**Title: Base64 Encoder & Decoder Converter**

**Tool Details:**

**Technology Stack:**

* **Frontend:** Web Components (Lit or Vanilla JS)
* **Backend:** Express.js (Node.js)
* **Processing:** Convert text or files to Base64 encoding and decode Base64 back to original format
* **Data Handling:** API requests & response parsing

**Goal:**  
By completing this assignment, candidates will:

* Learn how to build a **RESTful API** using **Express.js**.
* Implement **Base64 encoding and decoding** using Node.js.
* Work with **Web Components** to create an interactive frontend.
* Understand **client-server communication** via API calls.
* Gain experience in **handling text and file encoding transformations**.

**Assignment Description:**

Develop a **Base64 Encoder & Decoder** where users can input **text or upload a file** via a **WebComponent-based form**. The backend, built with **Express.js**, will process the data, allowing users to **encode** content to Base64 or **decode** Base64 back to its original form.

**Tasks & Steps:**

**1. Backend API Development (Express.js):**

* Set up an **Express.js** server to accept **text or file input**.
* Implement API endpoints:
  + **Encode API:** Accepts raw text or a file and converts it to Base64.
  + **Decode API:** Accepts Base64 input and converts it back to text or a file.
* Handle proper MIME type validation for file uploads.

**2. Frontend (WebComponent-based UI):**

* Create a **form** using Web Components that allows users to:
  + Input **text** or **upload a file** for encoding.
  + Input **Base64 data** to decode back to original content.
* Send the data to the backend using the **fetch API**.
* Display the **converted output** (Base64 or decoded content).
* Provide an option to **download the converted file**.

**3. Integration & Testing:**

* Ensure the frontend **properly communicates** with the backend.
* Handle errors gracefully (e.g., **invalid Base64 data or unsupported file types**).
* Test the **Base64 encoding and decoding process** to ensure accuracy.

**Mathematical Calculation/Steps (if applicable):**

* **Encoding:** Convert text or binary data into Base64 using a character set (A-Z, a-z, 0-9, +, /).
* **Decoding:** Convert Base64 data back to original format using the reverse mapping of Base64 encoding.

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

* express (for backend server)
* body-parser (for handling text input)
* multer (for handling file uploads)
* lit (for WebComponent development)

**Acceptance Criteria:**

* The **Express.js backend** should successfully **encode and decode Base64** data.
* The **WebComponent-based frontend** should have a **form** for input text/file upload and Base64 input.
* The converted output should be **properly formatted and displayed**.
* Proper **error handling** should be in place for **invalid or unsupported data formats**.

**Submission Guidelines:**

1. **Fork** the provided GitHub repository.
2. **Create a folder** named base64-converter-<your-name>.
3. **Implement the backend and frontend** in the respective subfolders.
4. **Push the code** to your forked repository.
5. **Submit a pull request** with a brief description of your implementation.

**Ensure that the backend correctly encodes and decodes Base64 data while integrating seamlessly with the WebComponent-based frontend.**