**Podział na etapy i zadania krok po kroku**

**1. Konfiguracja środowiska i podstawy SQLite**

✅ **Zadania:**  
🔹 Zainstaluj Python i SQLite (jeśli jeszcze nie masz).  
🔹 Zainstaluj wymagane biblioteki (sqlite3, tkinter lub flask).  
🔹 Utwórz plik projektu (todo.py).  
🔹 Przetestuj podstawowe zapytania SQLite w terminalu (sqlite3 CLI).

**🔍 Czego się nauczysz?**  
✔️ Podstawowe operacje w SQLite (CREATE TABLE, INSERT, SELECT).  
✔️ Tworzenie i zarządzanie bazą SQLite w Pythonie.

**2. Tworzenie bazy danych i połączenie z Pythonem**

✅ **Zadania:**  
🔹 Utwórz bazę danych tasks.db.  
🔹 Stwórz tabelę tasks (id, title, description, status).  
🔹 Napisz funkcję do łączenia się z bazą SQLite w Pythonie.  
🔹 Dodaj obsługę błędów dla połączenia SQL (try-except).

**🔍 Czego się nauczysz?**  
✔️ Tworzenie tabel i operacje CRUD.  
✔️ Obsługa błędów SQL w Pythonie.

**3. Implementacja operacji CRUD w Pythonie**

✅ **Zadania:**  
🔹 Stwórz funkcję do dodawania nowych zadań (INSERT INTO tasks).  
🔹 Napisz funkcję do pobierania wszystkich zadań (SELECT \* FROM tasks).  
🔹 Zaimplementuj usuwanie zadań (DELETE FROM tasks WHERE id=?).  
🔹 Dodaj funkcję do edycji zadań (UPDATE tasks SET title=?, description=?, status=? WHERE id=?).

**🔍 Czego się nauczysz?**  
✔️ Podstawowe operacje CRUD w SQL i Pythonie.  
✔️ Dynamiczne przekazywanie danych do SQL w Pythonie.

**4. Stworzenie interfejsu użytkownika (Tkinter/Flask)**

✅ **Zadania:**  
🔹 Jeśli używasz Tkinter:  
    ▫️ Stwórz okno główne aplikacji.  
    ▫️ Dodaj pola do wpisywania nazwy i opisu zadania.  
    ▫️ Dodaj przyciski "Dodaj", "Usuń", "Edytuj", "Zakończ".  
🔹 Jeśli używasz Flask:  
    ▫️ Stwórz pliki app.py i templates/index.html.  
    ▫️ Utwórz formularz HTML do dodawania i edycji zadań.  
    ▫️ Obsłuż przesyłanie danych (Flask.request).

**🔍 Czego się nauczysz?**  
✔️ Tworzenie GUI w Pythonie (Tkinter) lub aplikacji webowej (Flask).  
✔️ Integracja interfejsu z bazą danych.

**5. Obsługa statusu zadań (ukończone/nieukończone)**

✅ **Zadania:**  
🔹 Dodaj nową kolumnę status do tabeli (ALTER TABLE).  
🔹 Zaimplementuj funkcję do zmiany statusu zadania.  
🔹 Zaktualizuj interfejs użytkownika – dodaj checkbox lub przycisk "Zakończ".

**🔍 Czego się nauczysz?**  
✔️ Modyfikacja schematu bazy danych.  
✔️ Dynamiczne aktualizowanie interfejsu.

**6. Wyświetlanie i filtrowanie zadań**

✅ **Zadania:**  
🔹 Dodaj funkcję do sortowania zadań według statusu (ukończone/nieukończone).  
🔹 Zaimplementuj wyszukiwanie po tytule/opisie zadania (SELECT \* FROM tasks WHERE title LIKE ?).  
🔹 Dodaj możliwość filtrowania po dacie dodania.

**🔍 Czego się nauczysz?**  
✔️ Tworzenie dynamicznych zapytań SQL.  
✔️ Obsługa filtrów w interfejsie użytkownika.

**7. Zapisywanie dat dodania i zakończenia zadania**

✅ **Zadania:**  
🔹 Dodaj kolumnę created\_at (DATETIME DEFAULT CURRENT\_TIMESTAMP).  
🔹 Dodaj kolumnę completed\_at, która zapisuje czas zakończenia zadania.  
🔹 Zaimplementuj automatyczne uzupełnianie daty zakończenia.

**🔍 Czego się nauczysz?**  
✔️ Praca z datami w SQL (DATETIME).  
✔️ Automatyczne aktualizacje rekordów.

**8. Logowanie operacji do pliku**

✅ **Zadania:**  
🔹 Stwórz plik log.txt, do którego będą zapisywane operacje użytkownika.  
🔹 Dodaj funkcję do logowania każdej operacji (np. dodanie zadania, zmiana statusu).

**🔍 Czego się nauczysz?**  
✔️ Praca z plikami w Pythonie (open, write).  
✔️ Rejestrowanie operacji użytkownika.

**9. Tworzenie kopii zapasowej bazy danych**

✅ **Zadania:**  
🔹 Napisz skrypt do automatycznego tworzenia kopii bazy (cp tasks.db backup\_tasks.db).  
🔹 Zaplanuj automatyczne backupy przy starcie aplikacji.

**🔍 Czego się nauczysz?**  
✔️ Tworzenie kopii baz danych SQLite.  
✔️ Automatyzacja zadań w Pythonie.

**10. Rozbudowa interfejsu użytkownika**

✅ **Zadania:**  
🔹 Dodaj kolory/statusy dla różnych typów zadań (np. ważne zadania – czerwony).  
🔹 Dodaj obsługę ikon i lepszy wygląd GUI.  
🔹 Dodaj powiadomienia (messagebox.showinfo w Tkinter lub flash w Flask).

**🔍 Czego się nauczysz?**  
✔️ Personalizacja UI.  
✔️ Praca z komunikatami użytkownika.

**11. Tworzenie wersji webowej i API (zaawansowane)**

✅ **Zadania:**  
🔹 Utwórz API REST (Flask + Flask-RESTful).  
🔹 Stwórz frontend w HTML + CSS + JavaScript.  
🔹 Połącz backend z frontendem.

**🔍 Czego się nauczysz?**  
✔️ Tworzenie API w Flask.  
✔️ Tworzenie interfejsu webowego.

**Podsumowanie**

Po ukończeniu wszystkich kroków będziesz umiał:  
✅ Pracować z SQLite i operacjami CRUD.  
✅ Tworzyć interfejs użytkownika (Tkinter/Flask).  
✅ Obsługiwać statusy, filtry i wyszukiwanie zadań.  
✅ Zapis danych do logów i backup bazy.  
✅ Tworzyć API i aplikację webową.

To kompletny plan nauki – krok po kroku! 🚀 Który etap chcesz zacząć jako pierwszy? 😃