Presentation CMPT 353 report:

Part A:

* **Docker & Docker-Compose:** I used docker-compose to build a container for the server and database because enables more efficient use of system resources. The node container is dependent on the database container so that I do not have to download MySQL in my PC.
* **Folder & Files:** I organize my files by segregating them. So, I create two folders which is pages and public. Because in the server I used a method of app.use on to the public folder to be able to use the JavaScript, images, and CSS on the server. Then for the webpage I placed them in page folder for neat looking.
* **Database:** Before I run the server, I have to pre-build the database and tables.

Products table:  
CREATE TABLE products (prod\_id MEDIUMINT NOT NULL AUTO\_INCREMENT, drink\_name VARCHAR(255) NOT NULL, price VARCHAR(125) NOT NULL, pic\_src VARCHAR(255) NOT NULL, PRIMARY KEY (prod\_id) );

Orders table:  
CREATE TABLE orders (order\_id MEDIUMINT NOT NULL AUTO\_INCREMENT, list\_prod VARCHAR(255) NOT NULL, total\_price VARCHAR(125) NOT NULL, timestamp VARCHAR(512) NOT NULL, PRIMARY KEY (order\_id) );

Pending table:  
CREATE TABLE pending ( list\_prod VARCHAR(255) NOT NULL, total\_price VARCHAR(125) NOT NULL, timestamp VARCHAR(512) NOT NULL );

The order table will be where the past orders are. Next, the pending table is where the order has not been able to make yet or have not pick it up yet. The reason why there are two tables for orders because if the customer wants to cancel the order, the process of deleting it will be easy and easy to record if there is someone order after you. Then last, is the product table this is where the info of the drinks is stored.

* **Code**: After setting it up when the server runs it will try to connect to the database and it will send a text on the console for the user that it can now run webpages that will be needed for the database. I created mine by making a lot of post and get method so when a script is calling a xml it can just call the method to use the function. The names of the path are straight-forward.
* **Functions:** This is how I store the data to the database. From the webpage of the customer, the customer can choose a drink and have the choice to input on how many quantities they want. As the customer press purchase, in the JavaScript it will check how many items are in the cart and its quantity then it will place drink’s name in a string that will be sending it from the XMLhttp to the server. In the server, the send params will go to the product table to get the product ID only and place the IDs to a string and that will serve as the list\_prod field in the pending table. Next the customer can now go to the pending order page, the page will show the items that was purchased, and it will have a progress bar. When it reaches half the cancel order will disappear and then if it finishes loading the order is complete. After that, the items will not show anymore in the page and send a signal to the server to delete the data from the pending table and place it to the order.

Part B:

* For the front-end I picked react and for the back-end is python flask. The reason why I picked react is easy set up, easy to understand, simple coding. Then for the python flask, I picked that because since my level of coding in python is good it will be easy to build a back-end also it has a module database which is SQLalchemy. Also flask is considered more Pythonic than the Django web framework.

Part C:

* **Benefits:**
  + React – One key feature I like in react is the useState & useEffect when getting the json data from the server I can easily place it in the useState. The useState can be a string, int, array. And when I want to post the data to the webpage I can use the useState by using a .map this basically a for loop. Next is that when I want to get a data from other website the CORS of react is quite easy to get the API it is just few codes. Compare to the JavaScript there are a lot of lines of code just to get the data from the server. Another feature in react is the building components. These types of components can help organize creating a web. There are two types I know is the functional component and the other one is the class component. Last key feature is setting up a react folder, basically you just need a nodejs, then type a command. After that it will create a folder and files you need to start a web.
  + Flask – First key feature I like in flask is simple to build an API web. There are two ways first is by using the app.route(). Similarly, to nodejs app.post/app.get. Next way is a restful flask basically is a class app.route(). I used the restful because is more neat looking and easy to read. Next feature is the RequestParser, this is very helpful when receiving a data from the client what this do is that whenever a client send a Form the request parse automatically get the data and place it in the variable. The thing it makes it great is that you can define what kind of data you want to get. You can put a required tag this will check if the client input something if not it will pop the help tag to the client. Last thing I like is that when querying something from the database you don’t have to worry about the async, not like in nodejs you have to be aware. Because it sometimes annoys why you are not getting the data in order. In flask it will still be synchronous.

Part D:

* For this new technology, it will still need a pre-built database and tables, but this time for the tables will be having four. In SQLalchemy flask to create table it can be in the server file by creating a class Model and putting columns for the table. The different to this table to the MySQL is that you can put a relationship column. This will help organize the connection between table for example there is a table for authors and books. What the relationship column do is that if you want get the books of a specific author you can just do author.<name of column> this will go to the table of books and look for all the books of this author in just 1 line of code. The extra table is for the items of the order, and the order table will be the sets of those items. So, this way when getting the order history, I can easily access the items of the order. Unlike in the nodejs when I am getting the order history, I have to use the string products ID then loop to get the data one by one. But for flask since I have the relationship, I can just type 1 line of code to get all information of the drinks. In short it lessens the code line. Then moving on to the client-side when posting a data from server to data react surely lessen the hassle of it. In HTML JavaScript has more workload needs do because since they don’t have useState and .map its hard to put the data to the webpage. Based on the screen shots (presentation) there is a big difference of lines of code in react Its only 20 lines of code. Then in JavaScript HTML its 47 lines of code. From the chosen technologies I picked it really made my web developing easy.

TESTING:

* Prints out every status when inserting, deleting, selecting to the MySQL database.
* Restriction of the client when purchasing they can’t purchase when they are no items in the cart. It will pop an alert to notify them.