POC-1 — LiteLLM + SQLAlchemy Integration (Final Closeout Pack)

Repository: HX-Infrastructure-Ansible (https://github.com/hanax-ai/HX-Infrastructure-Ansible.git)
Folder to Save: `POC-1/`

---

1. POC Status

- ✅ **Complete** — All success criteria met.
- **Source:** Findings & Runbook outputs.

Key Evidence:
- LiteLLM Gateway running as a systemd service.
- PostgreSQL 17 backend with SQLAlchemy ORM integration.
- Requests and responses logged correctly to DB.
- Curl test requests return HTTP 200 and rows appear in `requests` + `responses` tables.
- <5ms DB logging overhead confirmed.

---

2. Files to Include in `POC-1/`

1. **FINDINGS.md** — Executive summary and technical findings【48†source】
2. **RUNBOOK.md** — Step-by-step setup, validation, troubleshooting【49†source】
3. **config.yaml** — LiteLLM Gateway configuration (secrets redacted)
4. **db_init.py** — SQLAlchemy schema + initialization/validation script【50†source】
5. **Evidence Bundle** (engineer to generate — see Section 5)

---

3. Replication Guide (Abbreviated)

Follow **RUNBOOK.md** in sequence:
1. Provision 2x Ubuntu 24.04 VMs (`192.168.10.18` , `192.168.10.19`).
2. Install PostgreSQL 17, configure `litellm_db` , user `litellm_user` , pg_hba for `192.168.10.18` .
3. Deploy LiteLLM Gateway + Python 3.12 venv, install `litellm[proxy] sqlalchemy psycopg2-binary` .
4. Place `config.yaml` in `/etc/litellm/` and `db_init.py` in `/opt/litellm/` .
5. Run `db_init.py` to create and validate schema.
6. Enable + start `litellm-gateway` systemd service.
7. Run curl test:

```bash
curl -sS -X POST http://192.168.10.18:4000/v1/chat/completions \
  -H 'Content-Type: application/json' \
  -H 'Authorization: Bearer TEST_KEY' \
  -d '{"model":"gpt-4o-mini","messages":[{"role":"user","content":"hello"}]}'
```

8. Verify DB rows in `requests` + `responses` using SQL queries from RUNBOOK.

---

4. Schema Summary (from `db_init.py`)

requests
- id (PK, int)
- created_at (datetime)
- request_id (string, unique, indexed)
- route, model (indexed)
- payload (JSON)
- status_code (int)
- user_id, api_key (strings)
- start_time, end_time (datetime)

responses
- id (PK, int)
- created_at (datetime)
- request_id_fk (FK → requests.id, CASCADE, indexed)
- latency_ms (int)
- content (JSON)
- prompt_tokens, completion_tokens, total_tokens (int)
- cost (string)
- response_model (string)

---

5. Evidence Bundle (Engineer Action)

Save outputs in `POC-1/evidence/`:
- `service_status.txt` — `systemctl status litellm-gateway | head -n 20`
- `gateway_db_connect.log` — `journalctl -u litellm-gateway | grep -i connection`
- `chat_call.json` — Output of curl test
- `requests_head.txt` — Top rows from `requests`
- `responses_head.txt` — Top rows from `responses`
- `join_check.txt` — Join query proving request→response link

---

6. Recommendations for Engineering

- **Lock Versions** — Document LiteLLM, SQLAlchemy, psycopg2, Postgres versions in Evidence Bundle.
- **Make Validation One Command** — Add `make validate` target to auto-run curl + SQL checks and dump artifacts into `POC-1/evidence/`.
- **Standardize Schema** — Treat `db_init.py` as single source of truth for table structure.
- **Bundle Outputs** — Always produce Evidence Bundle before sign-off.

---

7. Exit Criteria

- All 5 validation checks pass (service, DB connect, curl 200, DB rows, join query).

- Evidence Bundle present in `POC-1/evidence/`.
- All files committed to repo under `POC-1/`.

Instruction: Save this document and all supporting files into the `POC-1/` folder in the HX-Infrastructure-Ansible (https://github.com/hanax-ai/HX-Infrastructure-Ansible.git) repository.