**Title: PDF to Word Converter**

**Tool Details:**

**Technology Stack:**

* **Frontend:** Web Components (Lit or Vanilla JS)
* **Backend:** Express.js (Node.js)
* **File Processing:** PDF to Word conversion
* **Data Handling:** File uploads & API requests

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

* Learn how to build a RESTful API using **Express.js**.
* Implement **file handling and processing** in a backend service.
* Work with **Web Components** to create a modular frontend.
* Understand **client-server communication** via API calls.
* Gain experience integrating third-party packages for **file conversion**.

**Assignment Description:**

Develop a **PDF to Word Converter** where users upload a PDF file via a WebComponent-based form. The backend, built with **Express.js**, will process the PDF file and convert it to a Word document. The processed file should be available for download.

**Tasks & Steps:**

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

* Set up an Express.js server to handle file uploads.
* Use a suitable package to convert PDFs to Word format.
* Implement API routes for file processing and downloading the converted document.

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

* Create a form using Web Components that allows users to upload a PDF file.
* Send the uploaded file to the backend using **fetch API**.
* Display the status of the conversion process and provide a download link once the file is ready.

**3. Integration & Testing:**

* Ensure the frontend properly sends data to the backend.
* Handle errors gracefully (e.g., invalid file formats, missing files).
* Test end-to-end functionality.

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

* Not applicable, but logic for file handling should be implemented efficiently to handle multiple conversions.

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

* express (for backend server)
* multer (for handling file uploads)
* pdf2docx or similar (for converting PDF to Word)
* lit (for WebComponent development)

**Acceptance Criteria:**

* The Express.js backend should successfully process uploaded PDFs and return a downloadable Word document.
* The WebComponent-based frontend should have a responsive form that allows users to upload files.
* Proper error handling should be in place for invalid inputs.
* The application should work seamlessly across modern web browsers.

**Submission Guidelines:**

1. Fork the provided GitHub repository.
2. Create a folder named pdf-to-word-<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 handles requests, processes the PDF file, and seamlessly integrates with the WebComponent-based frontend.**