Here's a detailed assignment task designed for a fullstack development role focusing on the MERN stack, which includes using Express.js, React.js (or Next.js), and Node.js.

Full Stack Developer Assignment: MERN Stack

Project Name: Bridge

Overview:
Develop a full-stack application using the MERN stack. The application will interact with the XY Finance API to fetch and display cryptocurrency data. The backend will be built with Node.js and Express, and the frontend will be developed using React.js or Next.js.

Impress

Backend Requirements:
1. **Technology Stack**:
   - Node.js with Express for the API server.

2. **API Endpoints**:
   - `GET /tokens`: Fetches supported tokens for a specified blockchain from the XY Finance API and returns them to the frontend for user selection.
   - `POST /quotes`: Receives a user's token and chain selection, queries the XY Finance Quotes API for a quote, and sends the data back to the frontend.
   - `POST /params`: Triggered when a user accepts a quote (clicks on "bridge"). Calls the XY Finance Create Transaction API to fetch the transaction parameters and returns these to the frontend.

3. **Structure**:
   - Organize the backend code with separate directories for controllers, routes, services, utils, and constants to ensure modularity and maintainability.

4. **Testing**:
   - Include backend tests using a framework like Mocha or Jest to validate API endpoints and business logic.

Frontend Requirements:
1. **Technology Stack**:
   - Use React.js or Next.js for the frontend.
   - State management (if needed) can be handled by Context API or Redux.

2. **Features**:
   - Dynamic display of tokens fetched from the backend.
   - Form to submit token and chain selection, and display the fetched quote.
   - Display the gas needed to execute the transaction.
   - Confirmation screen to initiate the "bridge" action and display transaction parameters.

3. **Structure**:

- Organize the frontend code into components, utils, constants, and URLs to maintain a clean structure.

Clean Code Practices:
- Ensure code readability and maintainability with proper naming conventions, consistent coding style, and comprehensive comments.
- Implement error handling and data validation both on the frontend and backend.

Deployment:
1. **Repository**:
   - Push the complete code to a GitHub repository named `bridge`.
2. **Backend Hosting**:
   - Deploy the backend on Vercel or OnRender.
3. **Frontend Hosting**:
   - Deploy the frontend on Vercel.

Submission:
- Provide the GitHub repository URL and the deployed application URLs.

Support:
- For any issues or clarifications, the candidates are encouraged to get in touch with the hiring manager or the technical team.

Evaluation Criteria:
- Functionality: All endpoints and user interactions should work as expected.
- Code Quality: The code should be clean, well-organized, and easy to understand.
- Design and UX: The frontend should be user-friendly and visually appealing.
- Testing: The backend should have meaningful tests, showcasing the candidate's ability to write testable code.

This task is designed to assess a candidate’s full-stack development skills, attention to detail, and ability to work with modern web technologies and APIs. The focus on clean architecture and production-like structure reflects the importance of scalable and maintainable code in professional environments.