LOCATION INTELLIGENCE ALGORITHM NETWORK

Submitted in partial fulfillment of the requirements of the degree

BACHELOR OF ENGINEERING IN COMPUTER ENGINEERING

By

TADVI NAQUEEB AALAM 221250
UBAIDUR RAHEMAN ANSARI 221252
SAAD ASHIQUE ALI MOMIN 221233

Name of the Mentor

Prof. Anand Bali

Department of Computer Engineering

M H Saboo Siddik College of Engineering

Mumbai

University of Mumbai

(AY 2024-25)

CERTIFICATE

This is to certify that the Mini Project entitled **"LOCATION INTELLIGENCE ALGORITHM NETWORK"** is a bonafide work of TADVI NAQUEEB AALAM ABDUL SHAKIR (221250) ,UBAIDUR RAHEMAN ANSARI (221252)and SAAD ASHIQUE ALI MOMIN (221233) Submitted to the University of Mumbai in partial fulfillment of the requirement for the award of the degree of **"Bachelor of Engineering"** in **"Computer Engineering" .**

(Prof. Anand Bali)
Mentor

(DR MOHD AHMED)
Head of Department

(DR Ganesh Kame)
Principal

Mini Project Approval

This Mini Project entitled "**LOCAL INTELLIGENCE ALGORITHM NETWORK"** by TADVI NAQUEEB AALAM ABDUL SHAKIR (221250), ANSARI UBAIDUR RAHEMAN (221252), and SAAD ASHIQUE ALI MOMIN (221234) is approved for the degree of **Bachelor of Engineering** in **Computer Engineering.**

Examiners

1.Prof. Anand Bali………...........
(Internal Examiner Name & Sign)

2…………………………………………
(External Examiner name & Sign)

Date:

Place:

Contents

1 **Introduction** **7**

1.1 Introduction

1.2 Motivation

1.3 Problem Statement

1.4 Organization of the Report

2 **Literature Survey** **9**

2.1 Survey of Existing System/**SRS**

2.2 Limitation Existing system or Research gap

2.3 Mini Project Contribution

3 **Proposed System (e.g. New Approach of Data Summarization)** **14**

3.1 Introduction

3.2 Architecture/ Framework

3.3 Algorithm and Process Design

3.4 Details of Hardware & Software

3.4 Experiment and Results for Validation and Verification

3.5 Analysis

3.6 Conclusion and Future Work.

References **24**

Annexure

4.1 **Published Paper /Camera Ready Paper/ Business pitch/proof of concept**

Abstract

The Location Intelligence Algorithm Network (L.I.A.N) is a web-based platform designed to visualize and analyze pathfinding algorithms within real-world city environments. The project focuses on enhancing the understanding of various pathfinding techniques, including Dijkstra's, A*, and Greedy Best-First Search, by applying them to real-world geospatial data. Unlike traditional pathfinding visualizations, which rely on abstract grids or pre-defined maps, L.I.A.N integrates dynamic maps based on real cities, allowing users to explore algorithm performance under realistic conditions.

Users interact with L.I.A.N by selecting two locations on a map, which are treated as the source and destination. The platform provides a list of pathfinding algorithms for the user to choose from. Upon selection, the chosen algorithm calculates the optimal or heuristic-based path between the two points, which is then visualized on the map. The visualization shows not only the final path but also the step-by-step decision-making process, illustrating the algorithm's inner workings, including node exploration, cost analysis, and heuristic evaluations. This allows users to observe the computational complexities and trade-offs in terms of time and efficiency in different environments, such as city grids, irregular streets, or areas with obstacles.

The real-world applicability of L.I.A.N makes it highly relevant for a variety of domains, including smart city planning, navigation systems, and urban mobility research. By providing an accessible and interactive way to visualize algorithmic performance, L.I.A.N also serves as an educational tool, ideal for both students and professionals who wish to deepen their understanding of algorithms in practical scenarios.

Keywords:
Pathfinding algorithms, Dijkstra’s algorithm, A* algorithm, geospatial visualization, urban mobility, real-world maps, algorithmic efficiency, and smart cities.

Acknowledgments

We wish to express our sincere thanks to our director **Dr. Mohiuddin Ahmed** and our I/c principal **ER. Ganesh Kame**, M.H. Saboo Siddik College of Engineering for providing us with all the facilities, support, and wonderful environment to meet our project requirements. We would also take the opportunity to express our humble gratitude to our Head of the Department of Computer Engineering **Dr. Mohd Ahmed** for supporting us in all aspects and for encouraging us with her valuable suggestions to make our project a success. We are highly thankful for our internal project guide **Prof. Anand Bali** whose valuable guidance helped us understand the project better, her constant guidance and willingness to share her vast knowledge made us understand this project and its manifestations in great depth and helped us to complete the project successfully. We would also like to acknowledge with much appreciation the role of the staff of the Computer Department, especially the Laboratory staff, who gave the permission to use the labs when needed and the necessary material to complete the project. We would like to express our gratitude and appreciate the guidance given by other supervisors and project guides, their comments and tips helped us in improving our presentation skills. Although there may be many who remain unacknowledged in this humble note of appreciation, there are none who remain unappreciated.

Chapter 1

Introduction

1.1 Introduction

Pathfinding algorithms have been central to the development of numerous applications across various fields such as robotics, transportation, logistics, gaming, and urban planning. These algorithms, such as Dijkstra's, A*, and Greedy Best-First Search, are designed to calculate the most efficient route from one point to another based on factors like distance, cost, or time. Traditionally, these algorithms are visualized in idealized environments like grids or mazes, which, although useful for understanding algorithm behavior, do not accurately reflect the complexities of real-world situations.

Real-world environments, especially urban areas, introduce unique challenges for pathfinding. City streets are not perfectly aligned grids; they are irregular, with obstacles, one-way routes, and varying distances between nodes. These variations make urban pathfinding both more complex and more practical to study for applications like autonomous vehicles, delivery drones, and intelligent navigation systems.

The **Location Intelligence Algorithm Network (L.I.A.N)** addresses this gap by offering a platform that visualizes pathfinding algorithms on real-world maps, specifically in urban settings. The primary goal of L.I.A.N is to help users understand how various algorithms perform in real-world environments and to allow for comparisons based on factors such as computational efficiency, path accuracy, and adaptability to urban constraints.

1.2 Motivation

The rapid development of smart cities, autonomous vehicles, and intelligent transportation systems has made pathfinding a critical area of research. Accurate and efficient pathfinding is necessary for minimizing travel times, reducing congestion, and ensuring the optimal use of resources in urban environments. The ability to visualize and understand how different pathfinding algorithms behave in realistic cityscapes can have significant implications for improving these systems.

Current tools for visualizing pathfinding algorithms are mostly confined to abstract, grid-based layouts, which limit their applicability to real-world problems. These tools do not consider the irregularities and constraints posed by urban environments, such as uneven road networks, obstacles, and traffic rules. Furthermore, they often fail to offer insights into the strengths and weaknesses of each algorithm in different contexts.

L.I.A.N was developed to overcome these limitations by providing an interactive platform where users can simulate and visualize the behavior of different pathfinding algorithms on real-world city maps. This platform offers a unique opportunity to observe how these algorithms operate in complex environments, providing practical insights that can be used to enhance navigation systems, urban planning, and AI-driven mobility solutions.

1.3 Problem Statement

Pathfinding algorithms are widely studied and used in various domains, but a major gap is the lack of tools to visualize their performance in real-world, dynamic environments. The existing tools that visualize these algorithms typically operate on abstract grids or predefined, simplistic environments, which do not reflect the complexities of real-world urban landscapes.

The **Location Intelligence Algorithm Network (L.I.A.N)** aims to address this gap by providing a platform where users can:

- Visualize pathfinding algorithms on real-world maps of cities.
- Compare the performance of algorithms like Dijkstra's, A*, and Greedy Best-First Search in urban settings.
- Gain insight into how these algorithms deal with real-world constraints such as irregular street layouts, obstacles, and node distances.

By offering these functionalities, L.I.A.N allows users to evaluate the practical effectiveness of each algorithm in solving urban pathfinding problems, which is vital for applications in smart cities, autonomous driving, and urban logistics.

1.4 Organization of the Report

This report is organized as follows:

- **Chapter 2: Literature Review**
  This chapter provides an overview of the key pathfinding algorithms, such as Dijkstra's and A*, as well as heuristic approaches. It also reviews existing tools for visualizing these algorithms and explores the limitations of current visualization methods when applied to real-world scenarios.
- **Chapter 3: System Design and Architecture**
  This chapter outlines the architecture of the L.I.A.N platform. It covers the integration of real-world geospatial data, the choice of algorithms, and the design of the user interface, which allows for easy selection of paths and algorithms.
- **References:** Following Chapter 3, the report concludes with a list of references, citing the research papers and sources used in the Literature Review and the development of L.I.A.N.

Chapter 2

Literature Survey

2.1 Survey of Existing System/SRS:

1. Introduction

1.1 Purpose

The purpose of the **Location Intelligence Algorithm Network (LIAN)** is to provide an educational platform where users can learn how various pathfinding algorithms operate in real-world routing scenarios. The system will visualize pathfinding processes on real-world maps, offering insights into algorithmic behaviour. A secondary objective is to analyse and visualize global trade routes using AI-driven pathfinding techniques, prioritizing BRICS (Brazil, Russia, India, China, South Africa) perspectives.

1.2 Scope

LIAN is a web-based application that allows users to:

- Select different pathfinding algorithms (Dijkstra, A*, Best-First Search, Bellman-Ford).
- Visualize algorithm execution step-by-step with animations.
- Interact with real-world map data for route calculations.
- Integrate with geocoding services for start and end locations.
- Generate reports and export results in PDF format.
- Provide trade route analysis using AI techniques favouring BRICS countries.

1.3 Definitions, Acronyms, and Abbreviations

- **GIS (Geographic Information System):** A system for capturing, storing, analysing, and managing spatial data.
- **OSM (OpenStreetMap):** A collaborative project that provides free geographic data.
- **BRICS:** A political-economic alliance comprising Brazil, Russia, India, China, and South Africa.
- **Nominatim:** A geocoding service that converts addresses into coordinates.
- **Leaflet.js:** A JavaScript library for interactive maps.

1.4 References

- IEEE 830-1998 Software Requirements Specification Standard.
- OpenStreetMap (https://www.openstreetmap.org/).
- Leaflet.js Documentation (https://leafletjs.com/).
- Nominatim API Documentation (https://nominatim.org/).

2. Overall Description

2.1 Product Perspective

LIAN is an educational tool that leverages **geospatial visualization** to explain pathfinding algorithms. It integrates OpenStreetMap data, allowing users to observe algorithmic decision-making in real-world routing.

2.2 User Characteristics

- **Students & Educators:** Learn and teach pathfinding concepts.
- **Researchers & Data Scientists:** Analyze global trade routes.
- **Developers & GIS Experts:** Test and optimize algorithm performance in geospatial applications.

2.3 Operating Environment

- **Frontend:** HTML, CSS, JavaScript (Leaflet.js, React.js optional).
- **Backend:** Python (Flask/Django), MySQL/PostgreSQL.
- **External APIs:** OSMnx for road network data, Nominatim for geocoding.
- **Hosting:** Cloud-based services (AWS, GCP, or DigitalOcean).

2.4 Design and Implementation Constraints

- **Performance Limitations:** Handling large geospatial data efficiently.
- **Accuracy Constraints:** Route calculation accuracy depends on OSM data.
- **Security Measures:** User authentication for report generation and access control.

2.5 Assumptions and Dependencies

- Users will have basic familiarity with maps and algorithms.
- OSM and Nominatim APIs will be available for location services.
- Web browsers will support JavaScript-based visualization.

3. Specific Requirements

3.1 Functional Requirements

- **Algorithm Selection:** Users can choose pathfinding algorithms.
- **Route Calculation:** Compute shortest path based on selected algorithm.
- **Path Animation:** Display algorithm exploration using animated paths.
- **Geocoding Integration:** Convert user input into coordinates.
- **Exporting Results:** Save route visualizations as PDFs.
- **User Authentication:** Required for saving reports and preferences.

3.2 Non-Functional Requirements

- **Performance:** Routes computed in under 3 seconds for urban areas.
- **Scalability:** Support for high user concurrency.
- **Usability:** Intuitive UI for algorithm selection and visualization.
- **Security:** Encrypted authentication, secured API requests.

4. External Interface Requirements

4.1 User Interfaces

- **Dashboard:** Home screen with the following sections:
- **About LIAN:** Introduction to the platform and its mission.
- **Sign Up and Login:** User authentication and account access.
- **Our Goals:** Educational and analytical goals, including BRICS trade route analysis.
- **Research Papers:** Academic background and supporting literature.
- **Subscription Models:** Overview of available subscription plans.
- **LIAN Simulator:** Main interactive tool to visualize pathfinding algorithms on maps.
- **Global Trade Routes Visualizer:** AI-driven feature for analyzing international trade paths.
- **Map View:** Interactive map displaying routes.
- **Control Panel:** Start/End location input and execution control.

4.2 Hardware Interfaces

- Web-based, requiring standard browser compatibility.

4.3 Software Interfaces

- **Leaflet.js:** Map rendering and interactions.
- **OSMnx:** Fetching road network data.
- **Nominatim API:** Geocoding support.

4.4 Communication Interfaces

- HTTP RESTful APIs for data exchange.

5. System Features and Use Cases

5.1 Pathfinding Animation

- Users see nodes being explored and shortest path traced.
- Implement **leaflet-ant-path** for smooth animations.

5.2 Handling Duplicate Coordinates

- If start and end points are too close, system provides nearest accessible node.

5.3 Route Comparison

- Users can compare different algorithms' performance.

6. Appendices

6.1 Data Flow Diagram

(Include diagrams showing data exchange between user, UI, backend, and APIs.)

6.2 Sequence Diagram

(Illustrates step-by-step execution from user input to final visualization.)

6.3 Glossary

- **Graph:** A network of nodes (locations) and edges (roads).
- **Pathfinding Algorithm:** A method for finding optimal routes.
- **Geospatial Data:** Data with geographical components.

Conclusion

This SRS provides a comprehensive overview of the **LIAN** project, defining its functional and non-functional requirements, external interfaces, and system behavior. The project aims to serve as an **educational platform.**

2.2 Limitations and Research Gaps

Analysis: Teaching Pathfinding Algorithms with Real-World Maps: This comprehensive analysis explores the theoretical and practical underpinnings of the Lian (Location Intelligence Algorithm Network) project, an educational tool designed to visualize pathfinding algorithms—Dijkstra, A*, Bellman-Ford, and Best-First Search—on real-world maps. The analysis addresses existing systems, their limitations, and Lian's contribution, conducted as of April 4, 2025, synthesizing recent academic insights to inform the project's development.

Existing Systems for Teaching Pathfinding Algorithms

Existing systems for teaching pathfinding algorithms include VisuAlgo, Pathfinding.js, and Geographic Information Systems (GIS), each offering distinct approaches to enhance learning. VisuAlgo, accessible at VisuAlgo, provides visualizations for graph traversal and other algorithms, with 24 modules covering data structures and algorithms through animations, supported by usability studies showing effectiveness in educational settings. Pathfinding.js, exemplified by PathFinding.js, focuses on pathfinding algorithms like Dijkstra and A* on grid-based environments, allowing users to visualize processes interactively, as seen in GitHub projects for algorithm visualization. GIS, discussed in educational research, enhances understanding of spatial data processing, with applications in algorithmic tasks like density analysis, as noted in studies integrating GIS with AI and coding for high school geography education.

The following table summarizes the existing systems:

System	Focus	Visualization Type	Key Features	References
VisuAlgo	Graph traversal, algorithms	Animated, 2D	24 modules, online quizzes, mobile version since 2022	VisuAlgo
Pathfinding.js	Pathfinding (Dijkstra, A*, etc.)	Grid-based, interactive	Drag-and-drop obstacles, algorithm selection	PathFinding.js
GIS	Spatial data processing	Real-world maps	Density analysis, AI integration, fieldwork support	GIS in Education

Table 1.1

Limitations and Research Gaps

A key limitation of existing systems is their reliance on abstract grid-based visualizations, such as in VisuAlgo and Pathfinding.js, which lack integration with real-world maps, reducing practical relevance for navigation scenarios. For instance, VisuAlgo's animations are effective for theoretical understanding but do not connect to geospatial contexts, and Pathfinding.js, while interactive, is confined to grids, as confirmed by its visual interface. Additionally, while GIS is used in education, its application to teaching pathfinding algorithms in geospatial contexts remains underexplored, with studies focusing more on geography than computational concepts, as seen in research on disaster risk reduction education. This gap is evident in the lack of tools that bridge theoretical algorithms with real-world navigation, limiting engagement and practical learning outcomes.

2.3 Lian's Contribution

Lian contributes by visualizing pathfinding algorithms, including Dijkstra, A*, Bellman-Ford, and Best-First Search, on real-world maps, addressing the identified gap by providing a practical, interactive learning experience. This approach enhances engagement and understanding of geospatial navigation scenarios, offering a novel context compared to the abstract grids of existing tools. By grounding learning in real-world maps, Lian connects theoretical concepts to practical applications, potentially improving comprehension and retention, which is particularly valuable given the usual focus on abstract visualizations in educational tools.

In conclusion, this analysis positions Lian as a pioneering tool in algorithm education, leveraging geospatial visualization to address identified gaps, enhancing learning through practical, interactive experiences.

Chapter 3

Proposed System

3.1 Introduction

This project, **L.I.A.N: Location Intelligence Algorithm Network**, aims to provide a web-based platform for visualizing various pathfinding algorithms on real-world city maps. By leveraging technology algorithms through engaging visuals.

On the frontend, technologies like HTML, CSS, and JavaScript are used to ensure a user-friendly experience. The backend relies on Python with the Flask framework to facilitate efficient server operations and algorithm execution, utilizing various Python libraries for necessary computations.

The integration of the OpenStreetMap API allows for the display of detailed maps, enriching the visualization of algorithm behaviors. This chapter outlines the platform's architecture, the design of the algorithms, system requirements, and experimental results that validate the project's effectiveness as an educational tool for exploring pathfinding algorithms.

3.2 Architecture/Framework

The system architecture is structured in a client-server model. The client-side, built with HTML, CSS, and JavaScript, allows users to interactively select starting and ending points on a map, choose an algorithm, and visualize the results. The server-side, developed using Python and Flask, processes the pathfinding logic and communicates with the OpenStreetMap API to fetch and display maps.

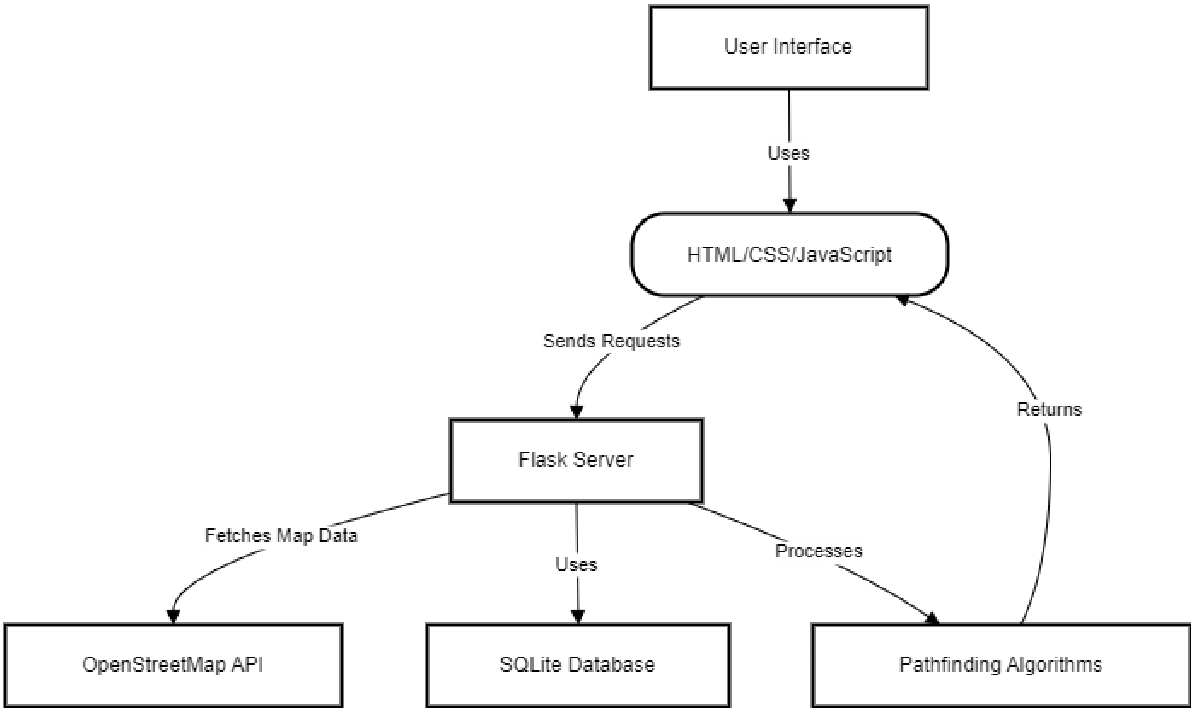


Fig:1.1:Architecture Diagram

Key components include:

- **Client-side**: HTML/CSS/JavaScript for user interface and interaction.
- **Server-side**: Python (Flask) for handling requests and implementing algorithms.
- **Database**: A lightweight SQLite database can be used to store user preferences and past queries.
- **API**: OpenStreetMap for mapping data and routes.

3.3 Algorithm and Process Design

The project implements three primary pathfinding algorithms: Dijkstra’s, A*, and Greedy Best-First Search. Each algorithm is designed to calculate the shortest path based on different strategies:

1. **Dijkstra's Algorithm**: This algorithm evaluates all possible paths, ensuring that the shortest path is found with an exhaustive search.
2. *A Algorithm**: Incorporating heuristics, A* balances speed and accuracy, making it suitable for real-time applications.
3. **Greedy Best-First Search**: Prioritizes the most promising nodes but may not always yield the optimal path.

The process flow is structured as follows:

1. User inputs starting and ending points.
2. User selects the desired pathfinding algorithm.
3. The server processes the request using the chosen algorithm.
4. The results are visualized on the map.

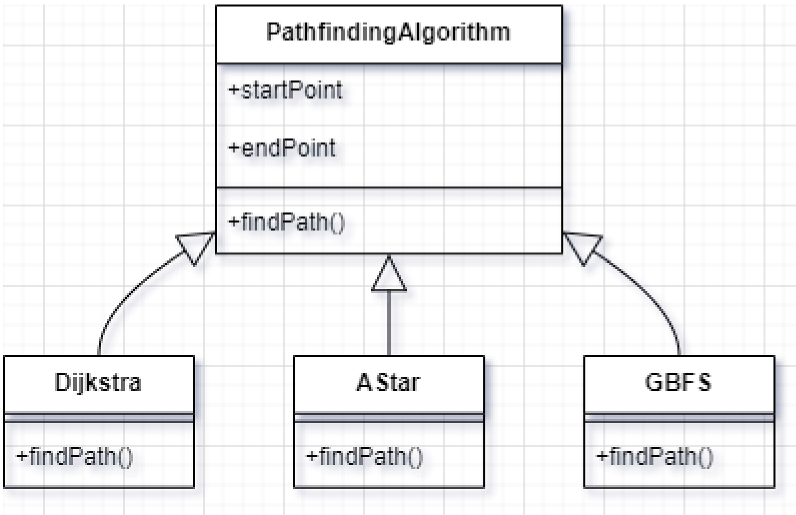


Fig:1.2: Class Diagram

3.4 Details of Hardware & Software

Hardware Requirements:

- A standard web server (e.g., Apache or Nginx) to host the Flask application.
- A machine with a minimum of 4 GB RAM and dual-core processor for local testing.

Software Requirements:

- Operating System: Any (Linux recommended for server deployment).
- Web Technologies: HTML, CSS, JavaScript.
- Backend: Python 3.x with Flask framework.
- Libraries:
  - `Flask`: Web framework.
  - `Requests`: For API calls to OpenStreetMap.
  - `NumPy` and `NetworkX`: For algorithm implementations.

Testing Tools:

- Postman: To test API endpoints and ensure that the server processes requests correctly.

3.5 Experiment and Results for Validation and Verification

To validate the system, we conducted several experiments comparing the performance of the three algorithms. Key metrics included execution time and the accuracy of the path generated.

- **Experiment Setup**: A set of predefined routes in a city map were tested with each algorithm under varying conditions (e.g., different traffic scenarios).
- **Results**: The A* algorithm consistently provided the shortest paths with the least execution time, followed by Dijkstra's, and lastly, GBFS.

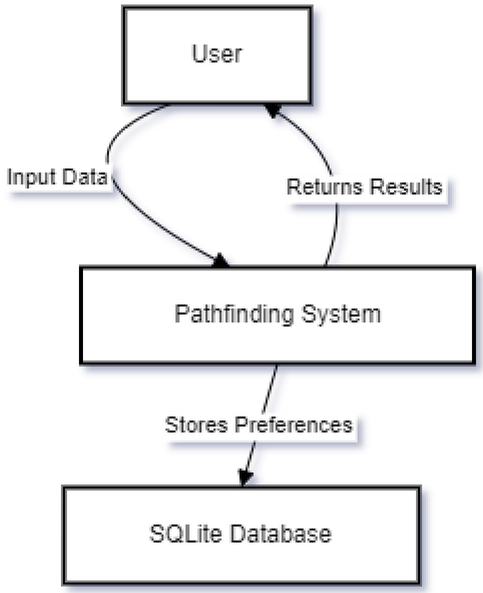


Fig:1.3: Data Flow Diagram Level 0

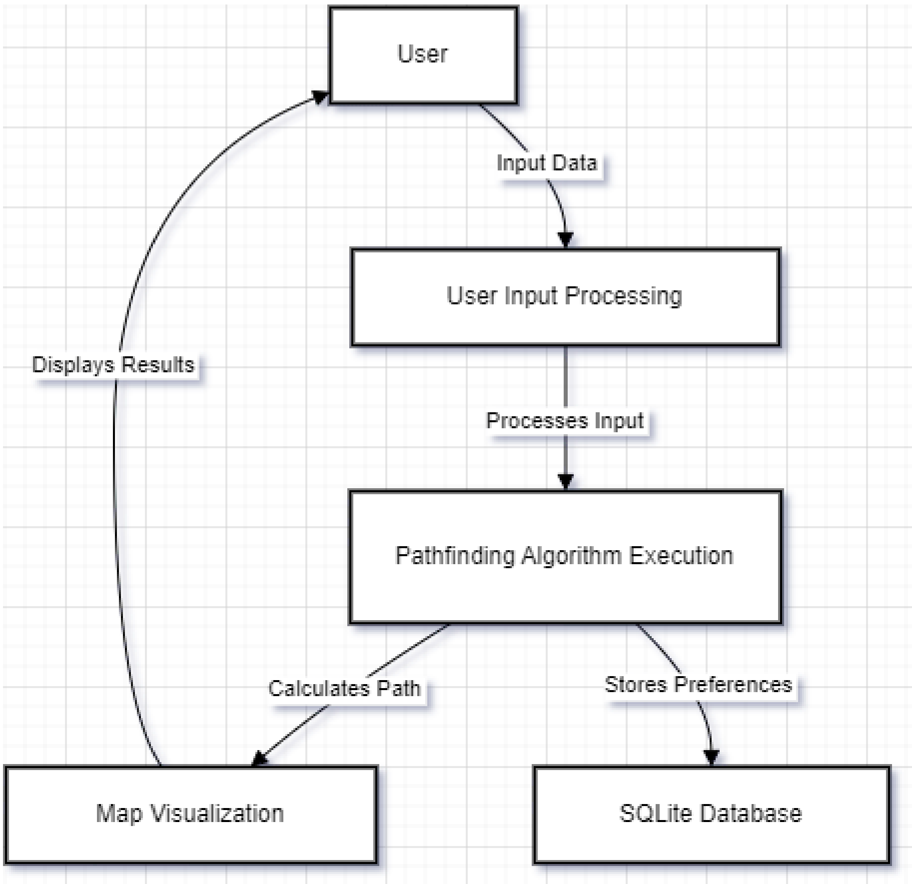


Fig:1.4: Data Flow Diagram Level 1

1. *User Input Processing (DFD Level 2)*

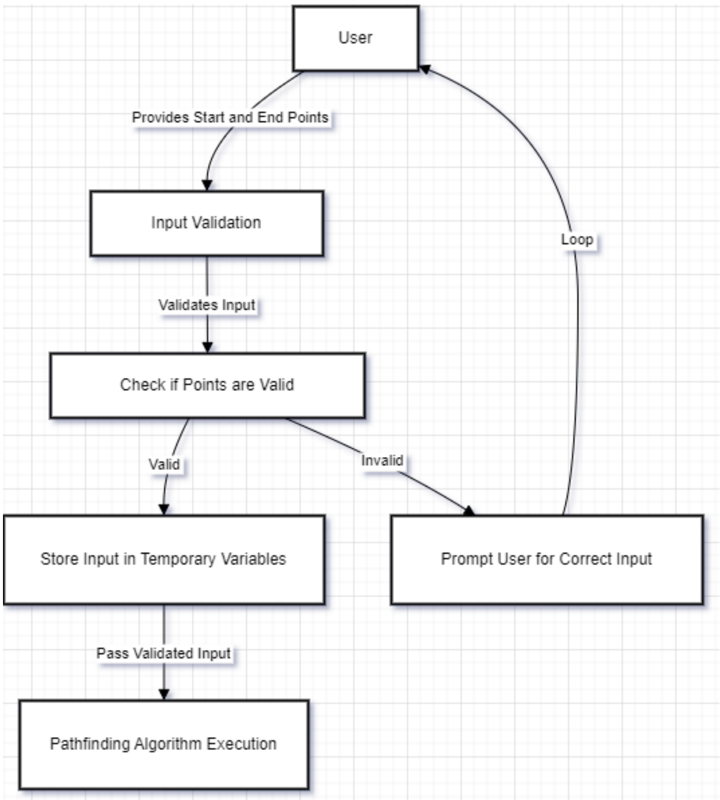


Fig:1.5: Data Flow Diagram Level 2.1

2. Pathfinding Algorithm Execution (DFD Level 2)

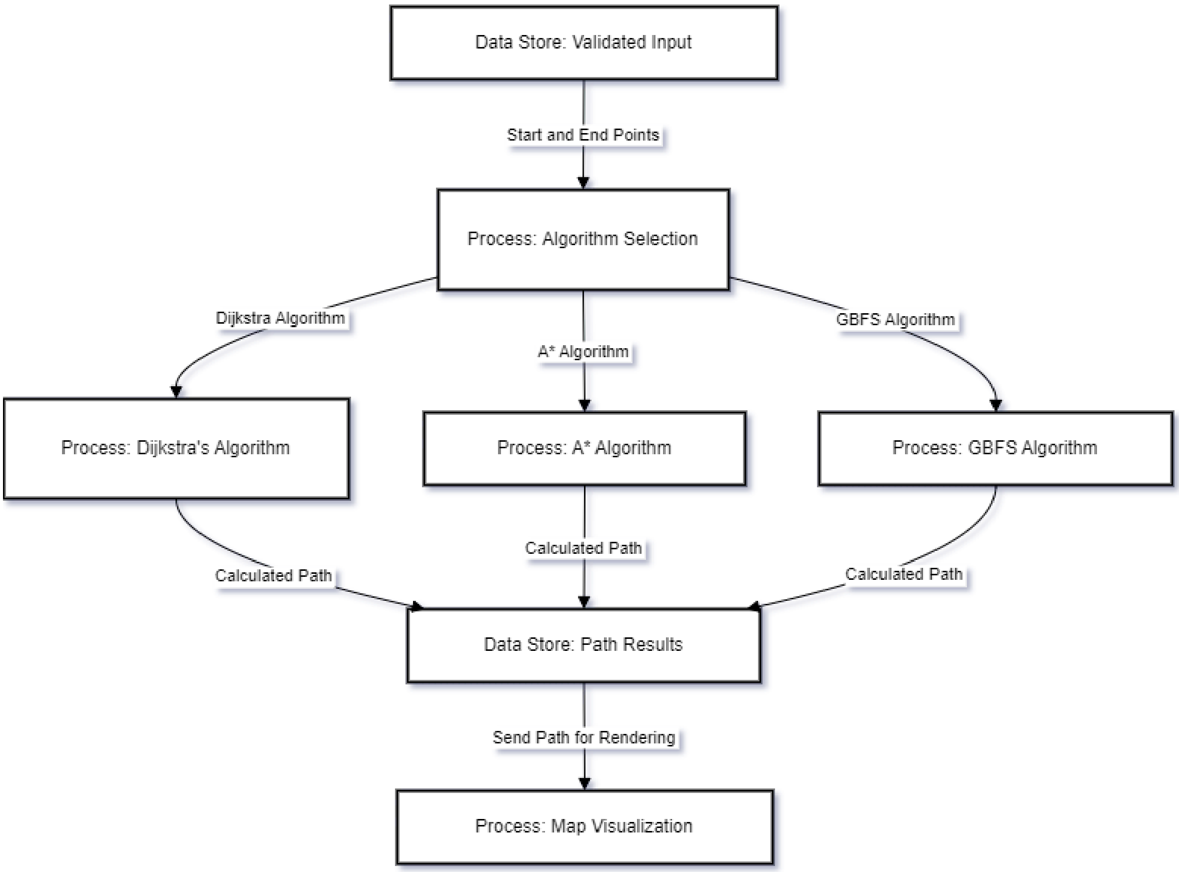


Fig:1.6: Data Flow Diagram level 2.2

2. Map Visualization (DFD Level 2)

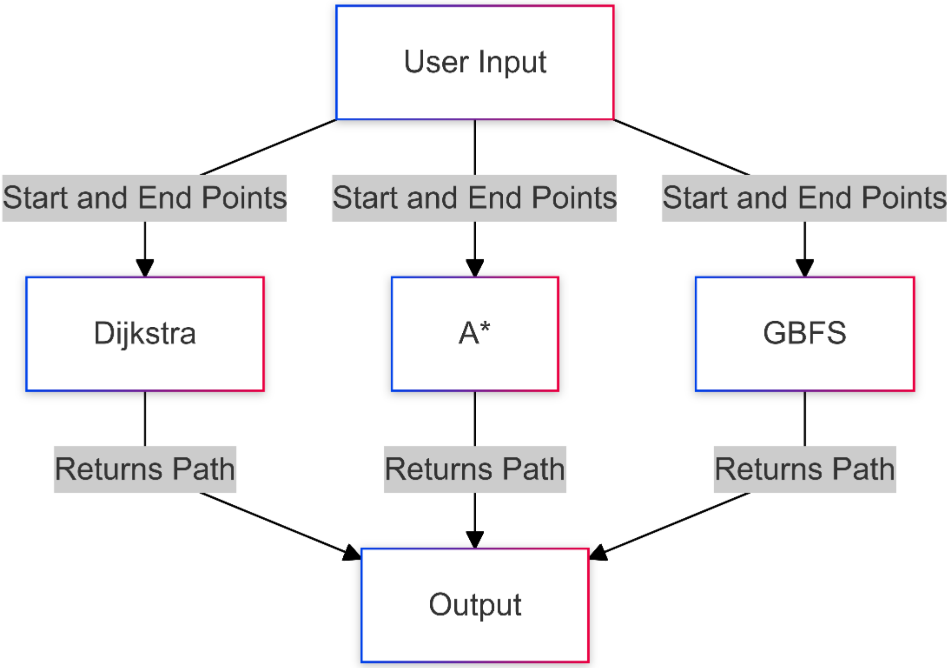


Fig:1.7: Data Flow Diagram Level 2.3

3.6 Analysis

The comprehensive analysis of pathfinding algorithms reveals significant distinctions in their performance characteristics, particularly in terms of efficiency and effectiveness. Dijkstra's algorithm is renowned for its ability to consistently deliver the shortest path between nodes in a graph. However, its inherent time complexity can make it less efficient when applied to larger maps or networks. This is primarily due to its exhaustive search method, which evaluates all potential paths, resulting in longer computation times as the graph size increases.

The analysis of the system shows that each algorithm has its strengths and weaknesses, and the choice of algorithm depends on the specific use case. The user-friendly interface allows non-technical users to experiment with various algorithms and observe their performance in real-time. The integration with OpenStreetMap API ensures the system works in real-world scenarios, providing real-time pathfinding solutions.

Pros:

- Real-time map visualization of paths.
- Easy comparison between different algorithms.
- Interactive user interface.

Cons:

- Dijkstra’s algorithm is slow for large-scale maps.
- GBFS, though fast, can give inaccurate results in certain cases.

3.7 Conclusion and Future Work

LIAN: Location Intelligence Algorithm Network is a unique web-based educational platform that enables users to visually understand how classical pathfinding algorithms operate on real-world maps. By integrating geospatial data with algorithmic logic, LIAN bridges the gap between theory and practical routing challenges. The platform’s focus on education and interactive visualization makes it an accessible tool for students, researchers, and developers alike.

While the primary goal is educational, LIAN also introduces an ambitious secondary dimension—leveraging these algorithms to analyze and optimize global trade routes, with an emphasis on BRICS countries. This positions LIAN at the intersection of **education, artificial intelligence, and geopolitics**.

Future Work

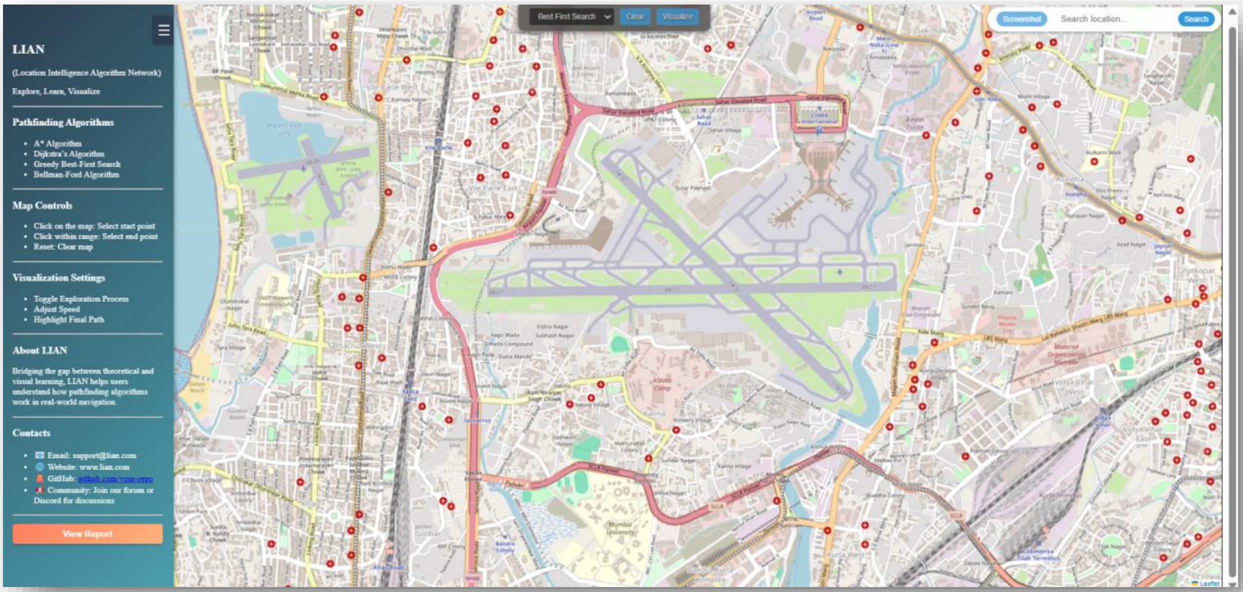
- **Enhanced Trade Route Analysis:** Integrate machine learning models to predict and optimize trade flows based on historical and real-time data.
- **User Custom Maps:** Allow users to upload their own geospatial data or draw custom paths for localized simulations.
- **Mobile App Version:** Extend the simulator to Android/iOS platforms for educational portability.
- **Real-Time Data Integration:** Connect with APIs providing live traffic, shipping, or logistics data.
- **Algorithm Expansion:** Introduce more advanced or hybrid algorithms such as ALT, Bidirectional Search, and Genetic Pathfinding.
- **Community Sharing:** Enable users to share simulations and export animated visualizations as GIFs or videos.

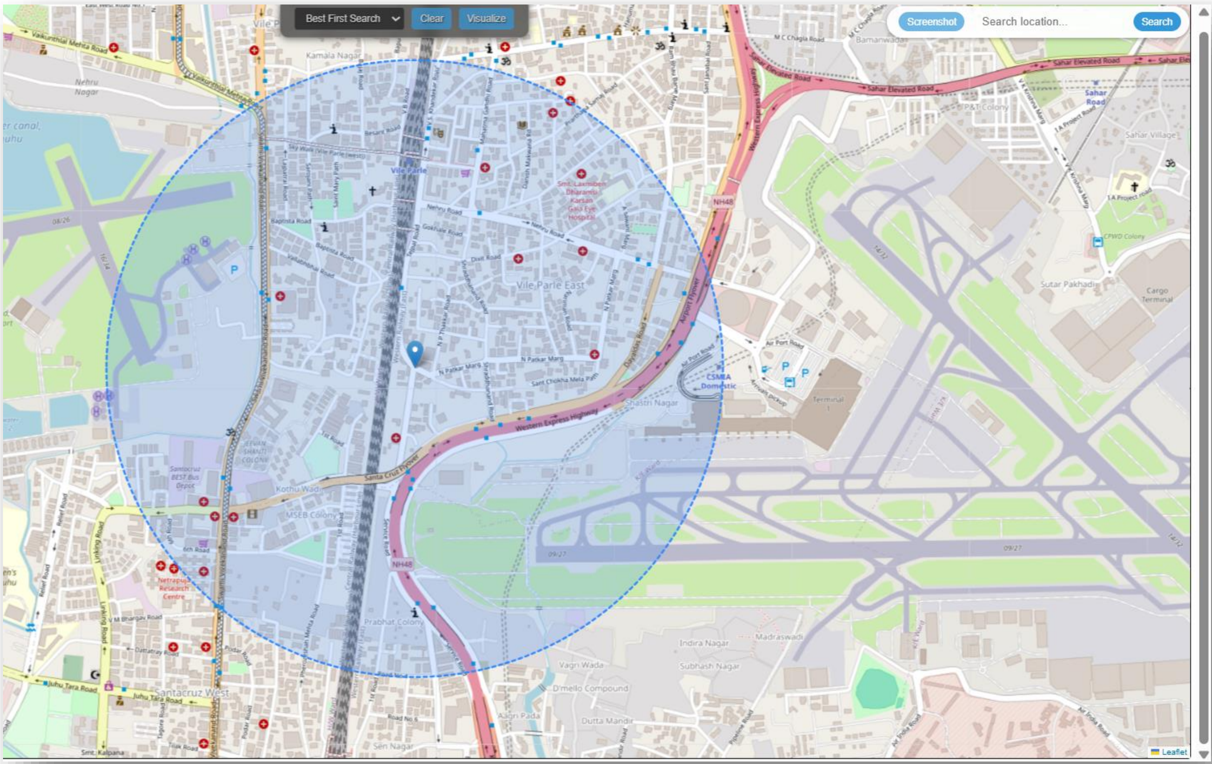
- **Gamification & Learning Modules:** Add challenges, quizzes, and tutorials to deepen user engagement and retention.

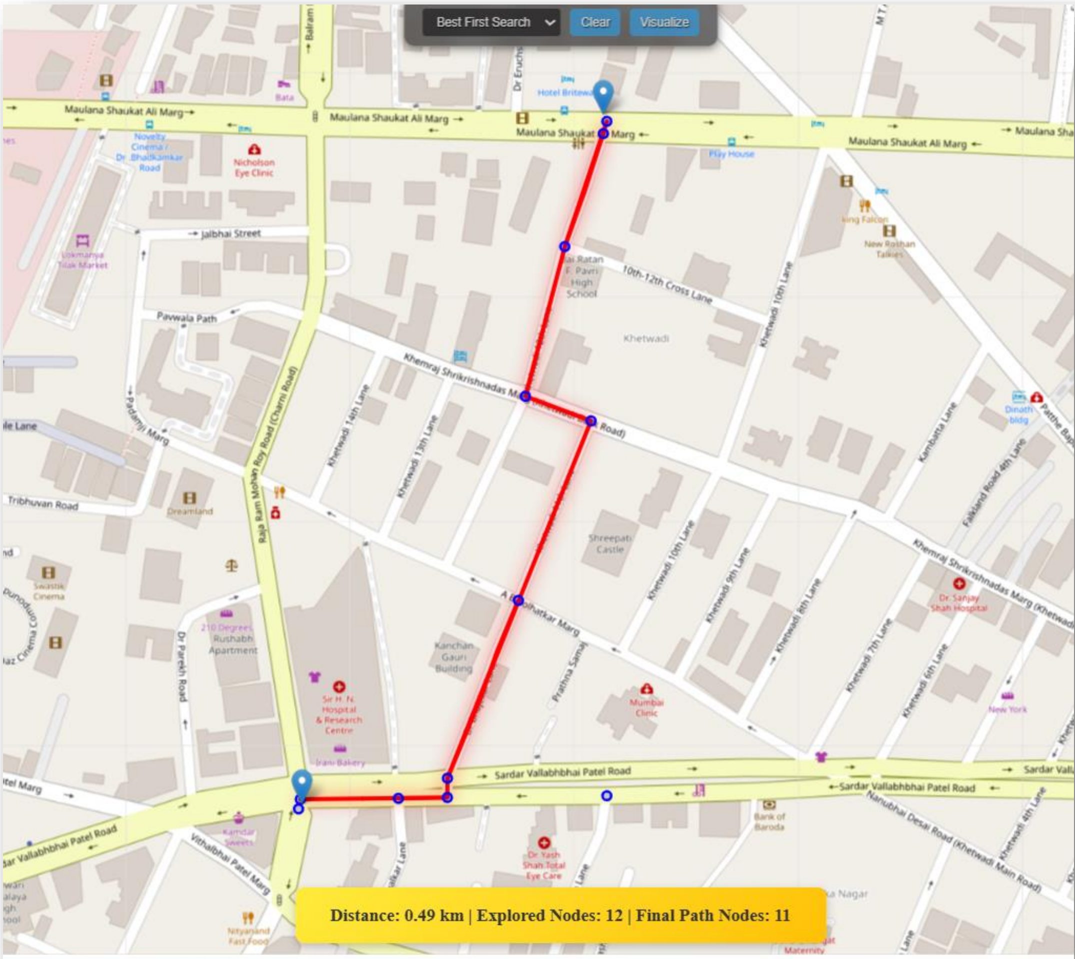
Secondary Goals

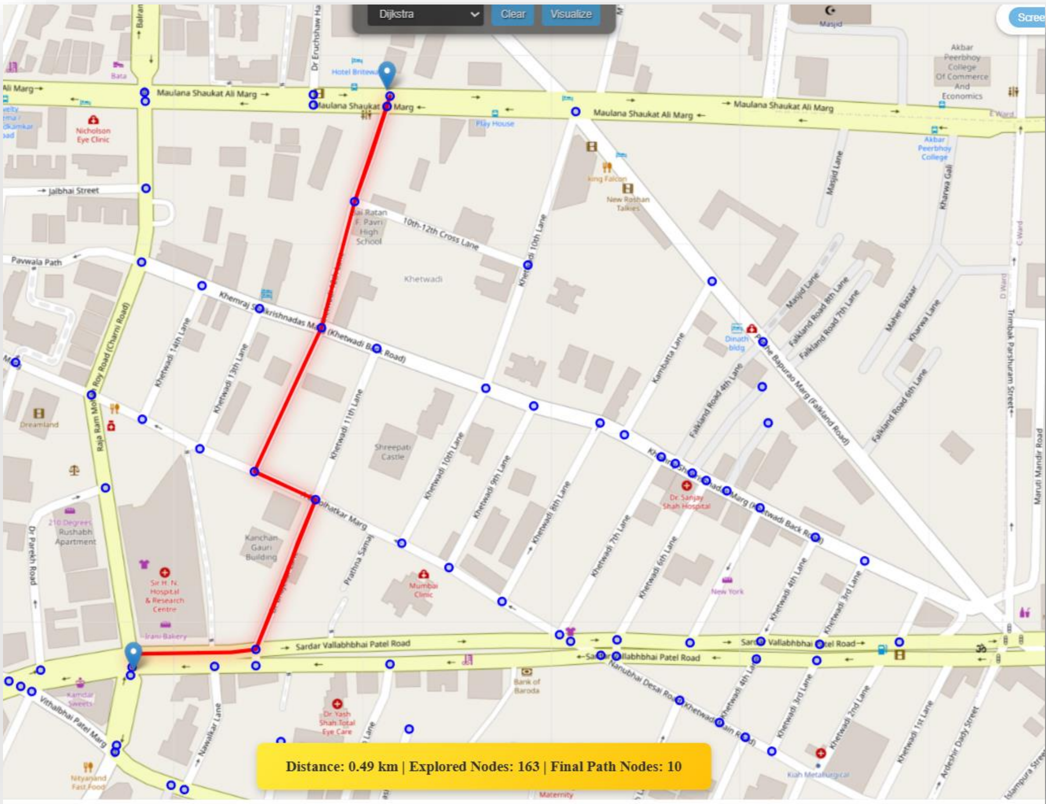
- Visualize and compare performance of pathfinding algorithms under different geographic and infrastructural conditions.
- Highlight and simulate BRICS-centric trade routes to explore geopolitical and logistical implications.
- Serve as an **open learning platform** for academic institutions and research groups in the fields of geoinformatics, AI, and transportation planning.
- Promote awareness of algorithmic fairness, global connectivity, and digital infrastructure gaps.
- Establish a foundation for future **AI-powered decision-making tools** in logistics and route optimization.

Website Prototype






Best First Search
Clear
Visualize
Maulana Shaukat Ali Marg
Khemraj Shrikrishnadas Marg
Sardar Vallabhbhai Patel Road
Distance: 0.49 km | Explored Nodes: 12 | Final Path Nodes: 11


Dijkstra
Clear
Visualize
Maulana Shaukat Ali Marg
Sardar Vallabhbhai Patel Road
Distance: 0.49 km | Explored Nodes: 163 | Final Path Nodes: 10

References

[1] G. J. e. al., “A Systematic Review and Analysis of Intelligence-Based Pathfinding Algorithms,” *Applied Sciences,* pp. -, 2022.

[2] P. E. N. N. J. R. B. Hart, "A Formal Basis for the Heuristic Determination of Minimum Cost Paths," *EEE Transactions on Systems Science and Cybernetics,* 1968.

[3] M. R. Wayahdi, “Greedy, A-Star, and Dijkstra’s Algorithms in Finding Shortest Path,” *International Journal of Advances in Data and Information Systems,* pp. 45-52, 2021.

[4] Unknown, “Foundation of A* Algorithm,” *International Journal of Creative Research Thoughts,* 2018.

[5] D. e. a. Delling, “Engineering Route Planning Algorithms,” *Algorithm Engineering,* 2018.

[6] R. P. J. Dechter, “Generalized Best-First Search Strategies and the Optimality of A*,” *Journal of the ACM,* 1985.

[7] U. 02, "A Comparative Study of Pathfinding Algorithms for Road Networks," *Transportation Research Part C: Emerging Technologies,* 2020.