# WalkSafe: A Safety Aware Pedestrian Routing Prototype for Urban Walking in Pune

Hitesh Patil[1], Mansi Kale[2], Samruddhi Gawade[3], Amey Chatane[4], Shraddha Gunjal[5], Amit Raheja[6]
[1,2,3,4,5,6]*AISSMS IOIT*

***Abstract*—Traditional pedestrian navigation solutions mostly prioritize either minimizing distance or optimizing travel time and consequently do not incorporate factors that could influence pedestrian safety perception. WalkSafe serves as an end-to-end prototype implementing safety sensitive scoring into pedestrian navigation planning for Pune, India. The project integrates offline processing of streets using OpenStreetMap data, heuristic scoring of walkable road segments, H3 spatial aggregation of neighborhoods' safety zones, a FastAPI server, a local installation of the Valhalla routing engine, and a Flutter based mobile client. Within the implemented framework, static safety scores of road segments include road classification, lighting, sidewalk presence, and penalties related to road speeds, whereas incidents reported by users are considered dynamically while computing routes in the backend. The mobile app offers features such as route visualization with safety scoring, heading sensitive navigation, route rerouting, incident reporting, and an emergency response workflow with trusted contact information.**

## I. INTRODUCTION

Pedestrian navigation is usually framed as an optimization problem over distance, time, or convenience. For many users, however, the best route is not always the shortest route. Streets with poor lighting, missing sidewalks, highspeed traffic, or recent reports of unsafe activity may be less desirable even when they reduce travel time. In dense urban settings, this mismatch between shortest path navigation and perceived safety can materially affect route choice.

WalkSafe was designed as a prototype navigation system that brings safety-oriented signals into the pedestrian routing workflow. The project focuses on Pune, India, and combines open geospatial data, a route serving backend, and a mobile client that exposes safety information directly during route planning and navigation. Rather than treating safety as a purely learned prediction problem, the current system adopts a transparent heuristic model built from street level attributes and moderated incident reports.

The contribution of this work is primarily systems oriented. WalkSafe integrates:

- offline extraction of a walkable urban street network from OpenStreetMap,
- static segment level safety scoring from map derived attributes,
- H3based aggregation of safety zones for map overlay and contextual awareness,
- backend route evaluation that combines Valhalla routing with dynamic incident aware penalties,
- a Flutter mobile interface for route planning, navigation, incident reporting, and emergency support.

## II. SYSTEM OVERVIEW

WalkSafe is organized as a three-layer architecture: an offline spatial data pipeline, an online backend service layer, and a mobile client.

### 2.1 Offline Data Preparation

The offline pipeline downloads Pune's walkable street graph from OpenStreetMap using OS Mnx. Each road segment is assigned a static safety score based on mapped infrastructure and road characteristics. These scored segments are stored in Post GIS. Segment centroids are then aggregated into H3 cells at resolution 9 in order to construct polygon-based safety zones suitable for interactive map display.

### 2.2 Backend Service Layer

The backend is implemented in FastAPI and uses a

selfhosted Valhalla instance for pedestrian route generation. Valhalla returns a feasible walking route between an origin and a destination. The backend then evaluates the returned route against the locally stored safety dataset, matches route geometry to cached road segments, and applies penalties derived from verified incident reports. The result is a route response enriched with safety metadata rather than a purely geometric path.

2.3 Mobile Client

The mobile application is implemented in Flutter. It provides destination search, route display, zone overlays, stepwise navigation feedback, rerouting, incident reporting, and an emergency workflow for trusted contacts. Map rendering uses raster tiles and the interface is designed to present safety context without obscuring the primary navigation task.

## III. SAFETY SCORING MODEL

WalkSafe uses a heuristic segment level safety model. The current implementation separates static road scoring from dynamic incident effects.

3.1 Static Segment Score

For each walkable road segment, the system computes a base score using OpenStreetMap attributes. The implemented static model follows the form

$$S_{static} = base(r) + b_{light} + b_{sidewalk} - p_{speed}, \quad (1)$$

where base(r) depends on road type, $b_{light}$ is a lighting bonus, $b_{sidewalk}$ is a sidewalk bonus, and $p_{speed}$ is a speed related penalty. The final value is clamped to the interval [0, 100].

Representative base values in the implemented prototype assign higher scores to walking friendly segments such as footway and pedestrian roads, moderate scores to residential and tertiary roads, and low scores to primary, trunk, and motorway segments. Lighting contributes a positive bonus when the road is explicitly tagged as lit. Sidewalk availability also contributes a positive bonus. Speed penalties increase above a nominal pedestrian friendly threshold.

3.2 Dynamic Incident Penalty

Incident reports are not baked into the static road dataset during offline ingest. Instead, they are applied dynamically during backend route evaluation after moderation. For a matched route segment, the adjusted score is longer unsafe segments to influence the final route assessment more strongly than very short segments.

## IV. SPATIAL DATA PIPELINE

4.1 Street Network Ingest

The spatial data pipeline begins by extracting Pune's walk graph using OSMnx. The resulting network contains the geometry and tags needed for segment scoring, including road class, lighting information where present, and sidewalk related annotations.

4.2 PostGIS Storage

Scored road segments are written to a PostgreSQL database with PostGIS enabled. Segment geometries are stored as LINESTRING features. Safety zones are stored as polygon geometries. Spatial indexing through PostGIS supports persistent storage and efficient querying.

4.3 H3 Safety Zones

In addition to segment level storage, WalkSafe aggregates segment scores into H3 hexagons at resolution 9, which corresponds to an urban scale cell size suitable for pedestrian awareness. The average score within each populated cell is converted into a categorical zone label such as safe, cautious, or risky. These polygon zones are served to the mobile client as map overlays.

## V. BACKEND ROUTING AND SAFETY EVALUATION

The backend combines external routing with local safety evaluation.

5.1 Valhalla Route Generation

For a requested trip, the server submits origin and destination coordinates to Valhalla in pedestrian mode. The request uses pedestrian oriented routing options and returns route geometry, estimated length, and travel time.

$$S_{final} = \max(0, S_{static} - P_{incident}). \quad (2)$$

The incident penalty depends on a weighted incident score derived from report confidence, category, and recency, and is capped so that a small number of reports cannot produce arbitrarily large penalties. Only verified incidents contribute to route time penalties.

## VI. ROUTE LEVEL SAFETY

Once segment level adjusted scores are available, the route level safety value is computed as a length weighted average across matched route segments. This allows

### 6.1 Geometry Matching and Cache

To support responsive scoring, the backend maintains an in-memory cache of the latest road segment and safety zone datasets. Shapely STR tree indexes are constructed over cached geometries. During route evaluation, consecutive route coordinates are interpreted as route segments, and each route segment is represented by a midpoint. The midpoint is matched to the nearest cached road segment within a defined tolerance, and it is also associated with the corresponding safety zone when available.

### 6.2 Dynamic Incident Integration

After route segments are matched to stored roads, the backend queries verified incident reports from the database and aggregates them by nearby matched road segment. These aggregates are then converted into dynamic penalties that reduce the segment's static safety score. This design keeps the base safety dataset stable while still allowing recently verified incidents to influence route evaluation.

### 6.3 Design Note

The present prototype evaluates safety after Valhalla returns a pedestrian route. What this means is that instead of incorporating the safety cost directly into Valhalla's graph search, the current system is simply scoring the routes after the search is finished using a backend process. The upside of this method is that it avoids having to modify the Valhalla codebase in any way.

## VII. MOBILE APPLICATION DESIGN

### 7.1 Route Planning and Visualization

The Flutter client requests routes from the backend, renders the returned path, and overlays contextual safety information on the map. The interface also presents route safety summaries so that users can assess the route before or during travel.

### 7.2 Safety Zone Overlay

Safety zones are loaded from the backend as h3 polygon features and cached on the device to reduce repeat fetches. This allows the user to view proximity-based safety context even when cellphone network conditions are inconsistent.

### 7.3 Navigation Behavior

The mobile client maintains a navigation system that tracks user location, route progress, and camera behavior. WalkSafe application supports:

- automatic map following by the viewport during active navigation,
- heading up orientation when a reliable heading is available,
- smoothing of heading updates to reduce visual jitter and improve upon experience,
- route progress update as the user advances,
- arrival detection near the destination with a defined threshold,
- rerouting when users go off route

Heading smoothing is implemented using an exponential moving average so that the displayed direction reacts to user movement without producing abrupt oscillations.

### 7.4 Search and Basemap Services

Destination lookup is performed through geocoding requests scoped to Pune. The map view uses tiled cartography appropriate for mobile route display. These supporting services are integrated into the client but remain logically distinct from the safety scoring pipeline.

## VIII. INCIDENT REPORTING AND EMERGENCY SUPPORT

### 8.1 Incident Reporting Workflow

The users can report any incident that relates to factors such as insufficient lighting, bullying, suspicious behavior, or unsafe infrastructure. The incidents will go into the database but will not be permitted to directly affect route penalties until the reports have been moderated. Any reports that pass verification will then be included in the route penalty calculation. This moderation first design is intended to reduce the effect of unreviewed or low-quality submissions on navigation behavior.

8.2 Emergency Workflow

WalkSafe also has an emergency support feature centered on trusted contacts. In the current prototype, the active emergency workflow is implemented on the mobile device and sends SMS messages with live location information to stored trusted contacts. The backend can record emergency alert events, but the notification path itself is device local in the current system design to minimize operation costs.

## IX. IMPLEMENTATION SUMMARY

Table 1 summarizes the principal technologies used in the prototype.

Table 1: Principal technologies used in WalkSafe.

| Layer | Technology |
|---|---|
| Mobile client | Flutter |
| Backend API | FastAPI |
| Routing engine | Valhalla |
| Persistent storage | PostgreSQL with PostGIS |
| Spatial caching | Shapely STRtree |
| Hexagonal aggregation | H3 |
| Street network in gest | OSMnx |
| Map data source | OpenStreetMap |
| Database migration | Alembic |

## X. LIMITATIONS AND FUTURE WORK

WalkSafe is an implemented prototype, but several limitations remain.

First, the safety model is heuristic. Its transparency is useful, but it is not yet calibrated against behavioral studies, recorded incidents, or user preference data. Second, the quality of the static score depends on the completeness of OpenStreetMap tags; missing lighting or sidewalk information can weaken the realism of the score. Third, the current routing design evaluates safety after route generation rather than performing a fully integrated safety aware shortest path search inside the routing engine. Fourth, incident reporting would benefit from stronger anti abuse controls, including server-side authentication, validation, and rate limiting. Finally, the system has not yet been evaluated through a formal user study or largescale field deployment.

Future work can therefore proceed in several directions:

- integrating richer contextual signals such as time of day, weather, or traffic intensity,
- replacing or complementing heuristic scoring with learned safety estimators,
- implementing stronger authenticated reporting and abuse prevention mechanisms,
- exploring tighter integration between the safety model and the routing engine,
- conducting user studies on route trust, usability, and acceptable detour tradeoffs,
- extending the pipeline beyond a single city deployment.

## XI. CONCLUSION

The WalkSafe framework serves as a practical prototype for pedestrians navigating safely through an urban area. WalkSafe makes use of roads derived from OSM, heuristic score evaluation for segments, H3 zoning aggregation, storing data using PostGIS, path calculation using Valhalla, backend evaluation of paths, and frontend mobile app implementation using Flutter technology. At present, the framework used by WalkSafe deliberately aims to be pragmatic, since the safety factors in each segment of the walkable road network are already calculated offline, while verification of any incident is done dynamically online when evaluating the paths in the backend. Despite WalkSafe being just a prototype and not yet being a fully validated solution for public safety, it is a noteworthy point of exploration in the field.

## REFERENCES

[1] Boeing, G., “OSMnx: New Methods for Acquiring, Constructing, Analyzing, and Visualizing Complex Street Networks.” Available: https://osmnx.readthedocs.io/
[2] Uber, “H3 Hexagonal Hierarchical Geospatial Indexing System.” Available: https://h3geo.org/
[3] Valhalla Project, “Valhalla Routing Engine Documentation.” Available: https://valhalla.github.io/valhalla/
[4] PostGIS Project, “PostGIS Documentation.” Available: https://postgis.net/
[5] Shapely Project, “Shapely STRtree Documentation.” Available:

https://shapely.readthedocs.io/
[6] Ramirez, S., "FastAPI Documentation." Available: https://fastapi.tiangolo.com/
[7] Google, "Flutter Documentation." Available: https://flutter.dev/
[8] OpenStreetMap Contributors, "OpenStreetMap." Available: https://www.openstreetmap.org/