# Passenger Clearance System

Yuhan Xu

**Summary**

## In this part of the project, I will try to go through the city where passengers go to the departure city. Boston's distance decides to release the order. The distance between the two will use the Dijkstra algorithm to calculate.

**Input Example**

passenger\_id,airport,city,state,country,airliner,distance

A,KLAX,Los Angeles,CA,US,AAL,4799

B,KORD,Chicago,IL,US,DAL,1580

C,KJFK,NewYork,NY,US,AAL,342

D,VHHH,Hong Kong,Hong Kong,China,CPA,12796

E,EGLL,London,London,UK,BAW,496

F,YSSY,Sydney,Sydney,Australia,AAL,16263

G,PHNL,Honolulu,HI,US,UAL,8200

H,PANC,Anchorage,AK,US,DAL,5420

**Output Example**

C NewYork 342 miles

E London 496 miles

B Chicago 1580 miles

A Los Angeles 4799 miles

H Anchorage 5420 miles

G Honolulu 8200 miles

D Hong Kong 12796 miles

F Sydney 16263 miles

**Business Requirement**

1. for dataset, it requires to get some information from passenger,including destination airport,destination city,destination state,destination country, and the airliner the passenger choose to take
2. it requires to get the direct distance for Boston to the destination City

**Output after Code running**

Passenger information sorted by shortest distance from Boston:

Passenger ID Destination City Distance from Boston

C NewYork 342 miles

E London 496 miles

B Chicago 1580 miles

A Los Angeles 4799 miles

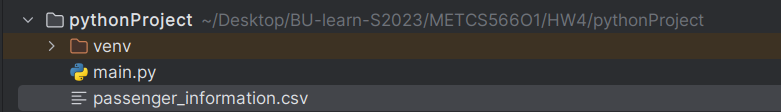
H Anchorage 5420 miles

G Honolulu 8200 miles

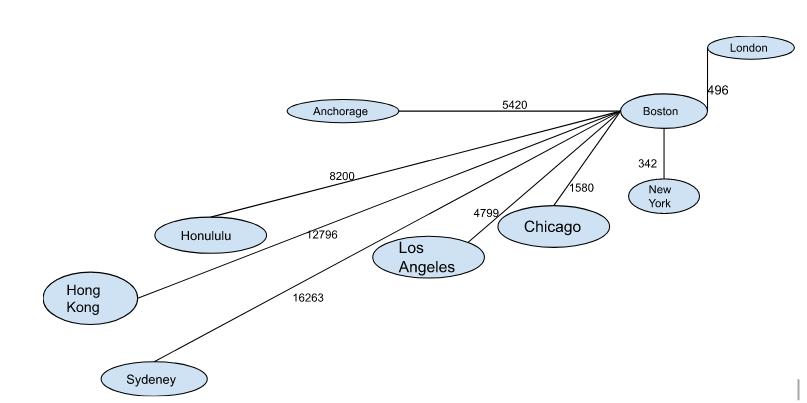
D Hong Kong 12796 miles

F Sydney 16263 miles

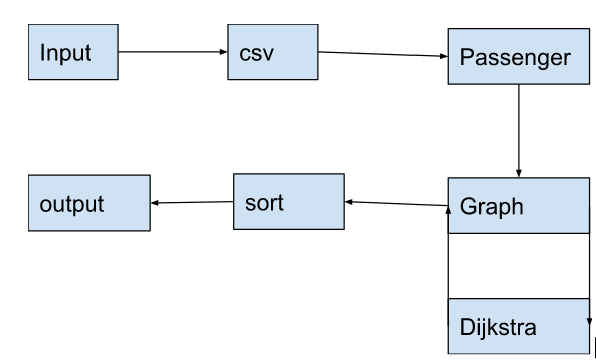
**Screenshot of the file structure**



**Graph for Dijkstra**

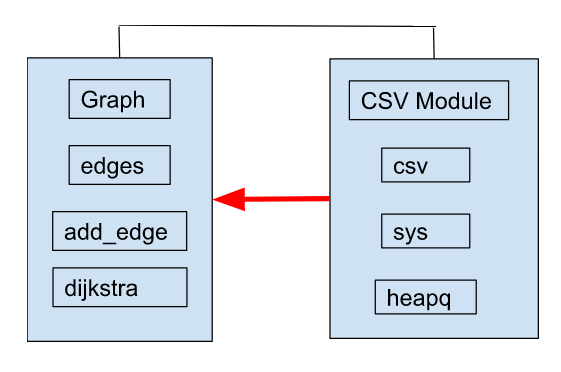
****

**Sequence Model**

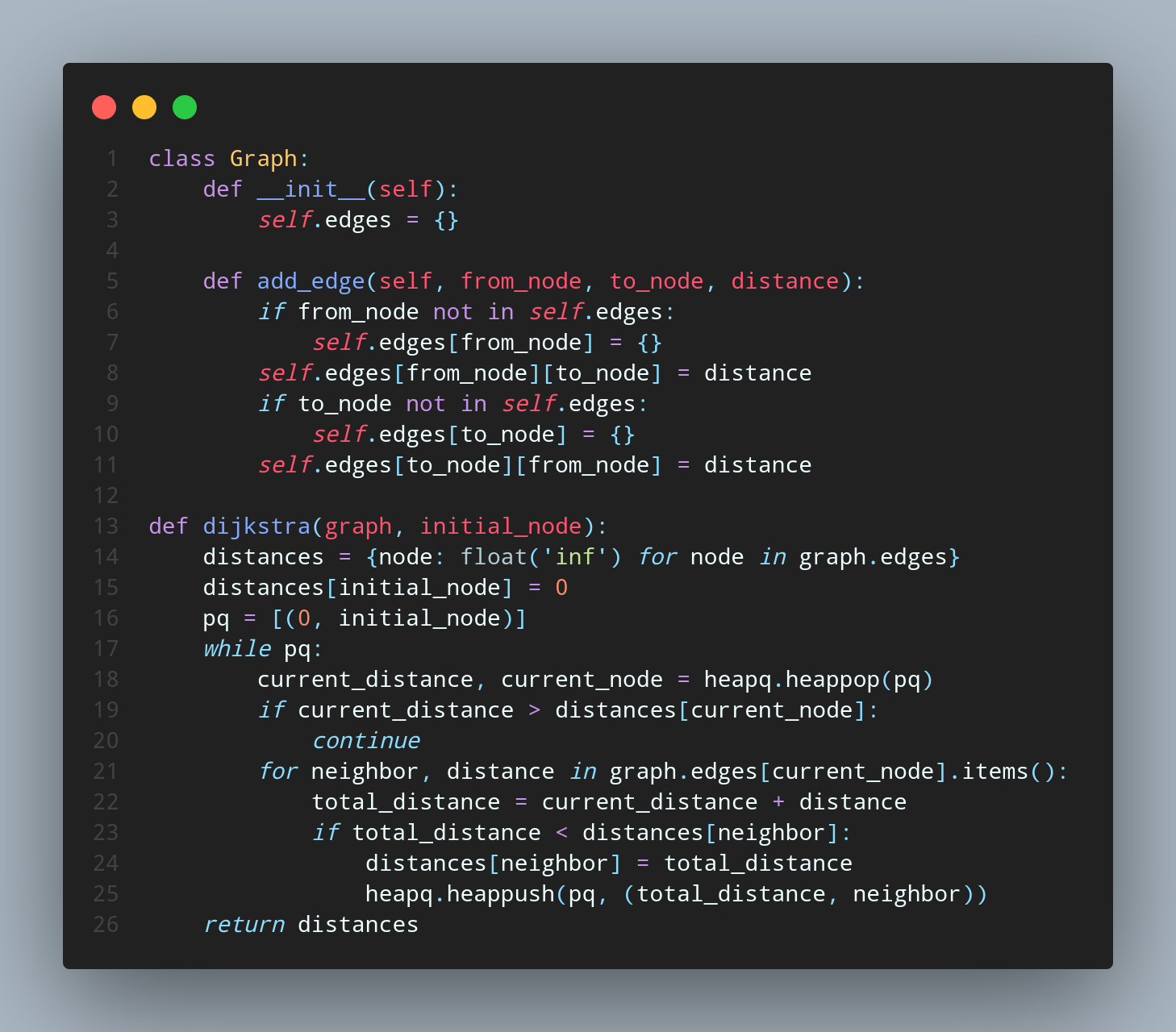
****

Sequence mdel

**UML**

****

**Code for Dijkstra Algorithm**



**Time Efficiency**

The time efficiency of the code depends on the input size, specifically the number of cities in the CSV file.

The dijkstra() function has a time complexity of O(E log V), where E is the number of edges in the graph and V is the number of vertices (or cities in this case) in the graph. Since each row in the CSV file adds an edge to the graph, and each city is a vertex, the time complexity of creating the graph is O(N), where N is the number of rows in the CSV file. Therefore, the time complexity of the dijkstra() function called for each passenger is O(E log V), where E and V are both O(N).

The time complexity of sorting the passenger distances is O(N log N), since it uses the built-in sort() function in Python.

Overall, the time efficiency of the code is O(N log N + N log N) = O(N log N).

**Comletely Code**

