# Authentication and sessions

How Exponential knows who is asking, for developers used to Symfony security firewalls, Laravel guards, Django auth and Devise - users as content objects, the anonymous user, login handlers, sessions and their handlers...

Source file: `doc/eng-us/help/web-framework-developers/authentication-and-sessions.md` Last changed 2026-11-05
https://doc.exponential.earth/learninminutes/exphelp/viewer/help/web-framework-developers/authentication-and-sessions.md

This page is for developers who configure a security firewall, guards or an authentication backend. In Exponential, users are **content**: each user is a content object of the user class with an `ezuser` field holding login, e-mail and password hash, placed in user groups in the content tree. Authentication is done by the kernel's `user` module and login handlers; the session keeps the signed-in user.

## Contents

1. Users and the anonymous user
2. Signing in
3. Sessions
4. Requiring login
5. Machines: API keys and OAuth2
6. The current user in code
7. Compared with other frameworks

## Users and the anonymous user

| Concept | In Exponential |
|---|---|
| User model | a content object of the user class (`site.ini [UserSettings] UserClassID`, 4 on a standard install) with an `ezuser` attribute; PHP class `eZUser` |
| Groups | user group objects in the tree; roles are assigned to groups or users |
| Guest | a real user, the anonymous user (`site.ini [UserSettings] AnonymousUserID`, 10 on a standard install), with its own role |
| Profile fields | attributes of the user class, edited like any content |

Because users are content, they get versions, translations of their profile fields, permissions and search like any other content.

## Signing in

- The `user/login` view signs a user in; `user/logout`, `user/register`, `user/forgotpassword` and `user/activate` do what their names say.

- **Login handlers** check credentials: `site.ini [UserSettings] LoginHandler[]` (`standard` shipped; `LDAP` and `textfile` exist as alternatives). The first handler that accepts the credentials signs the user in. Your own handler is an extension point (User login handler).
- **Two-step sign-in** (authenticator app or e-mail code) comes from the `sevenx_authentication_2fa` extension, a login handler placed before `standard` (Two-step sign-in).
- In code, `eZUser::loginUser( $login, $password )` checks credentials and signs in.

## Sessions

- A session keeps one browser signed in. It ends `site.ini [Session] SessionTimeout` seconds after its last use (31104000, 360 days, shipped).
- `site.ini [Session] Handler` empty (the default) means `ezpSessionHandlerPHP`: PHP's own session storage. `ezpSessionHandlerDB` keeps sessions in the table `ezsession`, which lets **Setup > Sessions** list and remove them (Sessions). A Symfony-based handler (`ezpSessionHandlerSymfony`) is used when the kernel runs inside Symfony.
- Read and write session values through `eZHTTPTool::instance()` (`setSessionVariable()`, `sessionVariable()`, `hasSessionVariable()`), not `$_SESSION` directly.
- Session cookies are named with the prefix of `[Session] SessionNamePrefix`.

## Requiring login

| Need | Setting |
|---|---|
| A whole siteaccess for signed-in users (the administration, an intranet) | `site.ini [SiteAccessSettings] RequireUserLogin=true` for that siteaccess |
| Views reachable without login on such a siteaccess | `[SiteAccessSettings] AnonymousAccessList[]` (`user/register`, `user/forgotpassword` …) |
| Sign in before an area of a public site | a request rule with `Action=login` (Middleware) |
| Who may use a siteaccess at all | the `user/login` policy limited by siteaccess (Authorization) |

## Machines: API keys and OAuth2

Machines do not sign in with a form. The REST API accepts **personal API keys** (`Authorization: Bearer expk_...`) and **OAuth2** tokens (`POST /api/auth/v1/oauth/token`); expservices has its own tokens (OAuth and API keys, Personal API keys). Whatever the credential, every call is checked against the user's policies.

## The current user in code

PHP

```php
$user = eZUser::currentUser();
if ( !$user->isRegistered() ) { /* anonymous */ }
$userID = eZUser::currentUserID();
```

On the command line and in workers the user is anonymous until the code signs one in, for

example a dedicated integration user with `eZUser::setCurrentlyLoggedInUser( $user, $user->id() )`; never the shared administrator (Access control in code). In templates: `fetch( 'user', 'current_user' )`.

## Compared with other frameworks

| Framework | Concept | Exponential |
|---|---|---|
| Symfony | user provider, authenticators, firewalls | the user class, login handlers, siteaccess login rules |
| Laravel | guards, providers, Sanctum tokens | session login, login handlers, API keys and OAuth2 for the REST API |
| Django | `User` model, authentication backends | user objects, login handlers |
| Rails | Devise, `has_secure_password` | the user module and login handlers, built in |
| Spring | Spring Security filter chain | login handlers, request rules, policies |

## Related pages

- Accounts and sign-in
- Member registration and areas
- Security and audit
- 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.