Our Project, Traits and Tastes, is based on React for the front end and Node.js, and Express for the backend.

The Database was collected from Kaggle, using data from Vivino, which is the biggest wine site.

TABLE `WineCharacteristics` has a FOREIGN KEY (`WineID`) REFERENCES `WineDetails` (`WineID`), FOREIGN KEY (`TypeID`) REFERENCES `WineTypes` (`TypeID`), and FOREIGN KEY (`StyleID`) REFERENCES `WineStyles` (`StyleID`)

TABLE `WinePricingVolume` has a FOREIGN KEY (`WineID`) REFERENCES `WineDetails` (`WineID`)

TABLE `WineSearchCounts` has a FOREIGN KEY (`wineID`) REFERENCES `WineDetails` (`WineID`)

Finally, there is a view for the further use of Weekly rankings.

Our project proposal was about craft beers, however, because it was hard to find expertized research about the relationship between beer and psychology, we had to change the beverage type to wine. According to several articles and research, there are some influences from human emotion in selecting a certain kind of wine. Based on the FOOD QUALITY AND PREFERENCE paper, we made a wine recommendation website that uses the Myers-Briggs (MBTI) test.

this is what the questions look like. (I AM GONNA CLICK PERSONALITYTEST.TSX, match this line with the video)

So, introducing the project,

This is the App that holds Routes. We have two pages which are Home and PersonalityTestPage, components Footer, Header, Modal, Rankings, and Searchbar.

The home page has sections divided into home, rankings, about, and contact.

The search bar is where the main event happens. Users search from the search bar and can do an advanced search using the sort and filter. Also, clients can be redirected to Google Shopping to buy the product by clicking the Go Shop button.

From the PersonalityTestPage, users take a test and get wine recommendations based on their traits. By clicking the Share button users can share the result with friends.

The server side is the most important folder in our project because all of the interaction happens here.

We talk with the database and the front end using queries such as search-wines, weekly rankings, record-selection, and API/recommendations.

Search-wines are for searching. It gets input from the user and tries to search from it. Also, when the user clicks sort or filter by, the query takes it as an input and calls the result.

After searching, clients will click the selected wine. With record selection, we count the clicks and save it into the table, so that we can get the weekly rankings. Because we are getting the weekly ranking, we use View and collect data from the starting week using

WHERE s.weekOfYear = WEEK(CURDATE(), 1) AND s.year = YEAR(CURDATE())

this line.

Finally, API/recommendations are for the test result. After the test, the test determines which trait the user has. Based on the trait, it gives two random wine recommendations that the user will enjoy. It provides the wine based on the grape because grapes are the most important factor that differentiates the wine taste. As explained before, each trait has a taste preference due to the personality, which is a grape type in wine.

Now I will show how the app works.

Users will use the search bar and sidebar to search for wines.

Below, there is a weekly ranking that shows popular wines this week.

If you click the MBTI button on the header, it moves to the page. It shows the questions.

After submitting the result is here. Clients can press Go shop to buy it, click try again to do it multiple times, or click Share to share the result.

Also from the searched wine, there is a go shop that redirects to Google Shopping.