Caddy has some **serious advantages** over Apache2, especially if you're running modern web apps. Here's a breakdown of why people often prefer Caddy:

---

1. Automatic HTTPS

- Caddy automatically provisions and renews SSL/TLS certificates with Let's Encrypt.
- No need to mess with `certbot`, cron jobs, or renewal scripts like Apache often requires.
- HTTPS is default, so every site is secure out of the box.

---

2. Simpler Configuration

- Caddy uses a **single `Caddyfile`**, which is way easier to read and maintain than Apache's multiple `.conf` files and modules.
- Reverse proxies, redirects, and rewrites are straightforward.
- No extra modules for basic features; Caddy includes most modern functionality by default.

---

3. Lightweight and Efficient

- Written in Go, Caddy is fast and handles high concurrency efficiently.
- Minimal memory footprint compared to Apache + mod_php + all the extras.
- Works great on low-resource VPS servers.

---

4. Built-in Features

- Reverse proxy, load balancing, caching, gzip compression, and HTTP/3 support—all built-in.
- Automatic file server, directory browsing, and Markdown rendering without extra plugins.
- Flexible JSON API for dynamic configuration if needed.

---

5. Less Maintenance

- Updates are simple: `caddy upgrade` or via system package.
- Fewer security headaches—Caddy doesn't load dozens of modules by default like Apache does.
- Logs and status are straightforward.

Bottom line:
If you want **ease, security, speed, and less hassle**, Caddy is excellent. Apache is great for legacy setups or complex .htaccess rules, but for modern PHP sites like yours, Caddy is usually simpler and lighter.
