**Title: Simple Reminder API**

**Tool Details:**

* **Backend:** Node.js with Express.js
* **Frontend:** Web Components (vanilla JavaScript, no frameworks)
* **Database:** In-memory storage (or optional database like MongoDB)
* **AI Integration:** Manage and retrieve reminders dynamically

**Goal:** Candidates will learn how to build an Express.js backend that allows users to set and retrieve reminders and integrate it with a WebComponent-based frontend. They will practice API request handling, user input validation, and dynamic UI updates.

**Assignment Description:** Candidates will create a Simple Reminder API where users can create, view, and delete reminders. The backend will handle storing and managing reminders, while the frontend will provide a user-friendly interface to interact with the API.

**Tasks & Steps:**

1. **Backend Development:**
   * Set up an Express.js server.
   * Create API endpoints to:
     + Add a new reminder.
     + Retrieve all reminders.
     + Delete a reminder by ID.
   * Store reminders in memory (or use a database like MongoDB for persistence).
2. **Frontend Development:**
   * Create a Web Component with an input field for entering a reminder and a submit button.
   * Send a request to the backend to save the reminder.
   * Retrieve and display the list of reminders dynamically.
   * Provide an option to delete reminders.
3. **Result Display:**
   * Show all saved reminders with timestamps.
   * Update the list dynamically when reminders are added or deleted.

**Mathematical Calculation/Steps:**

* Generate a unique ID for each reminder (e.g., using timestamps or a random string generator).
* Store reminders as objects containing { id, text, timestamp }.
* Implement a method to filter out reminders by ID when deleting them.

**Third-Party Packages (if required):**

* express (for backend server)
* cors (to allow frontend-backend communication)
* body-parser (to handle JSON requests)
* uuid (optional, for generating unique reminder IDs)
* mongoose (optional, if using MongoDB for persistent storage)

**Acceptance Criteria:**

* The backend should correctly handle storing, retrieving, and deleting reminders.
* The frontend should capture user input and dynamically update the UI.
* The API should return accurate reminder data.
* The Web Component should encapsulate input handling and result display.
* Users should receive an error message if they try to delete a non-existent reminder.

**Submission Guidelines:**

1. Fork the provided repository.
2. Create a folder with your name inside the repo.
3. Implement both backend and frontend in the designated folder.
4. Push the completed code to your forked repo.
5. Submit a pull request for review.

The assignment is complete when the backend correctly manages reminders, returns accurate responses, and the frontend successfully updates the UI based on API interactions.