**Especificación de las estructuras y algoritmos utilizados en la aplicación**:

**Clase “Vértice”**: Estructura base del grafo, que contiene la información básica de cada uno de los “nodos” o vértices del grafo: nombre, latitud y longitud, una Lista encadenada de arcos, y un Vértice con la información del siguiente vértice en el grafo.

**Clase “Arco”**: Estructura base del grafo, contiene la información de las uniones o arcos del grafo: un Vértice que representa el vértice destino del arco, el peso del arco expresado en Kilómetros, y un arco que representa el arco siguiente en el grafo.

**Clase “ListaVértice”**: Lista enlazada de vértices. Contiene un vértice que representa el inicio de la lista, y funciones para agregar, buscar y eliminar vértices, así como para convertir la lista enlazada en una lista de C#(List<T>).

**Clase “ListaArco”**: Lista enlazada de Arcos. Contiene un arco que representa el inicio de la lista, y funciones para agregar, buscar y eliminar arcos de la lista, así como para convertir la lista enlazada en una lista de C#(List<T>).

**Clase “ControladorGrafo”**: Clase que representa el grafo, con una lista de vértices como atributo y funciones para obtener la lista de vértices ya almacenados, buscarlos, accesar a la tabla de Hash con la información de los vértices guardados, funciones de inicialización del grafo, y para obtener la ruta más corta entre dos vértices. Esta clase además es la que comunica la interfaz gráfica con la lógica de la aplicación.

**Clase “HashingTable”**: Clase que representa una estructura de “Tabla de hash”, una tabla en la que se guardan valores utilizando una función conocida como “función de hash”, la cual genera índices para la tabla para el guardado de datos con la idea de poder accesar esos datos en la forma más rápida. Utiliza un diccionario de C#(Dictionary<T,U>) para guardar las claves y los valores asociados a esas claves, en la forma de una cadena de texto, y un objeto. Contiene funciones para contar los valores guardados, comprobar si una clave existe dentro del diccionario, para agregar y remover valores, obtener valores guardados, y para la conversión de datos en índices por medio de la función de Hash.

**Clase “Calculador”**: Clase con un método para transformar pares de coordenadas a su equivalente en distancia, expresada en kilómetros.

**Clase “Form1”**: Formulario con el mapa, que contiene las funciones para mostrar el mapa y los puntos guardados como vértices. Contiene como atributos el grafo, los puntos iniciales del cursor del mapa, un marcador de mapas, y un overlay. Entre sus métodos, se encuentran las funciones para cargar el mapa, los vértices, dibujar el grafo, así como los controles para cada evento de la interfaz, específicamente los botones y campos para escribir datos, y el menú dropdown con todas las ubicaciones en el mapa.  
  
**Clase “Program’**: Clase con la función principal “main”, que da inicio al programa.

Ambas listas, de vértices y arcos, son listas enlazadas cuyos nodos apuntan al siguiente elemento de la lista. Se utiliza el algoritmo más básico de implementación de lista, en la cual cada elemento simplemente se agrega al final de la lista, sin orden específico, ya que se utilizan otras estructuras para manejar búsquedas y demás funciones.

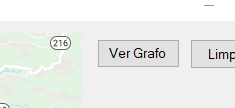
Para la búsqueda de la ruta más corta en el grafo, se utiliza el “Algoritmo de *Dikjstra*”, por ser uno de los más simples de implementar, y perfecto para el alcance del proyecto. Este algoritmo consiste en llevar cuenta de cuánto pesa cada arco relacionado con cada vértice, y sumar esos pesos, hasta llegar al más “liviano”, el cual representa la ruta más corta. En el caso de nuestra implementación, estos pesos son la distancia para llegar a cada nodo del mapa. Además se utiliza la implementación de grafo mediante la utilización de un “Grafo dirigido con lista múltiple de adyacencia”, el cual se escogió por tener definida una cantidad de vértices y por lo tanto encajar perfectamente para el alcance de la aplicación.

La tabla de Hashing contiene un algoritmo simple, utilizando la función SHA-256 incluída con el lenguaje C#, para convertir los datos guardados en índices de la tabla de Hash.

* **Problemas encontrados**:

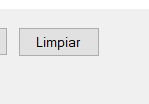
Se trató por mucho tiempo de representar los vértices de forma manual, pero debido a asuntos de tiempo y para evitar complicarse demasiado con la aplicación, se optó por utilizar la librería “GMap” de C# para manejar aspectos únicos del mapa.   
  
Se encontraron muchas dificultades para implementar el algoritmo de la ruta más corta/más larga, no se logró hacer trabajar el algoritmo con las estructuras utilizadas, y se priorizó la implementación del mapa y la estructura “Grafo”.   
  
- **Estado de la aplicación**:

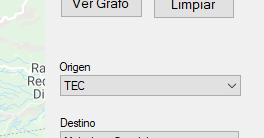
La aplicación se encuentra parcialmente operativa. Los botones para la obtener la ruta más corta y más larga existen en la interfaz gráfica, pero la lógica no está implementada por completo. La funcionalidad es decente pero hay muchas áreas de mejora, como por ejemplo se pueden incluir controles para funcionalidades como ampliar o reducir el mapa, crear rutas y por supuesto la opción de ver la ruta más corta/larga entre dos ubicaciones.

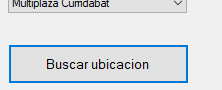
**Instrucciones** **de uso**:  
  
Para ver el grafo con todos los vértices, dé click al botón “Ver Grafo”:  
  
  
Luego utilice el “scroll” del mouse para hacer “zoom out”, lo que refrescará el dibujo del grafo:

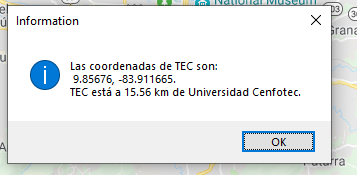


Para remover el grafo dibujado en el mapa, dé click al botón “Limpiar”:

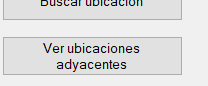


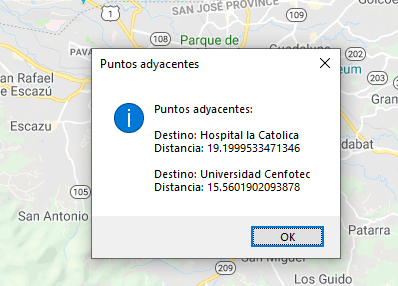
Para ver los detalles de las ubicaciones guardadas, seleccione una ubicación en el menú desplegable “Origen”, y luego dé click al botón “Buscar Ubicación”:  




Esto creará una ventana con la información solicitada:  


Para ver las ubicaciones adyacentes a la ubicación origen, dé click al botón “Ver ubicaciones adyacentes”:



Esto creará una ventana con la información solicitada:  
  
  
Para ver la ruta más corta entre dos puntos, seleccione una ubicación de origen y destino, y luego dé click al botón “Ver ruta mínima”:  
  
  
Para ver la ruta más larga entre dos puntos, seleccione una ubicación de origen y destino, y luego dé click al botón “Ver ruta máxima”:

