Testing strategy

How Exponential is tested and how to test your own changes - the layers from syntax checks and static analysis to DB-less PHPUnit suites, database tests, live-installation tests, design and quality tests, benchmarks and...

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

This page is for engineers who plan the tests of a change, review whether a change is tested enough, or evaluate how well Exponential itself is tested. The standard, with counts and known gaps, is Testing and continuous integration.

Contents

1. The layers
2. What runs where
3. Choosing the tests for a change
4. What a good test here looks like
5. Known gaps

The layers

Layer	Tool	Catches	Speed
Syntax	`php -l` (PHP 8.3 and 8.0), `node --check`	parse errors, syntax newer than PHP 8.0	seconds
File manifest	`bin/php/checkmanifest.php`	a file whose checksum is missing or stale in `share/filelist.md5`	seconds
Code style	`bin/php/checkcodestyle.php`, `phpcs.xml.dist`	new style violations, type declarations, PHP 8.0 incompatibilities	seconds
Static analysis	PHPStan level 1 with a baseline	new findings in `kernel/` and `lib/`	a minute
Unit and component tests without a database	PHPUnit, most suites	logic, parsing, rendering, security rules	minutes
Design and quality tests	PHPUnit suites `design` and `quality`	inline CSS or scripts in admin templates, token rules, translations, the sniffs themselves	minutes
Database tests	`tests/runtests.php --dsn ... --db-per-test`	SQL on MySQL 8.0 and PostgreSQL 15	longer
Live-installation tests	suites that need an installed site, or live servers named in environment variables	integration with a real site, MongoDB, HTTP	skipped without them
Boot smoke test	`tests/bin/bootsmoke.php` on PHP 8.0	a class that does not load, a missing compatibility function	seconds

Layer	Tool	Catches	Speed
Benchmarks	`bin/php/benchmark.php micro`	performance regressions (warning only)	a minute

What runs where

Where	What
`git commit` (after `make hooks`)	syntax, manifest, code style of staged files, commit message format
`git push` (after `make hooks` and `make devtools`)	PHPStan against the baseline
CI, Quality	syntax, manifest, commit messages, code style (warning), PHPStan, sniff tests, PHP 8.0 lint, install and boot
CI, PHPUnit	the DB-less run on PHP 8.1 to 8.5, and database tests on MySQL 8.0 and PostgreSQL 15
CI, Performance	the micro benchmark (warning)

Details: CI gates and Quality checks.

Choosing the tests for a change

1. **The tests that map to the changed files.** `tests/test-map.json` maps source files to the test files that cover them; with the tests named after the changed classes it selects the tests of a change.
2. **A new test that fails without the change and passes with it.** The contribution guide asks for exactly this.
3. **The suite of the area**, for example `php vendor/bin/phpunit --testsuite kernel-classes`.
4. **The whole DB-less run** before a larger push: `php vendor/bin/phpunit`.
5. **The database tests** when SQL changed.
6. **A check on a running site** for anything that renders: light and dark mode, the widths you support, no console errors.

What a good test here looks like

- It needs no database unless it tests SQL, so it runs on every CI leg (DB-less tests and test suites).
- It reads nothing from a developer's installation: no `settings/override`, no `var/` caches, no installed site. CI has none of them.
- It leaves no state behind: no static cache, INI value or file under `var/` that the next test sees.
- A test that needs a live server reads its location from environment variables and skips with the variable's name when it is not set; it carries no host name, account or password of its own.

Known gaps

From the standard: some test files are in no suite, the live-installation suites are skipped in CI, the database jobs cover MySQL and PostgreSQL only (not SQLite, MariaDB, Oracle or MongoDB servers), PHPUnit does not run on PHP 8.0, coverage is collected but not published or enforced, and the hooks are opt-in. Plan around them: if your change is specific to an engine without a CI job, test it on that engine yourself.

Related pages

- Testing applications
- Tests for extensions
- Code review 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.