DevifyX

Node.js Job Assignment

Assignment Title: PDF Merge/Split API

Assignment Description: Handle multi-part documents

Assignment Deadline: 7 Days

Objective

Develop a robust backend API using Node.js that enables users to merge and split PDF documents. The solution should be strictly backend-only; no frontend implementation is required. The API must efficiently handle multi-part PDF documents, support file uploads, and provide clear responses.

Core Features

1. **PDF Upload:** Accept PDF files via multi-part/form-data POST requests.
2. **Merge PDFs:** Merge two or more uploaded PDF files into a single document.
3. **Split PDF:** Split a single PDF into multiple documents based on specified page ranges.
4. **Download Result:** Provide endpoints to download merged or split PDF files.
5. **Error Handling:** Implement comprehensive error handling for invalid files, unsupported formats, and processing errors.
6. **API Documentation:** Provide clear API documentation (e.g., Swagger or Markdown).
7. **Logging:** Log all API requests and responses for debugging and auditing purposes.
8. **File Storage:** Store uploaded and processed PDFs temporarily and ensure timely cleanup.

Bonus Features

- Password-protect output PDFs.
- Support for merging password-protected PDFs (if passwords are provided).
- Allow splitting PDFs by bookmarks or detected chapters.
- Provide a status endpoint for long-running operations.

Technical Requirements

- Use **Node.js** (v14 or above) and **Express.js** for the API.
- Utilize reliable PDF processing libraries (e.g., `pdf-lib`, `pdfjs`, or similar).
- Ensure the API is RESTful and stateless.
- Write clean, modular, and well-documented code.
- Implement input validation and security best practices.
- Include a README with setup and usage instructions.
- Use Git for version control.
- Optional: Use Docker for containerization.

Deliverables

- Complete Node.js project source code.
- API documentation (Swagger/OpenAPI or Markdown).
- Sample requests and responses for each endpoint.
- Instructions for setup and running the API locally.
- (Optional) Dockerfile and docker-compose configuration.

Use of AI Tools

You are **permitted and encouraged** to use AI-based coding tools such as **GitHub Copilot**, **ChatGPT**, or similar platforms to assist with code generation, debugging, and documentation. However, the final submission should reflect your own understanding and structure.

Submission

- Upload your code to a public or private Git repository (e.g., GitHub, GitLab).
- Fill out the assignment submission form:
  `https://forms.gle/LAvLWFmHRLXswwsx5`
- Include the repository link and any relevant notes or credentials required for review.

Evaluation Criteria

- **Correctness:** All core features are implemented and work as specified.
- **Code Quality:** Code is well-structured, readable, and maintainable.
- **API Design:** Endpoints are RESTful, intuitive, and well-documented.
- **Error Handling:** Proper error responses and edge case handling.
- **Security:** Input validation and secure file handling.
- **Documentation:** Complete and clear documentation for setup and usage.
- **Bonus Features:** Implementation of any bonus features will be considered.
- **Timely Submission:** Assignment is submitted within the deadline.

Click here to read our Terms and Conditions