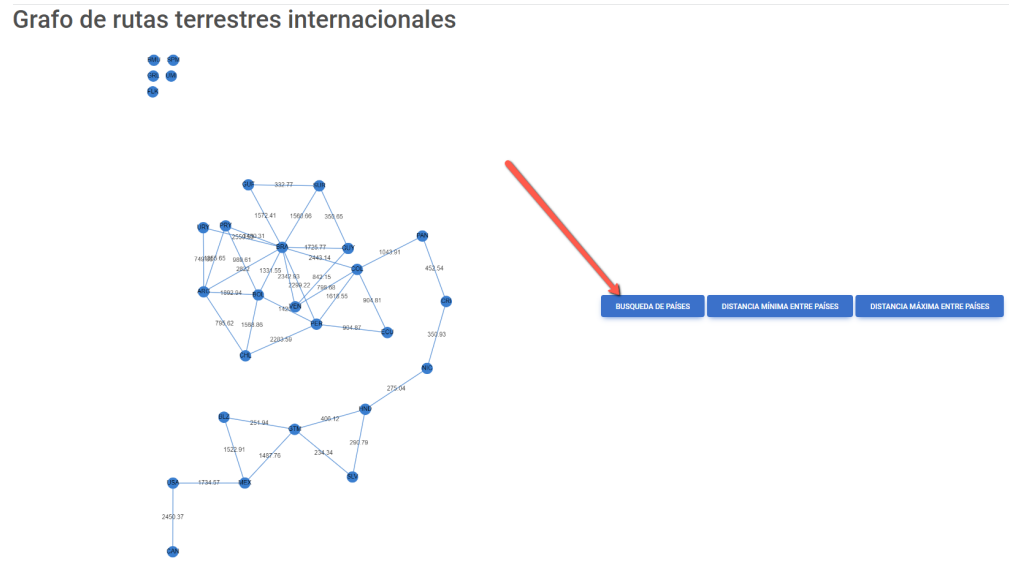
Ingresar a: https://mapgraph.azurewebsites.net/

Se muestra la página de inicio de la aplicación.

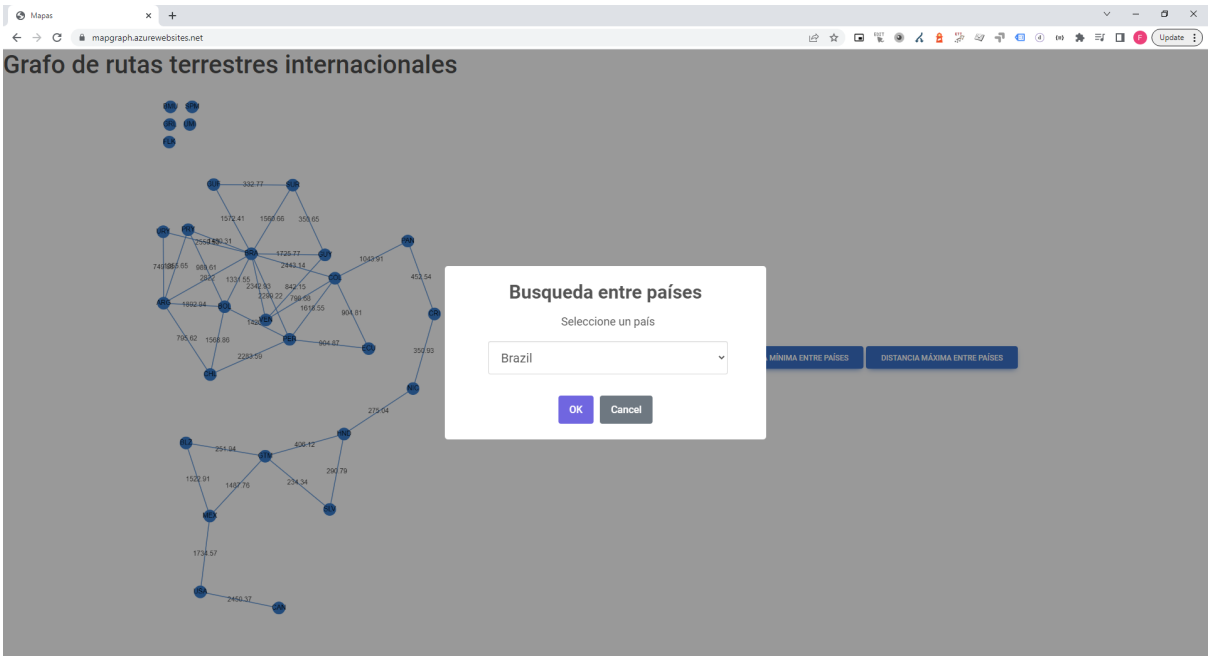


Búsqueda de países

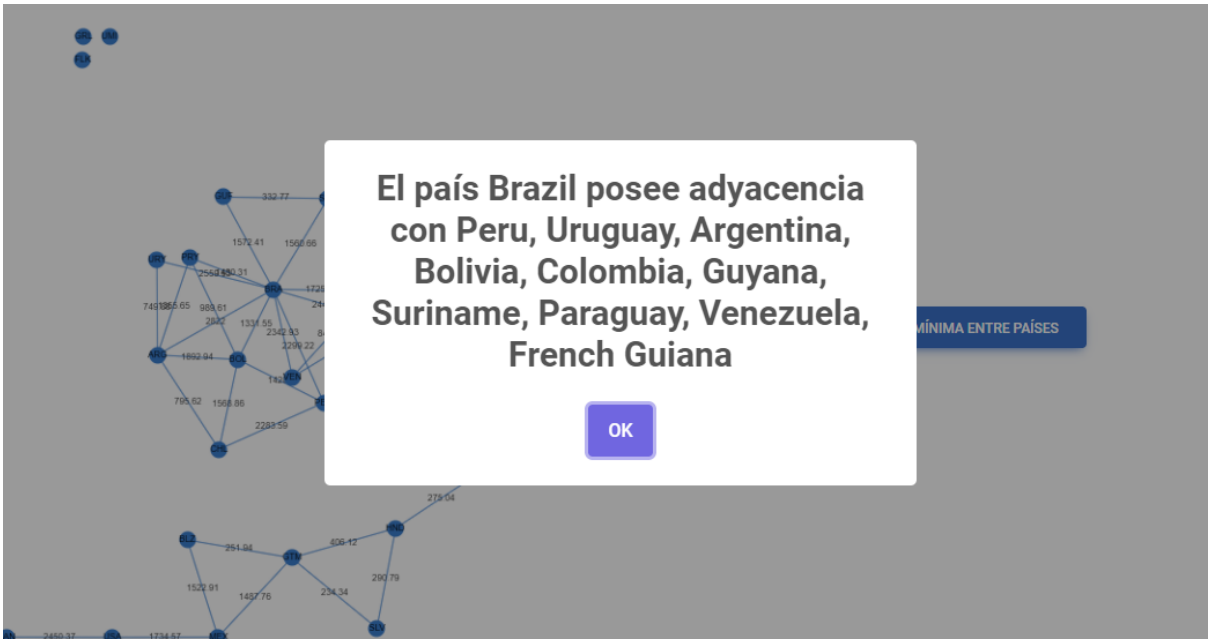
El usuario para buscar países, tendrá que realizar un click en el botón de “Búsqueda de países”.



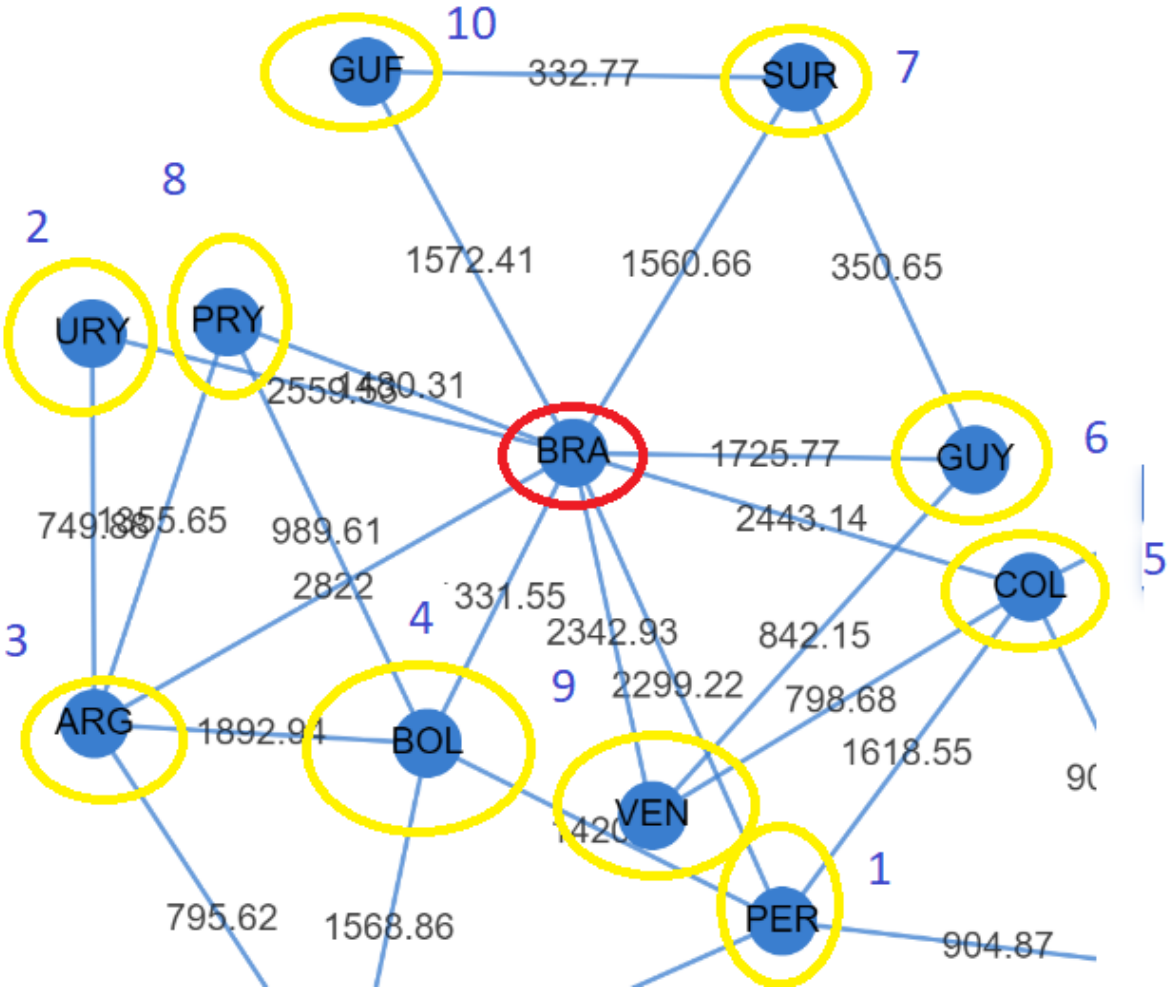
Luego se abrirá una ventana emergente para seleccionar el país. El usuario deberá seleccionar el país y hacer click en el botón de “OK”.



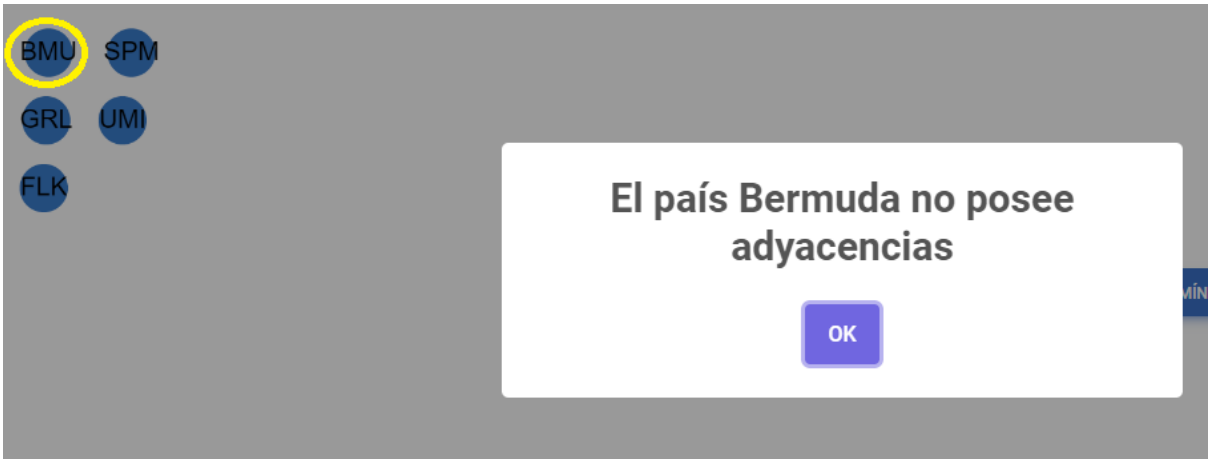
El sistema le mostrará cuáles son las adyacencias (límites) de ese país.



Se puede verificar el resultado con el grafo creado a la derecha.

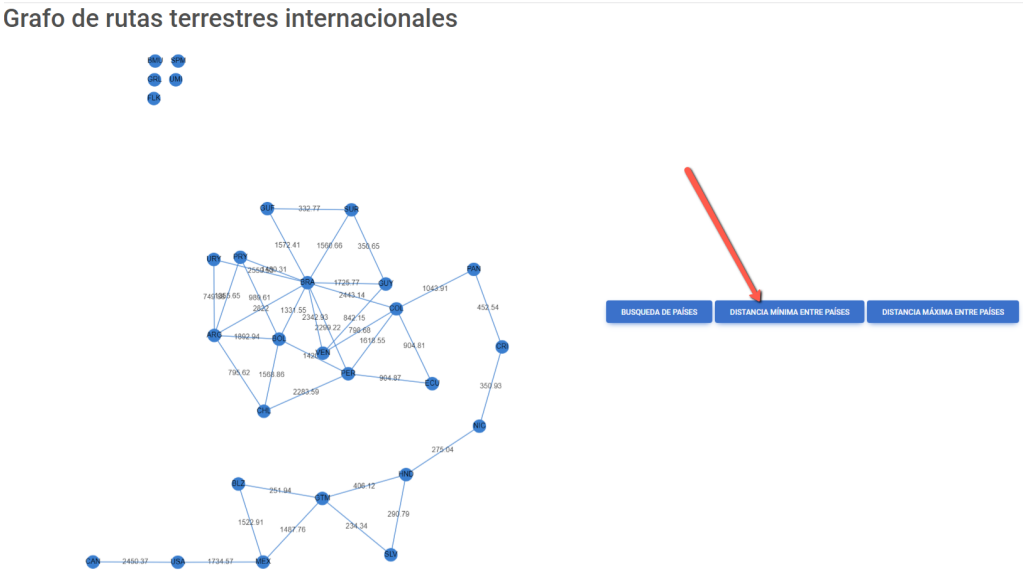


Y si el país no posee adyacencia se verá de esta manera

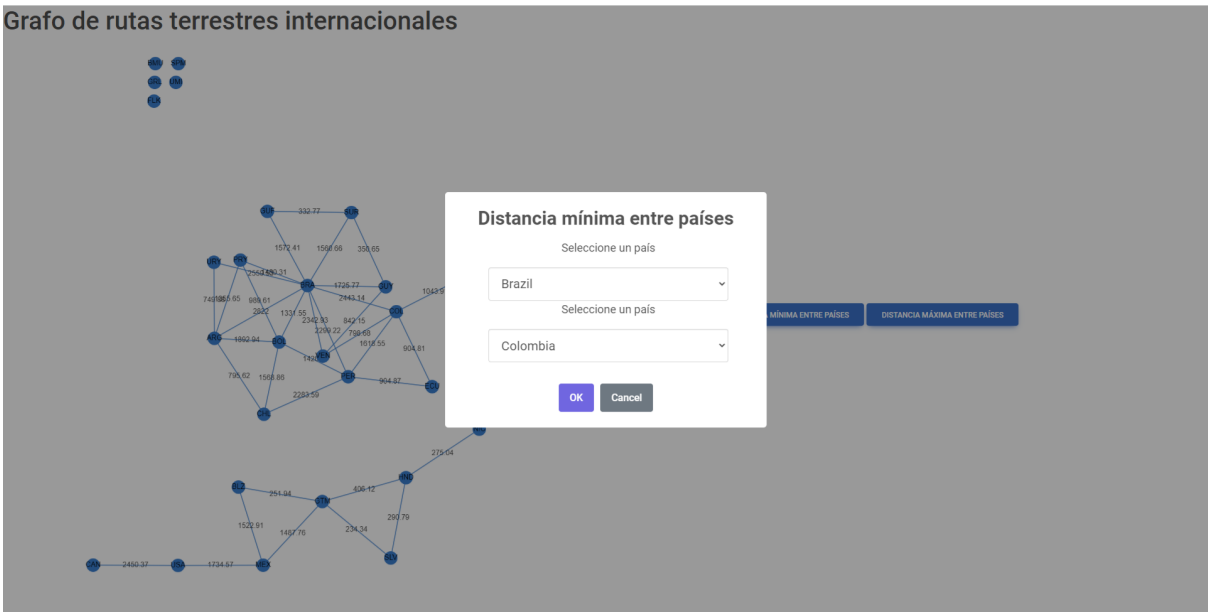


Distancia mínima entre países

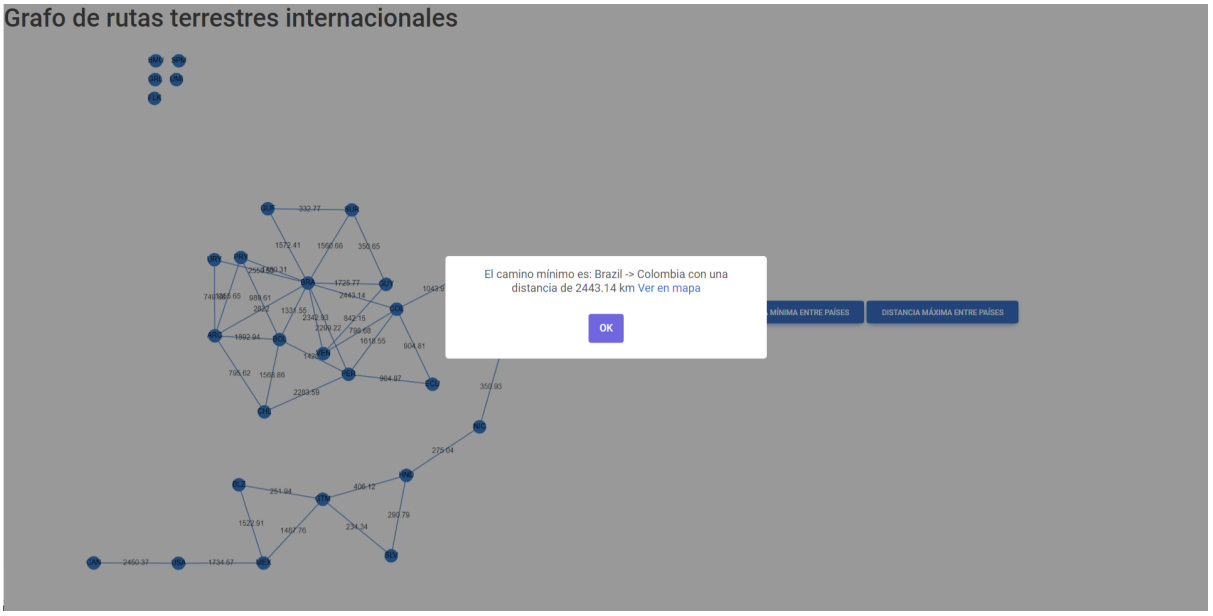
El usuario para buscar la distancia mínima entre países, tendrá que realizar un click en el botón de “Distancia mínima entre países”.



Luego se abrirá una ventana emergente para seleccionar el país de origen y el país de destino. El usuario deberá seleccionar los países y hacer click en el botón de “OK”.

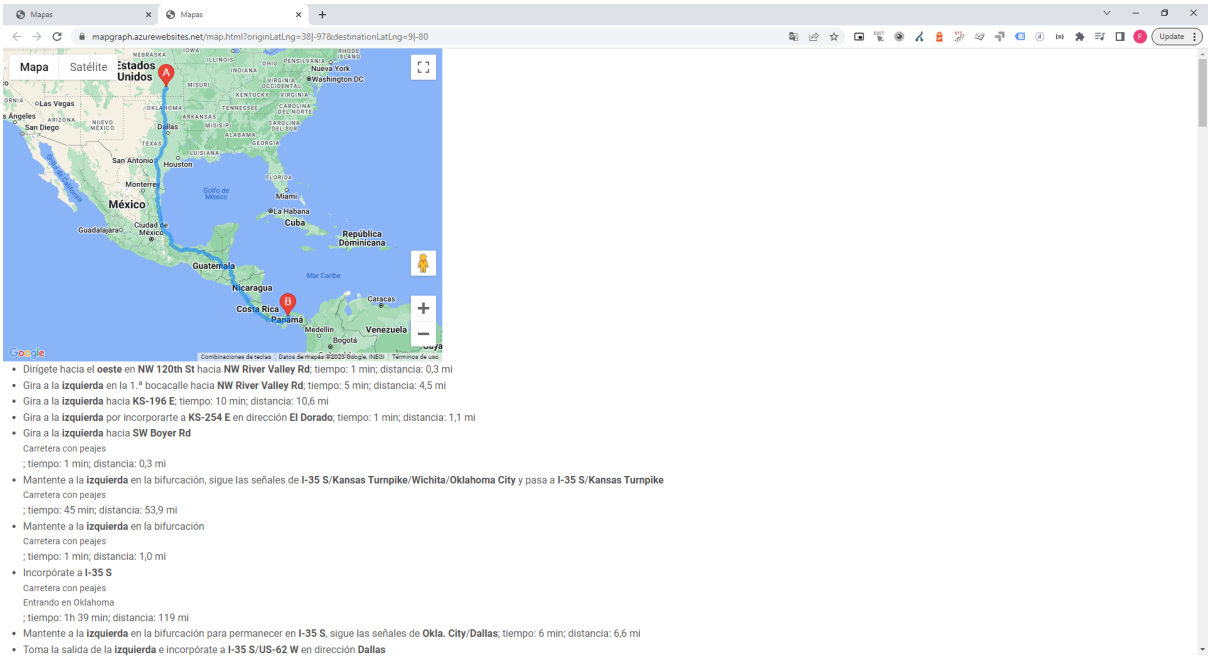


El sistema le mostrará cual es el camino mínimo para ir de un país a otro.

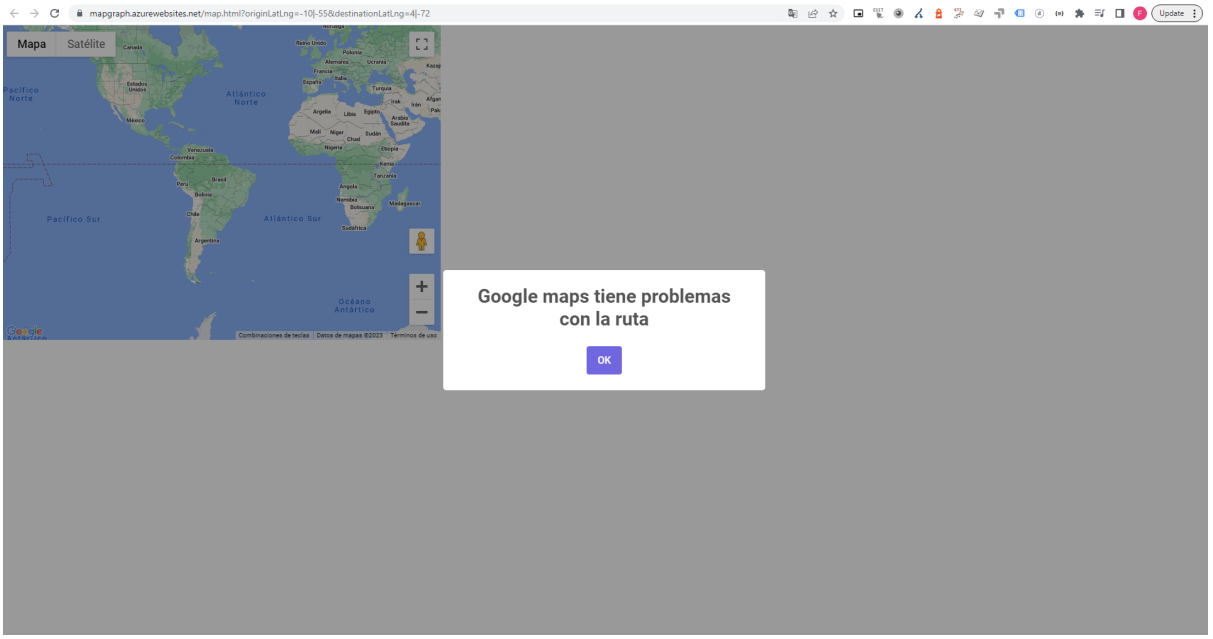


Si el usuario desea puede ver la ruta en el grafo creado o puede ver la distancia entre ambos países gracias a un link que lo redirecciona de Google Maps.

Esta parte le muestra la ruta terrestre para ir de un país a otro.

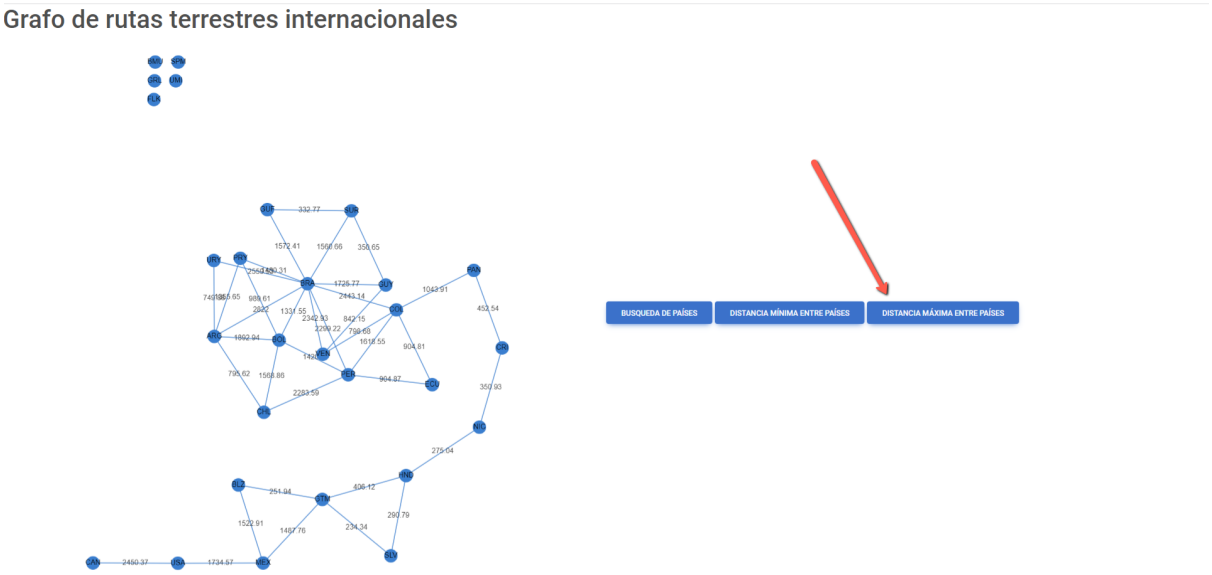


Importante mencionar que esta funcionalidad se encuentra limitada hacia algunos países por lo que es posible que en algunas ocasiones google maps alerte que no puede encontrar la ruta.

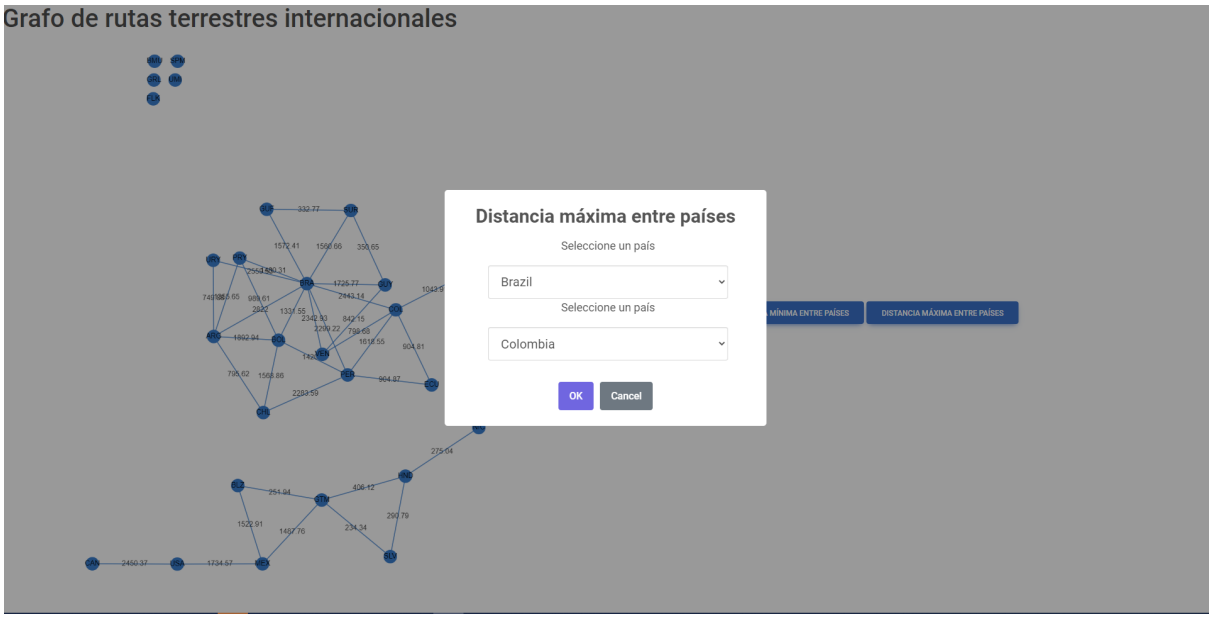


Distancia máxima entre países

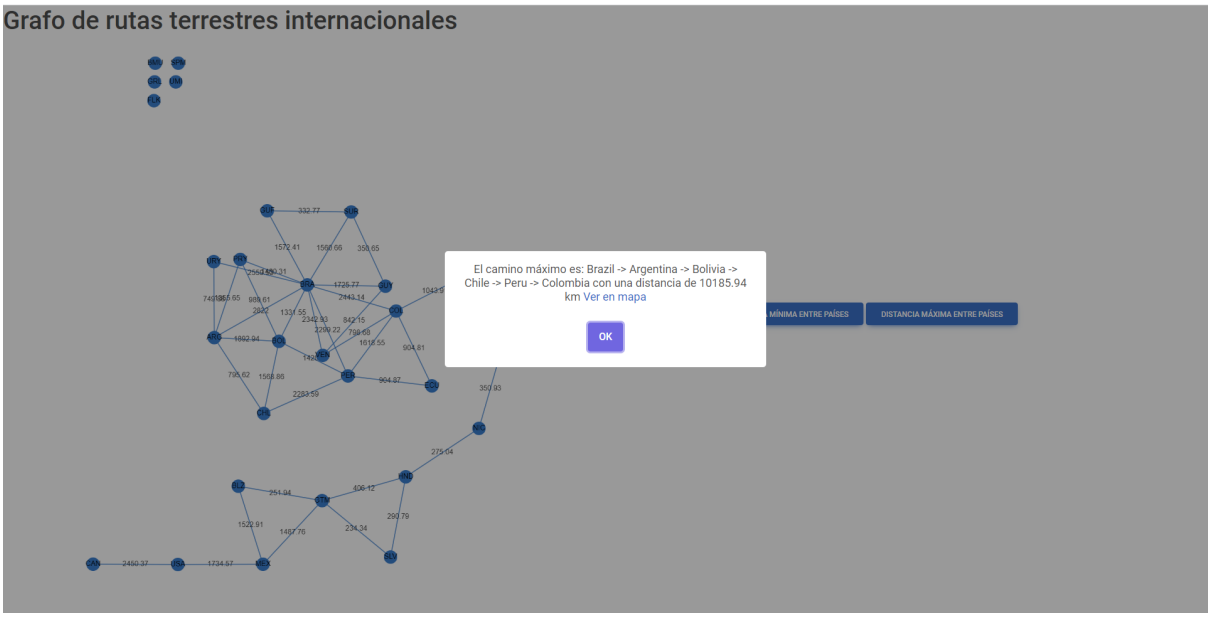
El usuario para buscar la distancia máxima entre países, tendrá que realizar un click en el botón de “Distancia máxima entre países”.



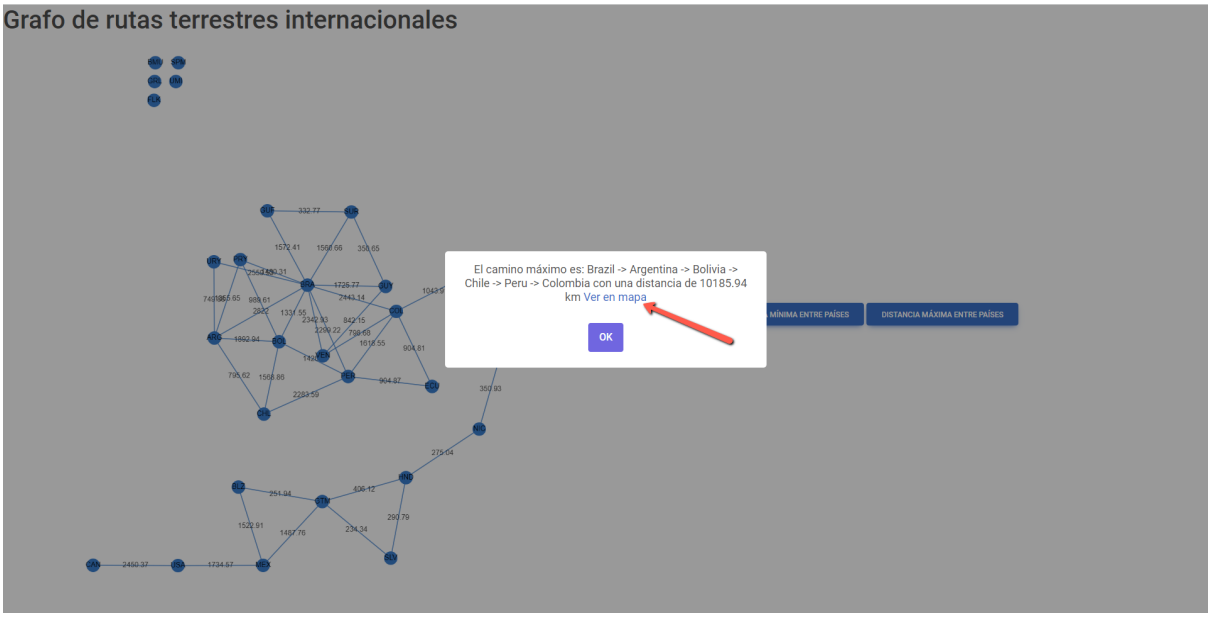
Luego se abrirá una ventana emergente para seleccionar el país de origen y el país de destino. El usuario deberá seleccionar los países y hacer click en el botón de “OK”.



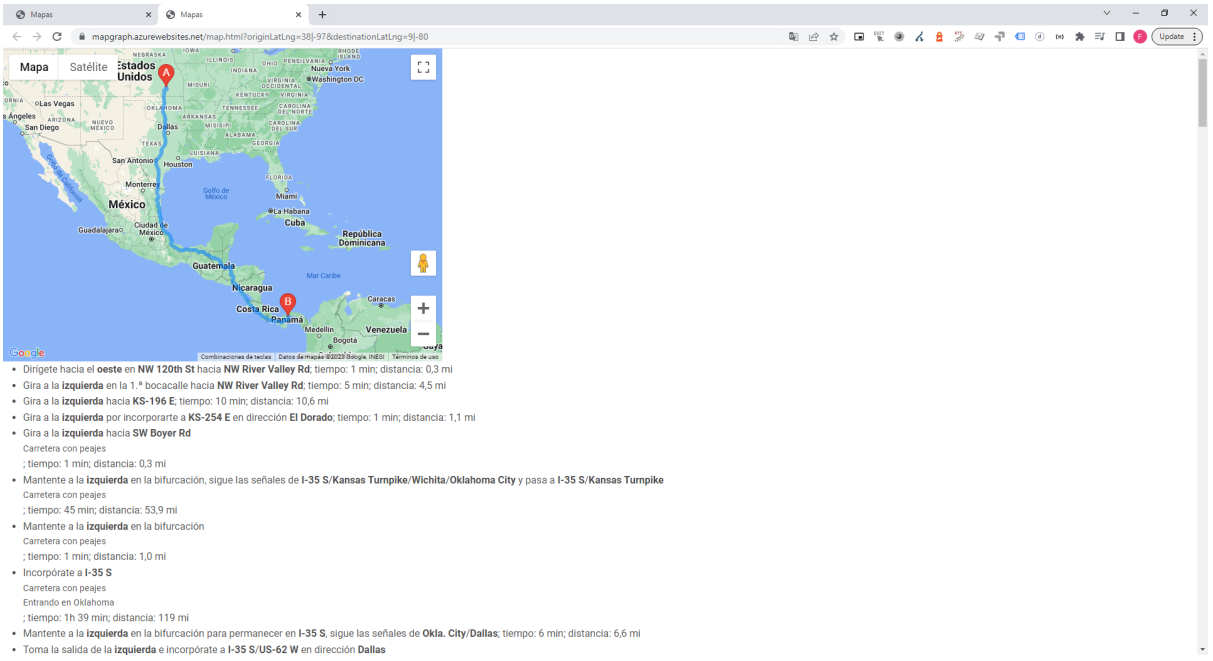
El sistema le mostrará cual es el camino máximo para ir de un país a otro.



Si el usuario desea puede ver la ruta en el grafo creado o puede ver la distancia entre ambos países gracias a un link que lo redirecciona de Google Maps.



Esta parte le muestra la ruta terrestre para ir de un país a otro.



- Dirígete hacia el **oeste** en **NW 120th St** hacia **NW River Valley Rd**; tiempo: 1 min; distancia: 0,3 mi
- Gira a la **izquierda** en la 1.ª bocacalle hacia **NW River Valley Rd**; tiempo: 5 min; distancia: 4,5 mi
- Gira a la **izquierda** hacia **KS-196 E**; tiempo: 10 min; distancia: 10,6 mi
- Gira a la **izquierda** por incorporarte a **KS-254 E** en dirección **El Dorado**; tiempo: 1 min; distancia: 1,1 mi
- Gira a la **izquierda** hacia **SW Boyer Rd**
  Carretera con peajes
  ; tiempo: 1 min; distancia: 0,3 mi
- Mantente a la **izquierda** en la bifurcación, sigue las señales de **I-35 S/Kansas Turnpike/Wichita/Oklahoma City** y pasa a **I-35 S/Kansas Turnpike**
  Carretera con peajes
  ; tiempo: 45 min; distancia: 53,9 mi
- Mantente a la **izquierda** en la bifurcación
  Carretera con peajes
  ; tiempo: 1 min; distancia: 1,0 mi
- Incorpórate a **I-35 S**
  Carretera con peajes
  Entrando en Oklahoma
  ; tiempo: 1h 39 min; distancia: 119 mi
- Mantente a la **izquierda** en la bifurcación para permanecer en **I-35 S**, sigue las señales de **Okla. City/Dallas**; tiempo: 6 min; distancia: 6,6 mi
- Toma la salida de la **izquierda** e incorpórate a **I-35 S/US-62 W** en dirección **Dallas**

Importante mencionar que esta funcionalidad se encuentra limitada hacia algunos países por lo que es posible que en algunas ocasiones google maps alerte que no puede encontrar la ruta.

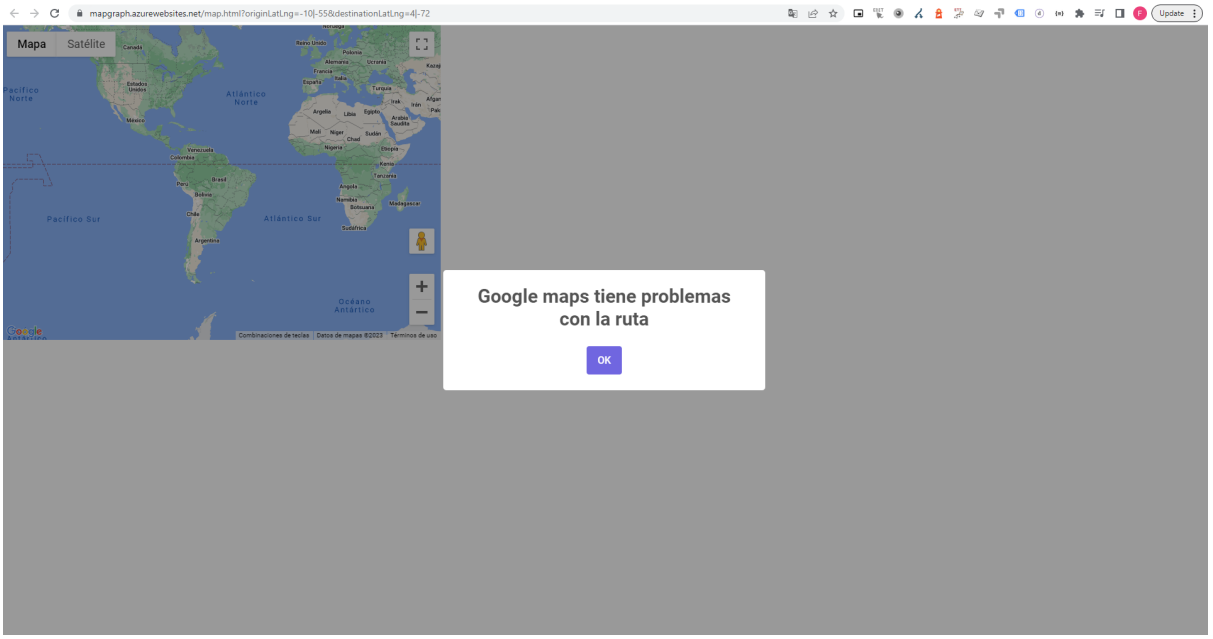
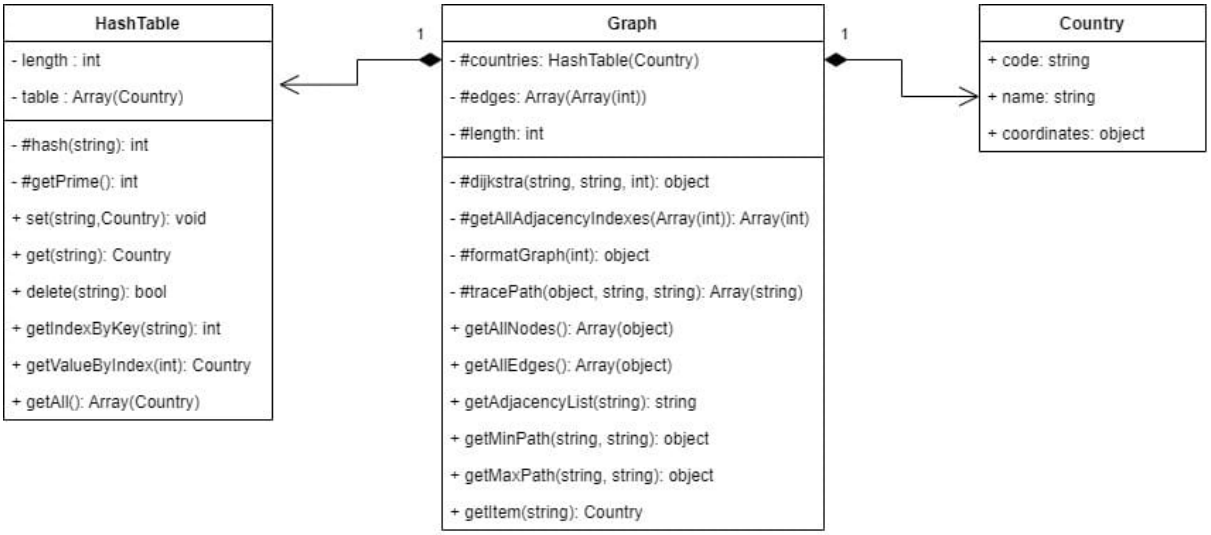


Diagrama UML

El código implementado se ve reflejado en un diagrama UML de esta forma:



Se observa que existe una composición entre las clases Country y HashTable en Graph, ya que el grafo para almacenar los países utiliza el hash y estos países se crean desde la data que devuelve el RestCountries API.

Estado actual de la aplicación

Actualmente, la aplicación está totalmente funcional en el ambiente de producción ya que las funcionalidades solicitadas:

- Búsqueda de países con hashtable
- Búsqueda de adyacencias de países
- Búsqueda del camino mínimo
- Búsqueda del camino máximo

Todas las funcionalidades fueron implementadas y no tienen ningún tipo de pulga.

Conclusión

En conclusión, el proyecto se basa en entender cómo funcionan los grafos para los mapas actuales y cómo pueden afectar mucho de lo que se utiliza cotidianamente.

Se puede deducir que, los grafos son importantes para los mapas porque permiten representar la información de manera estructurada y fácilmente procesable, ya que los nodos pueden representar ciudades, puntos de interés, carreteras, edificios, entre otros, mientras que las aristas representan las conexiones entre ellos, como las carreteras o caminos que los unen.

Al utilizar grafos para representar esta información, es posible realizar operaciones matemáticas y algoritmos sobre ellos para obtener información valiosa, como la distancia más corta entre dos puntos, la ruta óptima para llegar de un lugar a otro, y la detección de áreas de congestión de tráfico.

Además, no hay que dejar de lado la parte de búsquedas e interacciones rápidas un término que hoy en día es de suma importancia cuando se realiza un programa, en la eficiencia de cada función, en dicho código realizado, se muestra una variedad de métodos, como el hash, que en este caso, se aplicó uno cerrado, y de paso se añadió una parte para evitar las colisiones.

Finalmente se puede mencionar la parte de la arquitectura que se desarrolló en este proyecto en capas, donde la interfaz gráfica se comunicaba con un controlador para transportar los datos y la capa de negocio es la encargada de toda la lógica de búsqueda y relación de los datos con el grafo y el hashtable.

Referencias

Garg, P. (9 de 5 de 2018).
https://www.hackerearth.com/practice/data-structures/hash-tables/basics-of-hash-tables/tutorial/.

Google. (6 de 6 de 2015).
https://developers.google.com/maps/documentation/distance-matrix/overview.

Kruskal, J. (27 de 09 de 2018).
http://5010.mathed.usu.edu/Fall2018/THigham/Krukskal.pdf.

Mehlhorn, K. (3 de 10 de 2007).
https://people.mpi-inf.mpg.de/~mehlhorn/ftp/Mehlhorn-Sanders-Toolbox.pdf.

Shaffer, C. (19 de 1 de 2010).
https://people.cs.vt.edu/shaffer/Book/Java3e20100119.pdf.

Soe, N. C. (1 de 2 de 2020).
https://www.sciencepublishinggroup.com/journal/paperinfo?journalid=367&doi=10.11648/j.ijdsa.20200601.14.

Tyagi, N. (19 de 7 de 2022).
https://www.analyticssteps.com/blogs/dijkstras-algorithm-shortest-path-algorithm.

Ubah, K. (10 de 8 de 2021).
https://www.freecodecamp.org/news/html-css-and-javascript-explained-for-beginners/.