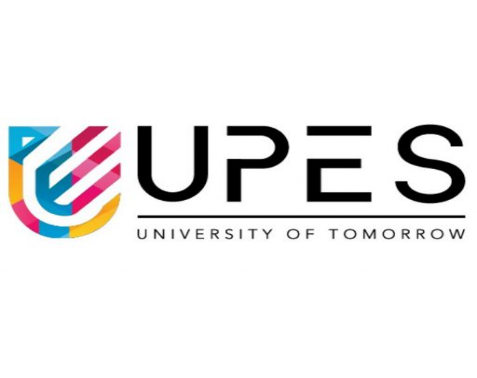
**Software Requirements Specification**

For

**PathSmart-Optimized Routes System**

Prepared by

|  |  |  |
| --- | --- | --- |
| **Specialization** | **SAP ID** | **Name** |
| CSE AIML | 500106274 | Harsh Bhartia |
| CSE AIML | 500107420 | Shailav Justa |
| CSE AIML | 500109572 | Vinayak Singh |
|  |  |  |



School of Computer Science

UNIVERSITY OF PETROLEUM & ENERGY STUDIES,

DEHRADUN- 248007. Uttarakhand

Table of Contents

|  |  |  |
| --- | --- | --- |
| **Topic** | | **Page No** |
| 1 | Introduction | 3 |
|  | 1.1 Purpose of the Project | 3 |
|  | 1.2 Target Beneficiary | 3 |
|  | 1.3 Project Scope | 3 |
|  | 1.4 References | 4 |
| 2 | Project Description | 5 |
|  | 2.1 Reference Algorithm | 5 |
|  | 2.2 Data/ Data structure | 5 |
|  | 2.3 SWOT Analysis | 5 |
|  | 2.4 Project Features | 6 |
|  | 2.5 User Classes and Characteristics | 6 |
|  | 2.6 Design and Implementation Constraints | 6 |
|  | 2.7 Design diagrams | 7 |
|  | 2.8 Assumption and Dependencies | 9 |
| 3 | System Requirements | 9 |
|  | 3.1 User Interface | 9 |
|  | 3.2 Software Interface | 9 |
|  | 3.3 Database Interface | 9 |
|  | 3.4 Protocols | 9 |
| 4 | Non-functional Requirements | 10 |
|  | 4.1 Performance requirements | 10 |
|  | 4.2 Security requirements | 10 |
|  | 4.3 Software Quality Attributes | 10 |
| 5 | Other Requirements | 11 |
| Appendix A: Glossary | | 11 |

**1. Introduction**

**1.1 Purpose of the Project**

The primary objective of an optimized routing system and navigation is to efficiently determine the best possible routes for transportation or data transmission by considering variable, such as distance. The goal is to minimize delays, fuel consumption, and resource use while maximizing convenience, safety, and operational efficiency.

Specifically, the objectives include:

1.Minimization of Travel Time and Distance: To calculate and recommend the fastest or shortest paths between origin and destination, ensuring users or vehicles spend the least time in transit.

2.Real-Time Adaptability: To provide dynamic routing that can adjust to real-time conditions such as traffic congestion, road closures, accidents, or weather changes, ensuring continued optimization during the journey.

**1.2 Target Beneficiary**

1. Businesses & Enterprises: Companies looking to streamline their document delivery processes, improving efficiency, and reducing costs.

2. Courier & Delivery Services: Companies that specialize in physical document transportation and need to optimize their routes for faster delivery.

3. Government Agencies: Organizations responsible for the distribution of official documents (e.g., legal, public records) that require timely, cost-effective delivery.

4. Legal Firms & Law Offices: Firms that handle important legal documents that must be delivered quickly and securely.

5. Educational Institutions: Schools and universities that need efficient systems for distributing academic records, certificates, and other documents.

**1.3 Project Scope**

The project’s scope includes:

1. Route Optimization: Developing algorithms to find the most efficient delivery routes, minimizing travel time, fuel consumption, and costs.

2. Real-Time Tracking: Integrating GPS and tracking systems for real-time monitoring of deliveries.

3. Dynamic Scheduling: Creating a scheduling system that adjusts to traffic conditions, delivery priorities, and other real-time variables.

4. User Interface (UI): Designing an easy-to-use interface for operators to input delivery details and monitor progress.

**1.4 References**

1. **Dijkstra, E. W.** (1959). *A Note on Two Problems in Connexion with Graphs*. Numerische Mathematik, 1, 269-271. doi:10.1007/BF01386390.
2. **Hart, P. E., Nilsson, N. J., & Raphael, B.** (1968). *A Formal Basis for the Heuristic Determination of Minimum Cost Paths*. IEEE Transactions on Systems Science and Cybernetics, 4(2), 100-107.

doi:10.1109/TSSC.1968.300136.

1. **Google Maps Platform Documentation.** (n.d.). *Routes API Overview*. Retrieved from https://developers.google.com/maps/documentation/routes

**2. Project Description**

**2.1 Reference Algorithm**

**1 Dijkstra's Algorithm** is used to compute the shortest path between two inputted based on distance. and hence routes will be optimized.

**2. PriorityQueue** will be used to store the distances and find the minimum distance using the min-heap algorithm

**2.2 Data / Data Structure**

**1. Node**: The fundamental unit of the linked list, containing the location and a reference to the next node.

**2. Priority Queue:** A **priority queue** is a specialized data structure that operates similarly to a regular queue, but with a key difference: each element has a priority level, and elements are dequeued based on their priority rather than their order of arrival. It would be used to take out minimum distance.

3.**Graph**: A **graph** is a data structure that consists of a set of nodes (also called vertices) and edges connecting these nodes. Graph would be used to create a network between locations. Locations would be nodes and their distances would be edges.

4. Data would be in form of adjacency matrix.

**2.3 SWOT Analysis**

**Strengths**:

1.User-Friendly Interface: The project has an easy front end for users to enter locations and see routes, improving their experience.

2.Real-Time Route Calculation: The backend calculates routes in real-time, giving users accurate and timely results for navigation and logistics

3.Graph-Based Algorithm: Using a graph-based method makes route calculations fast and accurate, even for complex networks.

4.Scalability: The system can be scaled to handle larger datasets and more complex networks as it grows.

**Weaknesses**:

1.Complexity in Implementation: Implementing real-time graph algorithms is complex and requires expertise, which can slow down development.

2.Resource Intensive: Real-time calculations in large graphs can be resource-heavy, leading to slower performance.

3.Dependence on Data Quality: Route accuracy relies on good, up-to-date data; poor data leads to wrong directions.

4.Limited Offline Functionality: If the system needs real-time data, it may not work well in areas with poor or no internet.

**Opportunities**:

1.Market Demand: There's high demand for optimized routing in logistics, transportation, and delivery, offering great market potential.

2.Integration with Other Systems: The project could work with GPS, traffic systems, or IoT devices to boost its features and market appeal.

3.Expansion into New Markets: The system can be customized for various industries or regions, like emergency response, city planning, or tourism.

4.Technological Advancements: Using AI and machine learning can improve routing and predict traffic, giving a competitive edge.

. **Threats**:

1.Competition: The routing market is competitive with established players, making it tough to gain market share.

2.Technological Changes: Quick tech advances might make parts of the system outdated or need major updates, causing disruptions.

3.Data Privacy and Security: Managing real-time data risks privacy and security issues, especially with sensitive location data, leading to potential legal challenges.

4.Dependence on External Data Sources: The system may rely on external data sources (e.g., traffic data, maps), and any changes, inaccuracies, or disruptions in these sources could negatively impact the system's performance.

**2.4 Project Features**

1. Automated Route Planning: Automatically calculates the most efficient routes for multiple deliveries.

2. Real-Time GPS Tracking: Provides live location tracking of delivery vehicles for operators and customers.

3. Dynamic Route Adjustments: Adapts routes based on traffic, weather, or unexpected delays.

4. Priority-Based Scheduling: Assigns priority to high-importance or time-sensitive deliveries.

5. Estimated Time of Arrival (ETA): Displays accurate delivery time predictions for each route.

**2.5 User Classes and Characteristics**

Administrators

* Access to all system features and settings.
* Can configure routes, schedules, permissions, and integrations.
* Monitors system performance and manages user roles.

Dispatchers/Operations Managers

* Responsible for route planning and scheduling.
* Monitors real-time tracking and updates routes as needed.
* Analyzes delivery metrics and generates performance reports.

Delivery Drivers/Couriers

* Access to assigned routes, schedules, and delivery instructions.
* Receives real-time updates and can communicate with dispatchers.
* Reports delivery status and completes delivery confirmations.

Customers/Clients

* Receives notifications and ETAs for document delivery.
* Can track delivery status and request delivery adjustments (if enabled).
* Limited access to delivery status, not to operational settings.

**2.6 Design and Implementation Constraints**

Real-Time Data Processing

* Requires low-latency data processing for accurate, real-time updates on routes and delivery status.

Integration with External Systems

* Must be compatible with existing CRM, inventory, GPS, and mapping services (e.g., Google Maps, Mapbox).

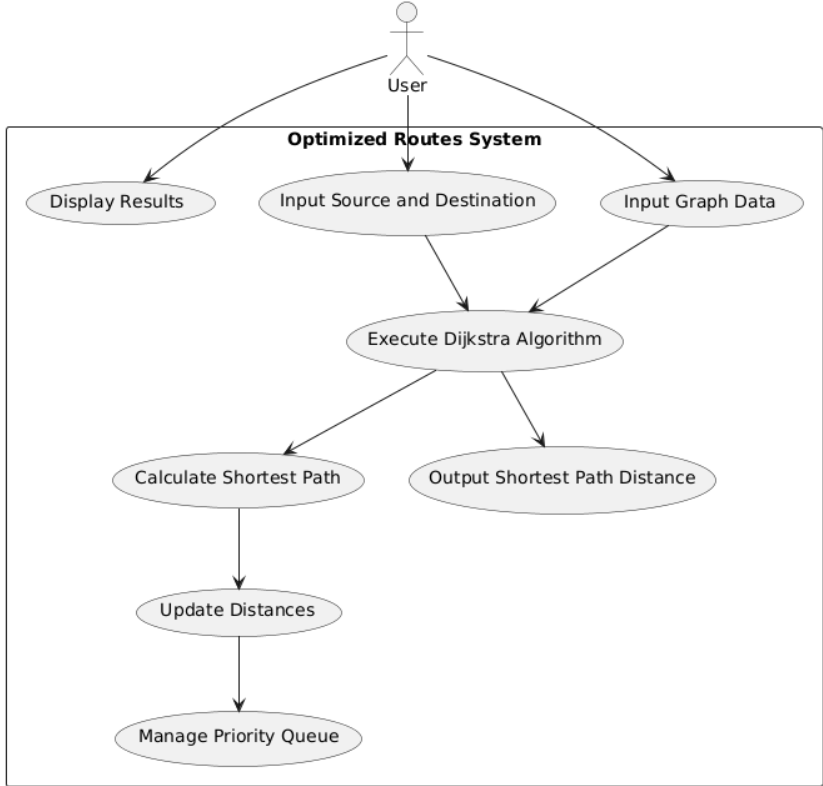
Scalability

* System should support a variable number of deliveries and users, from small operations to enterprise-level needs.

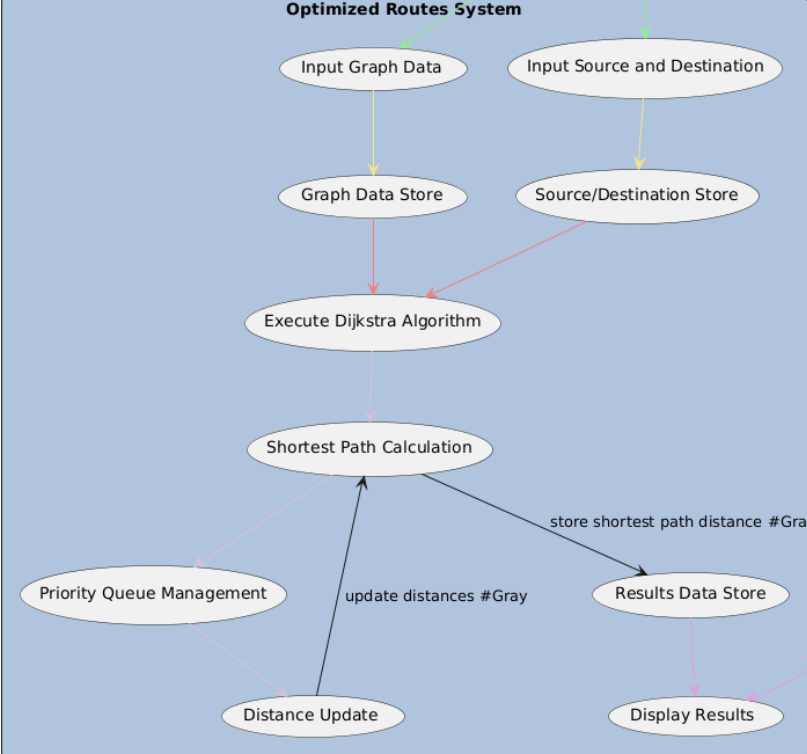
Data Privacy and Security

2.7 **Design Diagrams**

**Use Case Diagram**:



**Data Flow Diagram (DFD)**: Visualizes how customer data, booking data, and route details flow through the system.



**2.8 Assumptions and Dependencies**

Assumptions

1. Static Graph Input: The graph structure (nodes and edges) is predefined and static during each calculation.
2. Non-Negative Weights: Edge weights are non-negative to ensure Dijkstra's algorithm functions correctly.
3. Single Source, Single Destination: The system calculates the shortest path from a single source to a single destination per query.
4. Complete Input Data: All required inputs (graph, source, and destination) are provided before computation.

Dependencies

1. GPS/Mapping Service: Relies on accurate mapping and GPS data for route calculations.
2. Priority Queue Mechanism: Depends on efficient priority queue data structures (e.g., heap) to optimize the algorithm's performance
3. .Data Storage: Needs a data storage system to maintain graph data, calculated routes, and historical distances.

**3. System Requirements**

**3.1 User Interface**

The user interface will consist of web-based forms that allow users to:

**3.1.1** Input Locations

**3.1.2** Input distances in the form of adjacency Matrix

**3.1.3** Give Shortest Distance between any two locations.

The UI will be implemented using **REACT.JS.**

**3.2 Software Interface**

The system will use **Node.js** for backend logic, interacting with the algorithm, frontend and performing business logic such as distance calculations

**3.3 Database Interface**

**At the Moment we are not facing any requirements for a database.**

**3.4 Protocols**

The system will use:

**HTTP** protocol for communication between the client-side interface and the server.

**TLS/SSL** for secure communication to protect user data.

**4. Non-functional Requirements**

**Performance Requirements**

1.Low Latency in Route Calculation: The system should calculate the shortest path within milliseconds to seconds, depending on graph size, for real-time usability.

2. Scalability: Capable of handling a large number of nodes and edges (e.g., thousands of locations) without significant delays.

3. High Throughput: Able to process multiple routing requests simultaneously, supporting concurrent users and queries.

4. Efficient Memory Usage: Optimized to use minimal memory, especially important for large graphs or mobile devices..

**Security Requirements**

1. Data Encryption: Encrypt all sensitive data, such as user information, route data, and location history, during transmission and storage.

2. Authentication & Authorization: Implement secure user authentication (e.g., multi-factor authentication) and role-based access control to prevent unauthorized access.

3. Secure APIs: Protect all API endpoints, using secure protocols (HTTPS) and API keys or tokens to validate requests.

4. Data Anonymization: Anonymize personal data where possible to protect user privacy, especially for data used in analytics or reporting.

**Software Quality Attributes**

**Reliability**: The system must be available 99.9% of the time.

**Usability**: The system must be easy to use and require minimal training for users.

**Maintainability**: The codebase must be modular, allowing for easy updates and bug fixes.

**Scalability**: The system must scale to support more routes and users as the service expands.

**Portability**: The system should work across all major browsers and devices.

**Reusability**: Algorithms such as Dijkstra’s and profit maximization can be reused in similar transportation systems.

**5. Other Requirements**

Usability Requirements

1. User-Friendly Interface: Ensure a clean, intuitive UI for easy use by dispatchers, drivers, and other users.
2. Multilingual Support: Provide support for multiple languages to serve diverse users.
3. Error Handling & Feedback: Implement user-friendly error messages and feedback for invalid input or system issues.
4. Accessibility: Design the interface to be accessible for users with disabilities (e.g., screen reader compatibility).

Reliability Requirements

1. High Uptime: Target 99.9% uptime for system availability, minimizing disruptions.
2. Automatic Failover: Use failover mechanisms to switch to backup systems in case of hardware or software failure.
3. Data Backup: Regularly back up critical data, including user data and system configurations.

Maintainability Requirements

1. Modular Design: Structure the system in modules to facilitate easier updates and bug fixes.
2. Documentation: Provide thorough documentation for both users and developers for ongoing maintenance and support.
3. Error Logging: Log errors and exceptions for troubleshooting and system improvements.

Compliance Requirements

1. Audit Trails: Maintain an audit trail of route calculations and data access for compliance and traceability.
2. Adherence to Industry Standards: Follow industry standards (e.g., ISO 27001 for security) to align with best practices.

**6. Glossary**

**Graph: A collection of nodes (vertices) connected by edges that represent locations and routes in the system.**

**Node: A point in the graph, typically representing a location, such as a warehouse or delivery destination.**

**Edge: A connection between two nodes, usually with a weight representing distance or travel cost.**

**Shortest Path: The path between two nodes that minimizes the total distance or cost, calculated by algorithms like Dijkstra’s.**

**Dijkstra's Algorithm: A popular algorithm used to find the shortest path between nodes in a graph.**

**Priority Queue: A data structure used to efficiently select the next node with the smallest tentative distance in Dijkstra's algorithm.**

**Distance: The weight or cost associated with traveling along an edge between two nodes, usually representing time, distance, or fuel consumption.**

**Source Node: The starting point of the route from which the shortest path is calculated.**

**Destination Node: The endpoint of the route where the shortest path is calculated to.**

**ETA (Estimated Time of Arrival): The predicted time at which a delivery or journey will reach its destination based on calculated routes.**

**Graph Data Store: A storage system that holds the graph data (nodes and edges) for the route calculation.**

**Route Optimization: The process of determining the most efficient path based on various factors like distance**