**Title: Username Availability Checker**

**Tool Details:**

* **Backend:** Node.js with Express.js
* **Frontend:** Web Components (vanilla JavaScript, no frameworks)
* **Database:** Simulated in-memory storage (or optional database like MongoDB)
* **AI Integration:** Check username availability dynamically

**Goal:** Candidates will learn how to build an Express.js backend that verifies username availability 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 Username Availability Checker where users can input a desired username, send it to the backend via an API request, and receive a response indicating whether the username is available or already taken. The frontend will dynamically update the UI based on the response.

**Tasks & Steps:**

1. **Backend Development:**
   * Set up an Express.js server.
   * Create an API endpoint that checks username availability.
   * Maintain a list of existing usernames (either in-memory or stored in a database).
   * Return whether the username is available or taken.
2. **Frontend Development:**
   * Create a Web Component with an input field for entering the username and a check button.
   * Send the username to the backend via a fetch API request.
   * Receive the response and dynamically display the availability status.
3. **Result Display:**
   * Show "Username available" or "Username already taken" based on the API response.
   * Provide real-time validation feedback when the user types.

**Mathematical Calculation/Steps:**

* Check if the entered username exists in the stored list of usernames.
* If found, return "taken"; otherwise, return "available".

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

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

**Acceptance Criteria:**

* The backend should correctly verify username availability.
* The frontend should capture user input and dynamically update the UI.
* The API should return a valid response indicating availability.
* The Web Component should encapsulate input handling and response display.
* Users should receive real-time feedback when checking a username.

**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 processes username availability, returns accurate responses, and the frontend successfully updates the UI based on API responses.