This project shows how a front end (React) and a back end (Node.js with Express) can work together to share data.
On the front end, a user types their name and a message into a form. When they click "Submit," the data is sent to the back end using a **fetch POST request**.

The **back end** receives this data, stores it in a simple list (an array), and sends it back to the front end. The React app then updates the screen to show the new message right away. This process helps show how web apps let users send and see new data instantly.

Technical Details

- **Node.js** runs JavaScript on the server.
- **Express.js** makes it easier to set up routes and handle requests (like GET, POST, and DELETE).
- **CORS** lets the front end and back end talk to each other, even if they run on different ports (React usually runs on port 3000 and Express on port 5000).
- **Body-Parser** reads the JSON data that the React app sends.
- Data is stored in an **array** (not a database yet), so it's easy to test and understand.

In the React front end, the `fetch()` function was added to send the data to the Express server. When the server replies, React updates its state and re-renders the page, showing the new message.

My Experience with the Project

Working on this project helped me understand how the front end and back end talk to each other. At first, I didn't understand how to make the React app send data to the server or why I was getting CORS errors. After learning about Express middleware and testing the routes in Postman, it made a lot more sense.

I learned that the front end and back end are like two sides of the same conversation — the front end asks for something, and the back end answers with data. Seeing my data

appear right after I submitted the form was really exciting because it showed that my full stack system was working.

The hardest part was getting the communication between the two sides right, but once that worked, everything else felt much easier. I feel more confident about connecting my React app to a Node.js server now.