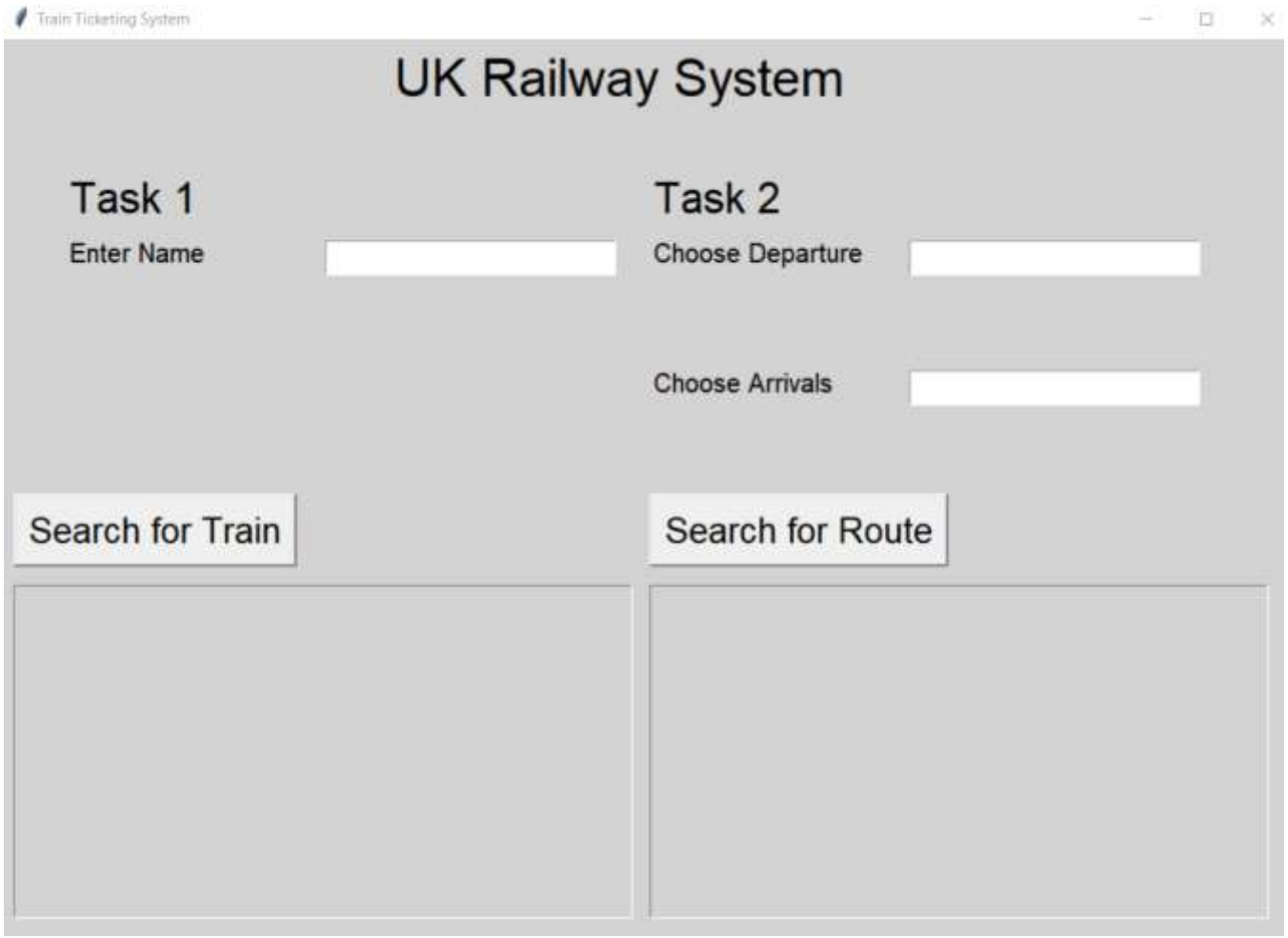
# Task 2: Design

Both Task 1 & 2 use a single unified interface which is show below:




As we know that Task 2 is related to efficient calculation path from one station to another, two classes are used for this purpose. The details of both the classes along with their relevant functions are listed below:

## 1. RailwayMap

This class implements a graph to store a network of railway stations. Following are the functions used in the class for task2:

- **add_edge**

  This function is responsible for adding new vertices and edges in the graph. For each new entry, it updates the adjacency_list and the cost_list which are the members of this class.

## 2. RailwayNetwork

This class uses an object of the RailwayMap class to load the data stored in railway_network.csv. Following are the functions used in task 2 from this class:

### 1. load_railway_stations_data

This function accepts a csv filename as parameter and loads the file contents into an object of *RailwayMap*. The *RailwayMap* uses adjacency_list and the cost_list to implement the graph. For each new entry, a new element is append into the adjacency list and corresponding cost is stored in the cost list.

### 2. search_railway_path

This function is responsible for performing the main functionality required in task 2 . It implements Dijkstra search algorithm for to find the shortest path between source and destination railway stations.

### OUTPUT