**Title: Compound Interest Calculator**

**Tool Details:**

* **Backend:** Node.js with Express.js
* **Frontend:** WebComponent-based UI
* **Database (Optional):** JSON storage or in-memory data handling

**Goal:** Candidates will learn to:

* Develop a RESTful API using Express.js
* Implement a WebComponent-based frontend for user input and result display
* Integrate frontend and backend through API calls
* Process and return compound interest calculations dynamically

**Assignment Description:** The objective of this assignment is to build a full-stack Compound Interest Calculator. The frontend will feature a form where users input the principal amount, interest rate, time period, and compounding frequency. The backend will process the data, compute the compound interest, and return the result for display on the frontend. AI should generate different input values dynamically for testing purposes.

**Tasks & Steps:**

1. **Backend API Development:**
   * Set up an Express.js server.
   * Create an endpoint to receive principal, rate, time, and frequency.
   * Perform compound interest calculations based on the input.
   * Return computed results to the frontend.
2. **Frontend Development:**
   * Create a WebComponent-based form with input fields for principal, rate, time, and frequency.
   * Send form data to the backend using fetch API.
   * Display the computed compound interest received from the backend.
3. **Integration & Testing:**
   * Ensure the frontend correctly sends user input to the backend.
   * Validate responses from the backend.
   * Handle errors gracefully in both frontend and backend.

**Mathematical Calculation/Steps:**

* **Compound Interest Formula:**
  + A = P (1 + r/n)^(nt)
  + Where:
    - A = Final amount after interest
    - P = Principal amount
    - r = Annual interest rate (in decimal form)
    - n = Number of times interest is compounded per year
    - t = Number of years
  + Compound Interest = A - P

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

* express (for backend API)
* cors (to handle cross-origin requests)
* body-parser (for processing JSON input)

**Acceptance Criteria:**

* The backend API correctly receives and processes input data.
* The WebComponent-based frontend interacts seamlessly with the backend.
* The compound interest calculation is accurate, considering different compounding frequencies.
* AI dynamically generates input scenarios for testing.
* The frontend displays results in a user-friendly manner.

**Submission Guidelines:**

1. Fork the provided repository.
2. Create a new folder with your name inside the repository.
3. Implement the solution within the folder.
4. Push the completed code to your forked repository.
5. Submit a pull request to the original repository.