c1) Orientation: What EasyEngine is (and why it changes your workflow) ?

- c1.1 — What EasyEngine CLI is (and why it’s worth switching) □□

c1.1 — What EasyEngine CLI is (and why it’s worth switching) ?

What EasyEngine CLI does

EasyEngine (`ee`) is a **server-side CLI** that helps you **provision and operate WordPress and other web stacks** using **containers** (commonly Docker-based). Instead of clicking around in a shared-hosting control panel, you manage sites with **repeatable commands**, which makes your workflow:

1. **Faster**
   - Create, update, and operate sites in seconds/minutes.
2. **More consistent**
   - The same commands produce the same setup every time (great for staging/prod parity).
3. **More operationally powerful**
   - You get direct access to services like **Nginx, PHP, DB, Redis**, and the ability to automate routines (backups, cron, cache clears, etc.).

How it “thinks”: sites as managed units

With EasyEngine, a **site** (like `example.com`) becomes a **managed unit** you can operate with subcommands such as:

1. **Lifecycle**
   - Create, enable/disable, update, delete
2. **Operations**
   - Reload/restart services, clear caches (`ee site clean`)
3. **Access**
   - Shell into the site’s containers (`ee shell`)
4. **Publishing & sharing**
   - Temporarily expose a local/secured site via a share link (`ee site share`) □□
5. **Scheduling**
   - Manage cron jobs with `ee cron` □

This aligns strongly with real DevOps-style workflows: **declare intent → run command → verify state**.

What changes vs shared hosting (the practical mindset shift)

You're moving from **"panel-driven hosting"** to **"infrastructure you control"**:

1. **You manage the server**
   - Updates, firewall, SSH keys, resources, backups become *your* responsibility.
2. **You gain performance & control**
   - Tuning PHP/Nginx, adding Redis, isolating sites, debugging via logs/shell access.
3. **You automate more**
   - Commands are scriptable; cron becomes explicit; troubleshooting becomes faster.

Core tools you'll use daily (from your excerpt)

Here's how the parts in your provided docs fit together:

1. `ee site ...`
   - Day-to-day site operations
   - Examples from excerpt:
     - `ee site clean` → clears page/object cache
     - `ee site share` → share a site via an online link (with `--refresh`, `--disable`, `--token=...`)
2. `ee shell ...`
   - Your "daily driver" to run WP-CLI, inspect configs, read logs inside the right container context
3. `ee cron ...`
   - Create/list/update/delete/run scheduled tasks cleanly

Outcomes you can expect after this course ?

By the time you complete the chapters, you should be able to:

1. Provision WordPress sites confidently (including PHP/DB choices)
2. Operate and debug sites via `ee shell`
3. Manage caching and routine maintenance (`ee site clean`, reload/restart)
4. Create reliable scheduled tasks with `ee cron`
5. Use share links safely and intentionally (`ee site share`)

Next: pick c1.2 or tell me your goal ?

If you want a natural continuation, I suggest:

1. **c1.2 — Installation & prerequisites** (server requirements, Docker, DNS, ports, safety basics), or
2. **c2.4 — Create a WordPress site end-to-end** (fastest path to “I’m productive”), or
3. Tell me: *Do you plan to run one server for multiple client sites, or one server per project?*