Objective: Build **Slack Connect**, an application enabling users to connect their Slack workspace, send messages immediately, and **schedule messages** for future delivery. This assignment assesses your skills in full-stack development (TypeScript/JavaScript, Node.js), OAuth 2.0 integration, and token management.

Core Requirements (Mandatory):

Your application must fulfill the following:

1. **Secure Slack Connection & Token Management:**
   - Implement the **OAuth 2.0 flow** to connect to a Slack workspace.
   - Your backend service must securely store both **access and refresh tokens**.
   - Implement **refresh token logic** to automatically obtain new access tokens when old ones expire, ensuring continuous service without user re-authentication.
2. **Message Sending (Immediate & Scheduled):**
   - Provide a UI to select a Slack channel and compose a message.
   - Allow users to **send the message immediately** to a channel.
   - Allow users to **schedule the message** for a specific future date and time. Your backend must persist scheduled messages and reliably send them at the designated time.
3. **Scheduled Message Management:**
   - Display a list of all currently scheduled messages.
   - Enable users to **cancel** a scheduled message before its send time.

Technology Stack:

- **Frontend: TypeScript**, JavaScript (ES6+), any modern JS framework/library (e.g., React, Vue.js, Angular).
- **Backend: Node.js** (with Express.js, **use TypeScript**).
- **Persistence:** Use a lightweight solution like SQLite, LowDB, MongoDB, or simple JSON for storing tokens and scheduled messages.

Deliverables:

1. **Public GitHub Repository:**
   - A **public GitHub repository** with all your source code.
   - A comprehensive `README.md` file including:

        - **Detailed Setup Instructions:** How to clone, install, configure (e.g., Slack credentials), and run both frontend and backend locally.
        - **Architectural Overview:** Brief explanation of your design, focusing on OAuth, token management, and scheduled task handling.
        - **Challenges & Learnings:** Any significant hurdles and how you overcame them.
2. **Live Project Deployment (Brownie Points!):**
    - Deploying your application online (e.g., Netlify/Vercel for frontend, Heroku/Render/Glitch for backend) will earn you **brownie points**.

Submission:

Please submit your assignment within 7 calendar days from the date you receive this assignment.

Please email the URLs for your **public GitHub repository** and (if applicable) your **live project deployment** to (add in cc):

- **devesh.anand@gocobalt.io**
- **careers@gocobalt.io**
- **saroj@gocobalt.io**

Evaluation:

We'll assess your **functionality and correctness** (especially refresh token logic and scheduling), **code quality**, **architectural design**, and **documentation**.