El presente trabajo tiene como propósito la implementación del algoritmo de Dijkstra para la búsqueda de rutas mínimas en un grafo que representa puntos de referencia dentro del campus Tec Guadalajara. La propuesta incluye además una visualización animada que permite observar de manera dinámica el proceso de exploración de nodos y la construcción del camino óptimo, lo cual facilita la comprensión del funcionamiento del algoritmo y su aplicación en un entorno real.

El algoritmo de Dijkstra es un método ampliamente utilizado para determinar el camino más corto en grafos ponderados con pesos no negativos. Su funcionamiento se basa en la inicialización de todas las distancias como infinitas, excepto la del nodo origen que se establece en cero, a partir de este punto se selecciona el nodo con la menor distancia acumulada y se marca como visitado, actualizando las distancias de sus vecinos si se encuentra un recorrido más corto. Este proceso se repite hasta que todos los nodos han sido visitados o se ha encontrado la ruta mínima hacia el destino y el resultado es un conjunto de caminos óptimos desde el nodo inicial hacia todos los demás nodos del grafo.

La justificación del modelo implementado se encuentra en la combinación de estructuras eficientes y herramientas de visualización. El grafo se construyó en Python utilizando diccionarios y listas, lo que permite almacenar las conexiones y sus pesos de manera clara y flexible. Para la selección del nodo con menor distancia acumulada se empleó una cola de prioridad mediante la librería heapq, que optimiza el proceso de búsqueda y garantiza la eficiencia del algoritmo. Finalmente, se incorporó una visualización con networkx y matplotlib.animation, que muestra paso a paso los nodos visitados y resalta el camino óptimo en color rojo, aportando un valor pedagógico y práctico al modelo. Este enfoque resulta adecuado porque representa fielmente un problema de rutas en un entorno real, es escalable a grafos de mayor tamaño y facilita la comprensión del proceso gracias a la animación.

En cuanto a la complejidad del proceso de asignación, el algoritmo de Dijkstra implementado con una cola de prioridad presenta una complejidad temporal de $O((V+E)\log V)$, donde V corresponde al número de nodos y E al número de aristas. Cada extracción del nodo mínimo y cada actualización de distancia se realizan en tiempo logarítmico, lo que asegura un rendimiento eficiente incluso en grafos grandes. La complejidad espacial es de $O(V+E)$, ya que se requiere almacenar las distancias, los nodos previos y la estructura completa del grafo. En el caso particular del grafo del campus, que cuenta con un número reducido de nodos y aristas, el algoritmo se ejecuta de manera prácticamente instantánea.