Kathmandu University
Department of Computer Science and Engineering
Dhulikhel, Kavre

A Project Report
on
"Travel Itinerary Planner"

[Code No: COMP 202]
(For partial fulfillment of II/I Year/Semester in Computer Science/Engineering)

Submitted by
Koishna Shrestha (53)

Submitted to
Mr. Sagar Aacharya
Department of Computer Science and Engineering

Submission Date:2/25/2026

Acknowledgement

The successful completion of this mini project, "*Travel Itinerary Planner*", would not have been possible without the valuable guidance, support, and encouragement we received from several individuals. We are grateful to all who contributed directly or indirectly to this journey.

First and foremost, I would like to express my sincere gratitude to our lecturer, **Mr. Sagar Acharya**, Department of Computer Science and Engineering, Kathmandu University. His constant guidance and support helped me stay focused and committed throughout this mini project.

I also thank the **Department of Computer Science and Engineering, Kathmandu University,** for providing the academic environment, tools, and encouragement necessary for such an initiative. The support of the department continually motivates us to think innovatively and push our limits.

I also thank my **friends and family** for their moral support, patience, and thoughtful input during the course of this mini project. Their encouragement was a great source of strength.

Abstract

This mini project is one of the Travel Itinerary Planner System that has been developed based on the main ideas of Data Structure and Algorithms (DSA). The system helps the users to plan traveling effectively by calculating a minimum spanning tree (MST) of two or more destinations, so that all the cities are linked together with an optimal total cost of travel. The destinations are represented as a graph node, and travel cost as a weighted edge. Prim's algorithm is applied to calculate the MST, which will give an optimal set of paths to use in multi-destination travel. Structured trip management functions are also provided as part of the system that enables users to add cities, route definition, trip enqueuing, and visualization of travel connections. The use of graph theory, adjacency lists, stacks, and queues shows that the theoretical concepts of DSA can be used to address real-life planning issues in the context of traveling.

Keywords: *Travel Itinerary Planner, Data Structures and Algorithms, Minimum Spanning Tree, Prim's algorithm, route definition, graph theory, adjacency lists, stacks, queue, real-life planning.*

Table of Contents

List of Figures

Acronym/Abbreviations

DSA	Data Structures and Algorithms
MST	Minimum Spanning Tree
GUI	Graphical User Interface
RAM	Random Access Memory
GB	Giga Byte
MB	Mega Byte

Chapter 1 Introduction

The Travel Itinerary Planner is a desktop-based tool that helps its users to plan multi-destination trips effectively. Multi-stop trip planning can be a complicated process, and an inefficient choice of route can raise the expenses and time of traveling.
It uses the general ideas of DSA to estimate a minimal-cost network of travel paths, where all destinations are linked using the minimum total travel budget possible.

1.1 Background

As more and more travel options and destinations become available, manual trip planning becomes tedious. Graph theory offers a mathematical approach to such networks: the cities are the nodes; routes are the weighted edges. Prim's Algorithm is a classical method of computing the Minimum Spanning Tree (MST) and is applied to connect all the cities at the minimum total cost. The given mini project implements these concepts of DSA in practice to exhibit the route optimization, dynamic data management, and interactive trip planning.

1.2 Objectives

This mini project has the following objectives:

- To develop and establish a graph-based travel itinerary system.
- To use Prim's Algorithm to calculate the MST of the network.
- To exhibit the practical uses of DSA principles.
- Dynamic management of trip data, stacks, and queues.

1.3 Motivation and Significance

Multi-destination planning may be exhausting and tedious, particularly when the travelers are trying to keep the costs low and also avoid using routes that are inefficient. Choosing the incorrect route to the destinations or unnecessary routes

may result in loss of time, increased costs, and a baffling itinerary. The goal of this project is to develop a structured system that makes travel planning easier by automatically linking all cities with the minimum total travel cost using the MST method. This system not only makes the planning process less demanding and likely to produce errors due to human factors but also illustrates the real-life implementation of the DSA theories.

Chapter 2 Related Works

There are several web resources and applications to plan their travels, optimize their routes, and manage their itineraries, such as:

Tripit: It is a popular itinerary management utility that aids users in managing all the travel reservations in just one ordered design. It is capable of automatically adding flights, hotels, and activities, which is made possible by parsing confirmation emails to enable users to edit their travel itinerary and share it. It also has functionalities like integration with calendar and reminders, and sharing, which can be effective in managing various destinations. This platform shows some practical applicability to the fact that Travel Itinerary Planner can organize the information about the trip and give a complete guide of all places.

Rome2rio: It is a route discovery system that is a globally based route discovery tool that computes the route between any two destinations in the world. It provides a variety of transportation, such as trains, buses, and flights, as well as the approximate time and cost of traveling. The platform gives a concise summary of the potential routes and alternatives for traveling. It is specifically effective in illustrating route planning and cost-conscious traveling control.

Wanderlog: It is an app that enables one to arrange road trip itineraries by location, route, and plan the trips. It has options like multi-destination route overview, joint planning, and optimization of a trip. Users are also able to follow schedules, open trips to others, and change routes in real-time. These characteristics are directly correlated to the MST-based solution of the Travel Itinerary Planner, management of the trip queue, and interactive visualization of the trip, demonstrating the way route optimization can improve the planning process.

2.1 Problem Statement

It is necessary to have a simplified system that will show how the Data Structures and Algorithms (DSA) concepts of graph modeling, adjacency lists, and optimization of a route using Minimum Spanning Tree (MST) algorithms are applied in practice. This

kind of system must enable users to control destinations, calculate the optimal routes to travel, and plan trips effectively in an interactive and understandable way. The Travel Itinerary Planner fills this gap by combining MST-based connectivity of routes, dynamic trip queue-management, and transparent depiction of the travel connections. It offers an efficient resource when it comes to small-scale itinerary planning, as well as a learning resource for applying theoretical DSA concepts to real-world travel issues.

Chapter 3 Design and Implementation

The layout and development of this project were well-considered so that every action led to the appropriate development of the systems. The Qt GUI and the C++ backend are divided into a modular structure. Adjacency lists, stacks, and queues are used to dynamically manage cities and routes, and the action stack is used to allow undo functionality, and the trip queue is used to arrange itineraries. The system is based on MST to compute optimized routes, is used, and all modules are well integrated with the GUI to interactively manage the routes and visualize them.

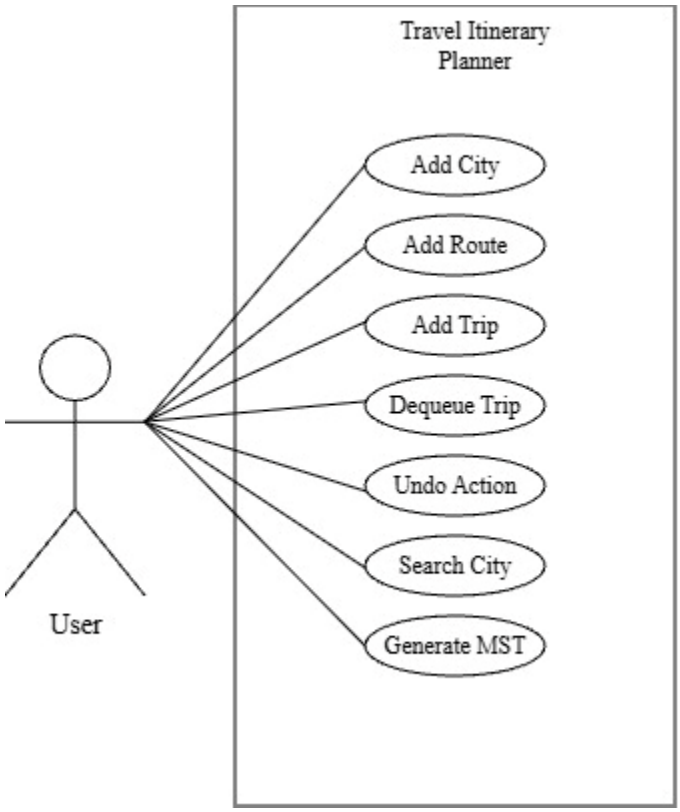

Figure 3.1: Use Case Diagram

This system was created in order to help the user to plan their travels efficiently, optimize routes between the various points of interest, and also to manage the trips through C++ and the Qt framework. The system also has a friendly user interface to manage routes and show the optimum routes on the basis of the graph.

3.1.1 Software Requirements

- **Functional Requirements**
  a. **Graph Constructions:** The system will be based on the definition of destinations as nodes and routes as weighted edges.
  b. **Route Optimizations:** The system will calculate optimum routes based on the MST algorithm.
  c. **Visualizations:** The system will showcase the network of destinations on the Qt interface and indicate the optimal routes.
  d. **Data Management:** The system will be able to store cities and routes efficiently through adjacency lists,
  e. **User Interaction:** The system will have the input forms and keys to add cities, the specifications of routes, and trips.

- **Non-Functional Requirements**
  a. **Performance:** The system will be able to calculate optimal routes within a short duration of time with a maximum of 15 destinations.
  b. **Reliability:** The system will be consistent in its operation without crashing during regular usage.
  c. **Usability:** The system would have an easy-to-use and simple Qt interface.
  d. **Scalability:** The system will support the addition of new cities and routes with a significant redesign.
  e. **Portability:** The system will be based on regular desktop environments with Qt support.

3.1.2 Hardware Requirements

a. Processor: Dual-core processor
b. Memory (RAM): 4 GB
c. Storage: 500 MB free disk space
d. Display: 1280 x 720 resolution
e. Input Devices: Standard keyboard and mouse

Chapter 4 Result and Discussion

The Travel Itinerary Planner was implemented successfully and helped users to plan and optimally travel directions between two or more destinations. The system allows one to add cities, build routes with budget calculations, control trips within a queue, and calculate optimal paths with the help of the MST algorithm. It guarantees effective travel planning, minimizes unnecessary costs in travel, as well as a well-organized visualization of paths. All the significant operations, such as adding cities and routes, enqueuing and dequeuing trips, undo operations, and computing MST paths, were all tested and ran successfully. Adjacency arrays, stacks, and queues are part of the data structures that are integrated to make it dependable and efficient

4.1 Features

4.1.1. Graph-Based Route Management

This tool enables the redesign of routes and enables the graphical representation of routes and their control over the network. The system is a graph of cities and routes in which the edges are weighted, and nodes are represented by cities. This facilitates easy storage and processing of travel information and access to related destinations at a fast rate.

4.1.2. The Minimum Spanning Tree (MST)

The system calculates the best travel links with the MST algorithm. This makes all the chosen destinations interconnected at the lowest possible overall traveling cost, enhancing the efficiency of the route and preventing unnecessary costs.

4.1.3. Trip Queue Management

The planned trips are managed in a queue data structure sequentially. Users are able to add and take away new trips, canceled or completed trips, and retain well-organized itinerary management.

4.1.4. Undo Functionality

Undo operations are provided by an action stack implemented. This enables the user to undo some recent steps (adding cities or routes), which makes it more reliable and eliminates the risk of making accidental errors.

4.1.5. Interactive Qt Interface

The system has a clean and easy-to-use Qt graphical interface. The ability to add destinations, cost route, optimization run, and visualization of results can be easily achieved without the technical difficulties.

4.1.6. Data Integrity and Input Checking

The data used in the system to identify the names of the cities and the cost of the routes is validated to eliminate duplication and the entry of wrong data so that the route management is uniform and correct.

Chapter 5 Conclusion and Recommendations

The Travel Itinerary Planner was then developed successfully to illustrate how Data Structures and Algorithms would be practically applied to the solution of a real-world travel planning problem. Graph structures are used to effectively model destinations as nodes and routes as weighted edges using the system. Using the Minimum Spanning Tree (MST) algorithm, the system will compute optimal travel connections that will result in a minimum overall cost while connecting all destinations.

Adjacency lists, stacks, and queues are integrated so that the data is well managed efficiently, they can be undone, and trips are handled in a structured manner. Living the graphical interface of the application written in Qt makes it more usable with its ability to add cities, define routes, manage trips, and see optimized routes in a visual state. On the whole, the project aims to integrate the theoretical concepts of algorithms with the practical application, which leads to the creation of a working travel planning system and a study.

5.1 Limitations

Despite its successful application, such a system is marked with limitations:

i. The system has been limited to small-to-medium networks.
ii. MST algorithm links the destinations together and does not calculate the shortest path between definite source and destination pairs.
iii. Information on traffic conditions, weather, or dynamic pricing is not provided in real time.
iv. It can be deployed with a limitation on data persistence (when storage is basic in the form of a file).
v. It is only visualized using the desktop Qt interface and lacks sophisticated graphical mapping.

5.2 Future Enhancement

The system has several possibilities to be improved and expanded:

i. Shortest Path Algorithms implementation like Dijkstra algorithm of specific source to destination route and optimization.
ii. Real-time data integration, e.g. traffic updates or cost-change.
iii. Design of graphical visualization of map to have a better use experience.
iv. Diversification to web or mobile platforms to become more accessible.
v. User authentication and managing of trip history.
vi. Addition of budget tracking and travel recommendation availability.

References

[1] Tripit. (n.d.). *Tript: Organize your travel plans.* https://www.tripit.com/web

[2] Rome2rio. (n.d.). *Rome2rio: Discover how to get anywhere.* https://www.rome2rio.com/

[3] Sue Where Why What. (n.d.). *Trip planning websites.* https://www.suewherewhywhat.com/trip-planning-websites

[4] Wanderlog. (n.d.). *Wanderlog: Trip planner and itinerary builder.* https://wanderlog.com/

[5] Google. (n.d.). *Google Maps.* https://www.google.com/maps

[6] Booking,com. (n.d.). *Booking.com.* https://www.booking.com/

[7] Skyscanner. (n.d.). *Skyscanner.* https://www.skyscanner.net/

Appendix

A.1 Adding Cities

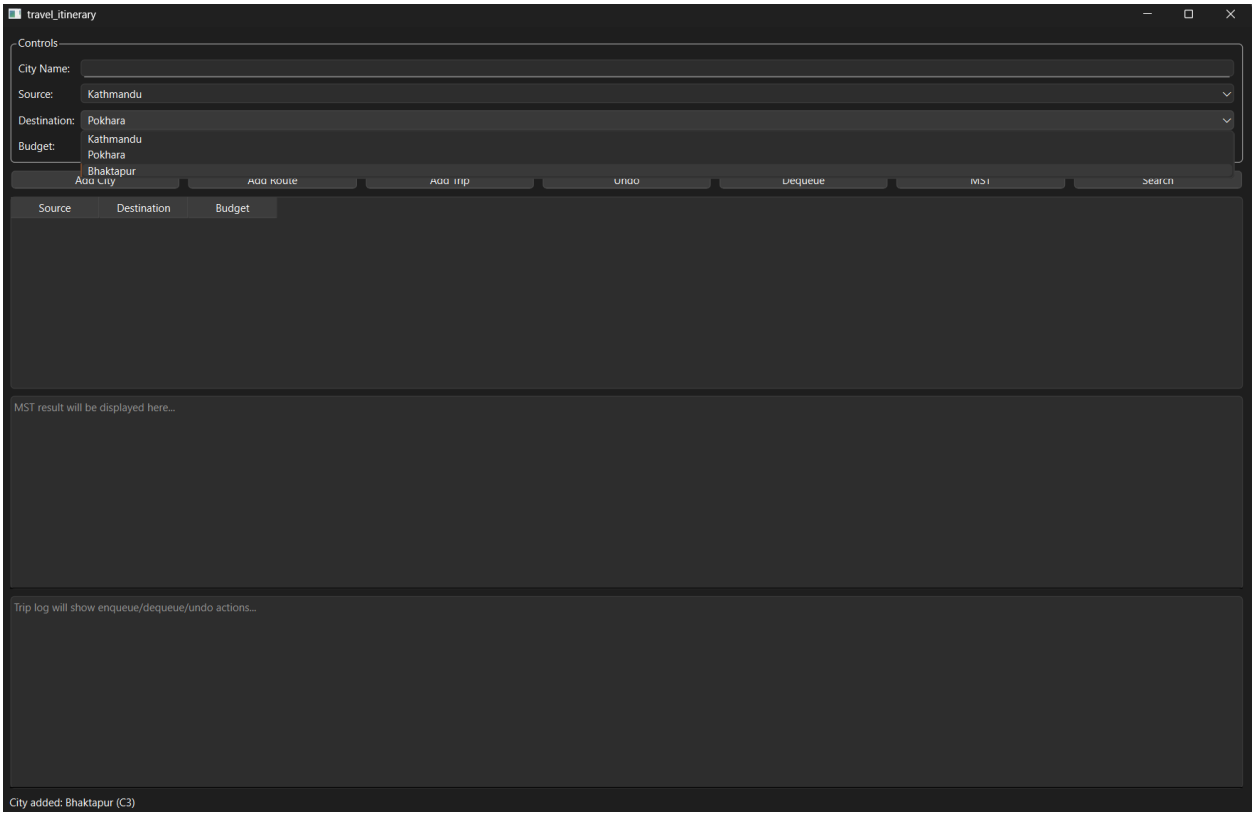

Figure 0.1:Adding Cities

A.2 Adding Route

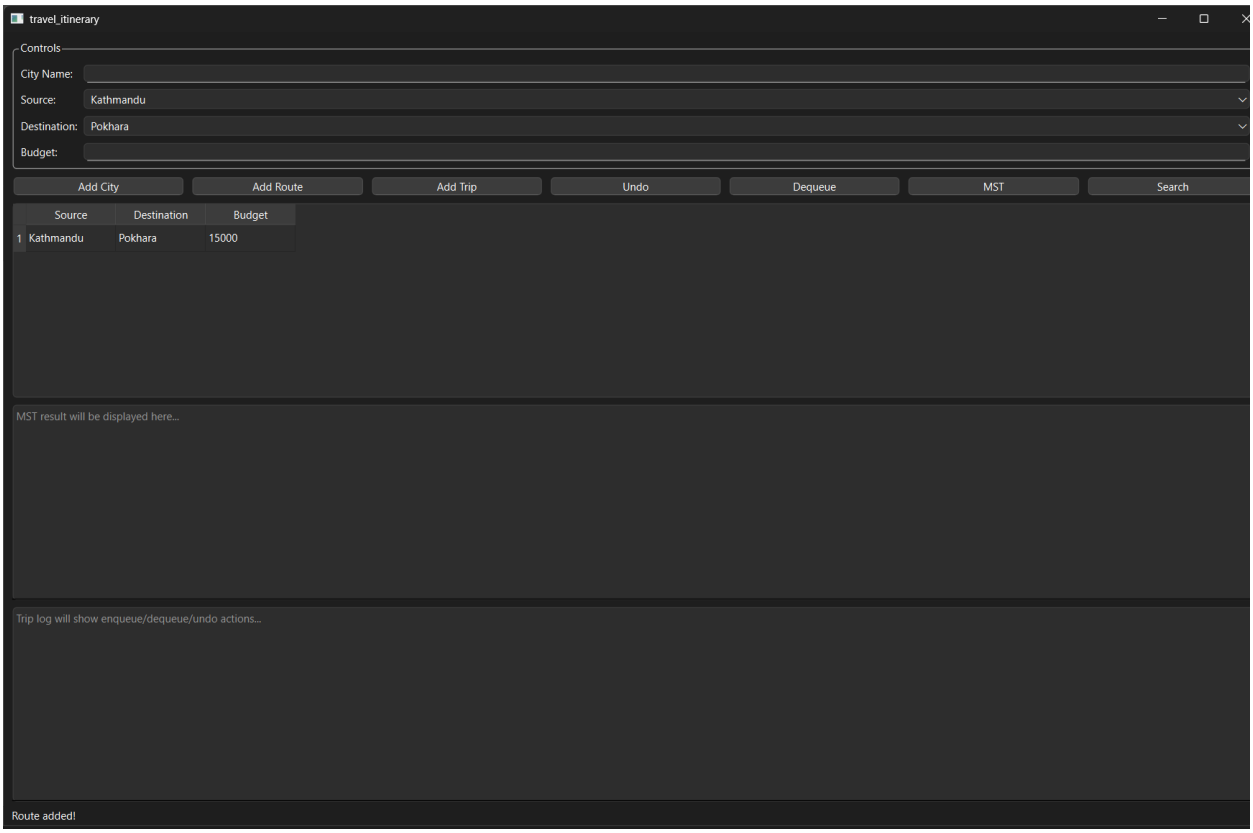

Figure 0.2: Adding Route

A.3 Enqueuing Trip

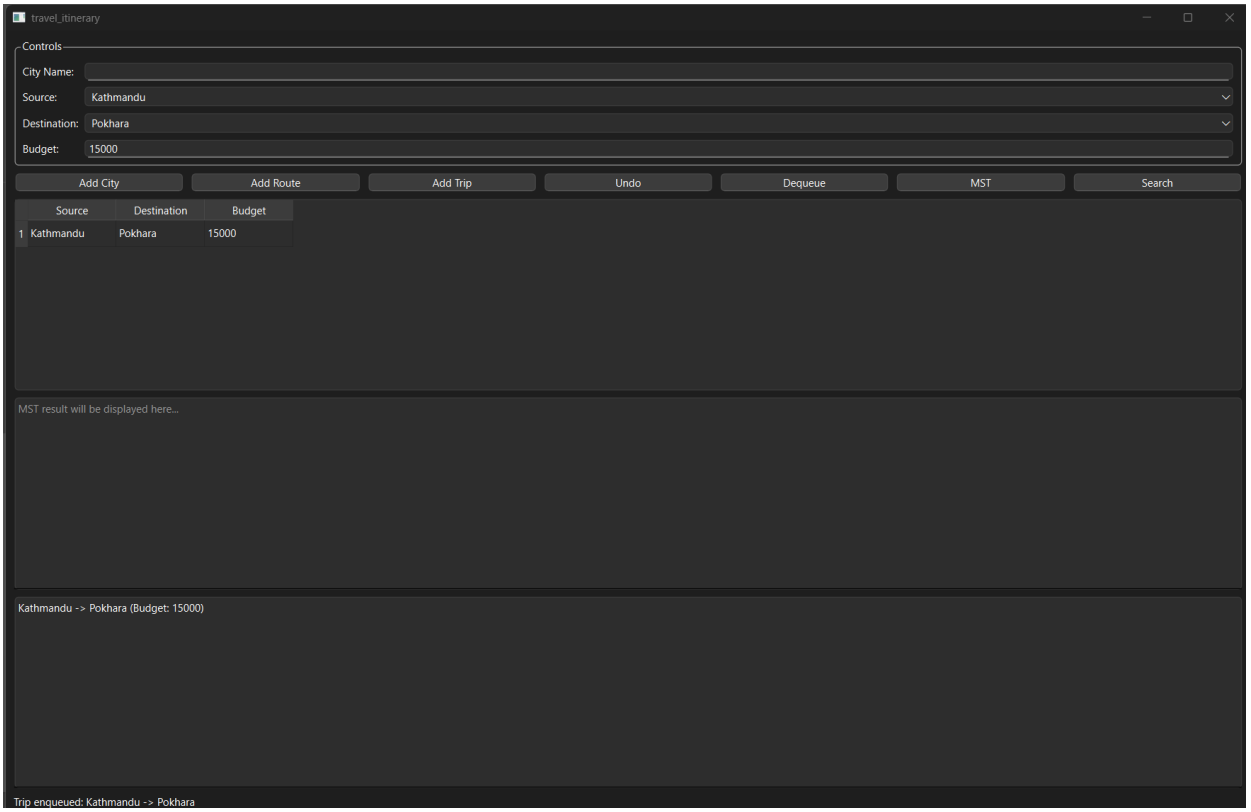

Figure 0.3:Enqueuing Trip

A.4 Minimum Spanning Tree (MST)

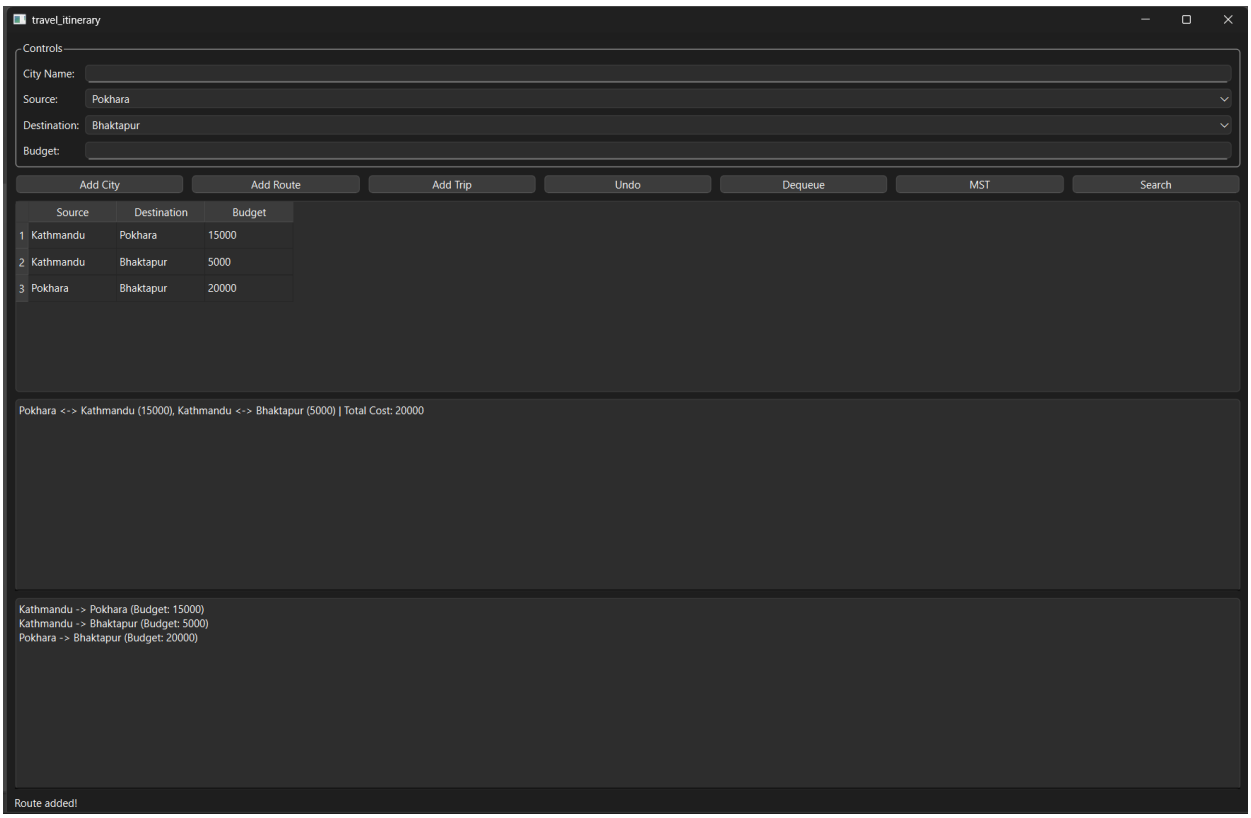

Figure 0.4: Minimum Spanning Tree

A.5 Undo Route

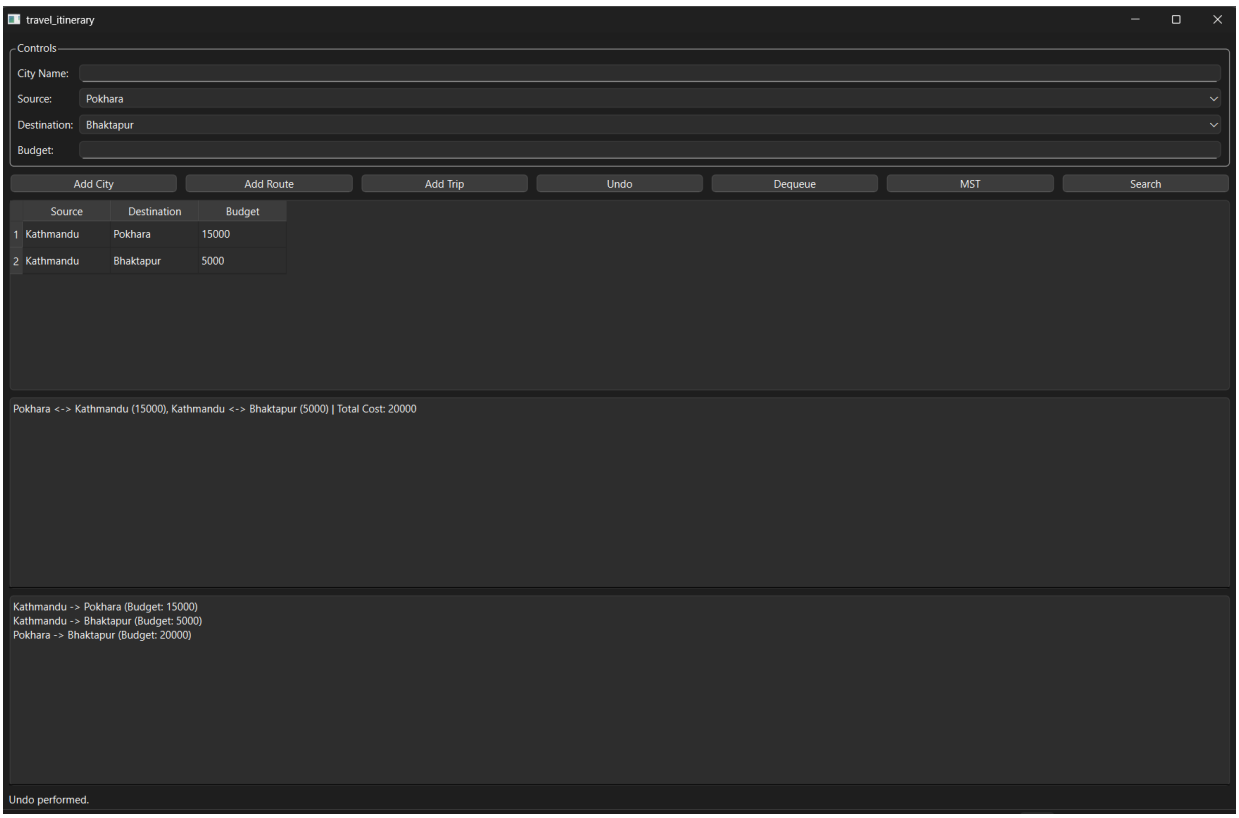

Figure 0.5: Undo Route

A.6 Dequeuing Trip

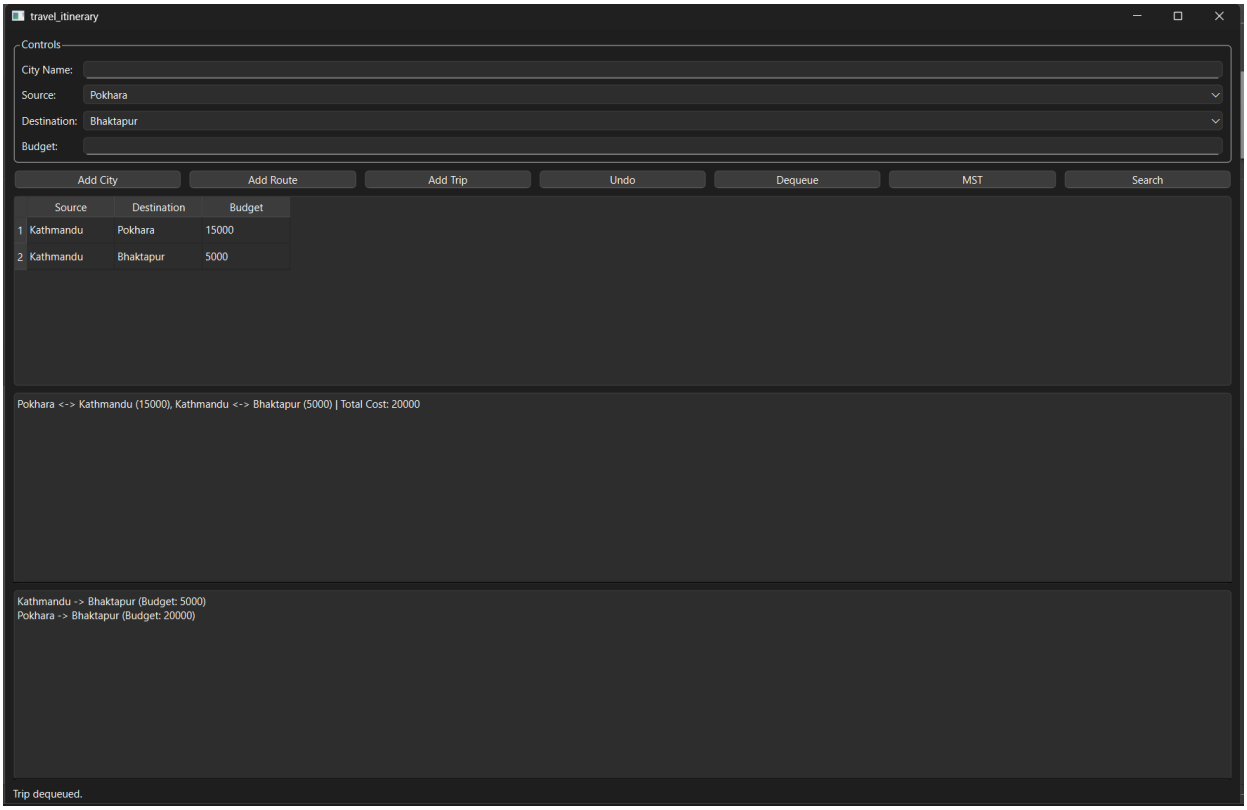

Figure 0.6: Dequeuing Trip

A.7 Searching City

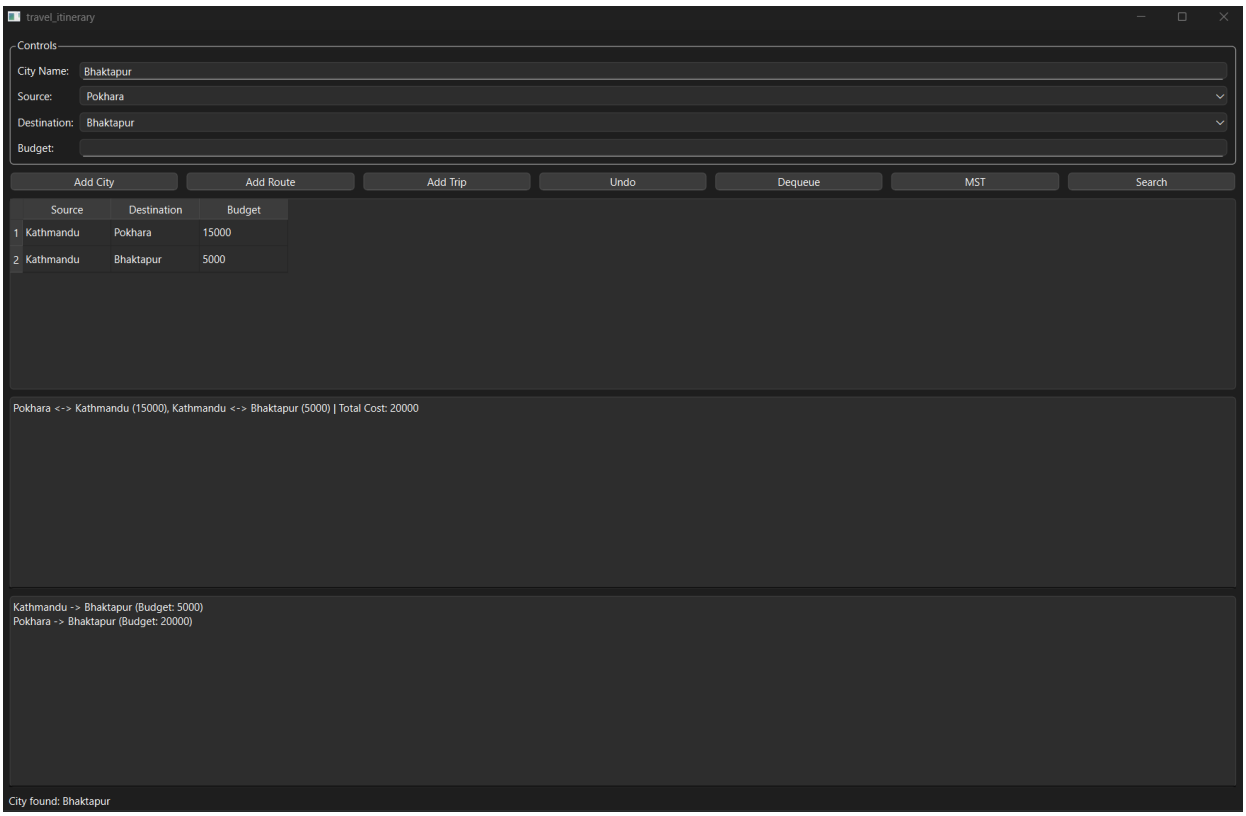

Figure 0.7: Searching City