DEVELOPMENT LOG:

November 14:

- First all team-member meeting on Discord at 2PM PST
- Team Contract: started, finished, and uploaded to our GitHub repository
- Project Goals: talked about potential implementations and ideas
- Current Idea: using Stanford SNAP, we can create a network of frequently purchased Amazon items to recommend to the user

November 17:

- Second meeting started at 6PM PST with all members
- We finalize the team contract
    - We noticed that the SNAP Stanford data set that we initially wanted to use didn't have sufficient metadata so the nodes and edges were codes but we didn't know what those codes meant.
    - Given this, we pivoted to choosing a project about finding the optimal path from a starting airport to an ending airport.

November 27:

- Third meeting started at 8PM PST with all members
- Assigned roles to each member
- We created a general timeline of when to finish initial tasks
    - Reading and translating the data into a graph / Data Set Parser by Dec 1
    - Find GUI library by Dec 1
    - The two algorithms and the graph class is dependent on the Data Set Parser so due date is TBA

December 1:

- We worked on parsing the data from OpenFlights into a graph
    - We created the Graph class
    - Created the constructor that reads from a routes file and an airplane file
    - Create necessary maps: one maps airports to latitude and longitude, another maps airport to all outgoing Edges (Edges contain start/dest airport and airline code)
- Next steps:
    - Create MakeFile
    - Add functions to graph class to create vertices and edges from the maps we created today
    - Find some GUI library for the graphical visualization

December 2 - afternoon:

- We build additional features in the graph class that supports creation on vertices and edges for the graph
- We are in the process of creating the MakeFile
- Also found potential methods to do the graphical output: CS225 pixel class, mapnik, a map API etc.

December 2 - night:

- The graph implementation is created and we successfully parse and store data into a graph structure
- Functions that printed airports and their latitude longitude as well as flight route information was created.
- main.cpp was created.
- Makefile was created.

December 3:

- Created Getters to publicly interact with the graph. This will be used in the algorithms as well as the graphical output.

Roles:

Note that group members can jump around between roles and this is just a loose idea.

Graph class (we can use parts of the classes from lab_ml):

2 Algorithms:

BFS and Dijkstra's Algorithm

Pranshu

Adj matrix/graph structure

Irfaan

Graphical Output Class (using a library):

Eric

Reading and translating the data into a graph / Data Set Parser:

Isaac and Eric

User-interaction give