April 20, 2020
CS3200

NU Student Database: Final Project Report

By Kenton Lee and Aidan Doherty

ReadMe:

Student COVID Database

This program serves as a web app that tracks important information about a Northeastern student who have had contracted COVID19 in the past. It can be used for any event organizer to review their guest list and evaluate the safety for inviting certain individual to a face-to-face event.

Libraries and languages used:
* Front-end: JavaScript, HTML, and React.js. CSS for styling.
* Back-end: JavaScript, Node.js, SQL(MySQL workbench)

The installation history can be found in package.json.

Front-end: Client

The Client is the front end of the program. It uses React.js to handle any user's attempts to create, retrieve, update, and delete a data entry in the database.
* Create: We created multiple forms for user to fill out. The forms contain dropdown selections based on other relevant data tables as well as some input option for other key information. The forms will take the data to the backend and add relevant data to the data table when the submit button is clicked.
* Retrieve: By clicking the refresh button, the program will go through query in the backend to retrieve the most updated data table and display it through an HTML table.
* Update: By clicking the edit button, a user can modify certain information about a data entry regarding to a student based on the given student id which is used as the primary key in the database.
* Delete: By clicking the delete button behind any data entry, the front-end will send the student id of the student to the backend and delete such data entry.

Important Buttons:
* Submit: calls on a function that leads to an INSERT sql statement in the backend to store all the data in the filled out form in our database.
* Edit: To open a pop-up form that allows user to enter certain information user wants to change about a student and uses the input student id to execute the UPDATE sql statement in the backend.

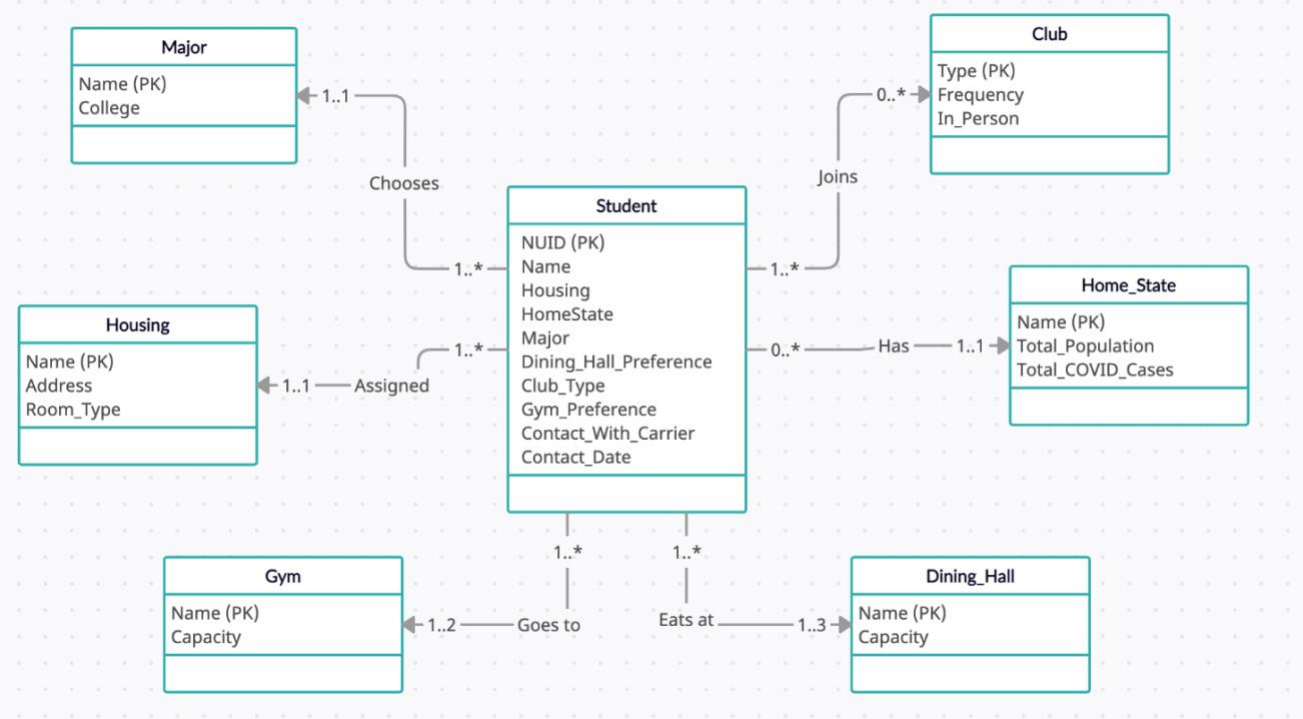
* Delete: When the button is clicked, the client sends the student id of the student in that row to the backend to execute the DELETE sql statement.
* Refresh: Sends an instruction to the backend to execute the SELECT sql statement and retrieve information from the student data table and use it to update the table displayed on the user interface.
* Homestate: Sends an instruction to the backend to execute the SELECT sql statement and retrieve information from the homestate data table and use it to update the pop-uped table displayed on user interface.
* Close: To hide the pop-uped form for "Edit" or the pop-uped table for "Homestate".

Back-end: Server

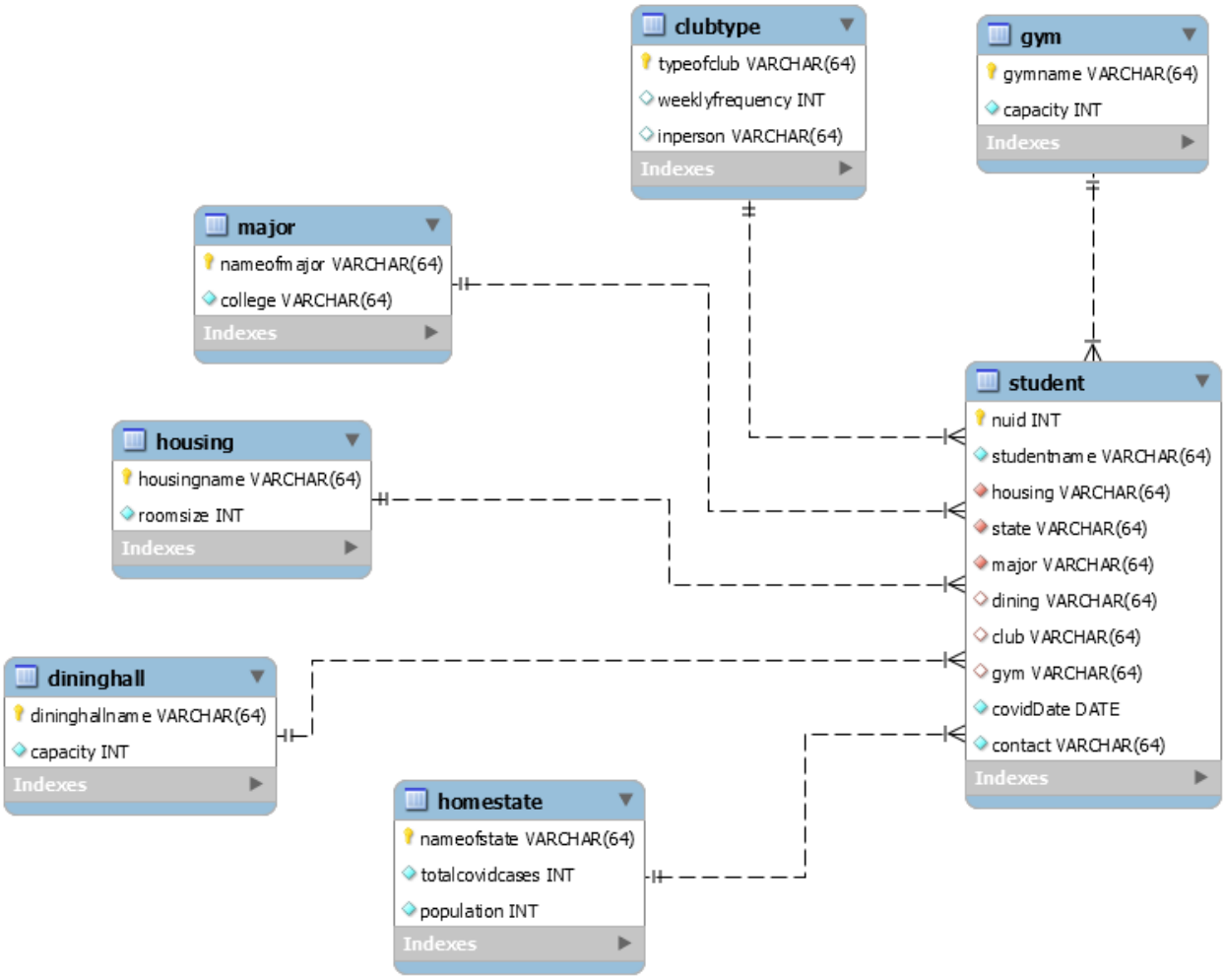
The server is the back end of the program. It uses Node.js to handle any instruction sent from the front-end and connect to the MySQL workbench server. Every time user clicks on certain button in the front-end, a function would be called that leads to another function in the backend which executes certain query to our database. Then the server would send the result of the query back to the front-end to update what the user can see on the user interface.

The details of how the front-end calls functions in the back-end is discussed in the buttons section of the client part.

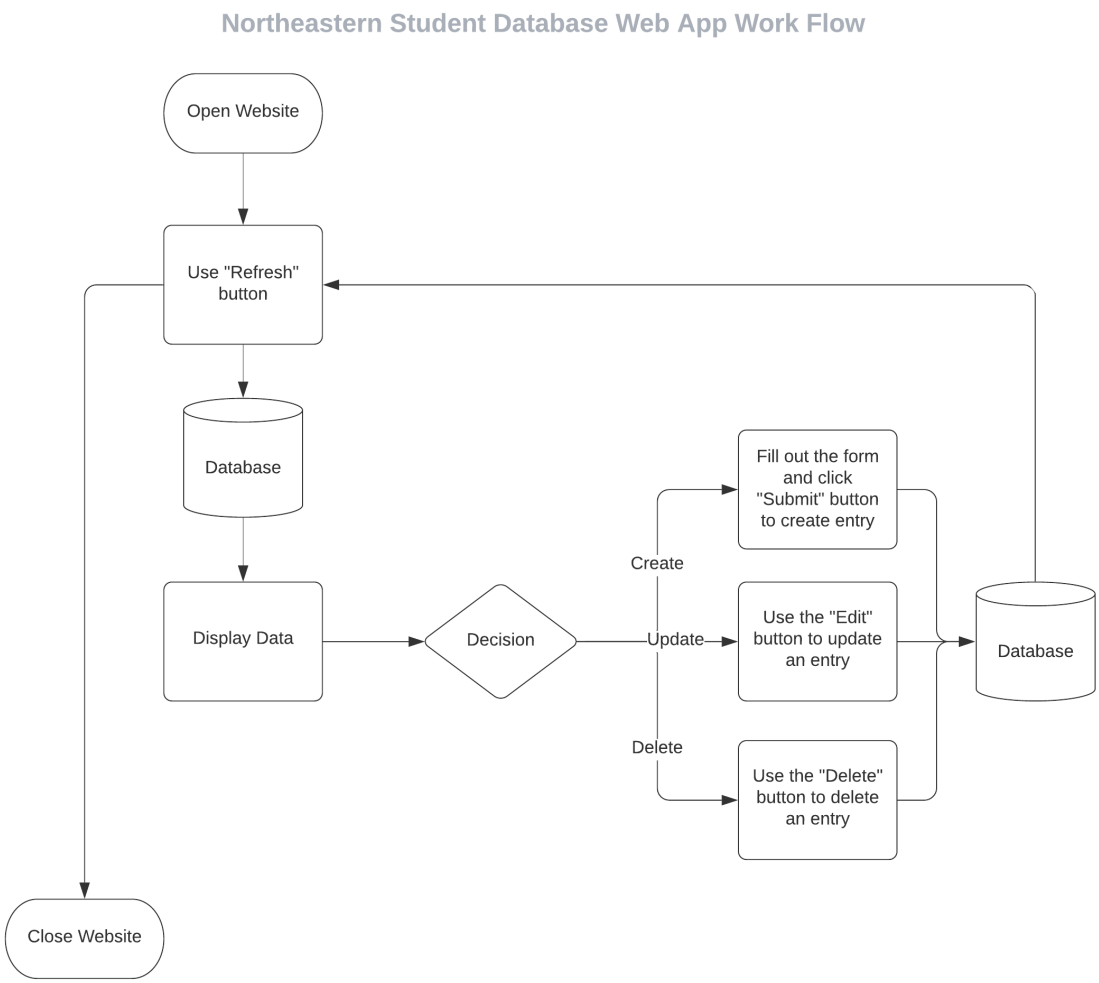
Conceptual UML Design:



Logical Design:



User Flow:



Lessons Learned and Future work:

A great deal of time was spent connecting the MySQL server to the front-end CRUD operations. Understandably, additional time was spent creating Node.js and building a query for the SQL database within the javascript. These two foundations of the project were the bulk of the workload and are the main technical knowledge and abilities that were gained. Time management was important per usual, but our commitment to the original architecture of the project proved useful. We were able to refine the technical specifications how we saw fit, without losing sight of the applicability and utility of the database as a whole.

After all, the purpose is to create something useful and effective, which we believe we've done. With that in mind, the purpose of this database is a way for Northeastern to gain insights into how COVID-19 has affected certain student populations. Specifically, we envision the database being used for on-campus events, club meetings, and other social gatherings in the summer and beyond. If 100 students are expected to attend an in-person event, using our database functionality Northeastern can require students to fill out certain information. This can then be a tool for organizers to complete risk assessments and determine if the event is safe for attendance. Additionally, it can be a way for organizers to quickly and efficiently aggregate demographic information for use at the event itself. For example, students with different backgrounds, different club affiliations, and different majors could all be placed in one group at the event, allowing students ample branching out opportunities. Vice versa could be applied if organizers are looking for students to be grouped by residence hall or home state. COVID-19 has drastically affected how students interact, and we believe our database could help give students safe social interactions that have been lacking.

Potential areas of added functionality essentially surround how niche topics could be explored. This could include applying this type of database to any major event, such a speaker coming to Northeastern that wants to know more about the students he/she are presenting for. If the functionality within the database included "fast facts" or data aggregation statistics on command it could have a wider range of uses. Overall, it can be a useful demographic descriptor in this sense and create meaningful connections that wouldn't have existed without.