Evaluating Exponential

A structured technical evaluation of Exponential for a project - the questions to answer about fit, runtime, data, performance, security, quality, maintenance and licensing, how to answer each one from the code and the...

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

This page is for engineers asked "should we build on Exponential?" It turns that question into smaller ones that the repository and the documentation can answer, so the result is evidence rather than impression. Write the answers down with their sources; the Design review checklist is a template for the result.

Contents

1. Set up a test installation
2. The questions, by area
3. Reading the project's own status statements
4. Risks to weigh

Set up a test installation

Evaluate on a real installation, not on screenshots. The quickest is SQLite, which needs no database server:

Shell

```
composer install
./console exp:install --help
```

`exp:install` uses SQLite when no `--db` is given (Database support). The installation book describes every install method (Choosing an install method). Serve it with Velocity to see the recommended runtime (Application servers).

The questions, by area

Area	Question	Where to find the answer
Fit	Does the content model express our data (typed attributes, versions, translations, relations, tree locations)?	The content model for engineers
Fit	Which product of the family do we need, and does it contain the parts we need?	What Exponential is
Runtime	Which PHP versions and extensions do our servers have?	`composer.json`, Requirements
Runtime	Which server will serve it?	Application servers
Data	Which database, and is it tested in CI?	Databases: choosing and running one

Area	Question	Where to find the answer
Performance	What does a cached and an uncached page cost on our hardware?	Performance measurement and benchmarks
Scale	Do we need several web nodes, and what must be shared?	Scaling and clustering
Security	What are the defaults, and how are vulnerabilities reported and fixed?	Security engineering overview
Integration	Which APIs do our other systems need?	API design: REST, ezjscore and headless
Quality	What do CI and the local gates check?	CI gates
Maintenance	What counts as a breaking change, and how is it announced?	Backward compatibility policy
Releases	How often are releases published, and how are they numbered?	Release engineering, `doc/changelogs/`
License	Is the GNU GPL, version 2 or later, compatible with how we distribute our code?	`LICENSE`, Contributing upstream

Reading the project's own status statements

The standards pages in `doc/eng-us/standards/` rate the project against public standards with **Complies**, **Partly complies**, **Does not comply** or **Unverified**, and list known gaps. They are the most honest single source for an evaluation, because they name what is missing:

- Security standards: OWASP Top 10, ASVS, CSP.
- Source control and releases: branch protection and signed commits are listed as gaps.
- Database and SQL standards and Testing and CI.
- Internationalization standards and Accessibility.

Risks to weigh

- **Inherited code.** The kernel is old and large; PHPStan runs at level 1 with a baseline of findings that existed when the check was introduced (Handling technical debt).
- **Database updates by hand on old installations.** Since 6.0.15 `exp:upgrade` and the migrations (`exp:migration:migrate`) apply and record every change; an installation upgraded by hand with the SQL update files has no record of which statements ran (Schema changes and migrations).
- **Policy, not enforcement.** Some repository rules (no force push, merge style) are policy without branch protection (Source control and releases).
- **A smaller ecosystem** than the largest PHP frameworks. Check the extension catalog for what you need before you plan to write it (The se7enxweb extension catalog).

Related pages

- Exponential for software engineers
- Design review checklist
- Software engineer FAQ

- 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.