“Agent-as-Coder” Challenge

Goal

Develop coding agent which write custom parsers for Bank statement PDF.

Fork `github.com/apurv-korefi/ai-agent-challenge` and add an `agent.py` that—when run via CLI—writes a new parser `custom_parsers/icic_parser.py`, and lets evaluators run the `agent.py` on their own statement file for a new bank (e.g., SBI) without manual tweaks.

Context and references

What’s an agent? In LLM land an agent is a loop that **plans → calls tools (code, file I/O, pytest, etc.) → observes results → refines itself**. It can keep short-term *memory* (conversation history, intermediate files) and run multiple correction cycles until the task is done.

A tiny open-source reference (≤ 100 LOC): **mini-swe-agent** – a 100-line Python agent that fixes GitHub issues or runs on the CLI (GitHub).
Quick reads on Anthropic-style agents & tool use:

1. Anthropic “Tool use with Claude” docs (Anthropic)
2. Anthropic Cookbook – patterns & notebooks (GitHub)
3. “Building Effective AI Agents” research note (Anthropic)

Core Tasks

#	What you deliver	Essentials
T1	**Design `agent.py`**	Use LangGraph *or* any lightweight SDK. Loop: plan → generate code → run tests → self-fix (≤3 attempts).
T2	**CLI**	`python agent.py --target icici` must: 1) read `data/icici/icic sample.pdf` & CSV; 2) agent writes parser `custom_parser/icici_parser.py`
T3	**Parser contract**	`parse(pdf_path) -> pd.DataFrame` matching the expected CSV schema.
T4	**Test**	Assert `parse()` output equals the provided CSV via `DataFrame.equals`.
T5	**README**	5-step run instructions + one-paragraph agent diagram.

Evaluation (100 pts)

Weight	Dimension
35 %	Agent **autonomy** (self-debug loops)
25 %	**Code quality** (typing, docs, clarity)
20 %	**Architecture** (clear graph / node design)
20 %	**Demo** ≤ 60 s showing fresh clone → `agent.py` → green `pytest`

Happy hacking—keep it simple, commit often.

Tips

Use following providers for free API credits
1- Google Gemini API
2- groq.com