Software engineers: start here

Where software engineers start who evaluate, architect, operate, review or contribute to Exponential as a system. The system in one paragraph, your first hour, five engineering rules, and a map of every page for…

Source file: `doc/eng-us/help/software-engineers/software-engineers-start-here.md` Last changed 2026-11-04
https://doc.exponential.earth/learninminutes/exphelp/viewer/help/software-engineers/software-engineers-start-here.md

This section of the help is for software engineers, whatever language you usually write: people who evaluate Exponential for a project, design a system around it, run it in production, review changes to it or contribute code back. It explains the system as a system: its parts and their contracts, how a request travels through it, where data lives, which caches sit where, how it is served, tested, released and secured, and which documents are the source of truth for each of those.

These pages do not repeat the detailed guides. They give you the engineering view and link to the page that has the details, usually in `doc/eng-us/standards/`, `doc/eng-us/bc/6.0/`, `doc/eng-us/features/6.0/` or the guides. Where the sources do not answer a question, the pages say so instead of guessing.

Contents

1. Exponential in one paragraph
2. Your first hour
3. Five engineering rules
4. Map of the software engineering pages
5. Pages from other audiences you will use
6. Finding these pages again

Exponential in one paragraph

Exponential is a free and open source PHP content management system and application framework, licensed under the GNU GPL, version 2 or later. Its README calls it "a professional PHP application framework with advanced CMS (content management system) functionality", whose most notable feature is "its fully customizable and extendable content model". The kernel (`kernel/`, `lib/`) runs on PHP 8.0 to 8.5 (PHP 7.4 usable without support), stores content in SQLite 3 (the default), PostgreSQL, MongoDB, MariaDB, Oracle or MySQL, renders through its own template language, and is extended through extensions in `extension/`. Since 6.0.15 it ships its own application server, Exponential Velocity, next to Apache or nginx with PHP-FPM and FrankenPHP. The code base was developed from 1999, first under the name eZ Publish, a product its creator discontinued after almost 18 years; 7x leads the development today and the Exponential Foundation operates and funds the project (Who is 7x?, Who is the Exponential Foundation?).

Your first hour

1. **Get the shape of it** (10 minutes): Exponential for software engineers and Architecture overview.
2. **Follow one request** (10 minutes): The request lifecycle.
3. **Know where state lives** (10 minutes): Databases: choosing and running one and Caching layers for engineers.
4. **Know how it runs** (10 minutes): Application servers and Concurrency, statics and long-lived processes.
5. **Know how change is controlled** (10 minutes): Testing strategy, CI gates and Release engineering.
6. **Know the security contract** (10 minutes): Security engineering overview.

Five engineering rules

1. **Never edit `kernel/`, `lib/` or another extension in place.** An update replaces them. Use the extension points; a kernel class override is the last resort (Events, extension points and plugins).
2. **Assume a long-lived process.** Under Velocity a worker serves many requests: no per-request data in statics without a reset, idempotent registrations (Concurrency, statics and long-lived processes).
3. **Write for every database.** Go through `eZDB::instance()`, escape and cast, and ship schema changes for every driver (Writing portable SQL).
4. **A published tag never changes.** A mistake in a release is fixed by the next version, never by moving a tag (Version numbers and immutable tags).
5. **Every break has upgrade notes.** A change that needs an action from an installation gets a page in `doc/eng-us/bc/6.0/` (Backward compatibility policy).

Map of the software engineering pages

Orientation

Page	What it helps you do
Exponential for software engineers	Understand the system, its stack and its constraints on one page
Reading path for software engineers	Read the pages in a sensible order, by role
Evaluating Exponential	Run a structured technical evaluation
Glossary for software engineers, Software engineer FAQ	Terms and quick answers

Architecture

Page	What it helps you do
Architecture overview	See the layers and their responsibilities

Page	What it helps you do
The request lifecycle	Follow a request from `index.php` to the response
Kernel, lib and extensions: the layering	Know which code may depend on which
Siteaccesses and configuration	Understand the INI cascade and siteaccess matching
Modules, views and access control	Understand routing and the permission model
The content model for engineers	Understand classes, objects, versions, nodes and datatypes
Persistence and the database layer	Understand `eZDB` and `eZPersistentObject`
The template engine and the design chain	Understand compiled templates, overrides and designs
Events, extension points and plugins	Extend without forking
Console, cronjobs and background work	Run work outside a web request

Data

Page	What it helps you do
Databases: choosing and running one	Choose among the six supported databases
Schema changes and migrations	Ship a schema change that every installation can apply
Writing portable SQL	Write queries that work on every driver

Runtime, performance and scaling

Page	What it helps you do
Caching layers for engineers	Know every cache and what it is keyed on
Cache invalidation and clearing	Invalidate correctly after a change
Application servers	Choose between Apache or nginx with PHP-FPM, Velocity and FrankenPHP
Velocity persistent workers	Understand the worker pool and its reset model
Concurrency, statics and long-lived processes	Write code that is safe in a persistent worker
Scaling and clustering	Plan several web nodes and shared storage
Performance measurement and benchmarks	Measure before and after a change

Operations

Page	What it helps you do
Observability: logs, debug bar and audit trail	See what the system is doing
Deploying a change	Ship a PHP, settings or template change in the right order
Incident handling	Respond to an outage or a regression
Multi-site installations	Run several sites from one installation
Internationalization and right to left	Handle languages, locales and text direction

Quality

Page	What it helps you do
Testing strategy	Know which kinds of tests exist and when each runs
DB-less tests and test suites	Write and run tests without a database
Static analysis and code style	Work with PHPStan, the baseline and phpcs
CI gates	Read a red check and reproduce it
Code review checklist, Design review checklist	Review a change or a design
Handling technical debt	Find, record and pay down debt

Release engineering and compatibility

Page	What it helps you do
Release engineering	Cut a release of the kernel or an extension
Version numbers and immutable tags	Choose the next number and never move a tag
Changelogs and upgrade notes	Write the record of a change
Dependency management with Composer	Pin, repin and resolve packages
Backward compatibility policy	Know what counts as a break and what must accompany it

Security

Page	What it helps you do
Security engineering overview	Know the security model, the defaults and the disclosure process
HTTP security: headers, CSP, request rules and proxies	Configure the HTTP layer safely
Authentication, sessions and two-factor sign-in	Understand sign-in and sessions
Security review checklist	Review a change for security

Interfaces and collaboration

Page	What it helps you do
API design: REST, ezjscore and headless	Choose and design an API
Git workflow	Work with branches, commits and pull requests
Contributing upstream	Send a change to the kernel or an extension
Documentation as code	Write and check help pages like code

Pages from other audiences you will use

Page	Why software engineers read it
Extension development	The full course from user to extension developer, with an API reference
Application creators: start here	Building applications on Exponential, with code
PHP developers: start here	The PHP side in depth
Web framework developers: start here	Exponential compared with other web frameworks
Hosting companies: start here	Servers, panels, operations and runbooks
Database support	The support statement for every database
Develop with Git, Update with Git	Working with clones of the kernel and extensions
Security hardening	The attacker-oriented hardening guide
What Exponential is	The product family and what each product contains

Finding these pages again

- In the help, filter by the audience **Software Engineers**, or open **All pages** and the folder `software-engineers`.
- Each page has **Edit on GitHub**. If a statement here is wrong or out of date, correct it there; the pages are checked like code (Documentation as code).

Related pages

- Exponential for software engineers
- Architecture overview
- Reading path for software engineers
- Migrations for software engineers, Migrations
- 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.