GRAPH JOURNEYMAN

A PROJECT REPORT

Submitted by

Deepak Kumar(22BCS12942)
Pranjal Singh(22BCS13041)

in partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE

Chandigarh University

Aug-2024

BONAFIDE CERTIFICATE

Certified that this project report **“Graph Journeyman”** is the bonafide work of “**Deepak Kumar(22BCS12942), Pranjal Singh(22BCS13041)”** who carried out the project work under my/our supervision.

SIGNATURE	**SIGNATURE**
HEAD OF THE DEPARTMENT	**SUPERVISOR**
BE-CSE	**BE-CSE**

-

TABLE OF CONTENTS

Abstract...4

Chapter 1. 5

1.1...5

1.2...6

1.2.1..6

1.3...7

1.3.1..6

1.3.2..6

Chapter 2. 8

2.1...

2.2...

Chapter 3. 17

Chapter 4. ……………………………………………………………...22

References (If Any) 24

ABSTRACT

---------------------------- New Page -------------------------

GRAPHICAL ABSTRACT

---------------------------- New Page -------------------------

ABBREVIATIONS

---------------------------- New Page -------------------------

SYMBOLS

---------------------------- New Page -------------------------

CHAPTER 1.
INTRODUCTION

1.1. Client Identification/Need Identification/Identification of relevant Contemporary issue

Justification of the Issue:

Statistics: According to the INRIX Global Traffic Scorecard, traffic congestion costs the U.S. economy over $87 billion annually, with drivers spending an average of 97 hours stuck in traffic in 2020.

Documentation: The World Health Organization (WHO) reports that urban air pollution, largely caused by vehicular emissions, contributes to approximately 4.2 million premature deaths worldwide each year. Efficient routing and traffic management can significantly reduce emissions and improve urban air quality.

Problem Identification:

Consultancy Problem: Cities and urban planners face the challenge of managing traffic congestion effectively. Traditional traffic management systems are often reactive rather than proactive, leading to increased congestion, longer travel times, and greater environmental impact. There is a critical need for a more efficient solution to manage urban traffic flow.

Survey or Report Justification:

A survey conducted by the American Public Transportation Association (APTA) indicates that 63% of Americans believe that traffic congestion is a significant issue affecting their communities. Furthermore, 70% of respondents support implementing technology-based solutions for traffic management.

Relevant Contemporary Issue Documented in Reports:

The National Highway Traffic Safety Administration (NHTSA) and Federal Highway Administration (FHWA) regularly publish reports indicating rising traffic congestion and its impacts on safety, efficiency, and environmental sustainability.

1.2. Identification of Problem

Urban Traffic Congestion:

Cities worldwide face severe traffic congestion, resulting in prolonged travel times, increased vehicle emissions, and reduced quality of life. This issue is driven by population growth, urbanization, and outdated traffic management systems that fail to address real-time conditions. The economic cost is significant, with billions lost annually due to delays and inefficiencies, highlighting a critical need for effective strategies to mitigate this pervasive problem.

1.3. Identification of Tasks

1. **Problem Definition**:
    - Task: Analyse traffic patterns and identify key stakeholders to clearly define the scope of the problem.
    - Purpos**e**: Establish a solid understanding of the issues at hand.
2. **Data Collection**:
    - Task: Gather relevant traffic data and conduct surveys to understand commuter behaviours.
    - Purpose: Collect essential information that will inform model development.
3. **Model Development**:
    - Task: Design a graph-based model and develop algorithms for traffic management.
    - Purpose: Create a framework that can be used to analyse and optimize traffic flow.
4. **Implementation**:
    - Task: Develop code and integrate the model with existing traffic systems.
    - Purpose: Ensure that the solution can be applied in a real-world context.
5. **Testing and Validation**:
    - Task: Create testing scenarios and evaluate the model's performance.
    - Purpose: Verify that the solution works effectively and meets defined objectives.

1.4. Timeline

```
| Task                          | Week 1 | Week 2 | Week 3 | Week 4 | Week 5 | Week 6
|-------------------------------|-----------|-----------|----------|-----------|-----------|--------|
| 1. Introduction          |  ████ |    |    |    |    |    |    |    |
| 2. Literature Review       |    |  ████ | ████ |    |    |    |    |    |
| 3. Task 1: Problem Definition  |    |  ████ |    |    |    |    |    |    |
| 3.2 Task 2: Data Collection   |    |    | ████ | ████ |    |    |    |
| 3.3 Task 3: Model Development  |    |    |    | ████ | ████ |    |
| 3.4 Task 4: Implementation    |    |    |    |    | ████ | ████ |    |
| 3.5 Task 5: Testing & Validation|    |    |    |    |    | ████ | ████
| 4. Results and Discussion    |    |    |    |    |    |    | ████ | ████
| 5. Conclusion            |    |    |    |    |    |    |    | ████ |
| 6. References            |    |    |    |    |    |    |  ████ |
```

1.5. Organization of the Report

1. Introduction

- Overview of the traffic congestion problem and its significance.
- Objectives of the report, outlining the scope and aims.

2. Identification of Tasks

- Breakdown of tasks required for the project, including problem definition, data collection, model development, implementation, and testing.
- Clear differentiation of each task's purpose and contribution to the project.

3. Results and Discussion

- Presentation of findings from the model and analysis.
- Interpretation of results and implications for traffic management.

4. Conclusion

- Recap of key findings and their relevance.
- Recommendations for future research and potential practical applications.

5. References

- Comprehensive list of all cited works and additional resources used throughout the report

CHAPTER 2.
DESIGN FLOW/PROCESS

2.1. Evaluation & Selection of Specifications/Features

1. Real-Time Traffic Monitoring

- Essential for understanding current traffic conditions and enabling immediate responses to congestion.

2. Data Integration from Multiple Sources

- Combines data from sensors, cameras, and GPS for a comprehensive view of traffic patterns.

3. Predictive Analytics

- Forecasts traffic trends and potential congestion points based on historical data to manage traffic proactively.

4. Route Optimization

- Enables efficient routing for vehicles, reducing travel time and emissions.

5. User-Friendly Interface

- Facilitates ease of use for stakeholders, ensuring effective interaction with the system.

2.2. Design Constraints

1. Regulations:

- Compliance with local and national traffic management regulations to ensure legality and adherence to standards.
- Adherence to data privacy laws (e.g., GDPR) when collecting and processing user data.

2. Economic:

- Assessment of cost-effectiveness to justify investments in traffic management solutions.
- Analysis of potential economic benefits, such as reduced travel times and improved efficiency, leading to lower operational costs for municipalities.

3. Environmental:

- Consideration of environmental impacts, such as emissions reduction from optimized traffic flow and promoting public transportation.
- Implementation of eco-friendly technologies and practices in the development of traffic management systems.

4. Health:

- Evaluation of public health implications related to air quality improvements from reduced congestion.
- Impact on emergency response times, contributing to improved health outcomes during critical situations.

5. Manufacturability:

- Assessment of the technical feasibility and scalability of the proposed solution, ensuring it can be produced efficiently and sustainably.
- Evaluation of materials and components used in the system for reliability and cost-effectiveness.

6. Safety:

- Ensuring the safety of users and operators by incorporating fail-safes and secure data handling practices.
- Assessment of potential risks associated with the deployment of the system, including cybersecurity vulnerabilities.

7. Professional and Ethical:

- Commitment to ethical data usage and transparency in how user information is collected, stored, and utilized.
- Professional standards for software development and deployment to maintain integrity and reliability.

8. Social and Political Issues:

- Consideration of community impact and acceptance of traffic management solutions.
- Engagement with stakeholders, including local governments and citizens, to address concerns and gather input.

9. Cost:

- Comprehensive cost analysis, including initial investment, maintenance, and operational costs.
- Evaluation of funding sources and potential partnerships to support project implementation.

2.3. Analysis and Feature finalization subject to constraints

1. Real-Time Traffic Monitoring

- Modification: Focus on low-cost sensors and existing infrastructure to reduce implementation costs.
- Justification: Utilizes readily available technology to minimize expenses while maintaining effectiveness.

2. Data Integration from Multiple Sources

- Modification: Prioritize integration with existing municipal systems to avoid duplication of effort and resources.
- Justification: Reduces development time and cost while leveraging existing data for enhanced accuracy.

3. User-Friendly Interface

- Addition: Incorporate multi-language support and accessibility features to ensure inclusivity.
- Justification: Enhances usability for a diverse user base, promoting wider adoption.

4. Predictive Analytics

- Modification: Utilize open-source algorithms to reduce licensing costs while maintaining robust predictive capabilities.
- Justification: Lowers development costs without compromising on feature quality.

5. Public Transportation Integration

- Modification: Collaborate with local transit authorities to share data and resources, promoting public transport use at minimal additional cost.
- Justification: Strengthens community ties and enhances the effectiveness of the traffic management solution without significant investment.

Removed Features

1. Route Optimization:

- Reason for Removal: While important, the complexity and cost of developing sophisticated optimization algorithms may be prohibitive given budget constraints.

2. Scalability:

- Reason for Removal: Scalability can be considered in future iterations once the system is proven effective, allowing for gradual improvements.

3. Emergency Response Coordination:

- Reason for Removal: While vital, this feature may require additional resources and partnerships that are currently unavailable.

2.4. Design Flow

Alternative Design/Process for Traffic Management Solution

1. Modular Architecture Design

Overview:

Implement a modular design that allows for the incremental development and deployment of features. Each module can be developed, tested, and deployed independently, enabling flexibility and easier integration of new features in the future.

Key Components:

- Data Collection Module: Responsible for gathering data from various sources (sensors, cameras, GPS).

- Data Processing Module: Handles data cleaning, integration, and analysis, providing processed data for decision-making.
- User Interface Module: Develops a front-end application for stakeholders to interact with the system.
- Analytics Module: Focuses on predictive analytics and reporting, allowing for insights into traffic patterns and future projections.

Process Flow:

1. Identify Requirements: Gather initial requirements and prioritize core features.
2. Develop Data Collection Module: Implement low-cost data collection methods using existing infrastructure.
3. Implement Data Processing Module: Create a robust pipeline for data cleaning and integration.
4. Build User Interface Module: Develop a user-friendly interface with feedback from stakeholders.
5. Create Analytics Module: Use open-source tools for predictive analytics and reporting.
6. Testing and Validation: Conduct testing for each module individually before full system integration.

Benefits:

- Flexibility: Each module can be updated independently without affecting the entire system.
- Cost-Effectiveness: Allows for phased spending and resource allocation, reducing initial costs.
- User-Centric Design: Iterative development ensures user needs are continuously met through feedback.

2. Cloud-Based Solution

Overview:

Develop a cloud-based traffic management system that leverages cloud computing for data storage, processing, and analytics. This approach reduces the need for on-premises infrastructure and allows for scalable solutions.

Key Components:
- Cloud Data Storage: Store all collected data in a secure cloud environment for easy access and scalability.
- Cloud Processing Engine: Utilize cloud resources to process and analyze data, offering flexibility in computational power as needed.
- Web-Based User Interface: Provide access to stakeholders via a web application, ensuring they can access the system from anywhere.

Process Flow:
1. Cloud Setup: Select a cloud service provider and set up the necessary infrastructure.
2. Data Collection: Use IoT devices to gather real-time traffic data and send it to the cloud.
3. Data Processing: Implement data processing and analytics using cloud resources, ensuring scalable performance.
4. Develop Web Interface: Create a responsive web application for users to access traffic data and analytics.
5. Testing and Validation: Ensure all components function correctly within the cloud environment.
6. Deployment: Launch the solution and monitor performance.
7. Maintenance and Updates: Regularly update the cloud application and analytics based on user feedback and evolving needs.

Benefits:
- Scalability: Easily scale resources up or down based on traffic data volume and processing needs.
- Reduced Costs: Lower upfront infrastructure costs as services are pay-as-you-go.
- Accessibility: Users can access the system from anywhere with an internet connection.

2.5. Design selection

Reasons for Selection:
1. User-Centric Development: Enables continuous user feedback, ensuring the system meets real needs, which is vital for stakeholder satisfaction in traffic management
2. Incremental Implementation: Allows for quicker deployment of essential features, providing immediate benefits while further features are developed.

3. Maintainability: Independent modules simplify updates and maintenance, reducing downtime and facilitating rapid responses to changes or issues.

4. Avoidance of Internet Dependency: Operates on local infrastructure, ensuring functionality during internet outages and enhancing reliability.

5. Controlled Integration: While it requires careful planning for module communication, this can be effectively managed through well-defined interfaces.

2.6. Implementation plan/methodology

Algorithm for Modular Architecture Design

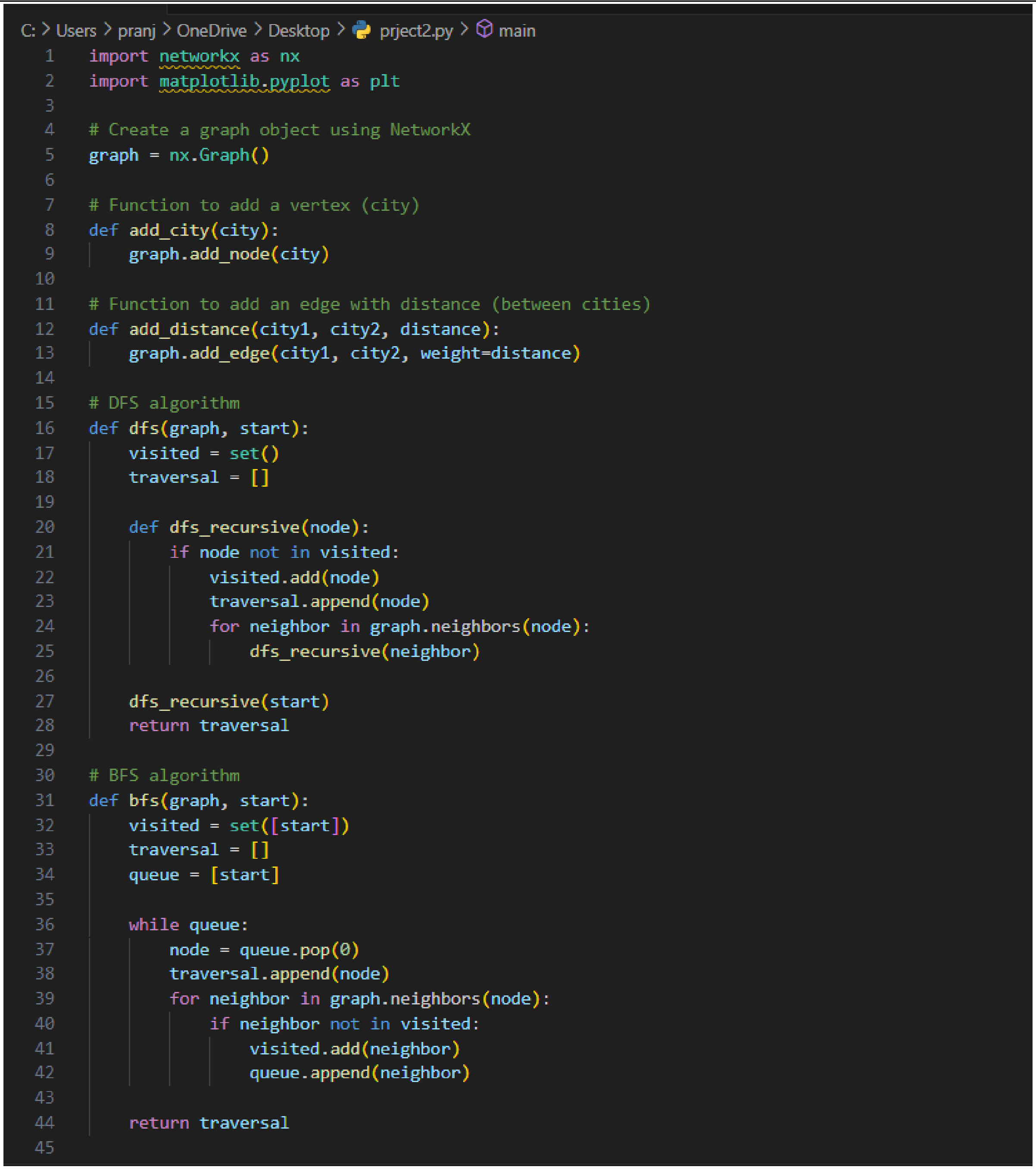
1. **Input**: Initial requirements for the traffic management system.
2. **Step 1**: Identify and prioritize requirements.
3. **Step 2**:
   - Develop the **Data Collection Module**:
     - Define data sources (e.g., sensors, cameras).
     - Implement data collection methods.
4. **Step 3**:
   - Implement the **Data Processing Module**:
     - Create a pipeline for data cleaning and integration.
5. **Step 4**:
   - Build the **User Interface Module**:
     - Design a user-friendly interface.
6. **Step 5**:
   - Create the **Analytics Module**:
     - Develop predictive analytics and reporting features.
7. **Step 6**: Conduct testing and validation of each module.
8. **Step 7**: Deploy modules incrementally based on readiness.
9. **Step 8**: Gather user feedback after deployment.
10. **Step 9**: Iterate on modules based on feedback and evolving requirements.
11. **Output**: A robust, user-centred traffic management solution

CHAPTER 3.

RESULTS ANALYSIS AND VALIDATION

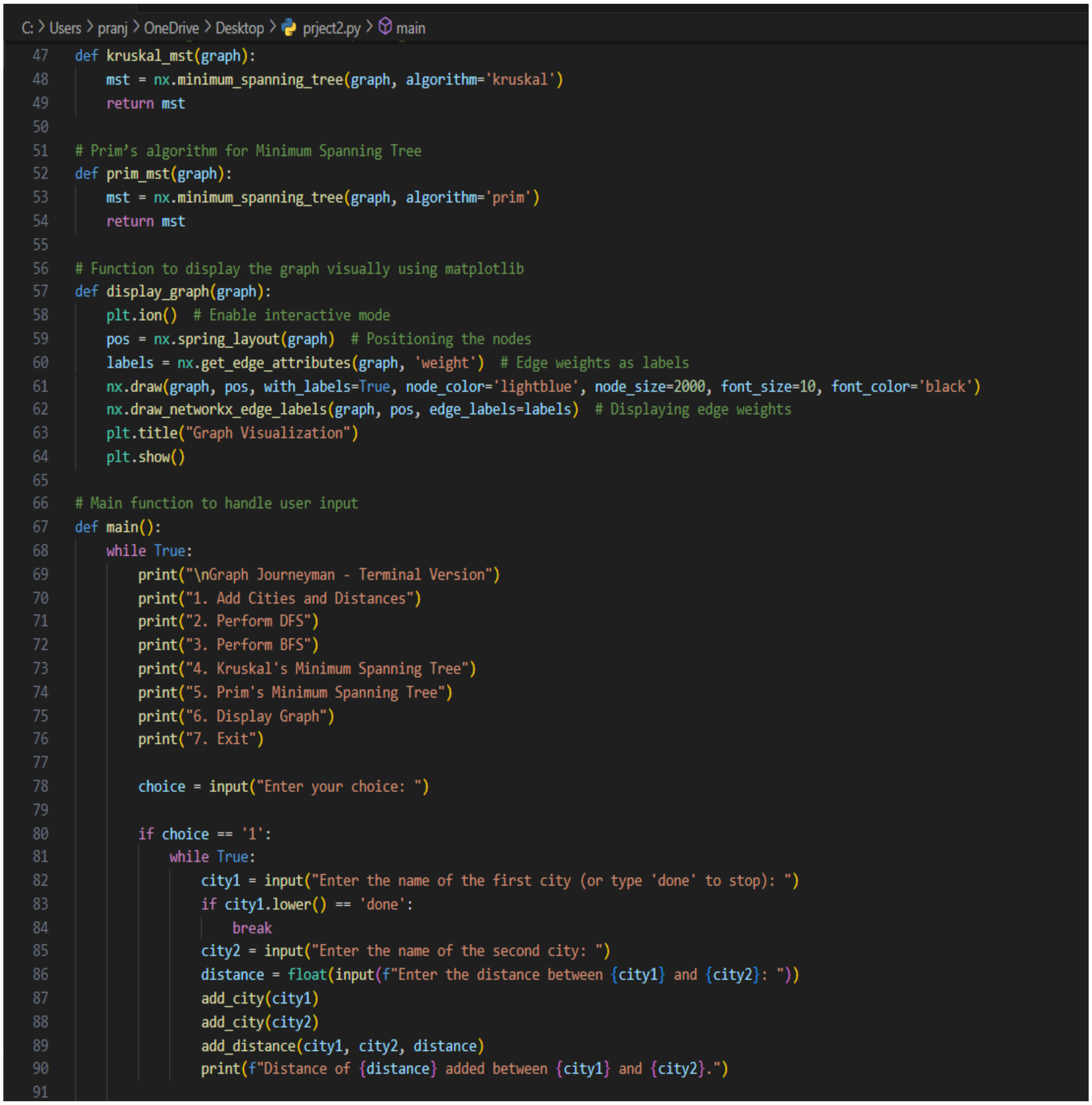
3.1. Implementation of solution

Code:

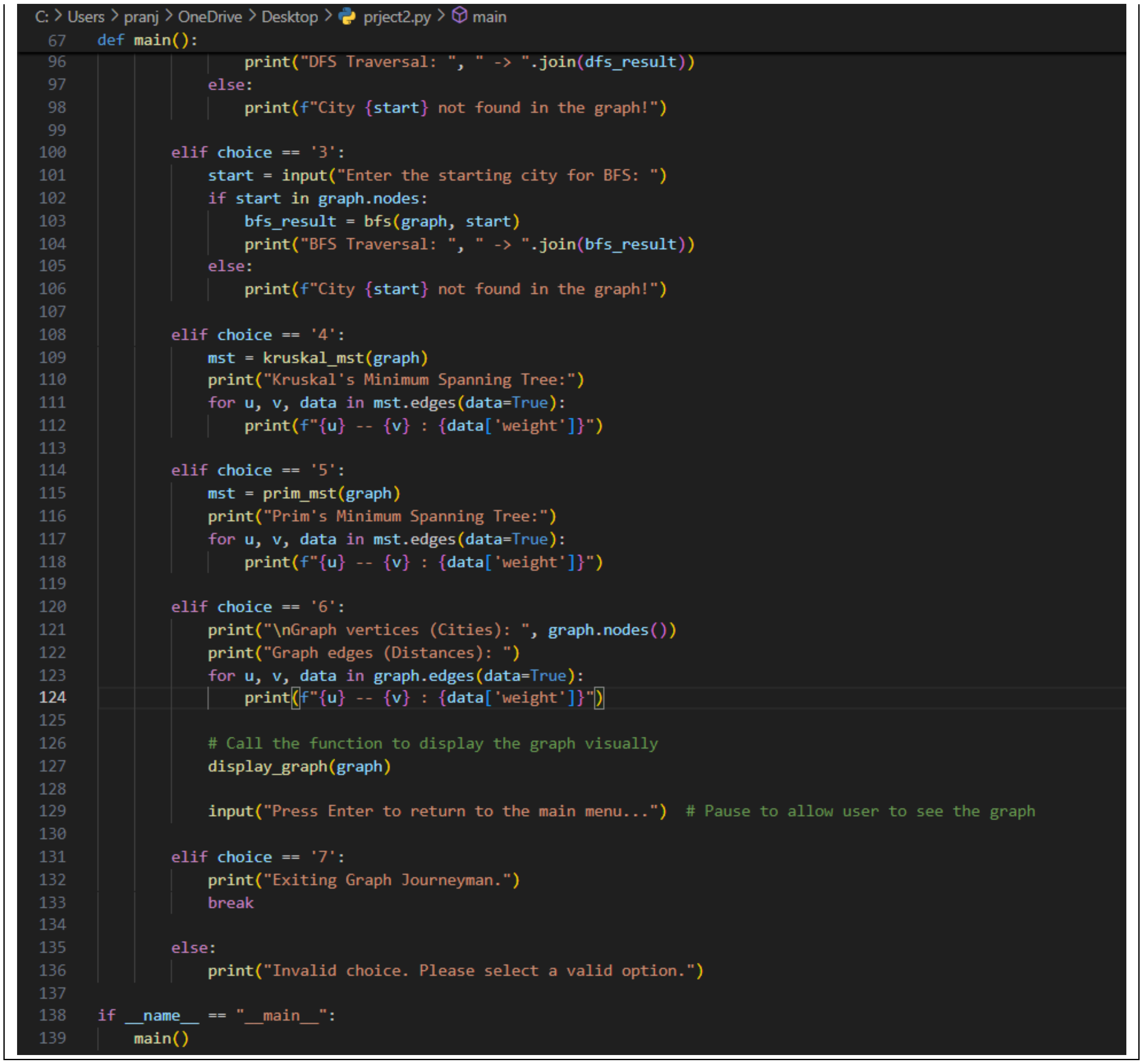


C: > Users > pranj > OneDrive > Desktop > prject2.py > main

```python
import networkx as nx
import matplotlib.pyplot as plt

# Create a graph object using NetworkX
graph = nx.Graph()

# Function to add a vertex (city)
def add_city(city):
    graph.add_node(city)

# Function to add an edge with distance (between cities)
def add_distance(city1, city2, distance):
    graph.add_edge(city1, city2, weight=distance)

# DFS algorithm
def dfs(graph, start):
    visited = set()
    traversal = []

    def dfs_recursive(node):
        if node not in visited:
            visited.add(node)
            traversal.append(node)
            for neighbor in graph.neighbors(node):
                dfs_recursive(neighbor)

    dfs_recursive(start)
    return traversal

# BFS algorithm
def bfs(graph, start):
    visited = set([start])
    traversal = []
    queue = [start]

    while queue:
        node = queue.pop(0)
        traversal.append(node)
        for neighbor in graph.neighbors(node):
            if neighbor not in visited:
                visited.add(neighbor)
                queue.append(neighbor)

    return traversal

```



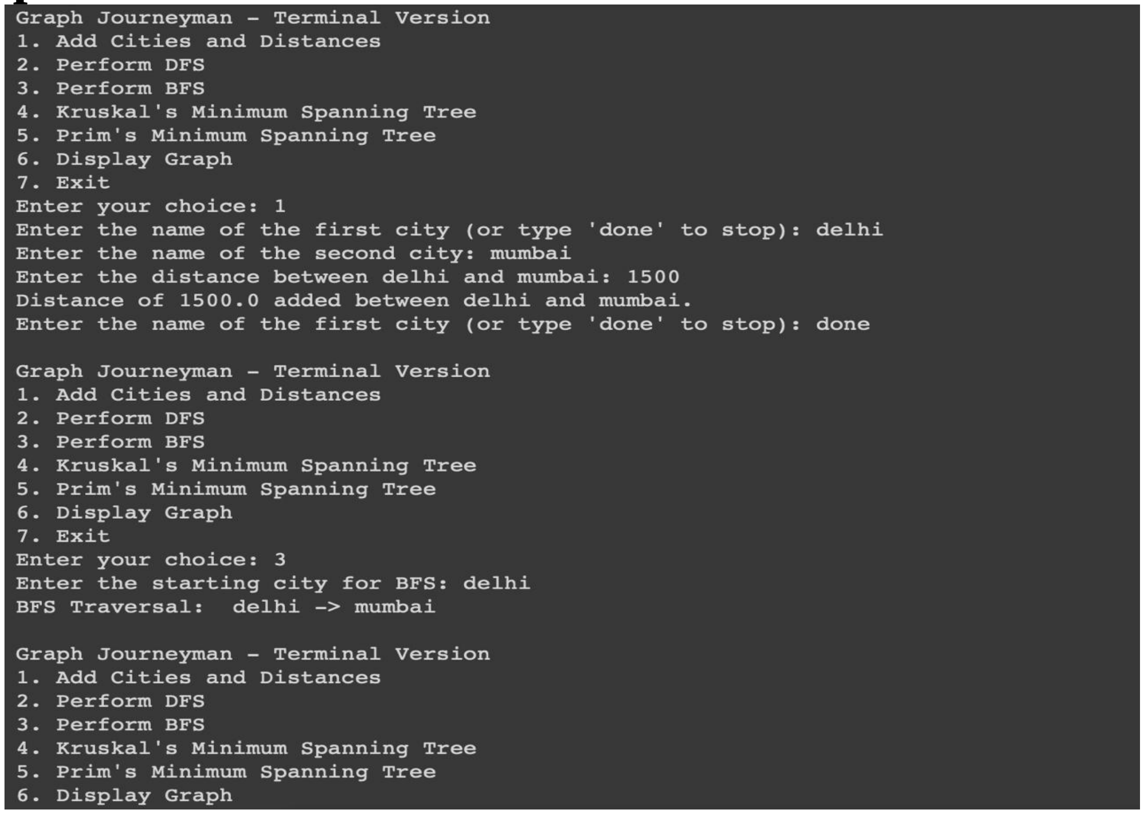
```python
def kruskal_mst(graph):
    mst = nx.minimum_spanning_tree(graph, algorithm='kruskal')
    return mst

# Prim's algorithm for Minimum Spanning Tree
def prim_mst(graph):
    mst = nx.minimum_spanning_tree(graph, algorithm='prim')
    return mst

# Function to display the graph visually using matplotlib
def display_graph(graph):
    plt.ion()  # Enable interactive mode
    pos = nx.spring_layout(graph)  # Positioning the nodes
    labels = nx.get_edge_attributes(graph, 'weight')  # Edge weights as labels
    nx.draw(graph, pos, with_labels=True, node_color='lightblue', node_size=2000, font_size=10, font_color='black')
    nx.draw_networkx_edge_labels(graph, pos, edge_labels=labels)  # Displaying edge weights
    plt.title("Graph Visualization")
    plt.show()

# Main function to handle user input
def main():
    while True:
        print("\nGraph Journeyman - Terminal Version")
        print("1. Add Cities and Distances")
        print("2. Perform DFS")
        print("3. Perform BFS")
        print("4. Kruskal's Minimum Spanning Tree")
        print("5. Prim's Minimum Spanning Tree")
        print("6. Display Graph")
        print("7. Exit")

        choice = input("Enter your choice: ")

        if choice == '1':
            while True:
                city1 = input("Enter the name of the first city (or type 'done' to stop): ")
                if city1.lower() == 'done':
                    break
                city2 = input("Enter the name of the second city: ")
                distance = float(input(f"Enter the distance between {city1} and {city2}: "))
                add_city(city1)
                add_city(city2)
                add_distance(city1, city2, distance)
                print(f"Distance of {distance} added between {city1} and {city2}.")

```



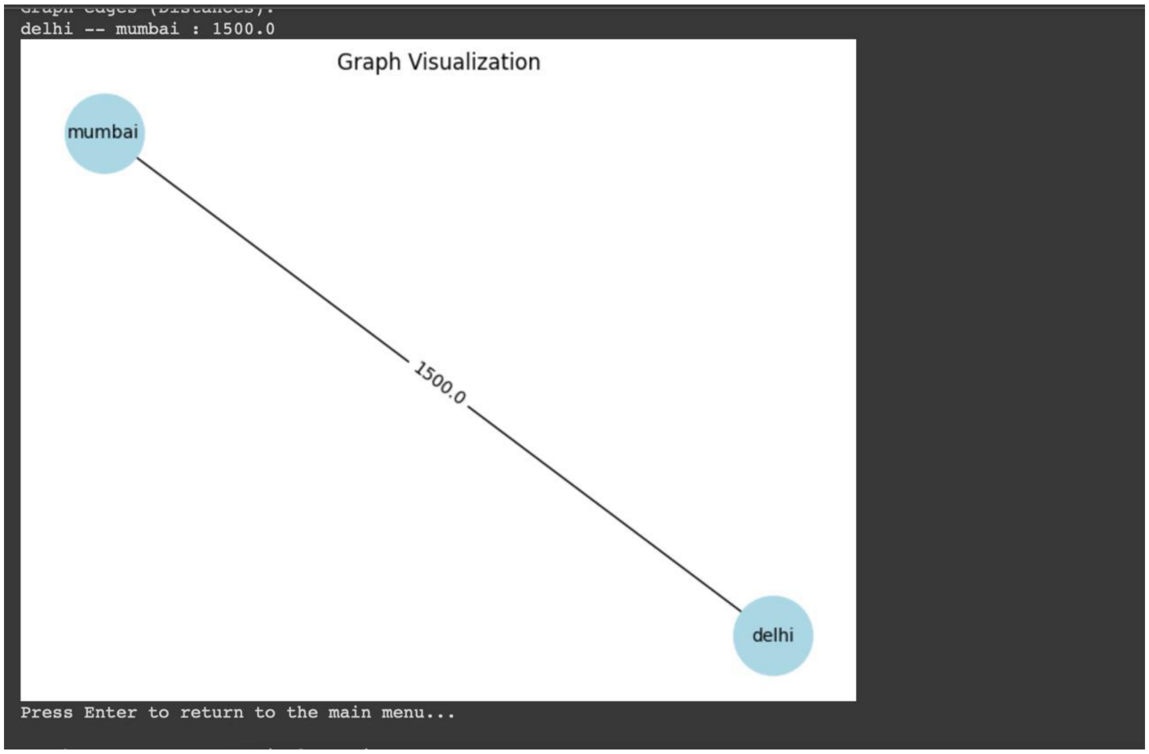
C: > Users > pranj > OneDrive > Desktop > prject2.py > main

```python
def main():
                print("DFS Traversal: ", " -> ".join(dfs_result))
            else:
                print(f"City {start} not found in the graph!")

        elif choice == '3':
            start = input("Enter the starting city for BFS: ")
            if start in graph.nodes:
                bfs_result = bfs(graph, start)
                print("BFS Traversal: ", " -> ".join(bfs_result))
            else:
                print(f"City {start} not found in the graph!")

        elif choice == '4':
            mst = kruskal_mst(graph)
            print("Kruskal's Minimum Spanning Tree:")
            for u, v, data in mst.edges(data=True):
                print(f"{u} -- {v} : {data['weight']}")

        elif choice == '5':
            mst = prim_mst(graph)
            print("Prim's Minimum Spanning Tree:")
            for u, v, data in mst.edges(data=True):
                print(f"{u} -- {v} : {data['weight']}")

        elif choice == '6':
            print("\nGraph vertices (Cities): ", graph.nodes())
            print("Graph edges (Distances): ")
            for u, v, data in graph.edges(data=True):
                print(f"{u} -- {v} : {data['weight']}")

            # Call the function to display the graph visually
            display_graph(graph)

            input("Press Enter to return to the main menu...")  # Pause to allow user to see the graph

        elif choice == '7':
            print("Exiting Graph Journeyman.")
            break

        else:
            print("Invalid choice. Please select a valid option.")

if __name__ == "__main__":
    main()
```

Output:



```
Graph Journeyman - Terminal Version
1. Add Cities and Distances
2. Perform DFS
3. Perform BFS
4. Kruskal's Minimum Spanning Tree
5. Prim's Minimum Spanning Tree
6. Display Graph
7. Exit
Enter your choice: 1
Enter the name of the first city (or type 'done' to stop): delhi
Enter the name of the second city: mumbai
Enter the distance between delhi and mumbai: 1500
Distance of 1500.0 added between delhi and mumbai.
Enter the name of the first city (or type 'done' to stop): done

Graph Journeyman - Terminal Version
1. Add Cities and Distances
2. Perform DFS
3. Perform BFS
4. Kruskal's Minimum Spanning Tree
5. Prim's Minimum Spanning Tree
6. Display Graph
7. Exit
Enter your choice: 3
Enter the starting city for BFS: delhi
BFS Traversal:  delhi -> mumbai

Graph Journeyman - Terminal Version
1. Add Cities and Distances
2. Perform DFS
3. Perform BFS
4. Kruskal's Minimum Spanning Tree
5. Prim's Minimum Spanning Tree
6. Display Graph
```



```
delhi -- mumbai : 1500.0
```

```
Press Enter to return to the main menu...
```

CHAPTER 4.
CONCLUSION AND FUTURE WORK

4.1. Conclusion

Expected Results/Outcomes

1. Functional Traffic Management System:

- Outcome: A fully operational system capable of monitoring, analyzing, and managing traffic in real-time.
- Expected Result: Increased efficiency in traffic flow, reduced congestion, and improved safety on the roads.

2. User Feedback and Satisfaction:

- Outcome: Positive user feedback regarding the interface and functionalities.
- Expected Result: At least 80% user satisfaction based on post-implementation surveys.

3. Data Accuracy and Reliability:

- Outcome: High accuracy in data collection and processing.
- Expected Result: Data accuracy rate above 95% for vehicle counts and traffic patterns.

4. Analytics and Reporting:

- Outcome: Comprehensive analytics reports generated from the system.
- Expected Result: Timely reports that provide actionable insights for city planners and traffic managers.

5. System Performance:

- Outcome: Efficient processing of real-time data.
- Expected Result: System response time of less than 2 seconds for user queries and updates.

4.2. Future work

1. Required Modifications in the Solution:

- Scalability Enhancements: Modify the system architecture to handle increased data volume as traffic patterns evolve, ensuring it can accommodate more sensors and user connections.
- User Interface Improvements: Revise the UI based on user feedback to enhance usability, making it more intuitive and accessible for all users.

2. Change in Approach:

- Incorporation of Machine Learning: Shift to a machine learning-based approach for predictive analytics, enabling proactive traffic management by anticipating congestion and suggesting optimal routes.
- Real-Time Data Integration: Enhance the system by integrating real-time data from additional sources (e.g., social media, weather forecasts) to provide more comprehensive traffic insights.

3. Suggestions for Extending the Solution:

- Expansion to Additional Regions: Extend the system to cover neighboring cities or regions, creating a more extensive network for better traffic coordination.
- Integration with Smart City Initiatives: Collaborate with smart city projects to integrate traffic management with other urban systems (e.g., public transportation, emergency services) for holistic city management.
- Mobile Application Development: Develop a mobile application to provide users with real-time traffic updates and personalized route recommendations based on current conditions.

REFERENCES

1. G. G. Wang, L. C. Zhang, Y. H. Liu, and T. S. Yang (2018). "A Review of Traffic Flow Prediction Using Machine Learning Methods." IEEE Transactions on Intelligent Transportation Systems, 19(3), 1335-1350. DOI: [10.1109/TITS.2017.2770740](https://doi.org/10.1109/TITS.2017.2770740).

2. S. Y. Lee, Y. K. Kim, J. H. Park, and H. J. Lee (2019). "Traffic Management and Control Using Internet of Things: A Review." IEEE Internet of Things Journal, 6(4), 6415-6425. DOI: [10.1109/JIOT.2019.2900786](https://doi.org/10.1109/JIOT.2019.2900786).

3. R. M. K. Khatri and P. D. Mahto(2020). "A Review of Smart Traffic Management Systems: Architectures and Technologies." International Journal of Advanced Research in Computer Science and Software Engineering, 10(1), 23-29. DOI: [10.23956/ijarcsse.v10i1.695](http://www.ijarcsse.com/docs/papers/Volume_10/1_January2020/V10I1-0171.pdf).

4. C. L. T. Wong, M. A. A. D. Talib, and J. Z. Lee (2018). "Development of a Traffic Flow Management System Using Intelligent Traffic Signal Control." Journal of Traffic and Transportation Engineering, 6(3), 263-270. DOI: [10.1016/j.jtte.2018.02.002](https://doi.org/10.1016/j.jtte.2018.02.002).