# Technical Decisions

### State Management: Bloc

We chose **Bloc** (flutter_bloc) for state management due to its clean separation between logic and UI. It helps manage complex UI states such as:

- Cart updates
- Product/category filtering
- Async state transitions (loading, success, failure)

### Local Persistence: SharedPreferences

Since the app has no backend, we used SharedPreferences to persist cart items locally. This allows the cart to survive app restarts without requiring a database or API.

### Project Structure

A **feature-first structure** was adopted:

- core/bloc — Business logic
- core/models — Data classes (Product, CartItem)
- core/services — Logic for local data handling
- ui/screens — Screen-specific widgets
- ui/widgets — Reusable UI components (product grid, category grid)

### Modularity

- Each Bloc is decoupled and testable
- Widgets are stateless wherever possible
- Reusable widgets accept input parameters (e.g., ProductGrid uses customizable button color)

# Challenges and Solutions

### 1. Challenge: Navigating tabs with Bloc updates

- **Problem:** Switching between Explore and Home sometimes caused UI state conflicts.
- **Solution:** Bloc events like LoadAllProducts and LoadCategoriesOnly were carefully dispatched based on tab index, ensuring fresh data every time.

### 2. Challenge: Product search in Home tab

- **Problem:** Needed real-time search functionality without affecting the original product list.

- **Solution:** Managed a filtered list within ProductState and implemented .where filtering logic triggered via search text input.

### 3. Challenge: Persistent cart with updates

- **Problem:** Keeping cart items consistent across app restarts, updates, and deletions.
- **Solution:** Wrote custom logic in CartService to serialize and deserialize Product and CartItem using SharedPreferences.

### 4. Challenge: Back navigation from category

- **Problem:** The back button in the Explore tab (category view) wasn't returning correctly to the category grid.
- **Solution:** Carefully updated the _selectedIndex and dispatched a LoadCategoriesOnly event with a slight delay to re-render the category grid correctly.

## Time Breakdown & Learnings

| Task | Time Spent |
|---|---|
| Project setup & dependency config | 1 hour |
| Product & cart models | 30 minutes |
| Bloc setup (Product & Cart) | 2.5 hours |
| UI for Home tab | 1.5 hours |
| Explore tab + Category view | 2 hours |
| Cart screen (with updates) | 2 hours |
| Persistent storage (SharedPrefs) | 1 hour |
| Final UI polish + search | 1.5 hours |
| Bug fixes & testing | 1 hour |
| **Total** | **~13 hours** |

### Key Learnings:

- Gained deeper understanding of Bloc event/state lifecycles
- Learned to manage state transitions with delayed rendering (Future.delayed)
- Reinforced concepts of local storage and serialization
- Improved widget modularization and navigation handling