SAM 2.0 AGI - One-Page Summary

What it is

SAM 2.0 is a hybrid Python/C multi-agent system with a web dashboard and slash-command interface. It combines a Python orchestrator with C-accelerated cores for meta-control and dual-system simulation.

Who it's for

Not found in repo.

What it does

- Runs a unified Python orchestrator, API server, and CLI via complete_sam_unified.py.
- Serves a web dashboard and terminal UI (default at http://localhost:5004).
- Provides slash-command controls for agents and tasks (e.g., /research, /code, /finance, /websearch).
- Uses C extensions for core subsystems: sam_sav_dual_system, sam_meta_controller_c, multi_agent_orchestrator_c, specialized_agents_c, and consciousness_*.
- Supports profile-based execution (full vs experimental) and loadable profiles in profiles/.
- Streams runtime events through JSONL logs and a live log panel (logs/sam_runtime.jsonl).
- Persists state snapshots per profile and reloads on start (sam_data//state.json).

How it works (repo evidence)

- Entry point: complete_sam_unified.py boots the system, API server, and UI.
- Core compute: C extensions implement the dual-system arena, meta-controller, agent orchestration, and consciousness modules.
- Interfaces: web dashboard and terminal UI; HTTP API endpoints such as /api/health, /api/agents, /api/command, /api/terminal/execute; SocketIO groupchat.
- Model providers: optional local Ollama or hosted APIs via OPENAI_API_KEY, ANTHROPIC_API_KEY, GOOGLE_API_KEY, GITHUB_TOKEN (when configured).
- Data flow: user/UI or API requests feed the Python orchestrator, which calls C cores; events are written to logs/sam_runtime.jsonl and state is saved under sam_data//.

How to run (minimal)

- pip install -r requirements.txt
- python setup.py build_ext --inplace
- Optional: install Ollama and pull a model (e.g., ollama pull codellama:latest).
- Start: ./run_sam.sh or python3 complete_sam_unified.py