Stage 4 Project Report

Video: https://youtu.be/Xy8resXDC1U
Link: http://35.209.169.228:3007/

- Please list out changes in the directions of your project if the final project is different from your original proposal (based on your stage 1 proposal submission)

  Our original project was supposed to analyze historical data to provide information on the best time of year to purchase a flight as well as provide live tickets with specific prices between certain airports. We were able to implement the live ticket feature to search for flights between exact airports, and even show the corresponding prices. However, we were unable to provide any timeOfYear usefulness in terms of analyzing the historical data and recommending the best time to buy tickets. In a way, our app is still good at helping users estimate a reasonable price for a ticket between 2 airports, as we do provide multiple flight/airline options between the locations. We were also not able to implement the creative component aspect of our project given the time constraint and lack of full stack experience between team members. Regardless of completing that extra component, our app still serves its primary use of providing informative flight ticket information.

- Discuss what you think your application achieved or failed to achieve regarding its usefulness.

  I think that due to not being able to implement the layover feature, we lost a big part of what made our app unique. While it is still useful for booking flights and finding what destinations are popular, there is more, if given time, that we would have liked to implement. Our app achieved being able to provide users with multiple flight options between airports to help them gauge a reasonable price. We also achieved being able to add tickets to a specific user, and ensuring that the number of tickets selected is not over the capacity of the flight (arbitrarily set to a number as we do not have access to the actual plane's capacity). We also set a limit to the total number of people that could be in an airport at a single time (arbitrarily set to a number as we do not have access to the actual airport capacity). We were able to keep track of stored flights for each user along with users being able to book multiple flights. Unfortunately, we were unable to provide any exact information on the time of the flight which is very important in terms of users going to the airport.

- Discuss if you change the schema or source of the data for your application

  We had to change our booking table by adding an extra element. The extra element we added was keeping track of the quantity of a ticket so that a user would be able to purchase more than one of the same ticket. This is relevant because if users are travelling with others, they may want to buy multiple seats on the same flight (e.g. a parent buying tickets for a family vacation) We had to use some generated data since not all of what we originally wanted was available. For example, in order to populate the User table, we had to create several random users of our application, as well as additional data required for each user. This allowed us to give these users a way to save flights for our demonstration.

- Discuss what you change to your ER diagram and/or your table implementations. What are some differences between the original design and the final design? Why? What do you think is a more suitable design?

  As described in the previous question, we updated the booking table to add a quantity value. This change was reflected in the ER diagram as well. Some other differences between the original design and the final include changing the Bookings table from a weak entity to a strong one. This is because we realized that we did not want bookings to rely on the user table. We wanted each booking to be uniquely identifiable, which meant we had to change it to a strong entity. Our final design was a more suitable design overall because it allowed us to most easily save, update, and delete bookings for each user while also being able to look at ticket data and high traffic areas. We did have to also implement a trigger for inserting and deleting from the Booking table to ensure that the SavedFlightIDs were also correctly inserted or deleted from the Booked_For table, but that ended up working well within our project. Finally, there was some data that we planned on using initially that did not end up making it into our final implementation, like the Company table or the time of year of a flight. However, overall the bulk of the airport and flight data was easily accessible and usable with our final model.

- Discuss what functionalities you added or removed. Why?

  We had to remove the original layover and the time of year features since the data from the databases did not include enough information to account for these features. We also removed some of the functionality in terms of providing a lot of future flights, as the consistent API calls to organize an up-to-date database of future flights would have cost money (we could not find any free API to collect

flight information from specific airlines). We did end up adding a functionality to limit the number of tickets available for a flight and the number of total people that can be at the airport at a certain time. We believe this makes sense, as the airport cannot be over capacity and the plane can certainly not have more passengers than seats. Additionally, we added a feature that reported how popular/busy each airport was. This“High Traffic” feature labels specific airports as high traffic and lists flights flying to it.

- Explain how you think your advanced database programs complement your application.

  Our advanced database programs allowed us to keep track of a lot of existing flight information and organize it in a way that enables users to look up specific flights between chosen airports. The database is highly useful for our app for this reason. More specifically, the advanced queries for the databases enabled us to compare information between tables and make more useful features. For example, we had a general table for flights and allowed users to select a departure and arrival airport and then add a specific flight for that information if a flight existed. The trigger was also very helpful since it allowed us to delete and insert bookings much easier. The bookings were related to the flight through a many-to-many relationship meaning that anytime we deleted a booking we also had to delete something from the Booked For table which the trigger allowed us to streamline.

- Each team member should describe one technical challenge that the team encountered. This should be sufficiently detailed such that another future team could use this as helpful advice if they were to start a similar project or where to maintain your project.
  - Cal - I feel like many of my team members could say the same, but this is the first time that i have been introduced to full-stack development so there was many things i had to learn including how to connect the database to different routes and how to implement each query so that we could see it at different routes
  - Neil: One technical challenge for me was understanding how to implement a front-end without any front-end experience. The best way that I was able to handle this was watch the video posted to prairielearn, as there are a lot of existing frameworks and help online. Specifically, rather than coding out an entire frontend in straight HTML, it is much easier to use a framework like React to organize components. If you are not taking CS411 without

the help of a React lecture, I would highly recommend watching YouTube tutorials because a lot of help exists for certain libraries/frameworks.
    - Evan: One technical challenge I encountered was organizing the data and creating all of the relationships between different datasets. I think it is very useful to plan ahead and find data that is cleaned and organized so that it can be easy to create relationships and do joins later on. We found a lot of data on Kaggle and were able to sift through it according to how popular it was (which was usually more organized for more popular datasets)
    - Michael: It was difficult in making sure the queries work on the database from the actual backend coding without testing the queries beforehand in GCP. I would recommend testing the query in the GCP terminal of the database to ensure the advanced query works, and then implementing it into the codebase of the app afterwards.

- Are there other things that changed comparing the final application with the original proposal?

  In our original proposal our creative component involved determining cheapest flight options based on layover flights. However, we were unable to implement this component because we did not have the knowledge or time to do so, and we did not have sufficient data on flights to support working with a layover feature. Instead, we implemented the High traffic airport feature and ticket quantity features which we did not originally plan to do. Besides that, overall we achieved the general goals of the original program. We provided users with flight information and allowed them to save the cheapest flight options to their account which was the overall goal of the application.

- Describe future work that you think, other than the interface, that the application can improve on

  There are a variety of things that could be added to improve this application. For example, including a real-time database that provided up to date flights would make the application slightly more relevant for users. It would also allow us to implement the layover feature described in our original proposal. This would also allow us to determine the most/least popular time of year to fly to certain destinations. Furthermore, adding a feature that could use a person's location to recommend different flights could be beneficial. This would have to be implemented with some sort of location tracking system or user input. Finally, we think that the application could be improved by providing future data, including exact dates and prices. Users typically want to look for future flights on specific dates, so to

make our app more functional, we could implement a departure date field while calling an API to get access to flights given these dates.

- Describe the final division of labor and how well you managed teamwork.
    - Cal - wrote stored procedures and implemented them in the backend, helped write the routes from our database to the front end. Implemented update and insert in the backend. Wrote 2 of the advanced queries
    - Evan - I did a lot of the backend work in NodeJS and connected it with Neil's front end. I wrote all the scripts sending JSON between the frontend, backend, and database, as well as handling the data at all points of the process. I built the CRUD operations for the frontend and backend, so I also worked on the trigger for the Booked and Booked_For tables. In terms of setting up mock data, I was able to generate random data for Neil to implement the front-end, which I later connected to the actual database in GCP; I worked on configuring the backend with the remote GCP database by sending SQL queries to the GCP database, and then processing the received data and sending it back to the frontend. Finally, I hosted the application so anyone with the specified link can access it (until the GCP subscription runs out). I created a VM instance also hosted on GCP and cloned the project files, and then packaged the frontend and backend to be production-ready before deploying the final build on the VM.
    - Neil - From stage 1, we all sort of built out the idea for the app and answered a specific question. From stage 2, I helped describe the purpose of each of the tables and the relationships. From stage 3, I helped analyze the costs from each of the indexes (as well as fix up the issues after the first submission). From stage 4, I mostly worked on the front end and understanding the React framework. I also helped a lot with creating the transaction at the end (built through another stored procedure).
    - Michael - I wrote a lot of the advanced queries (for searching for flights given certain user inputs) and tested them in GCP. I also helped out a lot with the stage 3 indexing and analysis. In the early stages, I helped a lot with finding data from Kaggle to help Evan later on with organizing the relationships.