1. Formalized Proposal
   1. The program is a ticket reservation system that allows Data Structures Airlines, which is a newly established airline, a way for their customers to book a flight on there airline. It is an easy to use system that allows the user to pick their departure airport, destination airport and the maximum number of stops they want and get the results back. The results come back giving the user the following info:
      1. Result
      2. Flight Number
      3. Price of Ticket
      4. Number of stops
      5. Number of seats left.
      6. Flight route
2. Time/Change Logs
   1. Week1: Research of how a flight network graph might look like
   2. Week2: How one would traverse that graph to get the results to find routes between the two vertices.
   3. Week3/4: I had to research how to keep limit the number of vertices you would visit in between each departure and destination to avoid a return with 8 stops. This help keep the project more real life applicable.
   4. Week4/5 on: Refreshed myself on Django.
   5. Week5/6: Had to figure out a solution of how to take the returned data and get it to display to the end user. A dictionary seemed to be the best result.
   6. Week7: Learned how to iterate over a dictionary using Django.
   7. Week8: Formatted the results so the UI was presentable. Wrote up final report.
3. Lessons Learned
   1. That I have only scratched the surface on how I am sure this is really done in real life. I tried to use a larger data set but was unable to get it to work without it taking a considerable amount of time to return the results. I may try and research this more so that I can come up with a way to use these real life airline routes that are open APIs online.
   2. How much I had forgotten about using Django from last semester to the current point. It took a good couple of days of several hours each to get myself somewhat back up to speed on how to utilize the framework again,
   3. Talk about the scope of your project and if it changed
      1. I think I did a pretty good job in staying in the scope of this project. Knowing how complicated the real world application of this app actually is I tried to make sure I didn’t end up going down to many rabbit holes and get side tracked. By utilizing things like random num generators to come up with price, and flight numbers it helped create a simple solution but gave the app a real life feel.
      2. What blockers you encountered and the solutions you found
         1. Django learning curve
         2. Complexity of graphs and graph traversal.
4. CODE including comments
   1. A link to your github repo (files on github that include comments and stylesheet adherence)
5. User's Manual
   1. Django Setup Instruction
      1. <https://djangobook.com/beginning-django-tutorial-lesson-3/>
      2. This is the resource I used to get started.
   2. Go to the starting page of the program:
   3. A screenshot of a cell phone

      Description automatically generated
   4. Select from the drop down menu the starting and destination city of your choice.
   5. Select the Max Number of Stops you would like from the drop down menu.
   6. Hit the submit button.
   7. A screenshot of a cell phone

      Description automatically generated
   8. Get the results:
   9. A screenshot of a cell phone

      Description automatically generated
   10. No Flights found page. Shown if no flights are available between the two cities.
   11. A screenshot of a cell phone

       Description automatically generated
   12. If you choose the same departure and destination city the following will display:
   13. A screenshot of a cell phone

       Description automatically generated
6. Conclusion/Summary
   1. This was a fun, but challenging program. It really made me think. I found myself diagraming out a lot on a white board trying to understand graphs that I had researched to get this to work.
   2. I feel like I have a better understanding of graphs, although I feel like I have a lot more I could learn about them. They are a very powerful and useful data structure in our everyday life.
   3. This project also served as a great distraction from the chaos of this world we currently live in. It served as a great outlet to get away from wanting to jump through my TV ☺
   4. A paragraph describing MERUSE (Michelle's principles of good programming found in an early Module folder Java Review) applied to your code.
      1. **M:** This program offers several different areas where the code is reusable. The add\_flights function can be used for initial setup but could be used if a admin view was setup and would allow employees access to add and change flights. The get\_route function could be used in several different aspects of the program as well.
      2. **E:** The program runs efficiently in its current version but does not run efficiently with a larger data set. I experimented adding over 2000 different routes from data I got offline. Although it worked the BigO for the program was awful. I ran out of time researching a solution. I thought I had one at one point but found it not to work well.
      3. **U:** The program does what it set out to accomplish of showing routes between to airports. I am not sure I would go putting my credit card info quite yet. But it will give you a flight result back.
      4. **S:** I think the code is readable. Python kind of forces you to right code that looks somewhat organized or it won’t run. I tried to add comments to the best of my ability.
      5. **E:** **See S.**
   5. A paragraph of future versions
      1. A future version would include some more of two-way routing. The current version just gives you a one way trip but doesn’t offer any way of getting home.
      2. Figure out how to stop visiting the same vertices twice when you increase the number of stops.
      3. How to implement an API with real-world flight routes that will actually give you results in under 5 mins.
      4. A better UI/UX.
      5. Change random number generators to actual real-world data.