Objective:

Task Management Application

features:

1. **Task Creation**: Users can create tasks with a title, description, and due date.
2. **Task Listing**: Users can view a list of tasks, showing titles, due dates, and completion status.
3. **Task Completion**: Users can mark tasks as completed.
4. **Task Filtering**: Users can filter tasks based on due dates and completion status.
5. **Database Integration**: Store task data in an SQLite database.
6. **API Development**: Create a FastAPI web service to interact with the task data.

Requirements:

1. Use Python 3.x for your project.
2. Utilise the following Python packages/modules:
    a. FastAPI for building the API.
    b. SQLAlchemy for database interaction.
    c. SQLite for database storage.
    d. datetime for handling due dates.
    e. Other packages as needed for API development and database management.
3. Create a modular project structure with separate files for routes, database models, and utility functions.
4. Implement validation to ensure data integrity, e.g., ensure due dates are in the future.
5. Allow users to interact with the application through a web-based user interface (HTML/CSS/JavaScript) and through API endpoints.
6. Host the application locally using uvicorn for the FastAPI server.
7. Provide proper documentation for your project, including code comments and a README.md file.
8. Upload the project to a git repository

Deliverables:

1. A Python project where the complete source code uploaded to a git repository
2. A SQLite database file with task data.
3. A FastAPI web service that provides CRUD (Create, Read, Update, Delete) operations for tasks.
4. A web-based user interface for managing tasks.
5. Documentation on how to set up and run the application.
6. A brief report on the design choices and challenges faced during the project.

Optional Enhancements (for an additional challenge):

1. User Authentication: Implement user authentication to allow multiple users to manage their tasks.
2. Deployment: Deploy the application on a cloud platform or web server for online access.
3. Task Reminders: Implement automatic email or notification reminders for upcoming tasks.
4. Should be able import tasks from an excel file

This assignment is designed to provide practical experience in building a complete Python application that covers web development, database interaction, and API creation using widely-used Python packages. The learner will be able to work on a real-world project and gain valuable skills.

Suggestions for fast learning

1. Learn Python (5 days) - https://www.geeksforgeeks.org/python-programming-language/learn-python-tutorial/
2. Write a program having following functions (Next 5 days)
    a. **Add task** - Create a function to add a task. The task should be added to a json file which is there in a folder. If we add a second task, it should add the task to the same json file.
    b. **List Task** - Write a function to list all tasks from the json file
    c. **Mark Task Complete** - Write a function to mark a task complete, given the task id as an input
    d. **Delete task** - Write a function to delete a task from the database (json file), given the task id

3. Next 5 days - Write another program to store data in a database rather than a JSON store. Replace the json file with a database (Can choose H2O, SQLite or Postgresql). Do the
4. Next 5 days - Convert the above program to FastAPI services

Hints

Create a Task class having the following attributes and object of this class should be used for transactions

- Id: int
- Name: str
- Description: str
- Due_date: DateTime

The json file should like below

```
{
  "tasks": [
    {
      "id": 1,
      "name": "Task 1",
      "description": "Complete project report",
      "due_date": "2023-11-15T10:00:00"
    },
    {
      "id": 2,
      "name": "Task 2",
      "description": "Prepare presentation",
      "due_date": "2023-11-20T15:30:00"
    },
    {
      "id": 3,
      "name": "Task 3",
      "description": "Review budget proposal",
      "due_date": "2023-11-10T14:15:00"
    }
  ]
}
```