**Title: Interest Calculator**

**Tool Details:**

* **Backend:** Node.js with Express.js
* **Frontend:** WebComponent-based UI
* **Database (if needed):** In-memory storage (e.g., JSON file)

**Goal:**

By completing this assignment, candidates will:

* Learn how to build and expose RESTful APIs with Express.js.
* Understand how to create and use WebComponents for frontend development.
* Gain experience in handling user input, processing data on the backend, and displaying results dynamically.
* Learn how to integrate AI-generated content dynamically.

**Assignment Description:**

Develop an **Interest Calculator** that allows users to input **principal amount, interest rate, and time period**. The **frontend** will send this data to the **backend**, which will calculate **simple and compound interest** and return the results. The **frontend** will then display the calculated values dynamically.

The assignment must include AI-generated hints, tooltips, or recommendations for user inputs (e.g., suggesting average interest rates).

**Tasks & Steps:**

**Backend Development (Express.js)**

1. Set up a Node.js project and install Express.js.
2. Create an API endpoint (POST /calculate-interest) to accept principal, rate, and time.
3. Implement logic to calculate **simple interest** and **compound interest** dynamically.
4. Integrate an AI-based recommendation for interest rates based on user input.
5. Send the computed results as a JSON response.

**Frontend Development (WebComponent)**

1. Create a **custom WebComponent** for the interest calculator form.
2. Implement input fields for principal, interest rate, and time.
3. Add a submit button that triggers an API call to the backend.
4. Display AI-generated suggestions for user input dynamically.
5. Fetch the API response and update the UI with calculated values.

**Mathematical Calculation/Steps:**

* **Simple Interest (SI):** SI=P×R×T100SI = \frac{P \times R \times T}{100}
* **Compound Interest (CI):** CI=P×(1+R100)T−PCI = P \times (1 + \frac{R}{100})^T - P

Where:

* **P** = Principal Amount
* **R** = Interest Rate
* **T** = Time in Years

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

* express (for API development)
* cors (to allow frontend-backend communication)
* body-parser (to parse incoming requests)
* axios or fetch (for frontend API calls)

**Acceptance Criteria:**

* The **backend** should correctly calculate simple and compound interest and return the results.
* The **frontend** should collect user input, send requests to the backend, and display results dynamically.
* AI-generated suggestions should be visible in the UI.
* The app should function **without page reload**.
* The solution must be modular and well-structured.

**Submission Guidelines:**

1. **Fork** the given GitHub repository.
2. Create a **new folder** with your name or unique identifier.
3. Implement the **backend** in a /backend folder and the **frontend** in a /frontend folder.
4. Ensure the project runs with npm install and npm start.
5. Push your code and submit a **pull request** with a clear description.

**Reference:**

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