Technical Assessment — Solution Design (CX AI Workflow)

🧩 **Goal**: This assessment is designed to evaluate solution design skills.

- Understand business needs and constraints
- Design an optimal workflow end-to-end
- Apply modern AI capabilities thoughtfully (quality, safety, cost, latency)

Background

Reap's CX team handles a high volume of customer requests and back-office operations. Many requests require:

- Gathering context from multiple sources
- Taking multi-step actions (some long-running)
- Communicating progress clearly
- Handling partial failures without confusing the user

We want to assess how you would design an AI-assisted workflow that is both effective for the business and safe for customers.

Business Case

Design an AI-assisted **CX Operations Assistant** that helps agents fulfill customer requests that involve **bulk operations**.

Example request

> “Please update the spending limits for 50 cards for the Marketing team to SGD 2,000, and notify the cardholders once done.”

What “good” looks like

- The workflow is **business-correct** (right policies, right approvals, right messaging).
- The system stays **responsive** during long-running operations.
- The user (CX agent or end customer) always knows:
  - What is happening
  - What has completed
  - What failed and why
  - What options exist (retry, cancel, escalate)

Constraints and assumptions

You may assume:

- Card operations are done via internal APIs (you can mock them).
- Some operations are slow and may fail for a subset of items.
- The system must be safe by default, and avoid unintended bulk changes.

Your Task

Design (and optionally implement) a solution that:

1. **Clarifies requirements** and defines what the system should optimize for.
2. Proposes the **optimal workflow** for handling the business case.
3. Designs a **system architecture** to support that workflow.
4. Demonstrates how you would apply **AI capabilities** responsibly.

You have full freedom over architecture and technology choices.

Scope

In Scope

- A clear end-to-end workflow, including:
  - Input understanding (intent + parameters)
  - Confirmation/approval steps (if needed)
  - Bulk execution with progress reporting
  - Partial failure handling and recovery paths
  - Final summary and notification behavior
- A minimal prototype
- For each major step in your proposed workflow, specify whether it is **"human only"**, **"assist"**, or **"fully auto resolved"**, and list 3–5 stop conditions that force escalation to a human.

Out of Scope

- No authentication or user management
- No production-level UI polish
- No real card API integration required (mock is fine)

Key Questions to Address (Solution Design)

Workflow design

- How does the system translate a messy request into a safe, executable plan?
- Where do humans confirm details, and what information is shown at confirmation time?
- What steps are synchronous vs asynchronous?

Long-running operations and UX

- How is progress communicated (streaming updates, receipts, statuses)?
- How do users interact while a job is running?
- How do you prevent accidental double-submits?

Reliability

- How do you handle partial failures (some cards succeed, some fail)?
- What is your retry strategy?
- How do you make the operation idempotent?
- How do you support cancellation?

AI usage (optional but encouraged)

If you use an LLM, describe:

- How you constrain outputs (schemas, validation, guardrails)
- How you evaluate quality and reduce hallucinations
- How you manage cost and latency

Technical Constraints

- **Language:** Your choice (Python, TypeScript, etc.)
- **LLM usage:** Your choice

Deliverables

Required

1. **Solution design write-up** (Notion page / PDF / Google Doc):
    - Problem framing and requirements
    - Proposed workflow (step-by-step)
    - Architecture diagram + explanation
    - Key design decisions and trade-offs
    - Failure modes and how you handle them
    - Observability plan (logs/metrics/traces you'd want)
    - Security and safety considerations (at least at a high level)

2. **Demo**
    - Option A: a short screen recording showing the UX
    - Option B: a functional prototype (that connects to models on OpenRouter, or you will need to supply a key to the providers of your choice)

Nice to Have

- Prototype (chat UI + backend) demonstrating:
    - Progress updates during bulk operations
    - Partial failure reporting
    - Retry/cancel flows
- Short code walkthrough video
- Hosted deployment

Evaluation Criteria

Criteria	Weight
Solution design — business understanding, workflow optimality, clarity of assumptions and trade-offs	Very High
AI application — appropriate use of AI, guardrails, evaluation approach, cost/latency awareness	High
UX for long-running work — progress, receipts, partial failures, user control (retry/cancel)	High
Reliability & correctness — idempotency, retries, failure handling, observability	Medium
Code quality (if implemented) — structure, readability, testability	Medium

---

Interview Discussion Guide (what we'll challenge)

Be prepared to discuss:

- Alternatives you considered and why you rejected them
- Where your design is most fragile

- How you would validate correctness and prevent unsafe actions
- How you would evolve the design if volume increased 10×

Time Limit

3 days from receiving this assessment.

Submission

Please share your proposed solution in any format (e.g. PDF, Notion page) to ivo@reap.global & chris.chan@reap.global