Frontend Technologies

1. HTML/CSS:

* For structuring and styling your web app.
* Consider using a CSS framework like Bootstrap or Tailwind CSS for responsive design.

JavaScript Framework:

- React: For building interactive UI components and managing state.

- Alternatively, you could use Vue.js if you prefer a more straightforward approach.

3. Data Visualization Library:

- Chart.js or D3.js: To create dynamic and interactive charts for displaying stock price trends.

Backend Technologies (Optional)

1. Node.js:

- For building the backend of your application.

- Use Express.js to create a simple RESTful API to handle requests.

2. Database (if you want to store historical stock data):

- MongoDB (NoSQL) or PostgreSQL (SQL): Depending on whether you prefer a NoSQL or relational database.

API and Data Management

1. API for Stock Prices:

- Use the official LuSE API or a third-party financial data provider that offers stock price data.

2. Axios or Fetch API:

- For making API requests from your frontend to retrieve stock data.

Real-Time Functionality (Optional)

1. WebSocket Library:

- Socket.IO: If you decide to implement real-time updates for stock prices.

Development Tools

1. Version Control:

- Git: For version control of your code.

- GitHub: To host your code repository.

2. Development Environment:

- Visual Studio Code: A popular code editor with excellent JavaScript support.

Deployment

1. Hosting Platform:

- Netlify or Vercel: For deploying your frontend application.

- Heroku or DigitalOcean: If you decide to host a backend server.

Optional Enhancements

- Testing Framework:

- Jest or Mocha for writing tests for your application.

- State Management:

- Redux (for React) or Vuex (for Vue) if your application grows complex.

This tech stack will give you a solid foundation for building your web app while also providing ample opportunities to learn and grow your JavaScript skills!