

Scientific AI Backplane Architecture

Agentic AI System for Asynchronous HPC Job Submission

Stage 1:
Request submission

User / Researcher

Natural language requests
Natural language

AI Agent

agent.py — agent_demo.py — agent_apps.py
Function Calling • Parameter Op-
timization • Result Interpretation

OpenAI-compatible

gpt-oss:120b120B params(Remote containerized)

gpt-oss:20b20B params

Stage 2:
LLM reasoning
& tool selection

qwen3:32b32.8B params

GET /job_status

POST /submit_job

Stage 3:
Async dispatch
(prevents timeout)

FastAPI REST API main.py

POST /submit_job
POST /submit_app_job
GET /job_status/{id}

Queue job

Redis Message Broker Queue + Result Backend

Task dispatch

Key Features:

Prevents LLM timeouts
Async job processing
5 chemistry apps integrated
100% success rate
Function calling support
Auto parameter tuning
Scientific interpretation
RDKit SMILES support
Production-ready

Models tested:

- gpt-oss:20b
- qwen3:32b
- gpt-oss:120b

Technology Stack:

- FastAPI (REST API)
- Celery (Async tasks)
- Redis (Message broker)
- Pydantic (Validation)
- OpenAI SDK (LLM)
- RDKit (Chemistry)
- Python 3.x

Code Statistics:
3,500 lines total

Stage 4:
Queue
management

Celery Async Worker tasks.py

submit_slurm_job()
run_simulation()

Execution: 10s - 30min

QE Wrapper 340 lines

CP2K Wrapper 300 lines

GPAW Wrapper 280 lines

LAMMPS Wrapper 360 lines

GROMACS Wrapper 370 lines

wrappers - Unified interface: 2,050 lines of code

Quantum ESPRESSO Plane-wave DFT

CP2K Mixed basis DFT

GPAW Real-space DFT

LAMMPS Classical MD

GROMACS Biomolecular MD

APPS/ directory - Production installations