🧪 Technical Test: Task Management REST API

💡 Objective:

Design and implement a **Task Management System API** using **Flask** and **PostgreSQL**. The API should allow users to manage projects and tasks with support for users, task dependencies, and status tracking. The focus is on RESTful design, database modeling, and handling logical constraints.

📌 Functional Requirements (Endpoints to implement):

1. **User Endpoints**
   - Create user
   - List users
   - Get user by ID
2. **Project Endpoints**
   - Create project
   - List projects
   - Get project by ID
   - List all tasks under a project
3. **Task Endpoints**
   - Create task under a project
   - Get task by ID
   - Update task status (only if all dependencies are completed)
   - List tasks assigned to a user
   - List tasks with a specific status

🔍 Logical Constraints & Checks:

- A task **cannot be marked as `completed`** unless **all its dependencies are already completed**.
- A user **cannot be deleted** if they are assigned to any pending or in-progress task.
- Circular task dependencies must be **detected and prevented** when creating or updating tasks.

💾 Technical Requirements:

- Use **Flask** (no Flask-RESTful or DRF-like libraries).
- Use **PostgreSQL** as the database (use SQLAlchemy as ORM).
- Provide proper **error handling** and HTTP status codes.

- Include basic **input validation** (e.g., for email format, status values).
- Provide a **README** file with instructions to run the application and test the API (via Postman or curl).
- Include a **sample database seed** script with initial users, projects, and tasks.

✅ Bonus:

- Add authentication using a simple token-based system (JWT or API key).
- Add support for pagination in list endpoints.
- Write basic unit tests for task dependency logic.

📦 Submission:

- A GitHub repository or zip file.
- README with setup steps.
- Postman collection (optional but preferred).