C++17 Console Projects with SQLite and Design Patterns

Contact Manager

Purpose: Add, update, delete, list people with names and mobile numbers.

SQLite Use: Store contact records.

Design Patterns:

- Singleton (Database connection)
- DAO (Person data management)
- Command (user operations)
- Factory (command creation)

Todo List App

Purpose: Create, update, delete, complete, and list todo items.

SQLite Use: Store tasks and their statuses.

Design Patterns:

- Singleton (DB access)
- DAO (Task operations)
- Command (each task action)
- Observer (notify UI/log when task changes)

Inventory Management System

Purpose: Manage products, stock levels, and prices.

SQLite Use: Persist inventory and transactions.

Design Patterns:

- Singleton (DB)
- DAO (ProductDAO, TransactionDAO)
- Strategy (pricing strategies)
- MVC (UI/input, model, and DB logic)

Student Grade Tracker

Purpose: Record student info and grades; calculate GPA.

SQLite Use: Store students, courses, and grades.

Design Patterns:

- Singleton (DB)
- DAO (StudentDAO, GradeDAO)
- Composite (nested subjects or assignments)
- Strategy (grading methods)

Microservice Simulator (Console)

Purpose: Simulate a small set of services (e.g. auth, data, logging).

SQLite Use: Persist logs or user data.

Design Patterns:

- Singleton (database/logging service)
- Mediator (service communication)
- Observer (monitor service events)
- Proxy (simulate network latency)

Expense Tracker

Purpose: Add, categorize, and track expenses.

SQLite Use: Store transactions and categories.

Design Patterns:

- Singleton (DB)
- DAO (TransactionDAO)
- Strategy (category filters)
- Command (add/delete/list expenses)

Library Management System

Purpose: Track books, borrowers, due dates.

SQLite Use: Store books, members, loans.

Design Patterns:

- Singleton (DB)
- DAO (BookDAO, MemberDAO)
- Factory (create users/books)
- Observer (alerts for due/overdue books)