AI Solution Engineer Test

Type: “Autonomy”, Updated at: October 2025

1. Goal

Design and implement a **minimal “AI CEO”** that can simulate running a small company for one workday. Your AI CEO should:

- Know about **employees**, **clients**, and **events**.
- **Receive inputs** (messages or changes).
- **Plan and make decisions** (who should do what, when, and why).
- **Take actions** (send emails, assign tasks, update CRM, etc.).
- **Explain its reasoning** and summarize what happened that day.

You can implement this **with any technology** —
→ custom code (Python, TypeScript, etc.),
→ orchestration tools (n8n, LangFlow, Flowise, Zapier, etc.),
→ or a **hybrid** setup (workflow tool + custom logic or APIs).

You are designing a **decision-making system**, not a polished app. We care about **autonomy, reasoning clarity, and clean design**, not the specific stack.

2. Scenario

You’re simulating one workday of a small software agency.

Throughout the day, the AI CEO faces three events:

1. A key client asks to **accelerate** a project.
2. A **new lead** arrives asking for a proposal.
3. One **employee becomes unavailable** mid-day.

Your AI CEO must autonomously:

- React to each event.
- Reassign or reschedule work if needed.
- Communicate appropriately (email/message/task).
- Keep decisions within resource/budget constraints.
- Produce an **end-of-day summary** explaining what happened and why.

3. Requirements

Core Functionality

Your system should be able to:

- **Run a one-day simulation** (either by running a script or triggering a workflow).
- **Maintain state** for employees, clients, and events.
- Follow an **"observe → plan → act → log"** cycle at least once.
- Include some notion of **constraints** (time, cost, or availability).
- **Log decisions** and reasoning behind them.

Actions (use any form)

Implement at least **three simulated actions**, such as:

- Sending an **email/message** (real or simulated).
- Creating a **task or calendar item**.
- Updating a **client/project record**.
- Generating a short **plan or report**.

You can implement these via:

- Code (functions, APIs, etc.)
- Workflow nodes (n8n flows, LangFlow blocks, etc.)
- Third-party connectors (Notion, Slack, Airtable, etc.)

4. Deliverables

At the end of a simulated day, your system should produce:

- A **summary or report** (text or markdown) describing:
    - What happened
    - What actions were taken
    - Why decisions were made
    - Any risks or follow-ups for tomorrow
- Logs or data that show actions were taken (e.g. JSON, console output, workflow logs).

You can submit:

- A **GitHub repo** (if code-based).
- Or a **shared workspace/export** (if using n8n, LangFlow, Zapier, etc.).