Results:

We started out with graph implementation. Once we realized that the graph should be final once built, we didn't include any functionality to modify the graph. We designed the Vertex and Edge model to hold all the information about the airports location, travel time and other miscellaneous. The edges were assigned with weight including the travel distance, stops and time to make the overall travel time more reasonable.

However, the data wasn't perfectly formatted. We figured out that some airports did not have IATA, which we initially decided to use for user's inputs. We also need another graph to store the IATAs and the vertices to retrieve then vertex when needed. A costume vertex class needs to be hashed to work in an unordered_map. It took us a while to identify this error. STL string has no split functionality, we had to implement ours. And the string entries in the airport data includes extra commas that messed up with the csv.

Our Dijkstra Algorithm can effectively find the shortest path from a single source to the destination. We've provided three different test cases checking for the desired functionality. We enhanced the feature of the landmark path by building it on top of Dijkstra so that it takes in multiple inputs. In this case, the function can take in multiple stops that the user desires.