Development Approach

For building StockSync, I started by breaking the project into smaller parts – stock price monitoring, live chat, and analysis charts. I first worked on setting up the backend using Node.js to handle data fetching from a real-time stock API. Once the backend was stable, I moved on to the frontend with React.js to create a clean and responsive interface. I used socket programming to enable instant updates for both stock prices and live chat messages. The MongoDB database was set up to store user accounts, chat history, and stock data. I tested each feature separately and then integrated them together to make sure everything worked smoothly.

Technologies Used

The backend was developed with **Node.js** and **Express.js**. For the frontend, I used **React.js** along with **HTML** and **CSS** for styling. **MongoDB** was chosen as the database for storing user and stock information. Real-time updates were handled using **Socket.IO**, and chart visualizations were implemented with a chart library to display stock trends.

Challenges Encountered

One of the main challenges was managing real-time updates without delays, especially when both stock data and chat messages were updating simultaneously. Another challenge was API rate limits, which required optimizing the way stock data was fetched to avoid exceeding the limits. Making the charts both responsive and accurate was also tricky, as they needed to update smoothly while still being readable. Despite these challenges, I was able to build a functional and user-friendly application that works in real time.