**Title:** GPA Calculator

**Tool Details:**

* **Backend:** Node.js with Express.js
* **Frontend:** WebComponent-based UI
* **Database:** Not required (use in-memory storage if needed)

**Goal:**  
Candidates will learn how to develop a backend API with Express.js, integrate it with a frontend using WebComponents, and dynamically process user input using AI-generated data. The assignment focuses on API development, frontend-backend communication, and form handling.

**Assignment Description:** Candidates will create a **GPA Calculator** where users input their course grades and credits. The backend will process this data and return the calculated GPA. The frontend will display the result dynamically using a WebComponent. AI will generate test input values for dynamic processing.

**Tasks & Steps:**

1. **Backend API Development (Express.js):**
   * Set up an Express.js server.
   * Create a POST API endpoint to receive grades and credit hours.
   * Implement logic to calculate GPA based on received data.
   * Send the calculated GPA back as a JSON response.
2. **Frontend Development (WebComponent):**
   * Create a WebComponent-based form to accept course details (course name, grade, and credit hours).
   * Validate and structure user input before sending it to the backend.
   * Use Fetch API to send data to the backend and handle the response.
   * Display the calculated GPA dynamically.
3. **Integration & Result Display:**
   * Ensure seamless data flow between frontend and backend.
   * Display error messages for invalid input.
   * Use AI-generated test data for dynamic processing and testing.

**Mathematical Calculation/Steps:**

* **Formula for GPA Calculation:**  
  GPA=∑(grade×credit)∑creditGPA = \frac{\sum (grade \times credit)}{\sum credit}
* Assign numeric values to letter grades (e.g., A = 4.0, B = 3.0, etc.).
* Multiply each grade by its corresponding credit hour.
* Divide the total weighted grades by the total credit hours.

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

* express – for creating the backend API
* cors – for handling cross-origin requests
* body-parser – for parsing request bodies (if needed)

**Acceptance Criteria:**

* The backend API correctly receives, processes, and returns GPA data.
* The frontend form successfully captures user input and displays the calculated GPA.
* Proper validation and error handling are implemented.
* The application runs without errors and provides accurate GPA calculations.

**Submission Guidelines:**

1. **Fork** the provided repository.
2. **Create a folder** with your name inside the repo.
3. **Implement** the backend and frontend components as per the requirements.
4. **Push** your code to your forked repository.
5. **Submit a Pull Request (PR)** with a short description of your implementation.

**Reference:**

1. [**https://www.calculator.net/**](https://www.calculator.net/)

Ensure that your implementation correctly integrates the frontend with the backend and handles all user inputs dynamically.