UNIVERSIDAD DE LAS AMÉRICAS PUEBLA

JULIO NOÉ HERNÁNDEZ TORRES

MATEMÁTICAS DISCRETAS/BASES DE DATOS

DEPARTAMENTO DE COMPUTACIÓN, ELECTRÓNICA Y MECATRÓNICA

PROYECTO PRIMAVERA 2025

Sistema de Navegación

Presentan:

CARLOS LOMELÍN VALDÉS 180625

SEBASTIÁN GARMENDIA RIVERA 181137

SANTIAGO IRIGOYEN VÁZQUEZ 180259 (MATEMÁTICAS DISCRETAS)

JOSÉ ANTONIO GONZÁLEZ MARTÍNEZ 181309 (MATEMÁTICAS DISCRETAS)

San Andrés Cholula, Pue. 5 de mayo de 2025

Sistema de Navegación

Introducción

Matemáticas Discretas

La Teoría de Grafos, aborda las características y cualidades de las gráficas o también llamados grafos, los cuales son conjuntos de estructuras discretas no vacías compuestas por objetos denominados vértices y por las conexiones entre ellos conceptualizadas como aristas; cabe destacar que, estas últimas pueden ser direccionadas o no.

Principalmente, los grafos son representaciones visuales que coadyuvan a plasmar la relación entre distintos agentes de una problemática, lo que permite identificar conexiones, dependencias y jerarquías entre los elementos involucrados; facilitando así, el análisis de sistemas complejos.

Los mismos, cuentan con un grado y orden, al igual que, se clasifican en muchos tipos como: subgrafos, grafos dirigidos y no dirigidos, ponderados, simples, multigrafos, entre otros. También, los vértices y nodos poseen un grado en conjunto con distintos tipos, lo cual posibilita el desarrollo de gráficos para distintas aplicaciones; las cuales, a forma general, pueden dirigirse a sectores como el de la informática, el tráfico aéreo y automovilístico, las redes sociales, las telecomunicaciones, la economía, etc.

Los sistemas de navegación forman parte del gremio al que se aplican los grafos, potenciando la representación de mapas y propiciando la planificación de rutas eficientes de manera visual, considerando factores de distancias, tiempos, o incluso condiciones de tráfico; lo cual, fundamenta la razón y el gran impacto que puede llegar a tener el presente proyecto.

Bases de Datos

Desde los años 70´s, los informáticos han desarrollado formas o áreas de almacenamiento para datos pequeños y masivos de manera digital, iniciando con mainframes, pasando por computadoras de escritorio, grid computing y computo en la nube; no obstante, debido a la evolución y a la aparición de empresas y el sistema económico se creó una familia de "software" que se enfoca en la gestión de datos increíblemente masivos. A esta se le denomina Base de Datos.

De forma puntual, las Bases de Datos son repositorios estructurados que permiten el almacenamiento, la manipulación y la recuperación eficiente de cualquier tipo de información; lo cual, posibilita la consulta y la actualización de estos al garantizar un sistema con seguridad, consistencia y escalabilidad.

Edificar una Base de Datos implica varios pasos. El primero consiste en modelar o definir la información que se quiere almacenar, para posteriormente enmarcarla gráficamente en un diagrama de Entidad Relación por medio de tablas y la definición de sus conexiones y del tipo de cada uno de los atributos. Finalmente, se tiene que desarrollar el código para crear la Base digitalmente (en el lenguaje de preferencia), a la vez que, se tienen que formular las consultas dependiendo de los datos que se quieren obtener.

En los últimos años, esta serie de pasos es realizada por cada vez más personas, instituciones, empresas y gobiernos, lo que muestra que las Bases de Datos se han convertido en una herramienta fundamental para la gestión, análisis y toma de decisiones basada en información estructurada y confiable. Por ello y mucho más, nos dimos a la tarea de implementarlas en el proyecto en cuestión, de forma en que se optimizará la gestión de la información relacionada con

edificios, rutas y distancias dentro del campus, facilitando así la navegación eficiente y precisa para los usuarios que empleen el sistema de navegación.

Objetivos

Matemáticas Discretas

- **General**
  - Construir un sistema de navegación que simule el movimiento de un estudiante a través de los distintos edificios de la UDLAP.
- **Específicos**
  - Consolidar los edificios (vértices) que engloban al grafo.
  - Determinar el camino inicial y único (aristas) a partir del cual puede existir movilidad entre los edificios de la universidad.
  - Desarrollar un programa con interfaz gráfica en JAVA que modele las mejores rutas que se pueden tomar de un punto a otro.
  - Realizar un análisis que compruebe que las rutas fueron erigidas de forma exitosa; es decir, que son las más cortas posibles.

Bases de Datos

- **General**
  - Edificar una Base de Datos que ofrezca el almacenamiento y la gestión de la información del sistema de navegación en cuestión.
- **Específicos**
  - Fabricar un diagrama Entidad Relación que coadyuve a la definición de la información que se manejará.

- Generar diferentes tablas que administren los edificios, las rutas posibles, las distancias y las rutas más cortas disponibles.
- Desarrollar un excelente código en lenguaje SQL.
- Construir diferentes consultas que posibiliten la obtención de datos de manera eficiente y veloz.

Integrantes

La construcción del proyecto en cuestión fue edificada en su totalidad gracias al trabajo colaborativo de los cuatro integrantes del equipo; puntualmente, se realizaron reuniones tanto síncronas como asíncronas a lo largo de 2 semanas para coordinar, asignar y ejecutar cada una de las secciones asignadas

Equitativamente, cada uno ejecutó tareas por igual de la totalidad del proyecto en el caso de Matemáticas Discretas; en cambio, en Bases de Datos ambos integrantes realizaron un 50% del total del trabajo. En cuanto al documento que se lee, cada uno nos encargamos de redactar 3 secciones de las 6 que engloban al documento; igualmente, los códigos en JAVA y en SQL, fueron desarrollados en asociación al proporcionar ideas y líneas de código por igual. Finalmente, el manual de usuario y las presentaciones en CANVA se construyeron en asistencia mutua. A continuación, se presenta la lista de tareas realizadas:

- Carlos
  - Reporte: Introducción, Objetivos y Desarrollo del Sistema.
  - Códigos: Ideas y programación por igual.
  - Manual de Usuario: 50% tanto de los materiales como de la sección ¿Cómo usarlo?

    - Presentaciones: 5 diapositivas de las 20 realizadas (Matemáticas Discretas) y 8 diapositivas de las 15 totales (Bases de Datos).
- Sebastián
    - Reporte: Integrantes, Evaluación y Conclusiones.
    - Códigos: Ideas y programación por igual.
    - Manual de Usuario: 50% tanto de los materiales como de la sección ¿Cómo usarlo?
    - Presentaciones: 5 diapositivas de las 20 totales (Matemáticas Discretas) y 7 diapositivas de las 15 totales (Bases de Datos).
- Santiago (Matemáticas Discretas)
    - Reporte: Introducción, Integrantes y Conclusiones.
    - Códigos: Ideas para el diseño de la interfaz.
    - Manual de Usuario: 50% del diagrama UML.
    - Presentaciones: 5 diapositivas de las 20 totales.
- José Antonio (Matemáticas Discretas)
    - Reporte: Evaluación, Integrantes y Conclusiones.
    - Códigos: Ideas para el diseño de la interfaz.
    - Manual de Usuario: 50% del diagrama UML.
    - Presentaciones: 5 diapositivas de las 20 totales.

Desarrollo del sistema

Matemáticas Discretas

En primera instancia, se determinaron los nodos o edificios que engloban al sistema de navegación de la UDLAP; estos, fueron 7 en su totalidad representando el auditorio Guillermo y Sofía Jenkins, la Biblioteca, la Cafetería, el edificio de Humanidades, Ingenierías, Negocios/Ciencias Sociales y

Medicina; de igual manera, se consolidó el camino restrictivo por el cual se puede llegar a alguno de los edificios por medio de aristas no direccionadas dando lugar a un grafo conexo. A continuación, se expone el diagrama formulado:

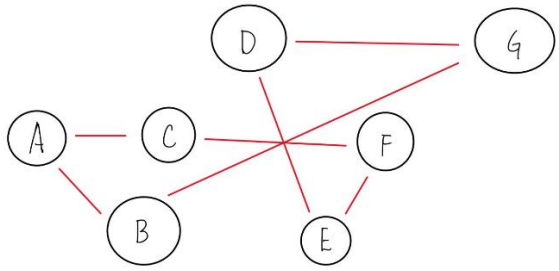

Figura 1. Diagrama inicial de nodos y aristas.

Donde:

A: Edificio de Humanidades.

B: Edificio de Ingenierías.

C: Auditorio Guillermo y Sofía Jenkins.

D: Edificio de Negocios/Ciencias Sociales.

E: Biblioteca.

F: Cafetería.

G: Edificio de Medicina.

Posteriormente, se afianzó la ponderación de los pesos de cada una de las aristas, lo cual se representó por medio de las distancias en metros entre cada uno de los edificios después de realizar una aproximación a pie: entre A y B la distancia es de 200m, entre A y C de 300m, de B a G es de 600m, de C a F existen 300m, entre D y E son 100m, de D a G son 300m, de E a F son 200m.

Una vez definido lo anterior, comenzamos con la estructuración del código en el lenguaje escogido, el cual fue JAVA debido a la gran familiaridad que poseemos con su sintaxis y lógica,

y a la facilidad que ofrece para realizar entornos interactivos. Primeramente, realizamos una clase llamada “ProyectoPrimavera2025”, la cual sirvió como host del main del programa; en ella, importamos java.util.* para poder recibir los datos que ingrese el usuario a través de scanner y System.in. Puntualmente, se formularon las preguntas sobre en qué edificio se inicia y en cual se termina, lo que se guardó en dos variables “inicio” y “fin”; después, se ejecutó una estructura de if, else if y else para manejar posibles casos en los que el usuario indicara un edificio inválido, y en el caso de que escogiese uno correcto se estructuró que se crease el grafo correspondiente y que se ejecutase el algoritmo de Dijkstra.

El siguiente paso en el desarrollo del código fue crear la clase “Ruta”, en la cual se administró todo lo necesario para obtener el camino más corto entre dos edificios del campus. En primera instancia, se inicializó el número de edificios como “n=7” en conjunto con el valor inicial de las conexiones entre nodos como “valorinicialC=infinito” y el arreglo de strings “nombrenodos” que sirvió para identificar a que letra de la Figura 1 corresponde cada edificio. A posteriori, se construyó el método “getIndex” para poder ver si lo ingresado por el usuario en el main, está contenido en el arreglo que tiene los nombres de los nodos; esto, por medio de un for para recorrer el arreglo y buscar la coincidencia de acuerdo a la letra que ingresó el usuario. Más tarde, se edificó el método de tipo int [] [] llamado “Grafo” que sirvió para plantear la matriz de adyacencia que administra el grafo en el que se basa este proyecto; para ello, creamos la matriz mencionada con el nombre de “grafo” con un tamaño de “n”x”n”, es decir, 7x7. Posteriormente, realizamos dos for anidados (para recorrer filas y columnas) que nos permitieron asignar el valor “valorincialC” a cada uno de los espacios de la matriz, para después crear un arreglo denominado “nodos” en donde se indicaron las conexiones entre nodos y los pesos de estas (las distancias entre edificios); y luego, se añadieron los datos mencionados a la matriz de adyacencia mediante un for. Finalmente, se

desarrolló el método “dijkstra” de tipo void que recibió como firma un arreglo “grafo” y dos variables enteras “inicio” y fin”; con ello, se inicializó un arreglo de tamaño “n” denominado “distancias” el cual sirvió para almacenar los 7 pesos de las conexiones desde el nodo inicial hacia los demás nodos. Igualmente, realizamos un arreglo de tipo booleano “nodovisitado” para saber con un true o false si un nodo ya se visitó en la ruta, a la vez que, creamos otro arreglo “nodoprevio” para almacenar el nodo previo a cada avance, abriendo así la posibilidad de poder reconstruir el camino al final del código; de igual forma, inicializamos estos arreglos con “valorinicialC” (distancias) y -1 (nodoprevio, indicando que el primer nodo no tiene predecesor). Una vez hecho lo anterior, nos dimos a la tarea de realizar el for que maneja el algoritmo de Dijkstra, en primer lugar hicimos un for global que recorre todos los nodos, para así inicializar una variable “d” en -1 para representar el nodo con menor distancia que, al usar otro for si no se ha visitado el nodo de la iteración y este cuenta con la distancia más “baja”, se convierte en el nodo más cercano; después de ello, se agregó un if por si la distancia es nula o no hay conexión posible con el siguiente nodo se rompiese el ciclo, de lo contrario se indica el nodo en cuestión como ya visitado. Ulteriormente, se construyó otro for para recorrer los nodos vecinos del nodo correspondiente a la iteración, y si existe conexión entre el vecino y d “nodo actual”, a la vez que, se encuentra que la ruta desde el nodo inicial es más corta que el camino conocido se actualiza la nueva distancia corta marcando que el nodo previo al nuevo nodo alcanzado es d. En adición, a forma de manejar un tipo de error, aplicamos un if en el que si después de lo anterior todavía no se da con el nodo destino se manda un mensaje en el que se indica que no hay una ruta disponible y, por tanto, existió un fallo en el algoritmo. Para finalizar, creamos una lista vacía para reconstruir el camino más corto formulado por medio de un for y varios add´s de los nodos previos, que después fueron reacomodados para

ver la ruta al derecho (al añadirlos en la lista la ruta se añade de forma contraria); lo que nos permitió mostrar en pantalla la ruta más corta en adición con la distancia que se recorre en metros.

Seguidamente, edificamos un diagrama de tipo borrador en el que plasmamos la idea que teníamos sobre nuestra posible interfaz (Figura 2 y 3), la cual fue lograda casi en su totalidad a través de un Frame Form llamado “Interfaz” que englobó 5 métodos a parte del main (“Interfaz”, “segundaVentana”, “redirecciondeSalida”, “dijkstraRuta” y “Boxsuperior”); en estos, se utilizó sintaxis como JFrame, JButton, JTextArea, JPanel, BorderLayout, JComboBox, JLabel, getImage, ActionListener para poder construir la interfaz gráfica. Específicamente, se inició el código con un arreglo llamado “Edificios” para indicar que letra correspondía con cada edificio, justo después se declaró el “areaResultados” como variable de clase y se crearon los colores “UDLAPVerde” y “UDLAPNaranja” que fueron utilizados a lo largo del diseño. El método “Interfaz” realmente se genera automáticamente por NetBeans y toma la función de un tipo constructor inicializar todos los elementos gráficos de la ventana. En cuanto al main, este funcionó como el desarrollador de la ventana de bienvenida en el que el se agregó el fondo y el logo de la universidad, al igual que, un botón para abrir la siguiente ventana; por otra parte, el método “segundaVentana” fue conformado por un panel de control, dos botones (calcular ruta y salir) y el espacio que muestra los resultados. Posteriormente, realizamos el método “redirecciondeSalida”, el cual sirvió para redirigir la salida estándar (System.out) hacia “areaResultados”, esto en lugar de mostrarla en la consola; seguidamente, se creó “dijkstraRuta” que contribuyó a traer desde la clase “Ruta” el algoritmo Dijkstra para implementarlo al oprimir el botón de Calcular Ruta. Ultimadamente, se implementó el método “Boxsuperior”, en el que se aplicó un estilo uniforme a los JComboBox de la GUI en cuestión.

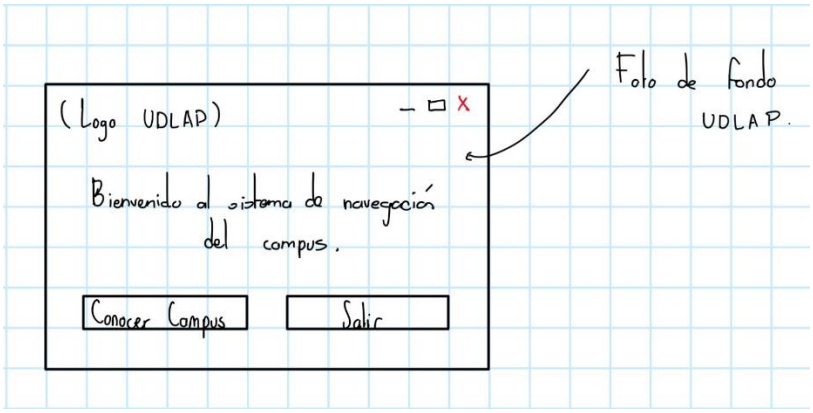

Figura 2. Diagrama de primera ventana a mano de la interfaz gráfica.

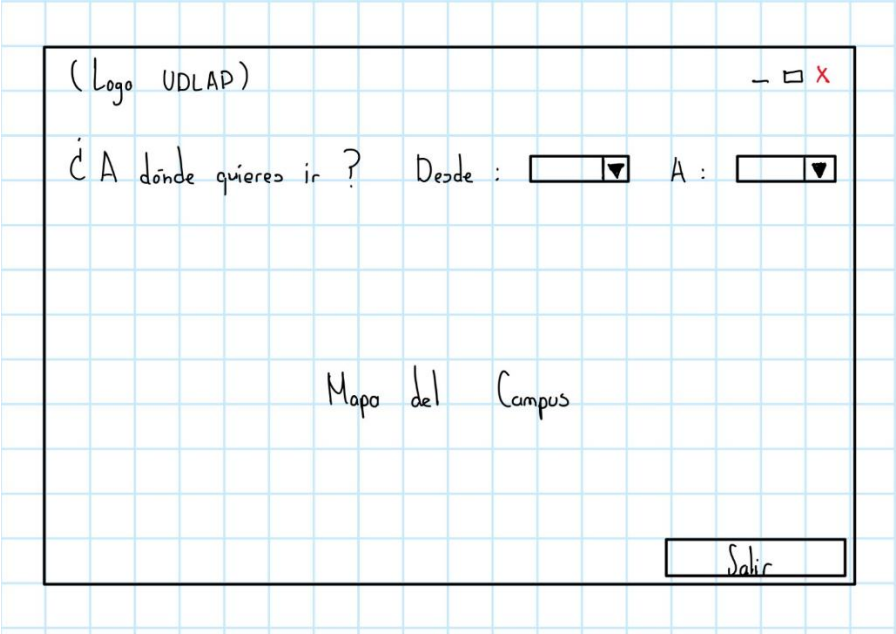

Figura 3. Diagrama de segunda ventana a mano de la interfaz gráfica.

Cabe destacar que, el diseño final sufrió algunas modificaciones al diseño propuesto inicialmente, no obstante, si se logró mantener la esencia casi en su totalidad. A continuación, se expone nuestra interfaz:

Figura 4. Ventana de bienvenida (Interfaz Gráfica).

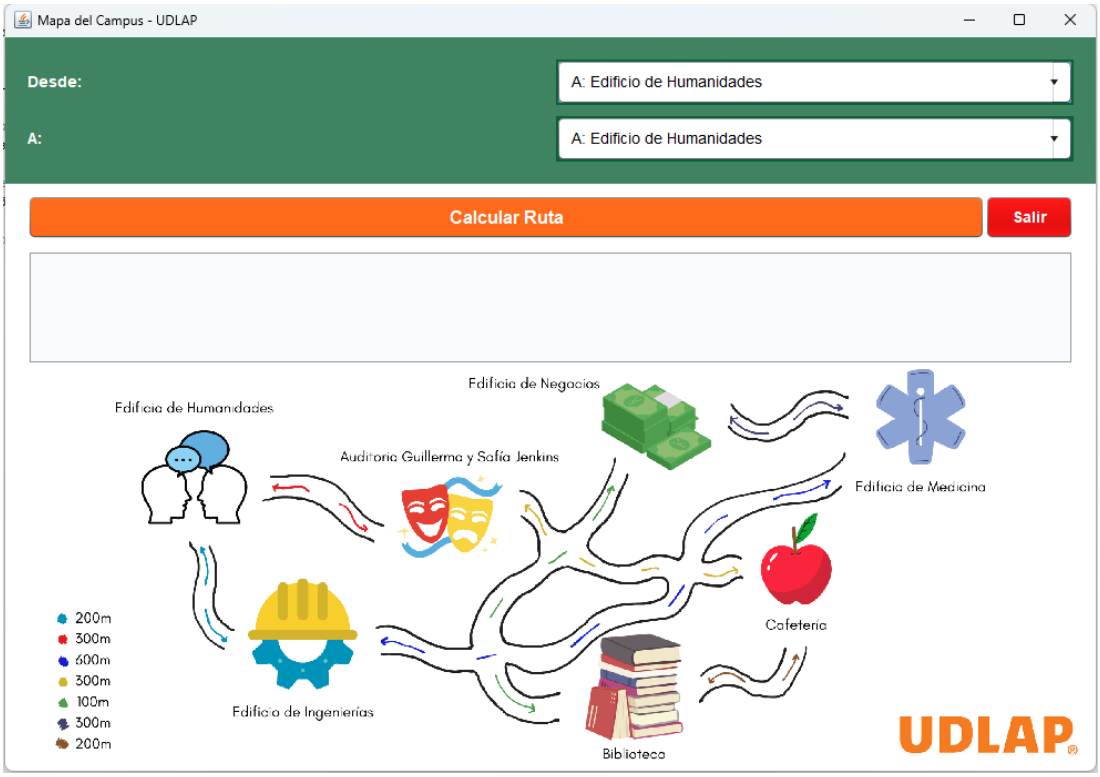

Figura 5. Segunda Ventana (Interfaz Gráfica).

Bases de Datos

Primeramente, se procedió al diseño gráfico de la base de datos empleada para modelar la información del sistema de navegación desarrollado. Para ello, se elaboró el correspondiente diagrama Entidad-Relación (Figura 6), con el objetivo de obtener una representación clara y estructurada de cómo se organizarían y gestionarían los datos.

En este diagrama se pueden observar las 4 tablas correspondientes a los edificios, a los caminos, a las mejores y a las peores rutas que se pueden realizar de acuerdo con la disposición de los nodos anteriormente explicados.

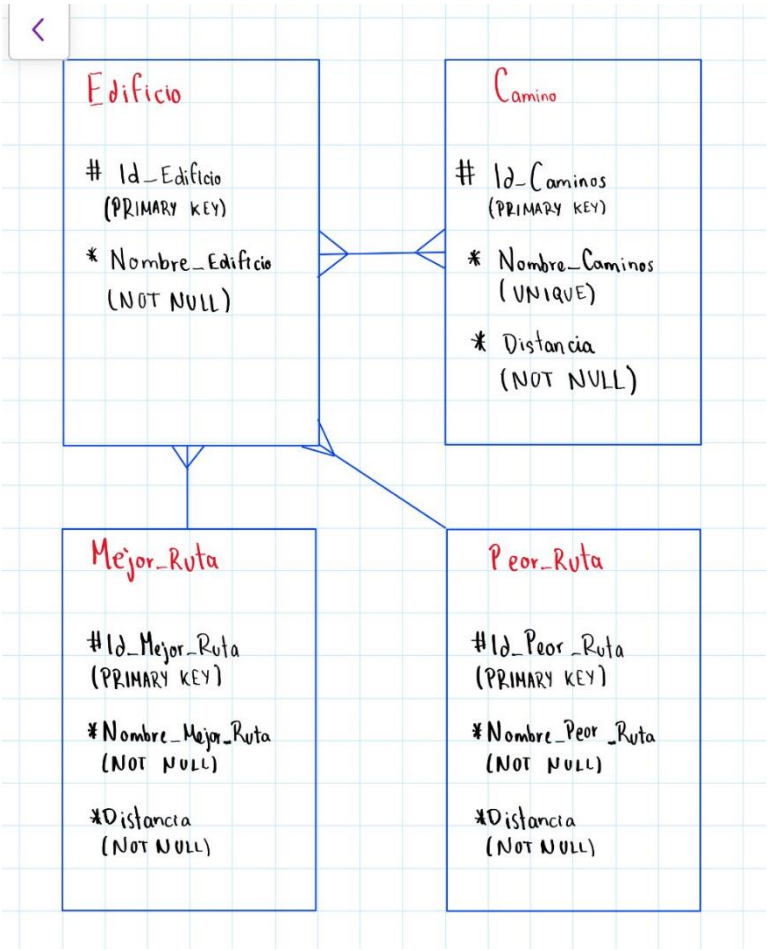

Figura 6. Diagrama Entidad-Relación para la creación de la base de datos.

Como se puede apreciar, las conexiones entre las tablas son de muchos a muchos debido a que un edificio tiene 2 posibles caminos, al mismo tiempo que, un camino tiene 2 posibles edificios como destino; y, en cuanto a las conexiones con la mejor y peor ruta se definió de muchos edificios a una peor/mejor ruta debido a que solo existen dos variaciones de los caminos en el grafo edificado.

En relación con el concepto de normalización, se puede afirmar que el esquema cumple con los requisitos de la Tercera Forma Normal (3FN); por tanto, se cumplen otras dos normas existenes. En primer lugar, todas las tablas están estructuradas con atributos atómicos, es decir, cada campo almacena únicamente un valor por celda. Por ejemplo, en la tabla Edificio, el campo Nombre_Edificio contiene un solo nombre por registro, sin listas ni valores compuestos. Además, cada tabla cuenta con una clave primaria simple (un solo atributo identificador), lo que garantiza que todos los campos dependen funcionalmente de la totalidad de dicha clave y no de una parte de ella. En el caso de la tabla Camino, Id_Caminos es la clave primaria, y los atributos Nombre_Caminos y Distancia dependen funcionalmente de ella, cumpliendo también así con la Segunda Forma Normal explícitamente. Dado que no existen dependencias transitivas (es decir, ningún atributo depende de otro atributo no clave), igualmente se satisface la condición para la Tercera Forma Normal. Estos principios y Formas Normales se extienden de igual forma a las tablas Mejor_Ruta y Peor_Ruta, donde los atributos Nombre_Mejor_Ruta / Nombre_Peor_Ruta y Distancia dependen directamente de sus respectivas claves primarias.

La primera tabla, correspondiente a los edificios, incluye un identificador único (id), el cual actúa como clave primaria de dicha tabla. Además, contiene el nombre del edificio, definido como un campo NOT NULL, ya que es indispensable contar con esta información para facilitar su identificación dentro del sistema. No obstante, no se impuso la restricción de unicidad sobre

este campo, dado que, al igual que en aplicaciones como Waze o Google Maps, es común que diferentes lugares compartan el mismo nombre sin generar conflictos en el funcionamiento del sistema.

En la segunda tabla se representaron los caminos construidos. Esta incluye un campo Id_Caminos que funge como clave primaria. También contiene el nombre del camino, que en este caso corresponde a los dos puntos que conecta. Este campo fue definido como único (UNIQUE), bajo el criterio de que cada camino une exclusivamente a un par de edificios y no debe repetirse. Finalmente, se incluye un campo para la distancia entre los puntos conectados, el cual es fundamental para el algoritmo de cálculo de rutas óptimas, ya que permite comparar trayectos y determinar la mejor opción disponible.

Por último, se diseñaron dos tablas adicionales para registrar la mejor y la peor ruta, respectivamente. Ambas comparten una estructura similar: cuentan con un identificador único como clave primaria, los nombres de los puntos que conecta la ruta (también definidos como campos NOT NULL para asegurar que siempre representen una conexión entre dos edificios), y la distancia total entre dichos puntos. Este último valor también debe ser no nulo, ya que implica la existencia de un trayecto real entre las ubicaciones.

Evaluación

Matemáticas Discretas

Para el testeo del sistema en cuestión, se realizaron tres pruebas puntuales. En la primera, se determinó como punto de partida la Cafetería y como destino el Edificio de Negocios, obteniendo así un recorrido de 300 m que pasa igualmente por la Biblioteca; lo cual, es el camino más corto debido a que el nodo con menos peso y el más cercano a la Cafetería es la Biblioteca (a 200m, en

comparación con el Auditorio que está a 300m), la cual solo tiene como camino al Edificio de Negocios (a 100m). La segunda prueba, consistió en un camino de 1000m que parte del Edificio de Ingeniería y finaliza en la Biblioteca, pasando por el Edificio de Humanidades, el Auditorio Guillermo y Sofía Jenkins, y la Cafetería; esto, se puede declarar como la ruta más efímera ya que el nodo del Edificio de Humanidades esta a tan solo 200m del de Ingenierías (y no a 600m como lo esta el otro nodo conectado, que es el de Medicina), después el Edificio de Humanidades te lleva forzosamente por el Auditorio que se encuentra a 300m, el cual te dirige obligatoriamente a la Cafetería que está a 300m y que, finalmente, te lleva de forma exclusiva a la Biblioteca en un tramo de 200m. Como último y tercer experimento, se llevó a cabo un paseo de 900m entre el Edificio de Medicina y el Auditorio Guillermo y Sofía Jenkins, que inició al pasar por el Edificio de Negocios por unos 300m (menores a los 600m de la otra conexión, que es el Edificio de Ingenierías) y culminó al recorrer la ruta obligatoria entre la Biblioteca (100m), la Cafetería (200m) y el Auditorio (300m).

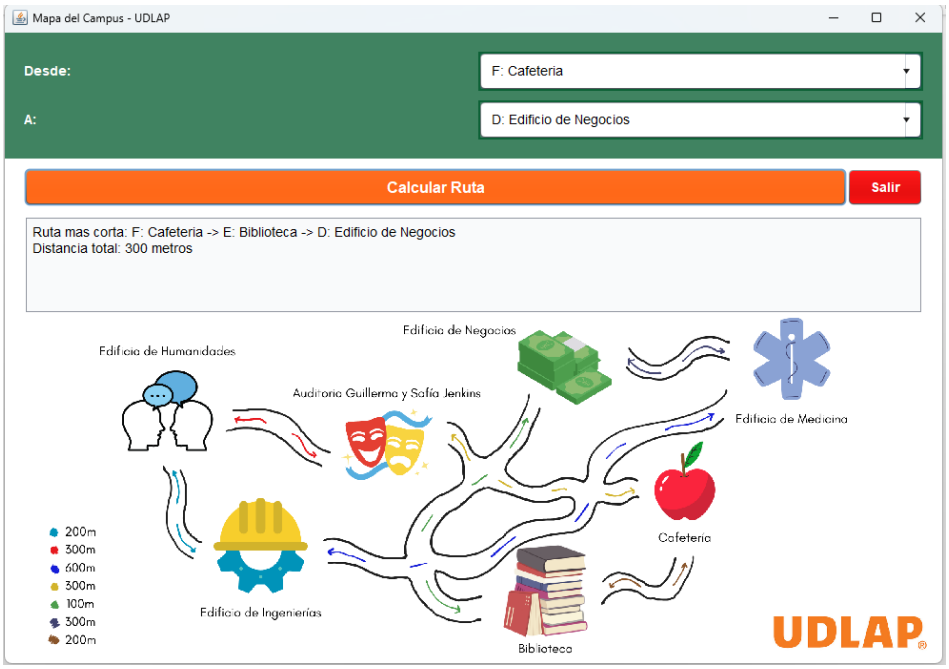

Figura 7. Prueba 1 – Cafetería a Edificio de Negocios (Ruta de 300m incluyendo a la Biblioteca).

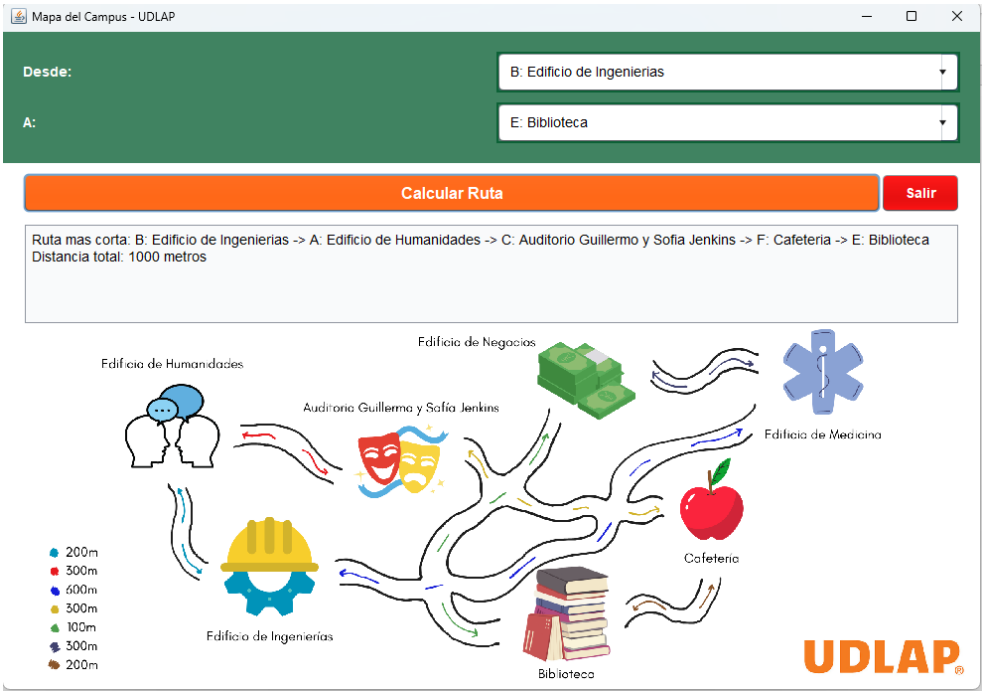

Figura 8. Prueba 2 – Edificio de Ingeniería a Biblioteca (Ruta de 1000m incluyendo al Edificio de Humanidades, al Auditorio Guillermo y Sofía Jenkins y a la Cafetería).

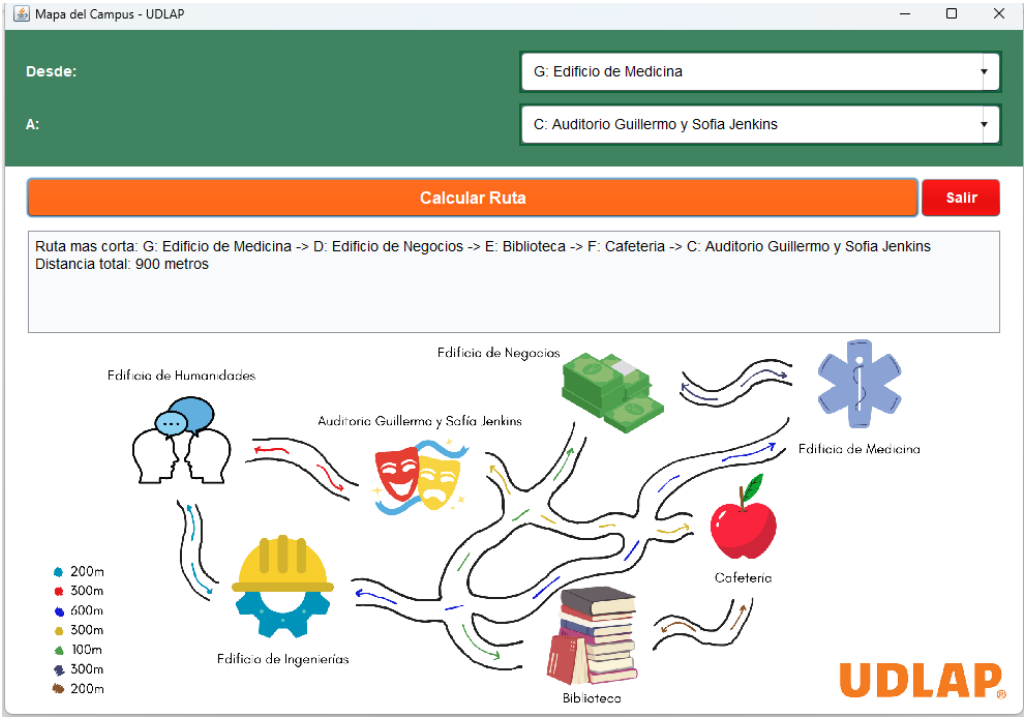

Figura 9. Prueba 3 – Edificio de Medicina a Auditorio Guillermo y Sofía Jenkins (Ruta de 900m incluyendo al Edificio de Negocios, a la Biblioteca y a la Cafetería).

Como se puede apreciar, las Figuras # y la explicación redactada fundamentan completamente que el código desarrollado trabaja de forma correcta y eficiente; esto, al proporcionar la ruta más corta posible entre dos puntos de la UDLAP en las tres pruebas formuladas.

Bases de Datos

En total se llevaron a cabo 10 consultas diferentes, en las cuales se ejecutaron búsquedas de edificios específicos mediante su nombre. Por ejemplo, se verificó la existencia de inmuebles denominados "Ingenierías" o "Auditorio Guillermo y Sofía Jenkins". Estas búsquedas puntuales permitieron al sistema ubicar lugares clave dentro del entorno, lo cual es fundamental para funcionalidades como marcación de destinos o verificación de datos.

Asimismo, se efectuó un filtrado de rutas que superan una determinada distancia, por ejemplo, aquellas superiores a 250 metros, lo que facilitó la identificación de trayectos relativamente extensos. En otro caso, se localizó una ruta específica mediante su identificador único, lo que permite acceder de manera directa a un recorrido concreto. Las rutas filtradas por longitud fueron posteriormente ordenadas de menor a mayor, brindando una visión clara de las opciones más cortas dentro del subconjunto de trayectos más largos.

Una de las consultas más interesantes es la que compara las distancias de cada ruta con la distancia promedio de todas. De esta forma, se pueden identificar los trayectos que están por encima de la media, lo cual es útil para detectar posibles ineficiencias o rutas que merecen ser optimizadas.

Otra operación clave desarrollada, se encargó de comparar la mejor y la peor distancia registrada para una misma ruta, vinculando ambos registros y mostrando el nombre de la ruta junto

con sus dos extremos: la mejor y la peor opción. Esto resultó muy útil para el análisis de calidad del sistema en cuestión o para entender cuánto puede variar un trayecto.

Finalmente, se llevaron a cabo acciones sobre la base de datos, tales como la inserción de un nuevo edificio denominado "Unicaja", la posterior consulta para verificar su existencia, la actualización de su nombre a "Hacienda" y la validación del cambio realizado.

En su totalidad, estas operaciones pusieron de manifiesto la relevancia crítica de asegurar que la información contenida tanto en el sistema de navegación como en la base de datos se mantenga actualizada, precisa y coherente en todo momento. Como se puede observar, la integridad de los datos no solo garantizó un funcionamiento eficiente y confiable del sistema, sino que también permitió que las consultas, análisis y funcionalidades asociadas se desarrollasen con la máxima fidelidad posible, evitando errores que pudiesen comprometer las rutas edificadas o la experiencia del usuario en ambos programas.

```
Id_Edificio     Nombre_Edificio
2       Edificio Ingenierías
Id_Edificio     Nombre_Edificio
3       Auditorio Guillermo y Sofía Jenkins
```

Figura 10. Consulta 1 y 2. Ejecución de consultas por nombre (donde sean igual a Ingenierías y posteriormente al auditorio).

```
Id_Caminos      Nombre_Caminos  Distancia
2       Humanidades-Auditorio   300
3       Ingeniería-Medicina     600
4       Auditorio-Cafetería     300
6       Negocios-Medicina       300
Id_Caminos      Nombre_Caminos  Distancia
4       Auditorio-Cafetería     300
```

Figura 11. Consulta 3 y 4. Búsqueda de caminos que excedan la distancia promedio, por otra parte, que tengan id=4.

Ingenierías – Cafetería	800	1300
Ingenierías – Medicina	600	1400
Auditorio – Negocios	600	1400
Auditorio – Biblioteca	500	1500
Auditorio – Cafetería	300	1700
Auditorio – Medicina	900	1100
Negocios – Biblioteca	100	1900
Negocios – Cafetería	300	1700
Negocios – Medicina	300	1700
Biblioteca – Cafetería	200	1800
Biblioteca – Medicina	400	1600
Cafetería – Medicina	600	1400

Figura 12. Consulta 5. Tabla inner join, para comparar el mejor y el peor caso respecto a la distancia.

Id_Caminos	Nombre_Caminos	Distancia
2	Humanidades-Auditorio	300
4	Auditorio-Cafetería	300
6	Negocios-Medicina	300
3	Ingeniería-Medicina	600

Figura 13. Consulta 6. Búsqueda de los caminos que excedan 250 de distancia, ordenándolos y mostrándolos de manera ascendente respecto a la distancia.

Id_Edificio	Nombre_Edificio
8	Unicaja

Id_Edificio	Nombre_Edificio
8	Hacienda

Figura 14. Consulta 7 a 10. Inserción de un nuevo edificio (Unicaja), recuperación de su información, posteriormente se actualiza (Hacienda) y se recuperan sus datos.

Conclusiones

Tomando en consideración lo explayado a lo largo del texto, podemos declarar que los programas codificados, compilados y ejecutados tanto en JAVA como en SQL fueron un rotundo éxito; esto, al poder ofrecer al usuario una interfaz funcional en conjunto con una base de datos relacional que permiten determinar y desplegar de manera eficiente la ruta más corta entre un punto A y un punto B dentro del campus de la Universidad de las Américas Puebla (UDLAP). De igual forma, conseguimos consolidar los conceptos abordados durante ambos cursos haciendo uso de ideas y herramientas tanto de las Matemáticas Discretas como de las Bases de Datos; logrando así, una sinergia entre distintas disciplinas que enriqueció la experiencia de nuestro aprendizaje. Específicamente, el uso de la Teoría de Grafos posibilitó la determinación de la ubicación de los edificios (nodos) y el modelado de las rutas (aristas) entre estos, lo que se tradujo en la optimización de los trayectos posibles a través del algoritmo Dijkstra. Similarmente, la aplicación de los diagramas, la codificación y las consultas en las bases de datos resultó esencial para almacenar y recuperar eficientemente la información de rutas, distancias y ubicaciones del sistema de navegación en cuestión.

En suma, el marco teórico ofreció una base estructurada para comprender los principios de los grafos, bases de datos y algoritmos de búsqueda de rutas, y la experimentación digital permitió llevar estos conceptos a la práctica. La integración de ambos factores no solo posibilitó una solución funcional, sino que también evidenció la aplicabilidad real de los conocimientos adquiridos.

Referencias

Hernández Torres, J., N. (2025). Clase 16 – Teoría de Grafos. UDLAP Departamento de Computación, Electrónica y Mecatrónica. Recuperado el 29 de abril de 2025 de https://click.udlap.mx/ultra/courses/_46095_1/outline/edit/document/_4038923_1?courseId=_46095_1&view=content&state=view

Hernández Torres, J., N. (2025). Clase 01 - Introducción a las BD. UDLAP Departamento de Computación, Electrónica y Mecatrónica. Recuperado el 1 de mayo de 2025 de https://click.udlap.mx/ultra/courses/_45682_1/outline/edit/document/_4175287_1?courseId=_45682_1&view=content&state=view