**Title: Rock Paper Scissors Game**

**Tool Details:**

* **Backend:** Node.js with Express.js
* **Frontend:** Web Components (vanilla JavaScript, no frameworks)
* **Database:** Not required (processing happens in memory)
* **AI Integration:** Simulate computer moves dynamically

**Goal:** Candidates will learn how to build an Express.js backend that processes Rock Paper Scissors game logic and integrates it with a WebComponent-based frontend. They will practice API request handling, game logic implementation, and dynamic UI updates.

**Assignment Description:** Candidates will create a Rock Paper Scissors game where users can play against a computer. The backend will randomly select a move for the computer and determine the winner based on user input. The frontend will allow users to select their move and display the game results dynamically.

**Tasks & Steps:**

1. **Backend Development:**
   * Set up an Express.js server.
   * Create an API endpoint that receives the user's choice (rock, paper, or scissors).
   * Generate a random choice for the computer.
   * Determine the winner based on game rules.
   * Return the result (win, lose, or draw) along with the computer's choice.
2. **Frontend Development:**
   * Create a Web Component with buttons for selecting rock, paper, or scissors.
   * Send the user's choice to the backend via a fetch API request.
   * Receive the response and dynamically display the game result.
3. **Result Display:**
   * Show the user's selected choice.
   * Show the computer's randomly generated choice.
   * Display the winner (User Wins, Computer Wins, or Draw).

**Mathematical Calculation/Steps:**

* The computer selects a random move from ['rock', 'paper', 'scissors'].
* Determine the winner using game rules:
  + Rock beats Scissors
  + Scissors beats Paper
  + Paper beats Rock
  + Identical choices result in a Draw

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

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

**Acceptance Criteria:**

* The backend correctly generates random moves and determines the game outcome.
* The frontend captures user input and dynamically updates the UI.
* The API correctly returns the game results.
* The Web Component encapsulates user interactions and displays the results properly.
* Users should see clear feedback for each round played.

**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 game logic correctly, returns accurate results, and the frontend successfully updates the UI based on API interactions.