Notepad Project Documentation

This document provides a comprehensive overview of the Notepad project, covering backend Flask API routes, data models, frontend JavaScript, and usage instructions.

1. Flask API Endpoints

- POST /api/v1/auth/register: Register a new user.
- POST /api/v1/auth/login: Authenticate user and create session.
- POST /api/v1/auth/logout: Logout user and clear session.
- POST /api/v1/notes: Create a new note (title, content, user_id).
- GET /api/v1/notes/<user_id>: Get all notes for a user.

Each endpoint expects JSON requests where applicable and returns JSON responses.

Errors are returned with appropriate HTTP status codes and error messages.

2. Data Models and DTOs

- User: id, username, email, phone, hashed password, createdAt, updatedAt
- Note: id, title, content (HTML formatted), user_id, createdAt, updatedAt
- DTOs: UserRegisterRequest, UserLoginRequest, NoteCreateRequest with validation via Pydantic

Password hashing is done using werkzeug.security for secure storage.

3. Frontend Functionality

- Pages: landing, register, login, dashboard (all served by Flask templates)
- Dashboard JS handles:

  * User authentication by reading user_id from localStorage

  * Loading user notes with fetch from backend API

  * Creating, saving, and 'Save As' for notes with rich text editing (bold, italic, font size, font family, color)

  * Export notes to PDF using jsPDF library

  * Logout clearing session and localStorage

Users can format notes via buttons and dropdowns that use document.execCommand.

4. Postman Testing Instructions

- Import Postman collection (if available) or manually test endpoints by setting Content-Type: application/json
- For register/login, send JSON body with required fields
- Save the returned user_id to localStorage for frontend interaction
- Test notes creation and retrieval using user_id
- Verify correct HTTP status codes and error handling

Example Register Request Body:

```
{
  "username": "testuser",
  "email": "test@example.com",
  "phone": "1234567890",
  "password": "password123",
  "confirm_password": "password123"
}
```

5. Running the Application

- Set up a .env file with MONGO_URI and SECRET_KEY
- Install Python dependencies (Flask, flask_pymongo, pydantic, python-dotenv, werkzeug, fpdf)
- Run the Flask app using `python app.py`
- Access app on http://127.0.0.1:5000/
- Use frontend forms to register, login, and manage notes