1. Introduction

This system is a Python-based decision support tool that determines optimal evacuation routes from various barangays in Leon, Iloilo to the central evacuation center, Poblacion. It applies fuzzy logic to evaluate road segment attributes such as slope, travel time, and curvature, while explicitly excluding a predefined set of blocked paths. The A* pathfinding algorithm is used to select the safest and most efficient evacuation routes based on these costs.

2. Barangays Covered

- Poblacion (Evacuation Center)
- Bobon
- Gines
- Barangbang
- Carolina
- Lonoc
- Bacolod
- Binolbog
- Ingay

3. System Architecture

3.1 Presentation Layer

- Built using Streamlit for interactive user interface.
- Displays barangay locations and evacuation routes on an Esri satellite-based Folium map.
- Highlights best evacuation routes, blocked paths, and alternative routes distinctly.
- Shows key metrics such as travel distance and estimated travel time.

3.2 Logic Layer

- Encapsulated in the EvacuationSystem class.
- Loads and parses KML files for route geometries.
- Reads Excel files for travel segment attributes: slope, travel time, and curvature.
- Manages a static graph of barangay-to-barangay connections.
- Defines a set of predefined blocked paths that are excluded from routing.

- Implements fuzzy cost evaluation for road segments.
- Executes the A* algorithm to find the minimal fuzzy cost path that avoids blocked roads.

3.3 Data Layer

- Route geometry stored in KML files.
- Travel data provided in Excel files per barangay route.
- Predefined blocked routes encoded in the system as a set for easy exclusion.

4. Key Components and Methods

4.1 Graph and Connections

_build_graph_connections() returns a dictionary defining adjacency ("graph") between barangays. Blocked paths are encoded as strings (e.g., "Poblacion to Bobon") in the set self.blocked_paths. Blocked edges are never considered in pathfinding.

4.2 Fuzzy Cost Evaluation

fuzzy_evaluation(slope, travel_time, curvature) applies fuzzy membership functions and rules to compute a cost value for each road segment incorporating steepness, travel duration, and curvature severity.

4.3 A* Pathfinding Algorithm

The astar_path(start, goal) method performs the following:

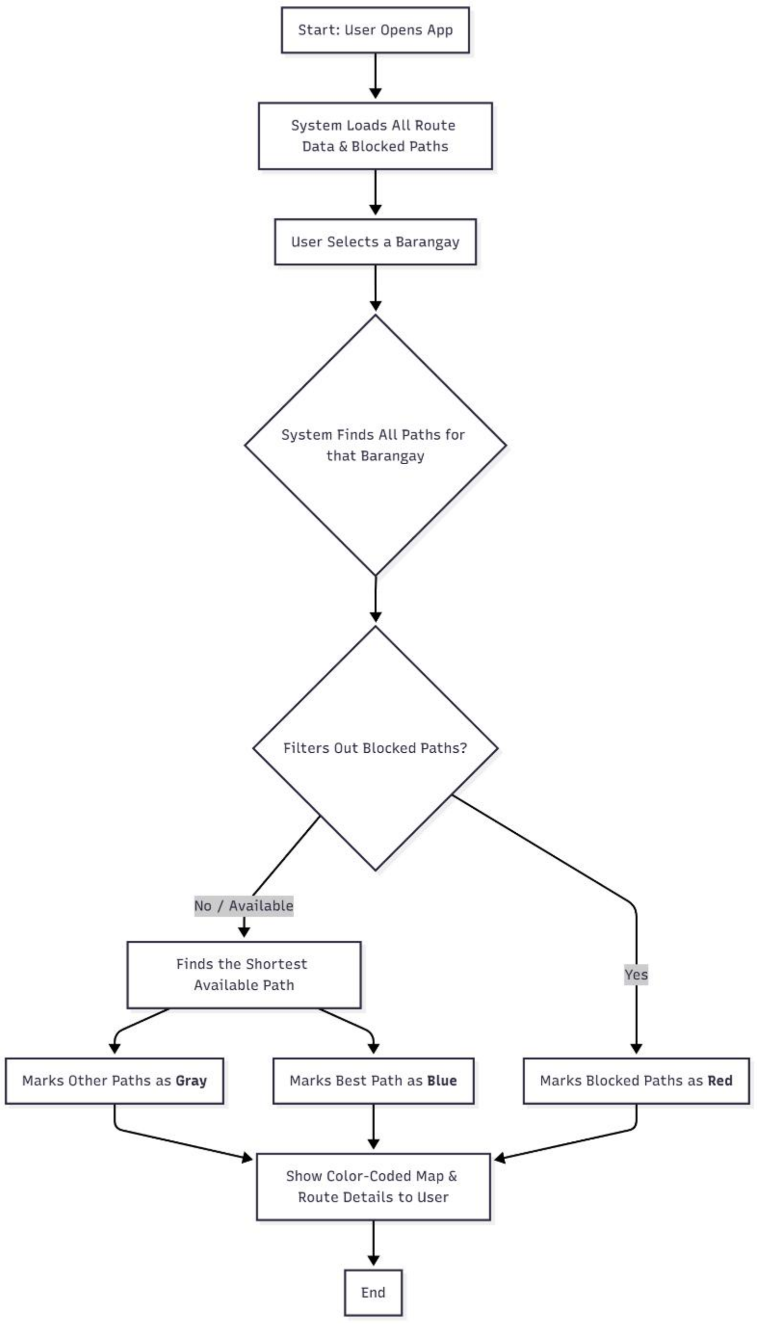
- Initializes open and closed sets for nodes pending exploration or already evaluated.
- Uses priority queue (heapq) ordered by f_score = g_score + h_score.
- Tracks:
    - g_score: actual fuzzy cost from start to current node.
    - h_score: heuristic estimate from current node to goal node, derived by heuristic() function calculating Haversine distance.
- Iteratively selects the node with minimum f_score from open set.
- For each neighbor:
    - Skips edges if the connecting path is blocked.
    - Computes tentative g_score by adding fuzzy cost from get_edge_cost.
    - Updates scores and predecessors if the tentative score is better.
- Returns the reconstructed optimal path on reaching the goal or None if no viable path exists.

4.4 Map Rendering

- Displays all barangays on the map.
- Colors blocked paths red, other paths gray, and the A* determined best path blue.
- Allows user to select a barangay and see the safe evacuation path from Poblacion.

5. System Workflow

1. Initialization:
    - Load route KML and travel attribute Excel files.
    - Build barangay adjacency graph.
    - Encode predefined blocked paths.
2. User Input:
    - Select barangay destination for evacuation.
3. Pathfinding:
    - A* algorithm finds the lowest-cost path avoiding blocked roads by summing fuzzy costs along edges.
    - Heuristic guides search using geographic distances.
4. Visualization:
    - Render map with all routes visually distinguished.
    - Provide distance and travel time metrics for the selected path.
5. Result:
    - User gets a safe, cost-effective evacuation route tailored to local road conditions and blockages.


Start: User Opens App
System Loads All Route Data & Blocked Paths
User Selects a Barangay
System Finds All Paths for that Barangay
Filters Out Blocked Paths?
No / Available
Yes
Finds the Shortest Available Path
Marks Other Paths as Gray
Marks Best Path as Blue
Marks Blocked Paths as Red
Show Color-Coded Map & Route Details to User
End

This is a Python script for a web application called SAFE Route (Smart Alert and Fast Evacuation with Rerouting Technology).

It's built using Streamlit to create the user interface. Its main job is to show evacuation routes on an interactive map.

Here's a simple description of what it does:

1. Loads Data: When it starts, the app loads all evacuation path coordinates from a .kml file and loads road data (like slope and travel time) from Excel files.

2. Sets Blocked Paths: The system has a hardcoded list of paths that are considered "blocked" (e.g., "Poblacion to Bobon").
3. User Selects Barangay: The user picks a barangay from a dropdown menu.
4. Finds Best Path: The system finds all available paths for that barangay. It then filters out any path on the "blocked" list. From the remaining *unblocked* paths, it selects the one with the shortest distance.
5. Displays Map: It shows a Folium map with all paths color-coded:
    - Red: Blocked paths.
    - Blue: The best available (shortest unblocked) path.
    - Gray: Other available paths that are not blocked but are not the shortest.
6. Shows Details: It also displays the distance (km) and estimated travel time (min) for the recommended Blue path.