**Title: Simple Poll System**

**Tool Details:**

* **Backend:** Node.js with Express.js
* **Frontend:** Web Components (vanilla JavaScript, no frameworks)
* **Database:** Not required (store votes in memory or log them to a file)
* **AI Integration:** Process and count votes dynamically

**Goal:** Candidates will learn how to create an Express.js backend for handling polls and integrate it with a WebComponent-based frontend. They will practice API request handling, dynamic vote counting, and UI updates.

**Assignment Description:** Candidates will develop a Simple Poll System where users can vote on a given question with multiple choices. The backend will store votes and return updated poll results, while the frontend will allow users to select an option and see real-time vote counts.

**Tasks & Steps:**

1. **Backend Development:**
   * Set up an Express.js server.
   * Create API endpoints to:
     + Fetch the poll question and options.
     + Submit a vote for a selected option.
     + Retrieve current poll results.
   * Store votes in memory or a simple JSON file.
   * Return updated poll results as JSON responses.
2. **Frontend Development:**
   * Create a Web Component with a poll question and multiple-choice options.
   * Allow users to select an option and submit their vote.
   * Send a request to the backend and update the UI with real-time vote counts.
3. **Result Display:**
   * Show the total votes for each option dynamically.
   * Prevent multiple votes from the same user (optional, using localStorage).
   * Display a success message after voting.

**Mathematical Calculation/Steps:**

* Count votes for each option and calculate the percentage of total votes.
* Example formula: (votes for option / total votes) \* 100.

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

* express (for backend server)
* cors (to allow frontend-backend communication)
* body-parser (to handle JSON request data)

**Acceptance Criteria:**

* The backend correctly stores and counts votes.
* The frontend allows users to vote and dynamically updates results.
* The API returns accurate poll results.
* Users cannot vote multiple times on the same poll (optional feature).
* The Web Component encapsulates the poll functionality properly.

**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 processes votes correctly, returns updated results, and the frontend dynamically updates based on user interaction.