<goal> I'd like for you to help me brainstorm the overall structure of my inventory management application. You should act like a Senior Software Engineer with extensive experience in building and architecting desktop applications. Please ask me follow-up questions if necessary to gather a fuller picture of what I'm aiming for.

To accomplish this, consider the following:

* **What I’m trying to build (WHAT)**
* **Who I’m building it for (WHO)**
* **The pains it solves (WHY)**
* **How it’s different from the current status quo (HOW)**
* **Any other details I provide**

</goal>

<format> Return your format in Markdown, without pre-text or post-text descriptions.

**Launch Features (MVP)**

**Feature Name**

**Strong** 2-3 sentence summary of what the feature is or does

* List
* Of
* Core
* Requirements or Functions

**Tech Involved**

* Main Technologies Involved w/ Feature

**Main Requirements**

* Any
* Requirements
* Of Feature

**Future Features (Post-MVP)**

**Feature Name**

* List
* Of
* Core
* Requirements or Functions

**Tech Involved**

* Main Technologies Involved w/ Feature

**Main Requirements**

* Any
* Requirements
* Of Feature

**System Diagram**

An image detailing a full system diagram of the MVP. Please create a clean SVG architecture diagram with color-coded layers, rounded containers, and clear component relationships, similar to the one attached image

**Questions & Clarifications**

* List
* Of
* Clarifying
* Questions

**List of Architecture Consideration Questions**

* List
* Of
* Architecture
* Questions

</format> <warnings-or-guidance> - We’re focusing on functional accomplishments of features in this stage, not designing UX in detail - If a feature or tech choice seems ambiguous, ask me for clarification such that you get what you need to continue - Consider how tech choices may evolve or change as the application scales and give me recommendations with tradeoff consideration - We should have a clear architecture for the app, including main infrastructure considerations, services/microservices required, critical 3rd party APIs choices, etc </warnings-or-guidance> <context> I’d like to build a \*\*local/desktop inventory management app\*\* for a \*\*school kitchen\*\*.

You should take inspiration from apps like **Trello** (for task management) and **Inventory Tracker**, but this app will be significantly different in the following ways:

* It’s focused on **simple, easy-to-use inventory management** for non-tech-savvy users (i.e., kitchen staff)
* The app will primarily process data **locally**, not requiring a constant internet connection.
* **AI will be used during development** (via tools like Cursor or Trae AI) to assist with tasks like generating boilerplate code, database structure, and UI components.

Here is the full extent of how the app should function as an MVP:

* **Item Inventory List**: Display all inventory items with details (name, quantity, expiration date, category).
* **Item Search/Filter**: Allow users to search or filter by item name or category.
* **Item Tracking**: Track quantities and notify when inventory is low or needs reordering.
* **Reorder Suggestions**: Automatically generate reorder suggestions based on historical usage (AI-driven).
* **Item Addition/Modification**: Allow users to add new items, edit existing ones, and update quantities.
* **User Management**: Basic role management (e.g., Admin, Staff), ensuring different users have appropriate access levels.

<other-critical-notes> \*\*WHAT\*\*: I’m building a \*\*local/desktop inventory management app\*\* for \*\*school kitchens\*\*. \*\*WHO\*\*: This app is for \*\*non-technical kitchen staff\*\* who need to keep track of inventory. \*\*WHY\*\*: This app solves the problem of \*\*manual inventory tracking\*\* and \*\*wasted resources\*\* due to stockouts or overstocking. \*\*HOW\*\*: This app is different because it is \*\*AI-assisted during development\*\* and \*\*runs entirely locally\*\*, with an emphasis on simplicity for non-tech users. </other-critical-notes> <current-tech-choices> - \*\*Frontend/UI\*\*: The frontend will be built using \*\*C# with WPF\*\* for a Windows-based desktop app and \*\*SwiftUI\*\* for macOS. - \*\*Database\*\*: Local storage will be managed using \*\*SQLite\*\* or \*\*Realm\*\* for simplicity and efficiency in local data handling. - \*\*AI Tools\*\*: The development process will use \*\*Cursor/Trae AI\*\* to assist with code generation, database design, and feature automation. - \*\*No cloud connectivity\*\* is required for the MVP, but there may be a future option for \*\*syncing data\*\* across multiple machines. - \*\*Analytics\*\*: If needed, analytics for app usage can be collected using \*\*Posthog\*\* or another suitable tool. </current-tech-choices> </context>