**Project Proposal**

Title: Food Delivery Routing Optimization

# 1. Introduction

The Food Delivery Routing Optimization system is designed to efficiently assign delivery orders to riders while considering constraints like maximum weight and optimal delivery paths. This application is implemented in C++ and uses graph-based algorithms to simulate a grid-based city map.

**2. Features of the Project**

### ****2.1 Grid Representation****

* **Feature**: A grid-based city map is implemented using a graph represented as an adjacency matrix.
* **Details**:
  + Each vertex represents a location on the grid.
  + Edges connect adjacent vertices to represent possible routes.
  + Users input the size of the NxN grid.

Here's a properly formatted project proposal for your provided code, including explanations of its features. You can copy this into a document editor to save it as a .docx file.

**Project Proposal**  
**Title:** Food Delivery Routing Optimization

**1. Introduction**  
The Food Delivery Routing Optimization system is designed to efficiently assign delivery orders to riders while considering constraints like maximum weight and optimal delivery paths. This application is implemented in C++ and uses graph-based algorithms to simulate a grid-based city map.

**2. Features of the Project**

### ****2.1 Grid Representation****

* **Feature**: A grid-based city map is implemented using a graph represented as an adjacency matrix.
* **Details**:
  + Each vertex represents a location on the grid.
  + Edges connect adjacent vertices to represent possible routes.
  + Users input the size of the NxN grid.

### ****2.2 Restaurants and Orders****

* **Feature**: Management of restaurants and their respective orders.
* **Details**:
  + Each restaurant is characterized by its name, location (vertex), and a list of orders.
  + Orders include details such as the destination vertex, maximum weight, and whether they are assigned.

### ****2.3 Riders****

* **Feature**: Management of riders and their activities.
* **Details**:
  + Riders have attributes like current location, total distance traveled, and availability.
  + Paths for riders are dynamically computed for each delivery.

### ****2.4 Graph Operations****

* **Feature**: Graph-based operations for pathfinding and distance calculations.
* **Details**:
  + Breadth-First Search (BFS) is used to calculate shortest paths between locations.
  + Paths and distances are determined dynamically during order assignments.

### ****2.5 Order Assignment****

* **Feature**: Dynamic allocation of orders to riders.
* **Details**:
  + Orders are assigned to the nearest available rider who satisfies weight constraints.
  + Riders' paths are updated, and assignments are logged in detail.

### ****2.6 Performance Metrics****

* **Feature**: Computation of total distance traveled by all riders.
* **Details**:
  + The system computes and displays the total distance covered by each rider and by all riders collectively.

Here's a properly formatted project proposal for your provided code, including explanations of its features. You can copy this into a document editor to save it as a .docx file.

**Project Proposal**  
**Title:** Food Delivery Routing Optimization

**1. Introduction**  
The Food Delivery Routing Optimization system is designed to efficiently assign delivery orders to riders while considering constraints like maximum weight and optimal delivery paths. This application is implemented in C++ and uses graph-based algorithms to simulate a grid-based city map.

**2. Features of the Project**

### ****2.1 Grid Representation****

* **Feature**: A grid-based city map is implemented using a graph represented as an adjacency matrix.
* **Details**:
  + Each vertex represents a location on the grid.
  + Edges connect adjacent vertices to represent possible routes.
  + Users input the size of the NxN grid.

### ****2.2 Restaurants and Orders****

* **Feature**: Management of restaurants and their respective orders.
* **Details**:
  + Each restaurant is characterized by its name, location (vertex), and a list of orders.
  + Orders include details such as the destination vertex, maximum weight, and whether they are assigned.

### ****2.3 Riders****

* **Feature**: Management of riders and their activities.
* **Details**:
  + Riders have attributes like current location, total distance traveled, and availability.
  + Paths for riders are dynamically computed for each delivery.

### ****2.4 Graph Operations****

* **Feature**: Graph-based operations for pathfinding and distance calculations.
* **Details**:
  + Breadth-First Search (BFS) is used to calculate shortest paths between locations.
  + Paths and distances are determined dynamically during order assignments.

### ****2.5 Order Assignment****

* **Feature**: Dynamic allocation of orders to riders.
* **Details**:
  + Orders are assigned to the nearest available rider who satisfies weight constraints.
  + Riders' paths are updated, and assignments are logged in detail.

### ****2.6 Performance Metrics****

* **Feature**: Computation of total distance traveled by all riders.
* **Details**:
  + The system computes and displays the total distance covered by each rider and by all riders collectively.

**3. User Input and Interaction**

* **Input Details**:
  + Grid size, number of restaurants, riders, and orders.
  + Locations and order details.
* **Output Details**:
  + Visualization of the grid and its connections.
  + Assignment of orders to riders with the computed delivery paths.
  + Summary of distances covered.

Here's a properly formatted project proposal for your provided code, including explanations of its features. You can copy this into a document editor to save it as a .docx file.

**Project Proposal**  
**Title:** Food Delivery Routing Optimization

**1. Introduction**  
The Food Delivery Routing Optimization system is designed to efficiently assign delivery orders to riders while considering constraints like maximum weight and optimal delivery paths. This application is implemented in C++ and uses graph-based algorithms to simulate a grid-based city map.

**2. Features of the Project**

### ****2.1 Grid Representation****

* **Feature**: A grid-based city map is implemented using a graph represented as an adjacency matrix.
* **Details**:
  + Each vertex represents a location on the grid.
  + Edges connect adjacent vertices to represent possible routes.
  + Users input the size of the NxN grid.

### ****2.2 Restaurants and Orders****

* **Feature**: Management of restaurants and their respective orders.
* **Details**:
  + Each restaurant is characterized by its name, location (vertex), and a list of orders.
  + Orders include details such as the destination vertex, maximum weight, and whether they are assigned.

### ****2.3 Riders****

* **Feature**: Management of riders and their activities.
* **Details**:
  + Riders have attributes like current location, total distance traveled, and availability.
  + Paths for riders are dynamically computed for each delivery.

### ****2.4 Graph Operations****

* **Feature**: Graph-based operations for pathfinding and distance calculations.
* **Details**:
  + Breadth-First Search (BFS) is used to calculate shortest paths between locations.
  + Paths and distances are determined dynamically during order assignments.

### ****2.5 Order Assignment****

* **Feature**: Dynamic allocation of orders to riders.
* **Details**:
  + Orders are assigned to the nearest available rider who satisfies weight constraints.
  + Riders' paths are updated, and assignments are logged in detail.

### ****2.6 Performance Metrics****

* **Feature**: Computation of total distance traveled by all riders.
* **Details**:
  + The system computes and displays the total distance covered by each rider and by all riders collectively.

**3. User Input and Interaction**

* **Input Details**:
  + Grid size, number of restaurants, riders, and orders.
  + Locations and order details.
* **Output Details**:
  + Visualization of the grid and its connections.
  + Assignment of orders to riders with the computed delivery paths.
  + Summary of distances covered.

**4. Code Features**

* **Dynamic Memory Management**: The system uses dynamic arrays for restaurants, riders, and orders.
* **Modular Design**: The system is structured with functions for graph creation, BFS, pathfinding, and order assignment.
* **Flexibility**: The grid size, number of restaurants, riders, and other parameters can be customized during runtime.

**5. Technology Stack**

* **Programming Language**: C++
* **Key Concepts**: Graph Theory, BFS, Dynamic Memory Allocation.