Team 071
Bernie Zhu
Joseph Jang
Rahul Setlur

Project Report

1. The biggest change to our project was the live feed of the bus positions and which buses are currently running. Rather than the buses being shown live, our app requires a refresh to re input all of the bus data into the database, rather than it being updated every minute or so. As a result, our idea of writing a more efficient algorithm of when the API will be recalled was also scrapped, as these two ideas go hand in hand with each other. Another big thing is that we didn’t actually store the buses current position into the database either. This results from the API being rather faulty, or at least weird, because the buses specifically only store a route on a map, and require another call to retrieve a “Trip”, which should’ve been the positions of the buses.
2. Like stated above, in regards to usefulness, our app did not provide as much information to users as it should’ve, like the actual position of the buses and a live feed of the bus positions. However, having a refresh function could be more useful to users, as users can instantly gain knowledge on where the bus currently is, rather than needing to wait a set amount of time for the position of the bus to be updated, not knowing where it is in between API calls.
3. The biggest change to our schema should have been the inclusion and addition of the “trip” data, which should have helped in the retrieval of the bus’s current position. We would also have needed to include a new “shape” attribute, where we already store the shapeID in the busLine information. The combination of these attributes should be able to help us find the actual position of buses. We also could have added more information to our tables, such as the stop name, or other stop points. The addition of these could have been helpful to the user in recognizing locations and destinations. The source of the data cannot be changed, however, as MTD is the only source in retrieving the information of buses in the CU area.
4. The original ER diagram and table implementation was created based on how we perceived the tables could be formatted in order to have a functional bus application, however, we had to change the tables and ER diagram in order to more accurately match the data given and retrieved by the MTD API. The data retrieved from the API would not have worked given our original interpretation, and thus we needed to change the tables and add tables so that it reflected the retrieved data better. Our later iterations/implementations of the database and its tables are certainly better as they better reflect the data given by the called API, and thus the database has better access to the more organized information.
5. In terms of functionalities removed, we removed the idea of the live feed/updates on the buses current positions. We removed this largely because it wasn’t necessary to our program, and writing the algorithms and updating the schema and other attributes necessary could have taken a lot of work that wouldn’t have contributed much to the required functions. However, it may have taken a lot of the usefulness and real-world use out of the application, which is a shame. In terms of added functionality, our app utilizes

Google Maps, and in turn our map functionality contains all of the functionality of Google Maps itself as well, which is easily one of the most widely used map applications with tons of functionality in of itself. We also added a function where depending on where you are on the map, your favorite stop will be highlighted in a different color, some the closer you are being one color, medium distance being another, and long distance being one last color. This adds a layer of complexity and just a cool feature overall.

6. Our trigger is essentially a quality of life feature, hoping just to use it to save some people's time. What it does, is that when a favorite stop is deleted, it will delete all favorite routes in which it originates at the favorite stop. Thus, it can just save the user a bit of time when clearing out unneeded favorites. Our stored procedure can be useful to the user to judge their own location and how far they might be from their favorite stops. They will be able to know when they are close, medium distance, and far away from the stops they find useful to themselves.
7. Technical Challenges:
    a. Bernie - A challenge that we encountered was the lack of data stored into our database, even if that data might not have technically been necessary. For example, when creating the map functionality, I wanted to be able to show what intersection each bus stop was at, but we didn't store the stop name into our database, but only the stopID, which is rather cryptic to a user. Thus, I needed to pull information into a json file to use in order to display the stop names. So, I think that it's best to store all information pulled from the API calls into the database, even if that information might not be perceived as useful to the application initially, because you never know when you might need it.
    b. Joseph - Another challenge we encountered was how we can better communicate about which APIs we will be utilizing. For example, the Google Maps API we used was an older API, and thus had more depreciated and extraneous information that we didn't need to use. Thus, we would recommend to research on the APIs you will be using beforehand with the team so these problems won't come out later on.
    c. Rahul - One last challenge encountered could be on the dependencies utilized in the project. As we worked on the project one by one, and more dependencies were added that needed to be installed, it got harder and harder for each member to work on the project. Also, be careful about systems that you might be working on and conflicts perhaps created from that. For example, we had one person working through macOS, one on Windows 10, and one on Windows 11, and we think that the Windows 11 in particular may have caused some issues in our application.
8. Not very many other things were different when compared to the original proposal. Other than the live updates to bus positions, most of the idea largely stayed the same, perhaps only the UI was a little bit different from the original idea.
9. The application can definitely include some more quality of life updates and more convenience features. For example, the map can have a button leading to the favorites page, rather than needing to type in the link address itself. Or, the application could have

more seamless transitions, for example, rather than having the favorites site, there could be a side bar which shows the favorites bar along with the search function.

10. Teamwork was not especially important or there was not very much of it in the project, as one team member completed one part of the project, then other team members can build off of that completed portion. Other than input from other team members on things like advanced queries/other ideas overall, there was not too much teamwork in the process. Bernie completed most of stage 2, created the tables, inserted the randomized data (users, favorites) into the tables, helped in the advanced queries, and created all of the indexes in stage 3. For the final application, the trigger was developed and implemented, as well as the map functionality along with the markers and the favorite system on the map itself. Joseph created a python script to insert the retrieved data from the API call into the database. He helped in the development of the advanced query as well. In the application, he created most of the front end, including the user login, favorites, and advanced query, along with the back end utilizing the API calls/routes. He also created the stored procedure and implemented it into the map. Rahul helped with the development of stage 2, and contributed with ideas on how to implement application and database. He also tried to work on the stored procedure part of the application and the backend as well.