Reflection / Output Report

1. What did you learn from this lab?

In this lab, I learned how to use Firebase Firestore to store and manage data in a web application. I was able to connect my HTML file to Firestore and perform CRUD (Create, Read, Update, Delete) operations. It was interesting to see how Firestore works differently from traditional databases, especially with how it handles real-time data updates. I also got more comfortable working with JavaScript's asynchronous functions, since Firestore operations don't happen instantly like in SQL.

2. What challenges did you encounter while connecting to Firestore?

One of the biggest challenges I faced was making sure my Firebase configuration was correct. At first, I had trouble getting my database to show up in the web app because I didn't properly set up Firestore imports. I also ran into some permission errors, which I later realized were caused by Firestore's security rules. Debugging these issues took some time, but I learned how to check Firebase's console for error messages and adjust the security rules when needed. Another challenge was updating and deleting data, since I had to correctly reference document IDs to modify the right records.

3. How does Firestore differ from traditional databases?

Firestore is different from traditional databases like MySQL because it's a NoSQL database. Instead of using tables with rows and columns, Firestore stores data in collections and documents. This makes it more flexible because I don't have to define a fixed structure for my data. Another major difference is that Firestore updates in real-time, so any changes made in the database show up instantly on the website. Also, Firestore is cloud-based and managed by Google, meaning I don't need to set up my own database server. This makes it easier to use but also means I have to rely on Firebase's built-in tools and pricing model.