Milla Brain Install Guide

Project Docs

December 2, 2025

Prerequisites

- OS: Ubuntu Linux (or similar). Works on macOS with analogous tools.
- Java (OpenJDK) and Clojure CLI (`clj`).
- SQLite available on the system.
- Ollama running locally at `http://localhost:11434` with a model name matching your config (e.g., `llama3.2`).

Install prerequisites on Ubuntu

```
sudo apt-get update
sudo apt-get install -y curl gnupg ca-certificates \
    openjdk-21-jdk sqlite3

# Install Clojure CLI
curl -O https://download.clojure.org/install/linux-install-1.11.1.1435.sh
chmod +x linux-install-1.11.1.1435.sh
sudo ./linux-install-1.11.1.1435.sh

# Install Ollama (from upstream; see https://ollama.com)
curl -fsSL https://ollama.com/install.sh | sh
```

Get the code

```
git clone <repo-url> milla-clj
cd milla-clj
```

Configure

Runtime configuration is read from YAML in this order: `MILLA_CONFIG` env var → `config/milla.yaml` → `milla-config.yaml` → `milla.yaml`. A sample is in `config/milla.yaml`:

```
db:
  path: milla_memory.db
node:
  id: home-milla
node:
  id: home-milla
  location: my-lab

ollama:
  url: http://localhost:11434/api/chat
  default_model: llama3.2
  keep_alive: 10m

chat:
  default_session: default
  history_limit: 50

prompt:
  max_tokens: 2000

thermal:
  enabled: false
  max_c: 85
  cooldown_ms: 120000
  sensor_path: /sys/class/thermal/thermal_zone0/temp

server:
  port: 17863
  pid_file: milla.pid
  heartbeat_ms: 5000

log:
  level: info
  file: milla.log
```

Environment overrides: `MILLA_DB_PATH`, `MILLA_NODE_ID`, `MILLA_NODE_LOCATION`, `OLLAMA_URL`, `OLLAMA_MODEL`, `OLLAMA_KEEP_ALIVE`, `CHAT_HISTORY_LIMIT`, `MAX_PROMPT_TOKENS`, `MILLA_DEFAULT_SESSION`, or `MILLA_CONFIG`.

Install dependencies

Dependencies are managed by Clojure CLI via `deps.edn`; no extra step is needed beyond having `clj` on PATH. The first run will download Maven deps automatically.

Initialize the database

- Fast path: `bin/milla-init-db`
- Or: `clj -M -e "(require ’milla.core) (milla.core/init!)"`

This creates `milla_memory.db` and required tables (see `doc/schema.sql`).

Run the server + prompt

Start the daemon (recommended for fast responses):

```
bin/milla-serve &
# or restart cleanly: bin/milla-restart-server
# stop daemon: bin/milla-stop
# health: bin/milla-health
# migrate schema: bin/milla-migrate
```

Optional first-run helper:

```
bin/milla-setup
```

All helper scripts accept `-h/-help`.

Then send prompts (CLI forwards to the server if running):

```
bin/milla llama3.2 "Hello, Milla"
```

Model argument is optional; if omitted, the configured `default_model` is used. For REPL use: `clj -M:repl` then `(require ’milla.core)` and call `(milla.core/ask!  :prompt "Hi")`.

Sync (kittens)

- Pull from a remote node: `bin/milla-sync-db pull user@host:/path/to/milla_memory.db`
- Push to a remote node: `bin/milla-sync-db push user@host:/path/to/milla_memory.db`

Pulling fetches `milla_memory.remote.db` and merges it into the local DB via `milla.sync/merge` (dedupe by content/time/source).

Merge two brains

```
bin/milla-merge /path/to/output.db /path/to/db1 /path/to/db2
```

Unions/dedupes statements/chat/chat_summaries and reruns summarization per session.

Testing

```
clj -M:test
```

Runs basic smoke tests for init/ask/fact and Ollama retry behavior.