Security checklist

For CMS developers, the security work of a site project as checklists. It covers the content model and roles, templates and output, forms and uploads, your own code, APIs, configuration and deployment, and the go-live...

Source file: `doc/eng-us/help/cms-developers/security-checklist.md` Last changed 2026-11-04
https://doc.exponential.earth/learninminutes/exphelp/viewer/help/cms-developers/security-checklist.md

This page is for developers who build sites and must hand them over secure. Exponential is secure by default in many places, and a project can still undo that with one careless role, template or setting. Go through these lists during development and again before launch; the hardening guide explains the attacks behind each item (Security hardening).

Contents

1. Content model and access
2. Templates and output
3. Forms and uploads
4. Your own code
5. APIs
6. Configuration and deployment
7. Accounts
8. Go-live hardening
9. Watching and reporting

Content model and access

☐ The Anonymous role reads only what visitors may read (`content/read` on the public sections), and nothing else beyond `user/login` and what the site's forms need.

☐ Members-only content is in its own section, read by a role assigned to the members' group, not just hidden.

☐ Editor roles are limited by class and subtree; no editor role has `setup/*`, `role/*` or `user/*` beyond need.

☐ Roles are assigned to groups; members and staff are in different groups.

☐ Access was tested with test accounts at `example.invalid`, and `./console exp:accesscheck` explains every surprise (Sections, roles and policies).

Templates and output

☐ Every value you print yourself goes through `|wash()`; datatype output comes from `attribute_view_gui` (Wash and escaping).

☐ No `<style>` blocks, `style=""` attributes, inline scripts or `javascript:` links in templates, so a Content Security Policy can be enforced (CSP-safe design, Content Security Policy).

☐ Editors cannot store markup: rich text has no tag for scripts; custom tags instead of pasted HTML (Rich text).

☐ No visitor-specific data in shared caches (Caching for CMS developers).

Forms and uploads

☐ `ezformtoken` is active and every form keeps the hidden fields the original template prints.

☐ Public forms have a captcha datatype; collected answers are removed on a schedule (Forms and information collection).

☐ Upload types are limited to what editors need; file attributes have a maximum size.

☐ Self-registration is on only where needed, with e-mail verification (Users and registration).

Your own code

☐ Every module view declares a policy function and checks it; writes need POST and the form token.

☐ SQL goes through `eZPersistentObject` or `eZDB` with values escaped (`escapeString()`) and IDs cast to integers (eZDB).

☐ Redirects go only to addresses of the site (Safe redirects).

☐ Content is created through the kernel API, as a dedicated user with exactly the rights needed (Content from PHP).

☐ The book's rules are met: Security in extensions.

APIs

☐ `rest.ini` `[Authentication]` `RequireAuthentication=enabled` (shipped) and `RequireHTTPS=enabled`.

☐ One account per client, with a role limited to what it needs; API keys with scopes and an expiry (The REST API, Personal API keys).

☐ Keys and tokens live in environment variables or secret stores, never in repositories.

Configuration and deployment

☐ Debug output off on staging and live, or limited to your address; PHP `display_errors=Off`.

☐ `settings/override/` is not in any repository and not copied between machines.

☐ Code is read-only for the site's user; workers and cronjobs never run as root.

☐ `ManagedByDeployment=enabled` where configuration comes from version control (Deploying a site).

☐ The rewrite rules of the web server keep `settings/`, `var/` internals and source files from being served (Security hardening, chapter 13).

Accounts

☐ The installer's administrator password changed and `var/log/initial-admin-password` removed (Change the admin password).

☐ A personal administrator account per person; two-factor sign-in for administrators (Two-factor sign-in).

☐ Password rules decided (`MinPasswordLength`, `WeakPasswordCheck`, `[PasswordSettings]`).

☐ Demo mode off (`EXP_DEMO_MODE` not set) (Demo mode).

Go-live hardening

☐ HTTPS everywhere, HSTS, secure cookies.

☐ The administration on its own host or restricted by network.

☐ Request rules or the request shield in front, first monitoring, then enforcing (Request rules).

☐ The setting-by-setting list: Go-live checklist.

Watching and reporting

☐ Audit alerts go to a person and the trail leaves the machine (Security and audit).

☐ Logs are read weekly; backups are off the machine and a restore was tested.

☐ Updates of Exponential and its extensions are followed and tested.

☐ Your team knows how to report a vulnerability (Reporting a vulnerability).

Related pages

- Security standards
- Security review checklist for hosts
- Project launch checklist
- Free documentation: this page is free to copy, translate and improve under the GNU FDL 1.3.

> **Become an Exponential Partner**
>
> Partners share their information, provide funding, contribute back and develop the products with us, and that is how Exponential grows. A partner website and dashboard are planned for mid-2027: please become a partner today to speed up our growth rate and land new features faster.
>
> Partners: start here | How to apply

Exponential Copyright © 1998-2026 Exponential Foundation and 7x
Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".
This documentation is free documentation, licensed under the GNU Free Documentation License 1.3 or later:
https://www.gnu.org/licenses/fdl-1.3.html.