Task - AI-Powered Checkout & Upsell (≈ 1 working day)

Problem

Build a **tenant-aware e-commerce mini-workflow** that

1. offers catalogue, cart and checkout APIs,
2. calls an LLM at checkout to suggest up-sell items,
3. persists completed orders, and
4. runs both locally via Docker Compose and on any free-tier cloud.

What to Build Component	**Core Responsibilities**	**Minimum Scope**
Catalog API	List products, prices and stock per tenant.	Read from in-memory list or JSON file; endpoints to list all products and fetch one by id.
Cart & Checkout API	Create/update cart; on checkout reserve stock and create order.	REST routes for cart creation/update and a checkout route that validates stock.
AI Upsell Service	When UPSELL_ENABLED=true, request ≤ 3 complementary products for the cart and add their explanations.	Decouple into its own module; may call OpenAI or a local model; must log prompt and response.
Order Store	Persist final orders.	In-memory map or SQLite file.
Public Storefront (SPA)	One-page UI: product list, “Add to cart”, Checkout button, upsell display, order confirmation.	Plain React, HTMX or vanilla JS.

1. Problem-Solving Approach:

How do you analyze the task requirements?

How do you plan and structure your solution before you start writing the main code?

What technical trade-offs do you make during development, and what is the logic behind them?

2. Architecture and Code Quality:

o Modularity, cleanliness, and clear structure of the code.

o Adherence to best practices and conventions for your chosen language and framework.

Is your solution easy for another developer to understand and build upon?

3. Efficiency and Resourcefulness in Using AI Assistants:

o High score: You demonstrate skillful use of AI to accelerate routine tasks (generating boilerplate, tests, documentation), to explore alternative solutions, for refactoring, or for debugging. You show critical thinking towards AI suggestions by verifying, adapting, and improving them.

o Low score: You simply copy large blocks of code without a clear understanding, rely on AI to do all the work for you, and struggle to proceed when the AI assistant provides an incorrect or incomplete answer.

4. Adherence to Requirements and Success Criteria:

Does the solution meet all functional and non-functional requirements described in the task (e.g., tenant isolation, real-time updates, security rules, RBAC)?

5. Testing and Quality:

Presence, adequacy, and coverage of automated tests (unit, integration), where applicable and required by the task.

6. Completeness of the Submitted Material:

o Quality and clarity of the README.md file.

o Ease of running and testing the project.