Full-Stack Developer Assessment

Objective:

The goal of this assessment is to evaluate candidates' proficiency in the MERN stack (MongoDB, ExpressJS, ReactJS, Node.js) and real-time communication with Socket.io. The assessment simulates a scenario in which they must build a mini version of a collaborative workspace application (similar to Discord/Slack), focusing on the following:

1. Real-time chat with different channels.
2. Real-time notifications for specific events.
3. User authentication with JWT.
4. Basic CRUD operations with MongoDB (e.g., creating/deleting channels).
5. Handling real-time updates in an active forex trading room (mocked with dummy price changes).
6. Role-based access control for channels and chat.

Instructions

Backend:

1. User Authentication (JWT-based):
   - Implement authentication using JWT.
   - Allow users to register, log in, and get an access token.
   - Store passwords securely using bcrypt.

2. MongoDB Schema:
   - Users: Each user should have the following fields:
     - _id, username, email, password, role (admin, trader, guest)
   - Channels: Define different chat channels for the forex trading room.

  - _id, name, created_by, members, isPrivate
- Messages: Store all the messages sent in each channel.
  - _id, message, sender, channel_id, timestamp
- Forex Data: Store a log of real-time forex data updates.
  - _id, pair (e.g., EUR/USD), price, timestamp

3. Role-Based Access Control:
- Only admins can create or delete channels.
- Traders can view and send messages in trading-related channels.
- Guests can only view certain public channels (e.g., General).
- Create middleware to enforce these permissions based on user roles.

4. Real-Time Functionality:
- Use Socket.io to handle real-time chat within channels.
- Implement a real-time forex feed (using random price generators).
- Notify users when someone joins or leaves a channel.
- Broadcast forex price updates to all users in the "Trading" channel.

Frontend:

The frontend should be built with ReactJS and include the following key features:

1. Login and Registration Page:
- Allow users to register and log in with their credentials.
- Validate the input and show error messages if there are issues.
- After login, store the JWT token in localStorage or cookies.

2. Channel Management:

- Display the available chat channels in a sidebar.
- Allow users with the "admin" role to add/delete channels.
- Non-admin users should only be able to join and leave channels.

3. Chat Interface:

- Implement a chat interface where users can send and receive messages in real-time using Socket.io.
- Messages should be persisted in MongoDB and loaded when the user joins a channel.
- Display timestamps and username next to each message.

4. Real-Time Forex Feed:

- In the "Trading" channel, display a panel showing real-time forex price updates for various currency pairs (simulate with random price changes).
- Use Socket.io to push these updates to all users in real-time.
- Show forex price data in a table format with currency pairs, bid/ask prices, and timestamp.

5. Notifications:

- Implement a notification system that alerts users when:
  - They receive a new private message.
  - A new user joins their channel.
  - Forex price updates occur in the "Trading" channel.
- Notifications should be shown in real-time.

6. User Presence:

- Track when users are online or offline.

- Show an "online" indicator next to usernames in the chat.

Wireframes:

Wireframe 1: Login Page

```
---------------------------------------
|           Forex Room                |
|-------------------------------------|
|   [Username]                        |
|   [Password]                        |
|   [Login]   [Register]              |
---------------------------------------
```

Wireframe 2: Main Application Layout

```
-------------------------------------------------------------------------
|   Channel Sidebar   |       Chat Area       |   Forex Price Feed Panel  |
|---------------------|-----------------------|---------------------------|
|  General            |  [User: You]          |  Pair | Bid  | Ask        |
|  Trading            |  [Messages...]        |  EUR/USD  1.05  1.06      |
|  Alerts             |  [Messages...]        |  GBP/USD  1.25  1.26      |
|  Create New Channel |  [Input message...]   |  ...                      |
-------------------------------------------------------------------------
```

Scoring Criteria:

- Correctness of functionality: Are all required features implemented? Are JWT authentication, role-based access, and Socket.io functioning as expected?

- Code quality: Is the code clean, readable, modular, and well-organized?

- Frontend design: Is the UI intuitive and consistent with the wireframes? Does it handle real-time updates well?

- Real-time features: Is the forex price feed updating in real-time? Do real-time notifications work correctly?

- Security: Are authentication and authorization implemented securely? Are passwords hashed? Are JWT tokens validated correctly?

- Edge cases: Does the application handle edge cases gracefully (e.g., disconnects, expired JWT tokens)?