# MINOR PROJECT

## Title: Net-Route Implementation and Comparative Study

**Presented by:**
R2142210034 - Abhishek Joshi
R2142210525 - Nihar
R2142210515 - Naman Jain
R2142210795 - Suraj Singh Bhandari

**Mentored By:**
Dr. Kingshuk Srivastava

## CONTENT

## INTRODUCTION

- In today's modern world, where data flows easily and seamlessly between devices and systems, efficient network routing plays a crucial role in data communication.

- This project relies on two fundamental elements i.e., data and algorithm used. The data is taken in the form of text files which is structured in a graph and then different path finding algorithms are used over the data.

- The project attempts to create a network routing system that not only finds the optimal or near-to-optimal path but also compares different approaches used by different algorithms for efficient and optimal solutions.

## TECHNICAL CONCEPTS

- Text File Handling
- Implementation of Graph Data Structure
- Various routing algorithms:
  - A* Algorithm
  - Dijkstra Algorithm
  - Bellman-Ford

## MOTIVATION

The human dependency on technology is growing day-by-day, leading to large number of data transfer from one place to another. This gives rise to various networking issues such as congestion, data loss, latency and much more. To solve these problems, our project brings a one-step solution providing the best path a data packet may follow for smooth transition and provides the best method possible for data transfer.

## PROBLEM STATEMENT

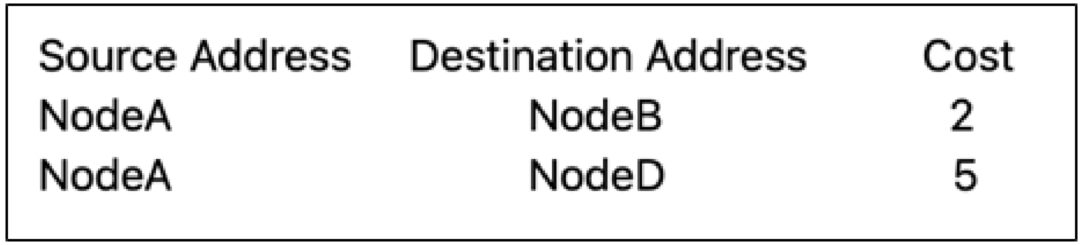
In today’s advanced world where communication depends totally on networking, the role of routing algorithms plays a vital role for fast and smooth transfer of data from one source to destination. Using algorithms which improve the efficiency of networking makes it easier to communicate smoothly.

## AREA OF APPLICATION

Network routing is used in transfer of data from one interconnected device to another. This project can be implemented by various communication-based organizations for smooth flow of data. Apart from this, it can be used to measure the various methods used for routing based on certain parameters.

## DATASET AND INPUT FORMAT

Random sampling of data using Python from different network topologies



| Source Address | Destination Address | Cost |
|---|---|---|
| NodeA | NodeB | 2 |
| NodeA | NodeD | 5 |

## LITERATURE REVIEW

| Title | Author | Algorithm used | Result |
|---|---|---|---|
| **Scaling and reliable sensor network** (Document) | Matthew S. Nassr Jangeun Jun Stephan J. Eidenbenz Anders A. Hansson Angela M. Mielke | DTRP (Directed transmission routing protocol) | 93% packets transfer at the rate of 1 packet per-second. |
| **Software defined network router in wireless** (Document) | Junfeng Wanga Yiming Miaob Ping Zhoub M. Shamim Hossainc SK Md Mizanur Rahmand | AODV, DLSR, SDN (novel) and GPSR | SDN provides the shortest path (novel algorithm). |
| **A survey of shortest path algorithms** (Document) | Madhumita Panda<br><br>Abinash Mishra | Dijkstra's algorithm, symmetrical Dijkstra's algorithms, A* algorithm, Bellman-ford algorithm, Floyd Warshall algorithm and genetic algorithm | Bellman-Ford algorithm is the most efficient whereas genetic algorithm with high accuracy takes more time. |
| **Performance analysis of Ad hoc networking routing protocol** (Document) | P. Chenna Reddy<br><br>Dr. P. Chandra Sekhar Reddy | DSDV, AODV, TORA AND DSR | DSDV performance was poor and DRS performed better than others but with more memory usage. |

## INFERENCE FROM LITERATURE

Few of the inferences from the literatures include the following:

- 93% of the packets transfer at 1 packet per second therefore limiting the chances of further optimization.
- Dijkstra Algorithm's accuracy could be improved using Genetic Algorithm but it takes more time.
- Understanding about newer algorithms such as AODV, DLSR, TORA, etc.

## SWOT ANALYSIS

**Strength:**

- Promotes efficient data transfer.
- This is independent of the network size to a very large network.

**Weakness:**

- Implementation of this algorithm can be complex and requires deep knowledge.
- Error handling may be a little complex to implement.

**Opportunities:**

- Ongoing advancement in routing can provide opportunities for improving routing efficiency and adaptability.
- Network Routing systems can play a crucial role in efficiently routing data.

**Threats:**

- This field is competitive with various routing protocols, so it makes it challenging to stand out.
- Integrating a subpar solution with an existing system might be unsuitable in some cases.

# OBJECTIVES

## MAIN OBJECTIVE

The primary objective of this project are as follows:

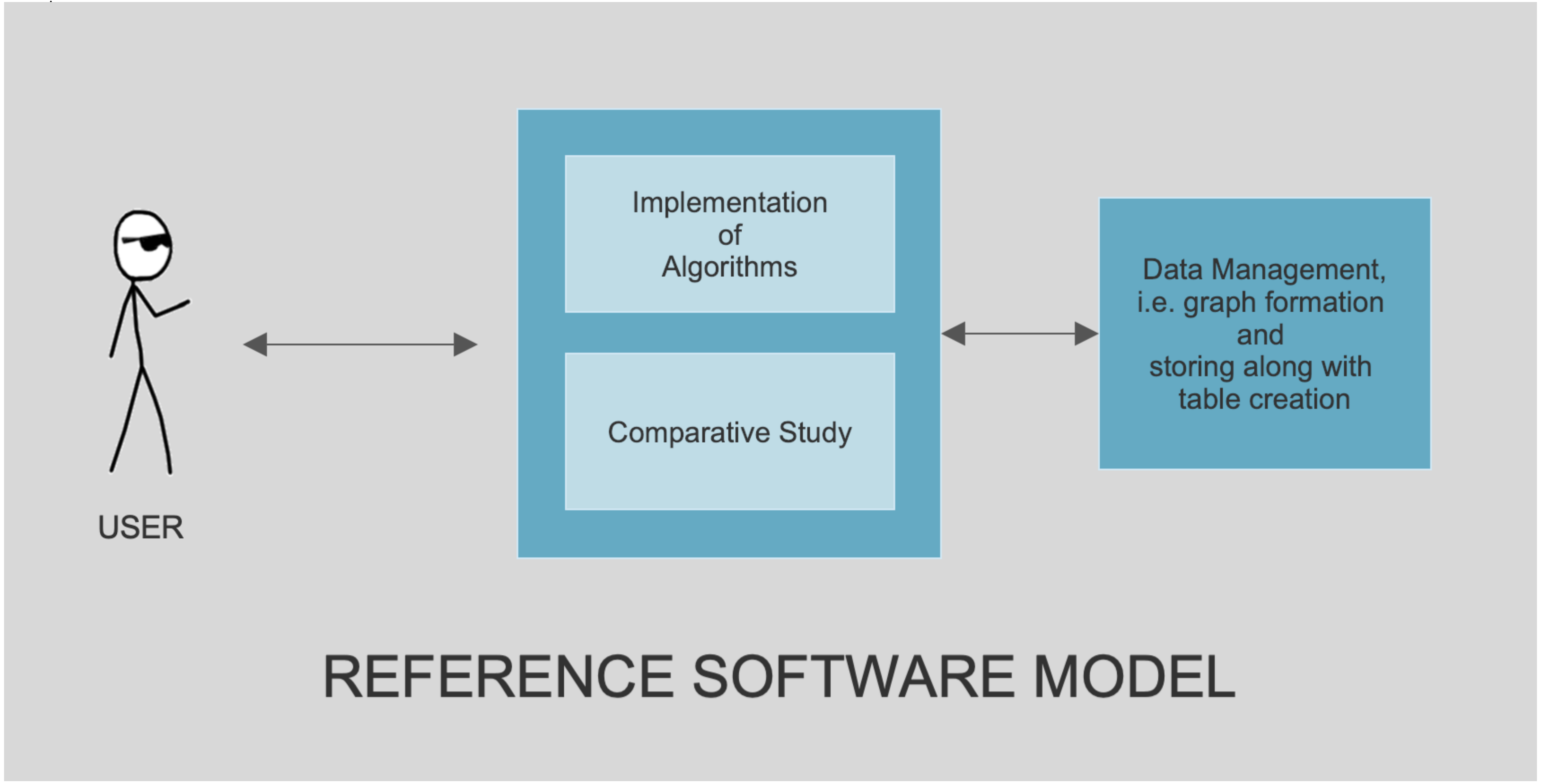
- This project aims to find the optimal path for data transfer to avoid latency in data packets transfer.
- This project focuses on how to process data into graph data structure to structure the topology information.
- Create a table containing shortest path from one node to another node which will help us perform the above and a comparative study.

## SUB OBJECTIVE

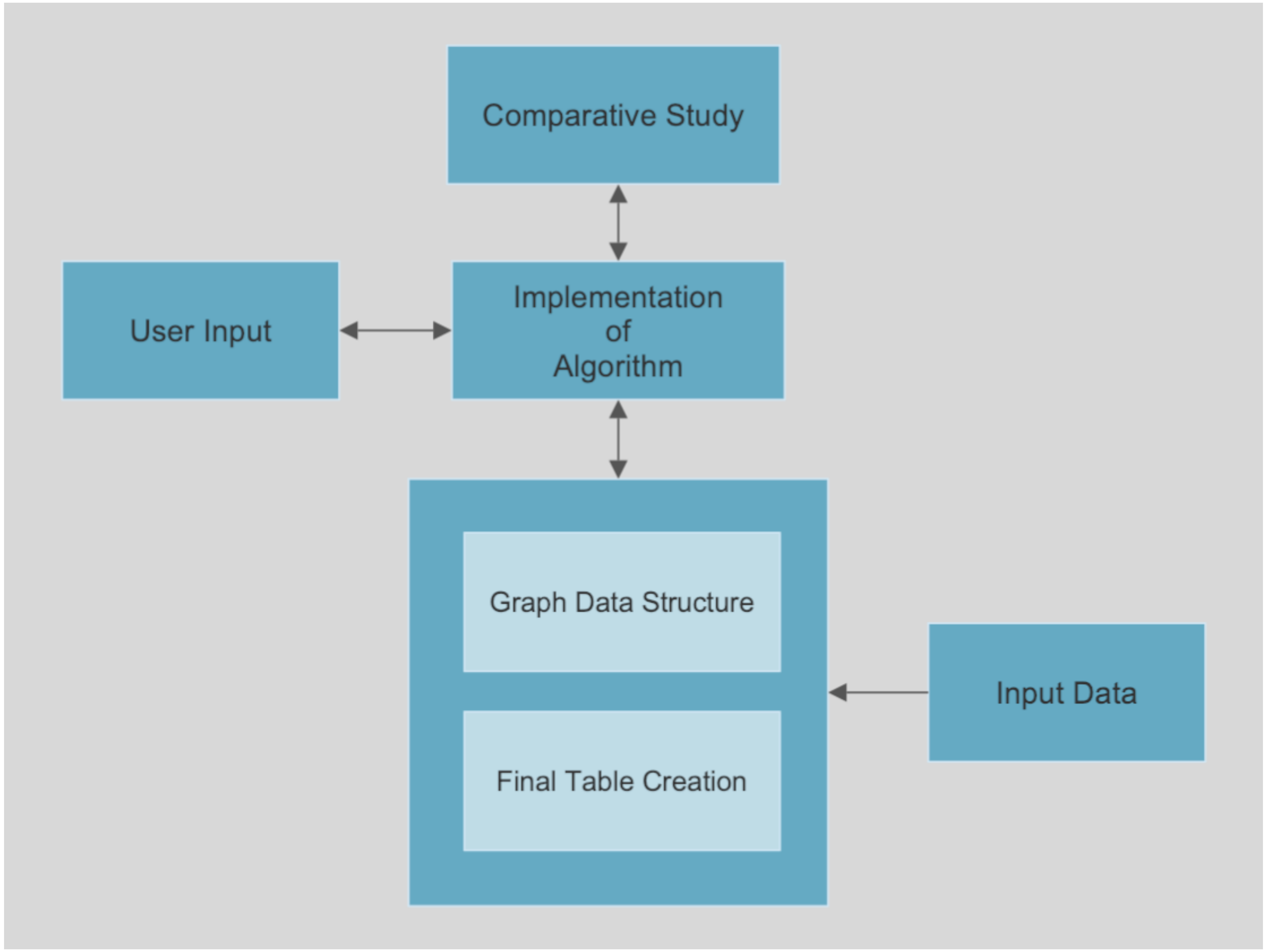
This project aims to achieve the following sub objectives:

- Review existing literatures which involve the implementation of network routing algorithms to diverse our understanding.
- It also speaks to identify any areas for future research and development.
- Document all of the project including the working, results and analysis and seek feedback from peers or experts in the fields.

## METHODOLOGY


USER
Implementation of Algorithms
Comparative Study
Data Management, i.e. graph formation and storing along with table creation
REFERENCE SOFTWARE MODEL

## STEPS INVOLVED

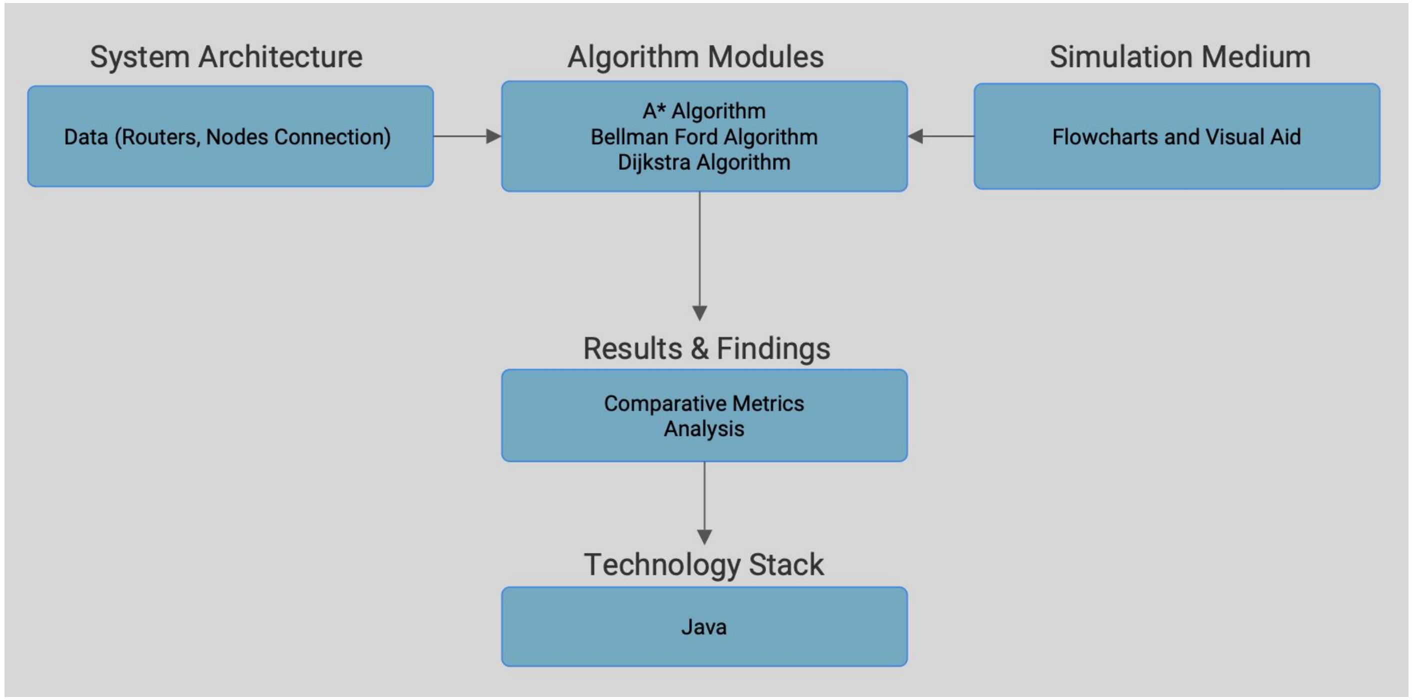


# WORKING MODEL

## REQUIREMENT ANALYSIS:

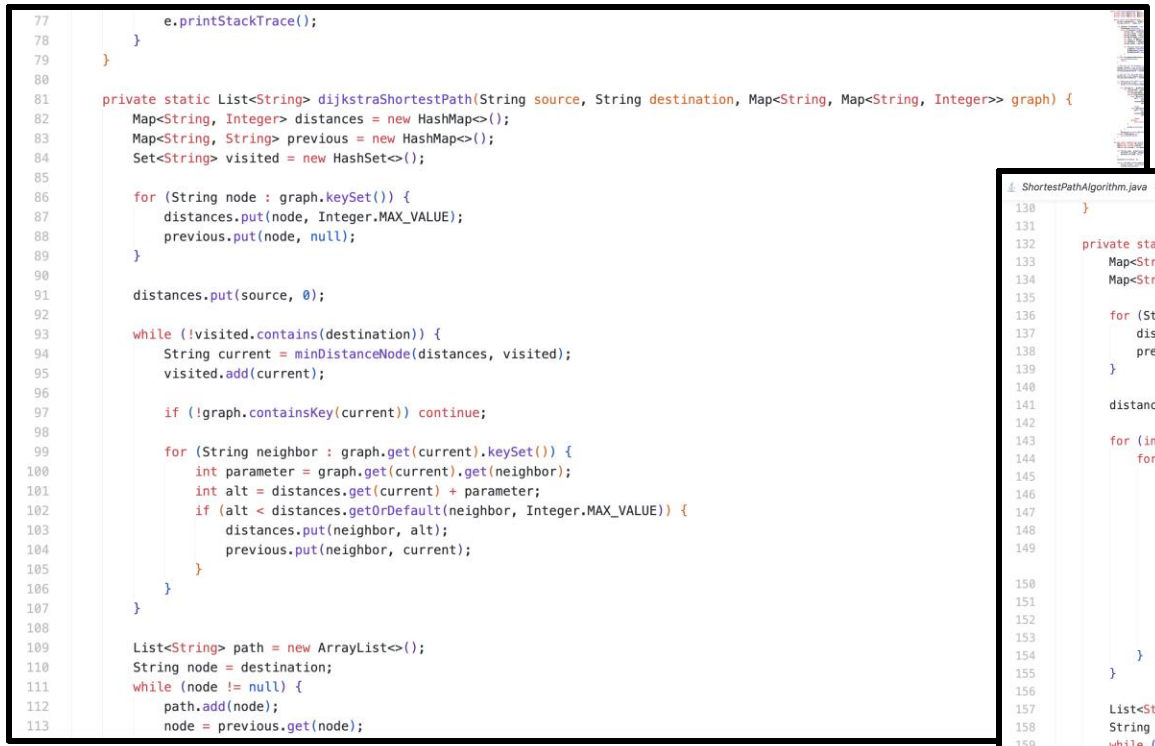
https://docs.google.com/document/d/1t0mKL227xQEApyoMAd-cX9zLy8aOE6tg-OGDjMcLTFQ/edit?usp=sharing

## TECHNICAL DIAGRAM

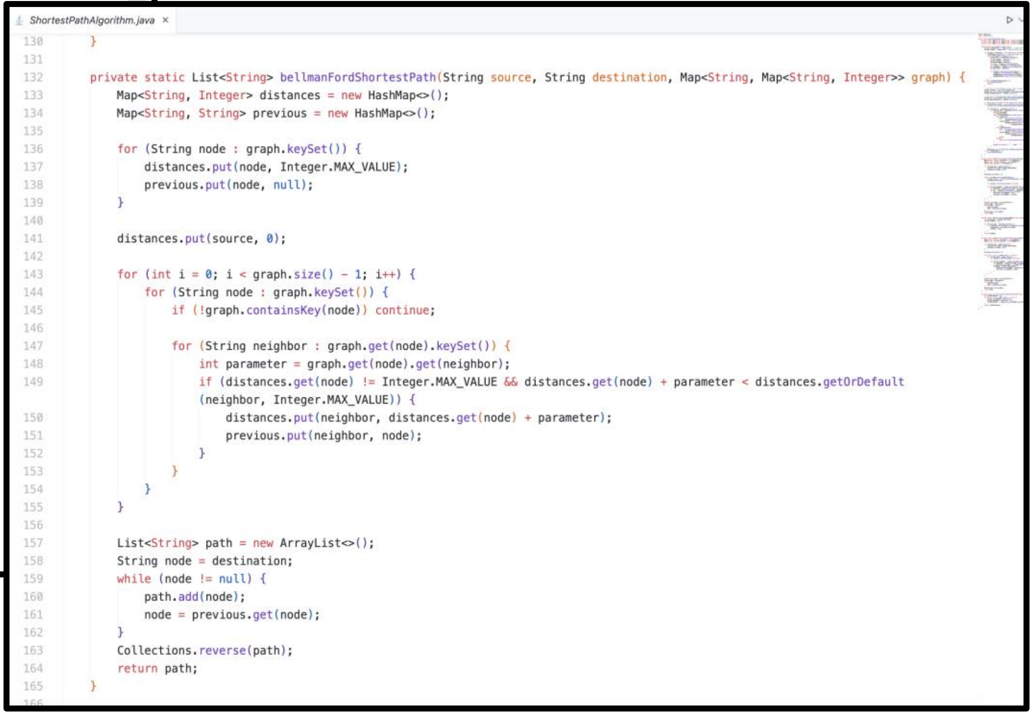


# WORKING MODULE & ATTAINED DELIVERABLE

Below are the part snapshots of the Dijkstra and Bellman Ford Algorithms, we have implemented in our project.



```java
            e.printStackTrace();
        }
    }

    private static List<String> dijkstraShortestPath(String source, String destination, Map<String, Map<String, Integer>> graph) {
        Map<String, Integer> distances = new HashMap<>();
        Map<String, String> previous = new HashMap<>();
        Set<String> visited = new HashSet<>();

        for (String node : graph.keySet()) {
            distances.put(node, Integer.MAX_VALUE);
            previous.put(node, null);
        }

        distances.put(source, 0);

        while (!visited.contains(destination)) {
            String current = minDistanceNode(distances, visited);
            visited.add(current);

            if (!graph.containsKey(current)) continue;

            for (String neighbor : graph.get(current).keySet()) {
                int parameter = graph.get(current).get(neighbor);
                int alt = distances.get(current) + parameter;
                if (alt < distances.getOrDefault(neighbor, Integer.MAX_VALUE)) {
                    distances.put(neighbor, alt);
                    previous.put(neighbor, current);
                }
            }
        }

        List<String> path = new ArrayList<>();
        String node = destination;
        while (node != null) {
            path.add(node);
            node = previous.get(node);
```



```java
    }

    private static List<String> bellmanFordShortestPath(String source, String destination, Map<String, Map<String, Integer>> graph) {
        Map<String, Integer> distances = new HashMap<>();
        Map<String, String> previous = new HashMap<>();

        for (String node : graph.keySet()) {
            distances.put(node, Integer.MAX_VALUE);
            previous.put(node, null);
        }

        distances.put(source, 0);

        for (int i = 0; i < graph.size() - 1; i++) {
            for (String node : graph.keySet()) {
                if (!graph.containsKey(node)) continue;

                for (String neighbor : graph.get(node).keySet()) {
                    int parameter = graph.get(node).get(neighbor);
                    if (distances.get(node) != Integer.MAX_VALUE && distances.get(node) + parameter < distances.getOrDefault
                    (neighbor, Integer.MAX_VALUE)) {
                        distances.put(neighbor, distances.get(node) + parameter);
                        previous.put(neighbor, node);
                    }
                }
            }
        }

        List<String> path = new ArrayList<>();
        String node = destination;
        while (node != null) {
            path.add(node);
            node = previous.get(node);
        }
        Collections.reverse(path);
        return path;
    }
```

## REFERENCE

1. *M. S. Nassr, J. Jun, S. J. Eidenbenz, A. A. Hansson and A. M. Mielke, "Scalable and Reliable Sensor Network Routing: Performance Study from Field Deployment," IEEE INFOCOM 2007 - 26th IEEE International Conference on Computer Communications, Anchorage, AK, USA, 2007, pp. 670-678, doi: 10.1109/INFCOM.2007.84. (Link to Document)*

2. *Wang, Junfeng & Miao, Yiming & Zhou, Ping & Hossain, M. Shamim & Rahman, Sk Md Mizanur. (2016). A Software Defined Network Routing in Wireless Multihop Network. Journal of Network and Computer Applications. 85. 10.1016/j.jnca.2016.12.007. (Link To Document)*

3. *Madkour, Amgad & Aref, Walid & Rehman, Faizan & Rahman, Abdur & Basalamah, Saleh. (2017). A Survey of Shortest-Path Algorithms. (Link To Document)*

4. *P. C. Reddy and P. C. S. Reddy, "Performance Analysis of Adhoc Network Routing Protocols," 2006 International Symposium on Ad Hoc and Ubiquitous Computing, Mangalore, India, 2006, pp. 186-187, doi: 10.1109/ISAHUC.2006.4290671. (Link To Document)*

UPES

UNIVERSITY OF TOMORROW

THANK YOU