DSA Project Presentation

Group Members:
Pranshu Desai (B24CS1057)
Shardul Diwate (B24CS1028)
Vansh Rathi (B24CM1065)
Katravath Kurumurthy (B24CM1035)

Campus Navigation System

Goal: Find the minimum travelling distances between two points on campus.

01. The campus is modeled as a weighted undirected graph

02. Nodes represent buildings/locations.

03. Edges represent paths with distance weights.

DSA Concepts Used:

- Graph Representation : Adjacency List
- Priority Queue (Min-Heap): To select the nearest unvisited node.
- Hash Map: To store distances and parent relationships.
- Dijkstra’s Algorithm: To find the shortest path efficiently.
- Time Complexity: O(E log V).

System Design

01. The campus is represented as a weighted graph with nodes and edges.

02. Each node corresponds to a campus building or landmark.

03. Edges connect nodes with weights representing distances in meters.

04. The user inputs a source and destination location.

05. Dijkstra’s algorithm computes the shortest path and minimum distance.

06. The result (path and distance) is displayed textually and visually on the graph.

Dijkstra’s Algorithm

01

Start from the source node with distance = 0.

02

Visit the unvisited node with the smallest distance.

03

Update its neighbors’ distances.

04

Repeat until the destination is reached.

Code Implementation:

Core Functions:

- addEdge(u, v, w) - Creates a bidirectional edge.
- dijkstra(source, destination) - Computes the shortest path.

STL Used:

- unordered_map
- vector
- priority_queue

Code Snippet:

```
while (!pq.empty()) {
    auto [d, node] = pq.top(); pq.pop();
    for (auto &nbr : adj[node]) {
        if (dist[node] + nbr.second < dist[nbr.first]) {
            dist[nbr.first] = dist[node] + nbr.second;
            parent[nbr.first] = node;
            pq.push({dist[nbr.first], nbr.first});
        }
    }
}
```

Conclusion and Future Scope

Conclusion:

- Implemented a working Campus Navigation System using Dijkstra's Algorithm.
- Demonstrated usage of graphs, priority queues, and adjacency lists.
- Achieved efficient shortest route computation.

Future Scope:

- Add GUI using real IITJ map.
- Integrate GPS for real-time navigation.
- Implement A* Algorithm for optimization.

Contributions:

1. Pranshu Desai – Graph Design & C++ Structure

Contributions:

Designed the overall graph structure of the campus.

Identified all nodes (buildings) and edges (paths) with accurate weights.

Implemented the initial Graph class in C++.

Wrote the addEdge() function and set up the adjacency list.

Helped in debugging C++ compile-time issues.

2. Shardul Diwate – Dijkstra Algorithm & C++ Implementation

Contributions:

Implemented the complete Dijkstra's Algorithm in C++.

Worked on the priority queue, distance map, and parent map logic.

Wrote the code for reconstructing and printing the shortest path.

Optimized the algorithm to ensure correct output and no infinite loops.

Assisted in terminal-based testing and edge-case checking.

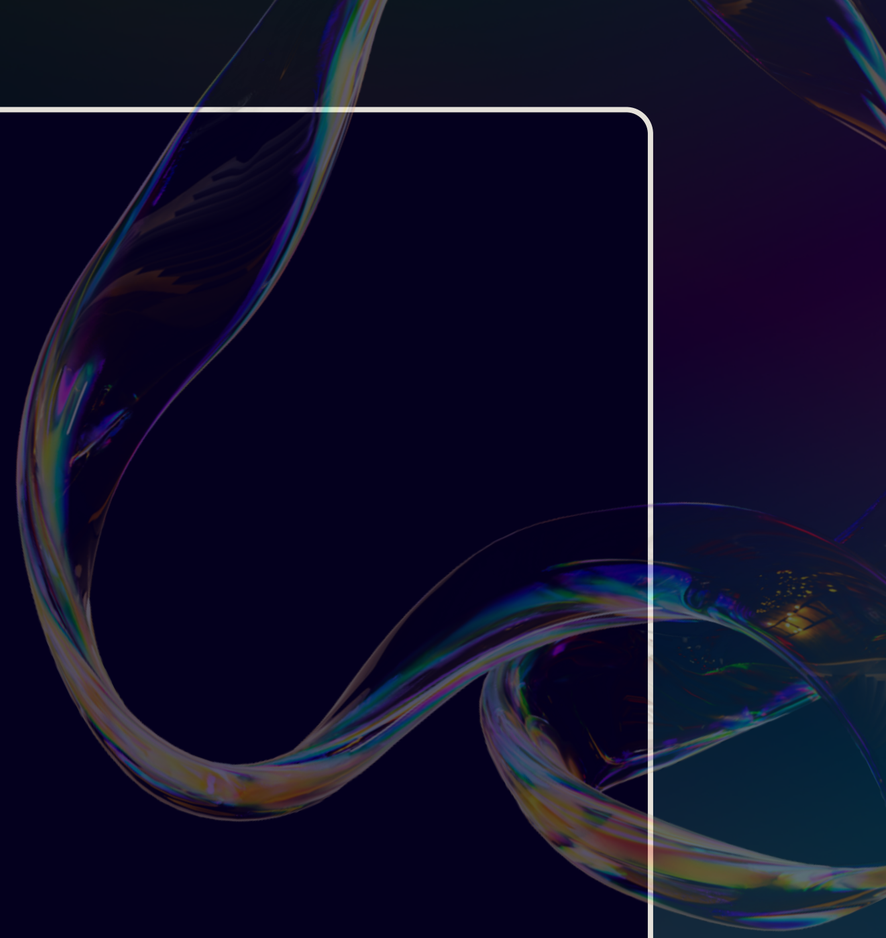
Contributions:

1. Pranshu Desai – Graph Design & C++ Structure

Designed the overall graph structure of the campus.
Identified all nodes (buildings) and edges (paths) with accurate weights.
Implemented the initial Graph class in C++.
Wrote the addEdge() function and set up the adjacency list.
Helped in debugging C++ compile-time issues.

2. Shardul Diwate – Dijkstra Algorithm & C++ Implementation

Implemented the complete Dijkstra's Algorithm in C++.
Worked on the priority queue, distance map, and parent map logic.
Wrote the code for reconstructing and printing the shortest path.
Optimized the algorithm to ensure correct output and no infinite loops.
Assisted in terminal-based testing and edge-case checking.



Contributions:

3. Vansh Rathi – Python Visualization & Map Layout

Built the Python version of the system using NetworkX and Matplotlib.
Implemented the dijkstra_path and plotted the graph layout.
Designed the coordinate positions for each building in the campus map.
Highlighted shortest path in red and ensured proper labeling.
Managed user input and cleaned the visual output.

4. Katravath Kurmurthy – Testing, Demo, and Presentation

Tested both C++ and Python codes with multiple location inputs.
Verified shortest paths manually to ensure correct results.
Created the PowerPoint presentation with system workflow diagrams.
Prepared the final demo sequence and handled formatting.
Wrote future-scope ideas and coordinated the explanation flow.