**MINOR-1 PROJECT**

**SYNOPSIS on**

Travel Route Chatbot

Submitted By:

|  |  |  |
| --- | --- | --- |
| **Name** | **Roll No** | **Branch** |
| Dishant Saini | R2142220485 | BTech CSE AIML |
| Palak Jain | R2142220583 | BTech CSE AIML |
| Sahil | R2142220622 | BTech CSE AIML |
| Akshat Rajput | R2142210209 | BTech CSE AIML |

**Under the guidance of**

Dr. Bhavana Kaushik

(Assistant professor at School of Computer Science)

A picture containing text, clipart

Description automatically generated

**School of Computer Science**

**University of Petroleum & Energy Studies, Dehradun**

**Dehradun-248007**

**Aug-Nov 2024**

**Approved By**

**Project Guide Cluster Head**

**INDEX**

|  |  |  |
| --- | --- | --- |
| **S NO.** | **TOPIC** | **PAGE NO.** |
| 1 | Introduction | 1 |
| 2 | Problem Statement | 2 |
| 3 | Literature Review | 3 |
| 4 | Objective | 4 |
| 5 | Methodology | 5 |
| 6 | Timeline | 6 |
| 7 | System Requirement | 7 |
| 8 | References | 8 |

# Introduction

In each day existence, humans normally face many problems in finding which direction leads. People usually discover each viable solution in finding their manner inside the campus, however not each answer can produce the finest direction. Shortest route trouble is a trouble in finding the quickest course or route from a directed graph. The “Chatbot” is used for navigating the person from one area to some other inside the campus. All navigation device takes user vicinity as a place to begin and gives the exceptional superior direction to be accompanied. Every path in a graph has a price to be calculated. This shortest course hassle is a way to discover a new course or direction in a graph with a minimum sum of weight travelled thru the course. This shortest path trouble is solved by way of using Dijkstra’s set of rules of finding the best aspect route between vertices in a graph. There are several versions of set of rules that can be used to determine the node that was pursued based at the route in given graph.

Variations of the shortest path may be outstanding from single-source Objective, pair direction and generalization. A pair of shortest paths is finding the shortest path for two points of nodes. All pair of shortest paths is a method to locate the shortest direction among all directed nodes. Single-source shortest route is finding the shortest form travelled, beginning from a certain node to all different nodes inside the graph.

Single source shortest direction problem is to find the shortest direction from any node at the graph are directed to a single destination node. Intermediate shortest route is finding the shortest course between nodes decided on via different nodes. Generalization is extensively greater green than the simple method to run one-pair of shortest routes set of rules on all pairs of vertices that are relevant.

# Problem Statement

Most of the advanced navigation systems are unable to provide precise routes as well as information of building within a small region such as a hospital, university campus or shopping mall. As people get more and more connected to technology, they lose their human touch. People feel more convenient to look for the problem themselves than to ask someone for help. This technological era requires a reliable and precise guidance system. It should be able to navigate the user no matter where they are in the world. The guidance system needs to be easy to use. Students spend most of their time travelling between different buildings on a college campus since the size of the campus can vary from 30 acres to 200 acres. New students have a hard time searching their way inside the campus.

To find the optimal path within the campus travel guide chatbot is required.

# Literature Review

|  |  |  |
| --- | --- | --- |
| Reference​ | Methods​ | Year of Research​ |
| [1] | Application of Dijkstra's algorithm for railway route optimization in transportation systems​ | 2014​ |
| [2]​ | Dijkstra's algorithm combined with node combination for shortest pathfinding in Geographical Information Systems (GIS)​ | 2018​ |
| [3] ​ | Integration of Dijkstra’s algorithm with the Haversine formula for landmark-based shortest path detection using real-world coordinates​ | 2013​ |
| [4]​ | Development of a web-based chatbot management system for creating chatbots without programming knowledge​ | 2021​ |
| [5]​ | Review and evaluation of various shortest path algorithms, including Dijkstra’s algorithm, A\*, and others​ | 2013​ |

# Objectives

The program aims to develop a chatbot that simplifies campus navigation by providing users with the shortest and most efficient routes. Using Dijkstra’s algorithm, the chatbot ensures accurate and efficient route calculations. Designed for students, staff, and visitors, it enhances user experience with a responsive interface that offers real-time, clear directions. The scalable design allows adaptation to other campuses, and its seamless integration with web technologies ensures accessibility across various devices, making campus navigation quicker and more convenient.

* To build a responsive chatbot interface that provides real- time, accurate direction for campus navigation.
* To develop an algorithm framework using Dijkstra’s algorithm for efficient and precise route calculations.
* To create a user-friendly system designed specifically for students, staff and visitors to reduce time spent navigating the campus.
* To ensure scalability of the program for potential adaptation to other campuses or similar environments.

# Methodology

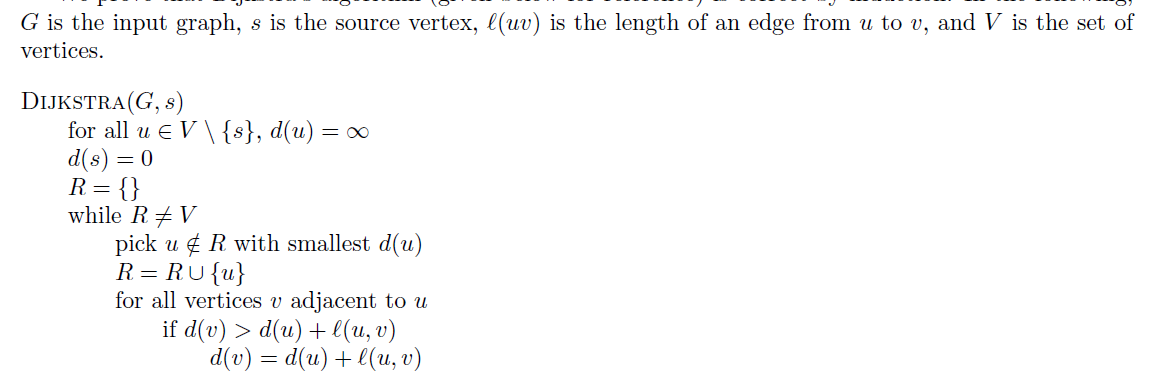
Dijkstra’s algorithm is for minimum spanning tree. In Dijkstra’s we generate a*SPT (shortest path tree)* with given source as root. We maintain two sets, one set contains vertices included in shortest path tree, and other set includes vertices not yet included in shortest path tree. At every step of the algorithm, we find a vertex which is in the other set (set of not yet included) and has a minimum distance from the source.

The Dijkstra’s algorithm maintains two sets of vertices which are:

*L*:Labeled vertices (shortest path is known)

*C*: Candidate vertices (shortest path not yet known)

One vertex is moved from *C* to *L* in each iteration



The steps to be followed for completing the proposed project are as follows:

1. **Identification of Key Blocks and Coordinates in Campus:** key areas (blocks) are identified on campus and their geographic coordinates (latitude and longitude) are collected. These positions are the nodes of the graph used in the Dijkstra algorithm. Nodes represent blocks such as classrooms, libraries, hotels, or other important landmarks.
2. **Distance and Direction Calculation Between Nodes:** Distances to neighboring nodes are determined using geographic information (Haversine formula for real-world coordinates) or manually calculated distances Furthermore, distances to neighboring nodes are determined using directional data (e.g. "left", "right", "straight"). (blocks) between the connected nodes. This data is a weighted graph where each edge represents the distance and direction between two blocks.
3. **Finding the Shortest Path and Providing Step-by-Step Directions:** Using Dijkstra’s Algorithm, the shortest distance between the user’s start block and destination block is calculated. The program will not only provide the user with the shortest route but also provide the user with step-by-step navigation instructions (e.g. turn left, go straight) and intermediate distances between the two, and help them navigate their environment better

# System Requirement

**Software Requirements**

Operating System : Windows 11/10 (32-bit or 64-bit)/ Linux

Software : Text Editor/Turbo C/Dev C++

Compiler : GCC

**Hardware Requirements**

Processor : Dual Core 2.7 GHz or better

RAM : 2GB or higher

# Timeline

|  |  |  |
| --- | --- | --- |
| Task | Timeframe | Description |
| Project Start | 20-21 August | Official start of the project |
| Requirements Analysis (T1) | Last Week of August | Analyzing requirements for the project |
| Estimate Distance (T2) | Last Week of August | Estimating distances between campus infrastructure |
| Map Designing (T3) | Mid-Semester | Designing the campus map |
| Algorithm Implementation/Coding (T4) | Last Week of September | Developing algorithms for map functions |
| Map Design & Algorithm Integration (T5) | Last Week of October | Integrating the map design with algorithms |
| Final Testing (T6) | Mid-November | Testing the integrated system |
| Project End | End of November | Project conclusion |

# References

1. Pandey, Pramod, and Sunanda Dixit. "Railway Route Optimization System Using Dijkstra Method." *International Journal on Recent and Innovation Trends in Computing and Communication (IJRITCC) Volume* 2 (2014): 435-440.
2. Fitro, Achmad, et al. "Shortest path finding in geographical information systems using node combination and dijkstra algorithm." *SHORTEST PATH FINDING IN GEOGRAPHICAL INFORMATION SYSTEMS USING NODE COMBINATION AND DIJKSTRA ALGORITHM* 9.2 (2018): 755-760.
3. Ingole, P. V., and Mangesh K. Nichat. "Landmark based shortest path detection by using Dijkestra Algorithm and Haversine Formula." *International Journal of Engineering Research and Applications (IJERA)* 3.3 (2013): 162-165.
4. Hasyim, M. W., and S. Pramono. "Web-based telegram chatbot management system: create chatbot without programming language requirements." *IOP Conference Series: Materials Science and Engineering*. Vol. 1096. No. 1. IOP Publishing, 2021.
5. Magzhan, Kairanbay, and Hajar Mat Jani. "A review and evaluations of shortest path algorithms." *Int. J. Sci. Technol. Res* 2.6 (2013): 99-104.