**Title: Login System**

**Tool Details:**

* **Backend:** Node.js with Express.js
* **Frontend:** Web Components (vanilla JavaScript, no frameworks)
* **Database:** MongoDB (for storing user credentials securely)
* **AI Integration:** Basic authentication handling

**Goal:** Candidates will learn how to implement a secure login system with user authentication. They will practice API request handling, hashing passwords, validating credentials, and dynamically updating the UI based on authentication responses.

**Assignment Description:** Candidates will create a Login System where users can register and log in securely. The frontend will collect user credentials, send them to the backend for authentication, and display appropriate messages. The backend will handle user registration, password hashing, and authentication.

**Tasks & Steps:**

1. **Backend Development:**
   * Set up an Express.js server.
   * Create API endpoints for user registration and login.
   * Store user credentials securely in MongoDB using password hashing (bcrypt).
   * Validate login credentials and generate a JSON Web Token (JWT) for authentication.
2. **Frontend Development:**
   * Create Web Components for login and registration forms.
   * Send user input to the backend via a fetch API request.
   * Receive authentication responses and update the UI dynamically.
3. **Authentication & Security:**
   * Hash passwords using bcrypt before storing them in the database.
   * Verify hashed passwords during login attempts.
   * Use JWT for authentication and session handling.

**Mathematical Calculation/Steps:**

* Hash password using bcrypt.hash(password, saltRounds).
* Verify login with bcrypt.compare(enteredPassword, storedHashedPassword).
* Generate a JWT token for authenticated users.

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

* express (for backend server)
* bcrypt (for password hashing)
* jsonwebtoken (for generating authentication tokens)
* mongoose (for MongoDB database interaction)
* cors (to allow frontend-backend communication)
* body-parser (to handle JSON requests)

**Acceptance Criteria:**

* Users should be able to register securely with hashed passwords.
* Users should be able to log in using valid credentials.
* JWT authentication should be implemented for session management.
* The frontend should capture user input and dynamically update the UI.
* Error messages should be displayed for incorrect login attempts.

**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 securely processes login requests, correctly authenticates users, and the frontend successfully updates the UI based on authentication responses.