**REFLECTION OF THE PROJECT.**

In this project, the aim was to find a valid path given a start city country and an end city country from a database of airports, airlines, and routes in.csv file extensions. I dissected the issue and approached it gradually in order to effectively understand what was expected. In this report, I'll go over how I read files, the proper data structures for storing the data, the search algorithm I used, and the project's lesson learned.

Making header files for the Route, Airport, and Airline objects was the first step towards deconstructing the issue. I read the CSV files first. I developed a helper function class with a function to read files and return them as a vector of vectors in order to facilitate modular coding.

In order to store the data, I needed an efficient data structure. Using a data structure that stores data in a key, value pair format will enable constant time complexity due to the data size, as opposed to using a list, which has, in the worst case, an O(n) time complexity. I had three unordered maps. one where the city and country were the key, and the value was a vector of all the airport codes in that city and country. The second one used a city and country vector as the value and an airport code as the key. The last map used a Route object as the value and the source airport code as the key. I populated each of these maps.

Implementing the search algorithm came next. Numerous search techniques, including depth-first, breath-first, and uniform-cost search, sprang to mind. I selected the breath-first search because I wanted the search to be computed in levels, and I needed the shortest path from any start point to any endpoint. The start location and the end location are the two parameters that the BFS accepts. There was a frontier that stored nodes which consisted of all the successor states (the next available flights) and an explored set that tracked all the visited states. With BFS, while the frontier is not empty, it pops the first element and explores it. After reaching the destination, it returns a solution path of all the nodes it visited to get there.

Overall, I learnt so much from doing this project. It improved my problem-solving abilities and gave me the experience of implementing the breadth-first search algorithm, considering the code's efficiency.

References to useful sources.

<https://devdocs.io/cpp/>