Native Orchestration Layer Task

Goal

1. Build a **simple native orchestration layer** using **TypeScript**.
2. No orchestration frameworks or tools.
3. The **orchestrator must fully control execution**.
4. LLMs (if used) act **only as workers**, never as controllers.

Core Principle

Orchestrator = control
LLM = function call

Rules

1. TypeScript only (Node.js runtime is fine)
2. No LangGraph, CrewAI, Temporal, Step Functions, etc.
3. Focus on **control flow and state**, not tools

What to Build

Create a small **orchestrator** that controls how tasks run and how state moves between them.

Core Requirements

1. **Shared Context**
   A single object that holds input, intermediate results, and errors.
2. **Task / Step**
   - Receives the context
   - Does one unit of work
   - Returns which task should run next
3. **Orchestrator**
   - Runs one task at a time
   - Updates the shared context
   - Decides the next task
   - Stops execution cleanly

4. **Dynamic Flow**
   Tasks must be able to choose the next task based on context.

5. **Error Handling**
   - Orchestrator catches errors
   - Routes execution to a final or error step

Evaluation Focus

- Orchestration thinking
- Explicit control flow

Plus

You may build the example using any kind of project or domain. A **code-generator or developer-tool–style project** is a plus, but not required.