

# operator-guide

## Operator guide

### Install

```
cd /Users/marchon/research/modelcontextprotocol/verae-mcp-timestamping-server
uv sync --group dev
```

### Run stdio

```
uv run verae-mcp-timestamping-server
```

### Run Streamable HTTP

```
VERAE_MCP_TRANSPORT=streamable-http uv run verae-mcp-timestamping-server
```

The listener is `http://127.0.0.1:8000/mcp`.

For the sample page on another origin:

```
VERAE_MCP_CORS=1 VERAE_MCP_TRANSPORT=streamable-http uv run verae-mcp-timestamping-server
```

A page served from the public internet still cannot call 127.0.0.1 from current browsers. Open `site/index.html` on the same machine, or put TLS and an access control in front of a non-loopback bind you choose yourself.

### Global token

A new machine runs `uvx verae-mcp-timestamping-server setup`. That creates an account or logs in and writes `~/.verae/auth.properties` (mode 600). It prints username, role, and expiry, not the JWT. `session` checks that file. Starting the server with no login prints the setup command on stderr.

`signup` creates a role `user` account with no bearer token. `create_user` adds an account when the stored role is `admin`.

Set `VERAE_MCP_OAUTH=1` and `VERAE_MCP_PUBLIC_URL` to the public HTTPS origin to require a browser login on the HTTP MCP endpoint. That login's JWT is the caller's bearer for `stamp` and is not written over the host file.

```
# or skip the tool and export a token yourself:  
export VERAЕ_GLOBAL_TOKEN=...
```

## Tests

```
uv run pytest
```

Stamp traces (live VeraeTime) are a separate script. They write  
debug-trace/<YYYY-MM-DDTHHMMSSZ>-<slug>/:

```
uv run python scripts/transaction_log_test.py
```

## Build these documents

```
bash scripts/build-docs.sh
```

HTML lands in docs/html/. PDF lands in docs/pdf/.