

TITRE PROFESSIONNEL DÉVELOPPEUR WEB ET WEB MOBILE

Soutenance
David SAADOUN – Knowledge

Sommaires



I. PRÉSENTATION
DU PROJET



II. CONCEPTION



III. DÉVELOPPEMENT



IV. RÉALISATIONS
PERSONNELLES



V. DÉMONSTRATION
DU PROJET



VI. EXEMPLE DE
RECHERCHE



VII. CONCLUSION

Organisation du travail



David

Product owner



David

Scrum master



David

Dev front



David

Dev back



David

Git master





Missions réalisées

- Conception de la base de données (MCD / MLD)
- Création des interfaces
- Développement Symfony (back-end & front-end)
- Sécurité, validation, gestion des rôles
- Envoi de mails, gestion des erreurs
- Tests, débogage et déploiement en local

CONSTAT

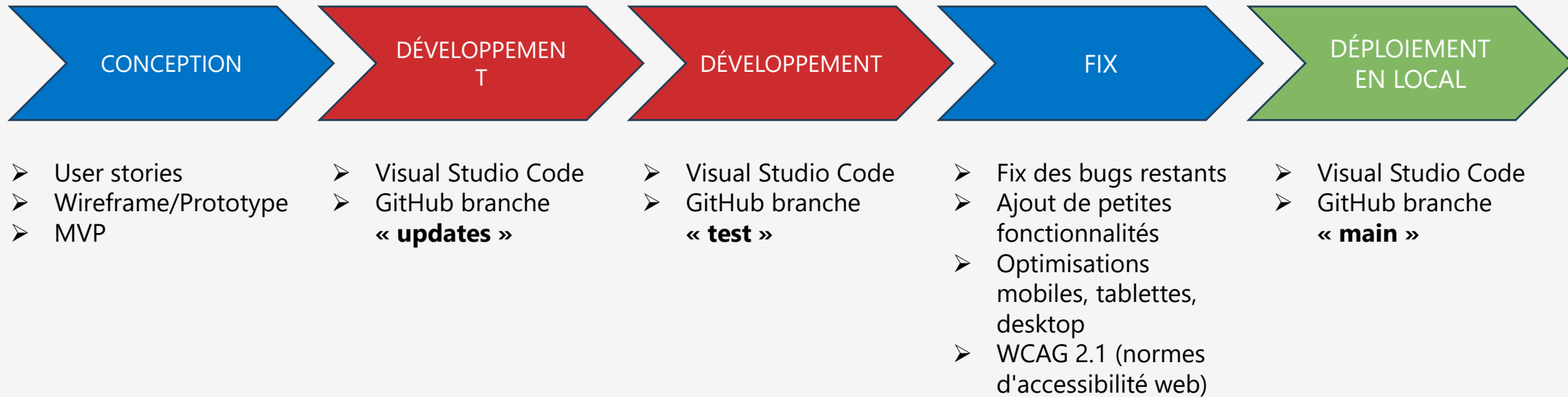
De nombreux élèves en formation ont du mal à suivre leurs parcours, accéder aux contenus ou interagir avec leurs formateurs via des outils simples et accessibles.

AUDIENCE

-  Étudiants en formation
-  Formateurs
-  Centres de formation
-  Administrateurs pédagogiques

Organisation du travail

- **5 sprints**, du vendredi au vendredi
- Utilisation de Teams, communication via Discord
- Daily meeting à 9 heures.
- Journée type de **9h à 19h**



Fusion sur la branche **main** puis déploiement le vendredi après-midi. Développement effectué avec Visual Studio Code

Conception



User stories

➤ 3 types d'utilisateurs

- Visiteur
- Étudiant
- Administrateur

➤ En vert: **MVP**

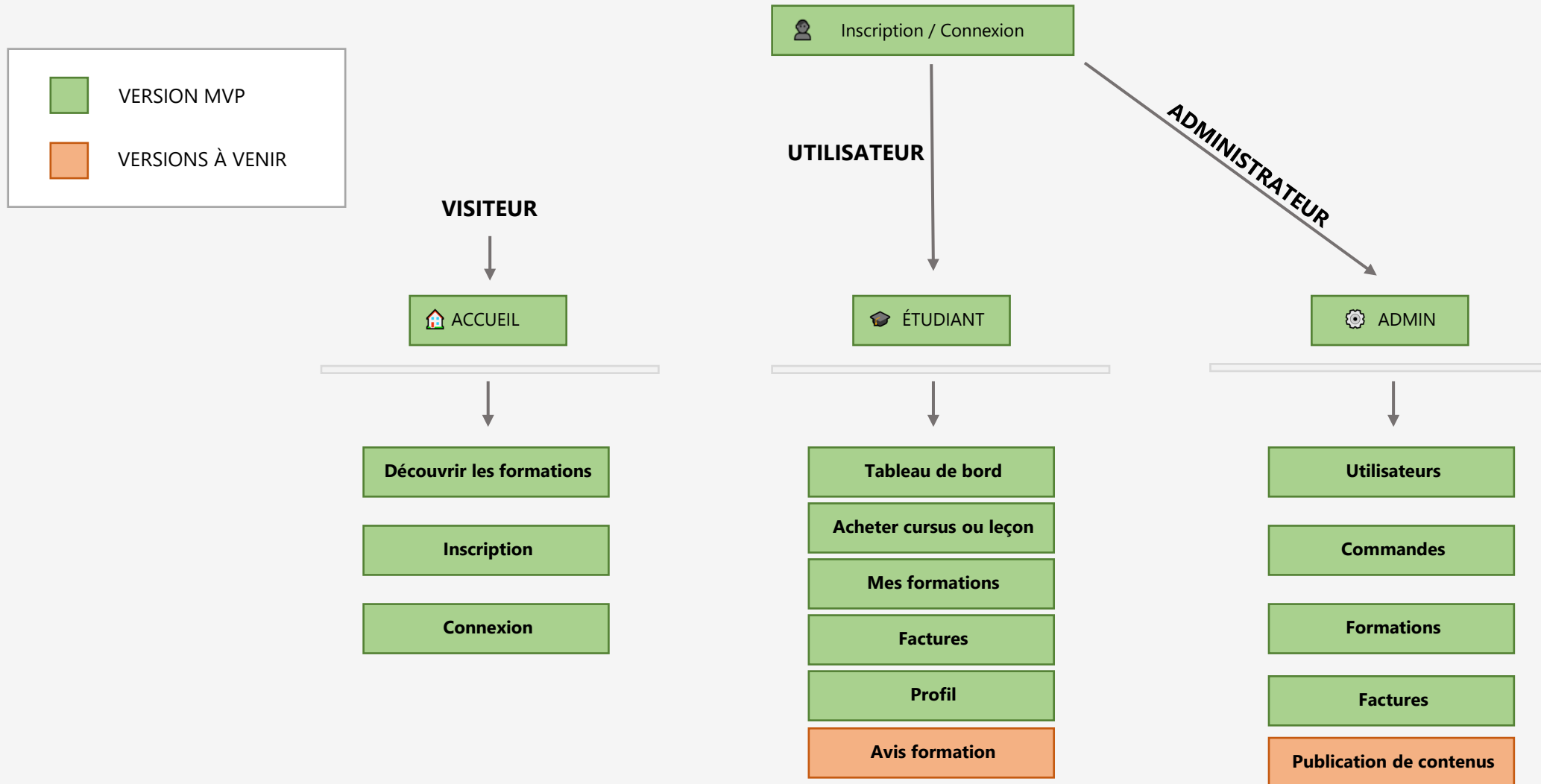
➤ En orange les versions futures à venir

En Tant que	Je souhaite	Afin de	Critère d'acceptation
visiteur	créer un compte	accéder à l'espace élève	L'utilisateur peut remplir un formulaire, valider, et recevoir un email de confirmation.
visiteur	me connecter	consulter mes formations	Un utilisateur existant peut saisir ses identifiants et accéder à son espace personnel.
visiteur	découvrir le contenu de la plateforme	comprendre l'intérêt de m'inscrire	Un visiteur non inscrit peut visualiser une page d'accueil avec exemples de cours.
Étudiant	accéder à mes cours	suivre ma progression	L'étudiant voit la liste des cours achetés avec accès aux contenus.
Étudiant	acheter un cursus ou une leçon	enrichir mes connaissances	L'étudiant peut choisir un cours, l'ajouter au panier et payer en ligne.
Étudiant	consulter mes factures	garder une trace de mes achats	Une page personnelle affiche l'historique des paiements téléchargeables.
Étudiant	modifier mes informations personnelles	mettre à jour mon profil	L'utilisateur peut modifier ses données (email, nom...) et sauvegarder.
Étudiant	changer mon mot de passe	sécuriser mon compte	Un champ permet de saisir un nouveau mot de passe avec confirmation.
Étudiant	obtenir une certification	valider mes acquis	L'étudiant peut cliquer sur un bouton pour générer un certificat PDF.

User stories

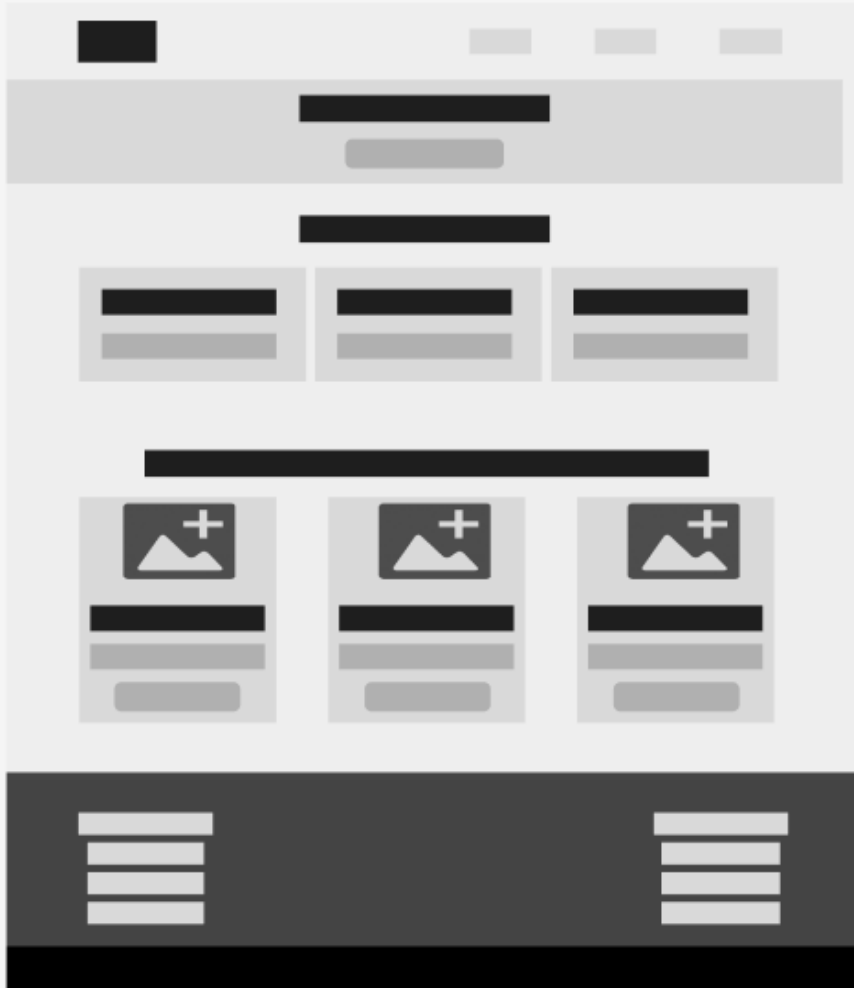
En Tant que	Je souhaite	Afin de	Critère d'acceptation
Administrateur	voir toutes les commandes et factures des utilisateurs	assurer le suivi administratif complet	Une interface d'administration liste toutes les commandes triables et exportables.
Administrateur	gérer les comptes utilisateurs	activer, désactiver, ou corriger des accès	L'administrateur peut activer/désactiver ou réinitialiser un compte utilisateur.
Formateur	publier des contenus	proposer des ressources pédagogiques	Un éditeur de contenu est accessible pour créer, modifier et publier une ressource.
Formateur	suivre l'avancement des élèves	adapter son accompagnement pédagogique	Un tableau de bord affiche les progrès par élève, par cours.
Étudiant	laisser un avis sur une formation	aider les autres à faire leur choix	Un formulaire de notation et commentaire est disponible après avoir suivi une formation.

Arborescence



Wireframes

Desktop Wireframe



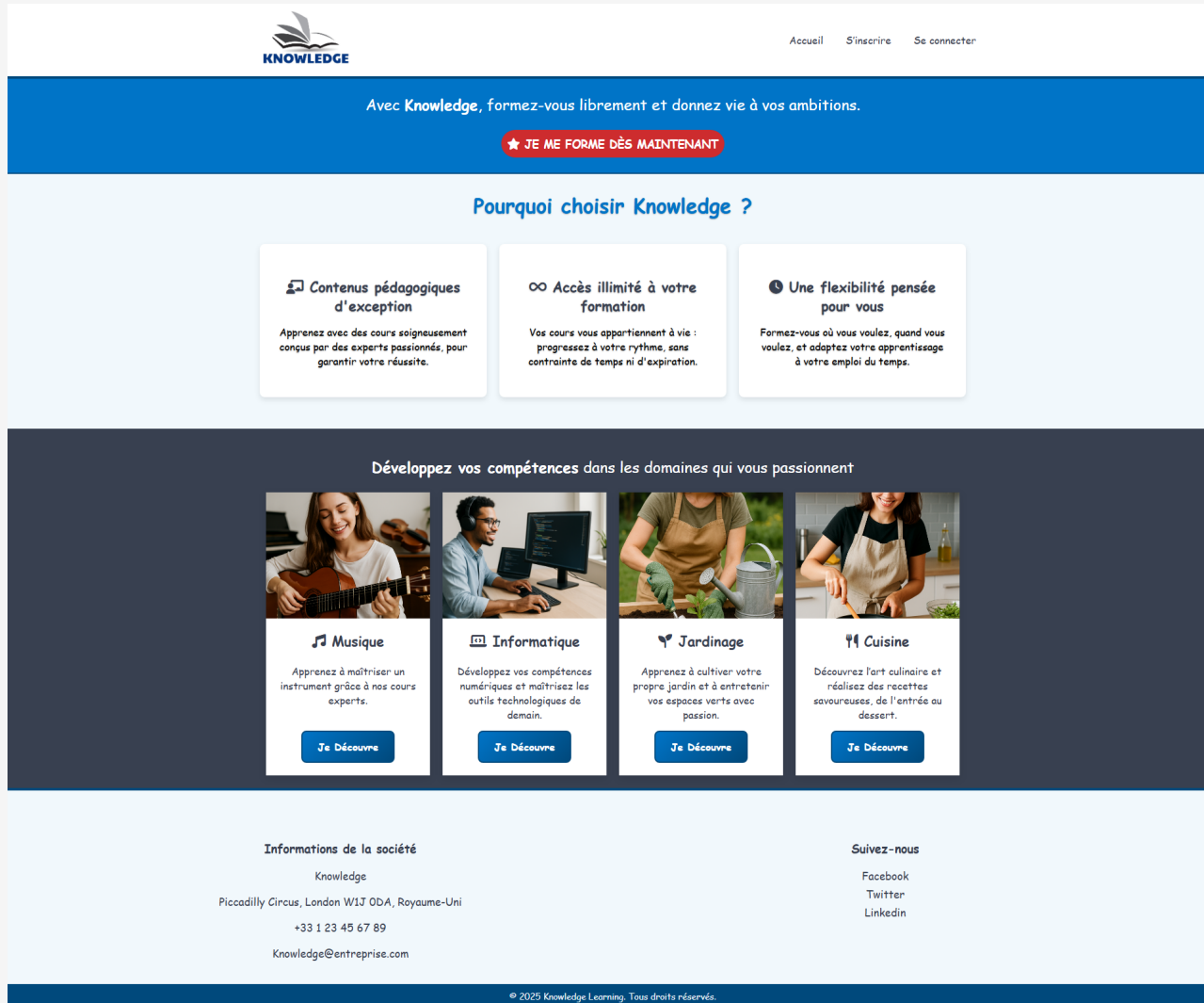
Desktop

Mobile Wireframe



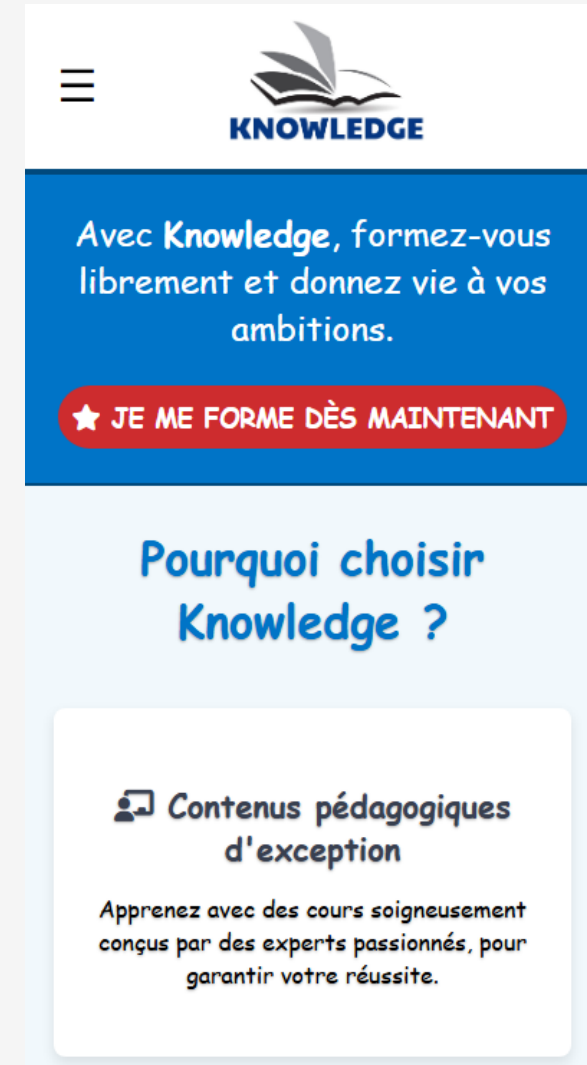
Mobile (rogné)

Maquettes



Desktop

Extrait des wireframes avec FIGMA



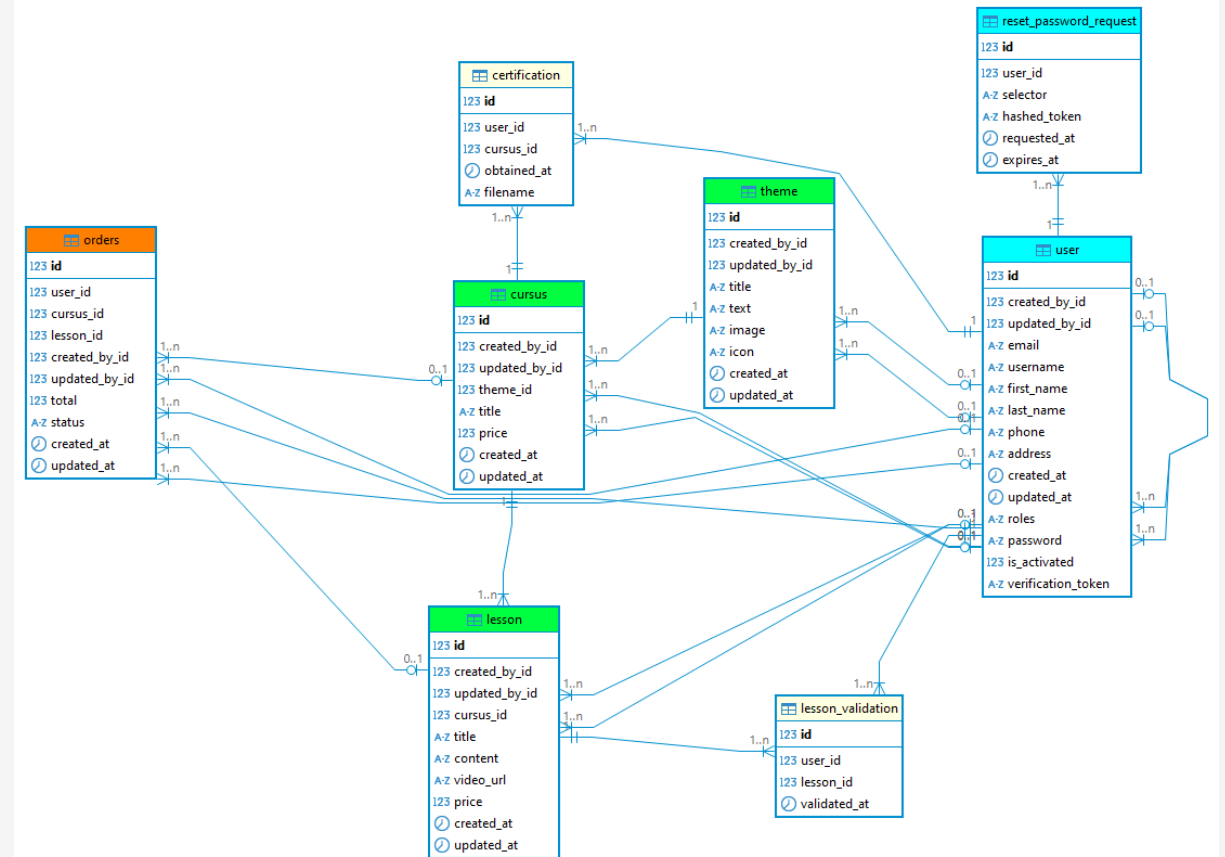
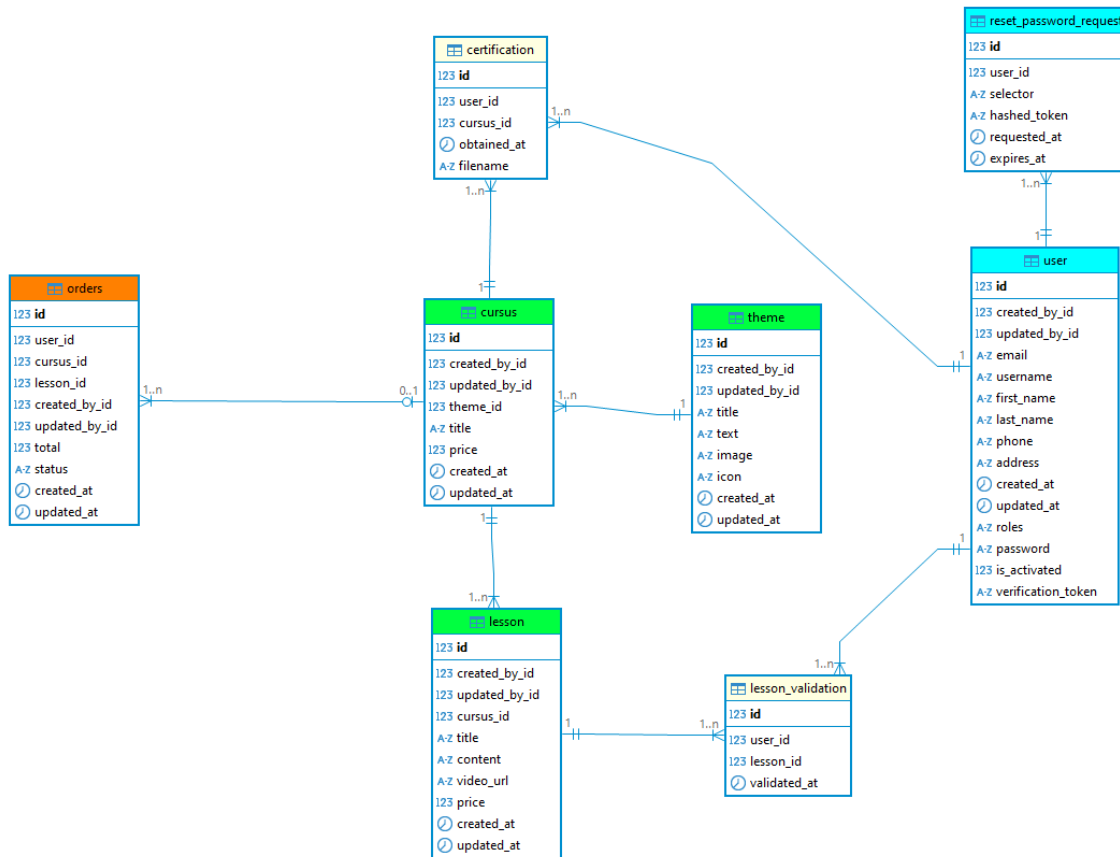
Mobile (rogné)

Dictionnaire des données

Nom de la donnée	Désignation	Type	contrainte
username	Nom d'utilisateur	varchar(255)	not null, unique
email	Adresse email	varchar(255)	not null, unique
password	Mot de passe haché	varchar(255)	not null
roles	Liste des rôles (ex: ROLE_USER, ROLE_ADMIN)	json	not null
is_activated	Statut d'activation du compte	boolean	not null, default false
created_by_id	Créé par un autre utilisateur	int	foreign key to user.id, nullable
updated_by_id	Modifié par un autre utilisateur	int	foreign key to user.id, nullable
created_at	Date de création	datetime	not null, default current_timestamp
updated_at	Date de modification	datetime	nullable
verification_token	Jeton de vérification email	varchar(255)	nullable
first_name	Prénom de l'utilisateur	varchar(255)	nullable
last_name	Nom de famille de l'utilisateur	varchar(255)	nullable
phone	Numéro de téléphone	varchar(20)	nullable
address	Adresse postale	varchar(255)	nullable

Extrait du dictionnaire des données

MCD



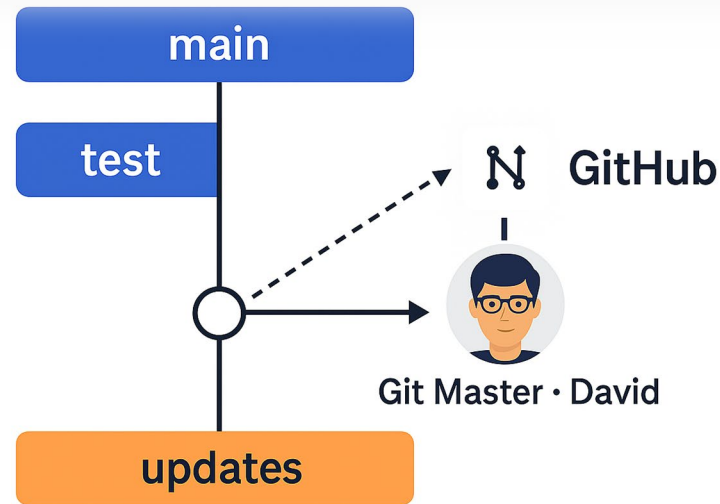
Modèle conceptuel de données (MCD) réalisé avec l'outil DBeaver – version simplifiée à gauche, version complète à droite.

Stack Technique

FRONT	BACK	DEVELOPER EXPERIENCE
<ul style="list-style-type: none">➤ HTML5 / CSS3➤ SCSS➤ JavaScript➤ Webpack Encore➤ Bootstrap 5	<ul style="list-style-type: none">➤ Symfony 7(PHP 8.1)➤ Doctrine ORM➤ Twig➤ MySQL (via DATABASE_URL)➤ JWT Auth➤ Services tiers : Stripe, MailJet	<ul style="list-style-type: none">➤ Docker (via compose.yaml)➤ Git + GitHub➤ Visual Studio Code➤ Symfony Profiler➤ PHPUnit➤ .env et configuration Symfony➤ Composer➤ Symfony CLI➤ FIGMA

Gitflow

Branche de **updates**
ex: add_login_user



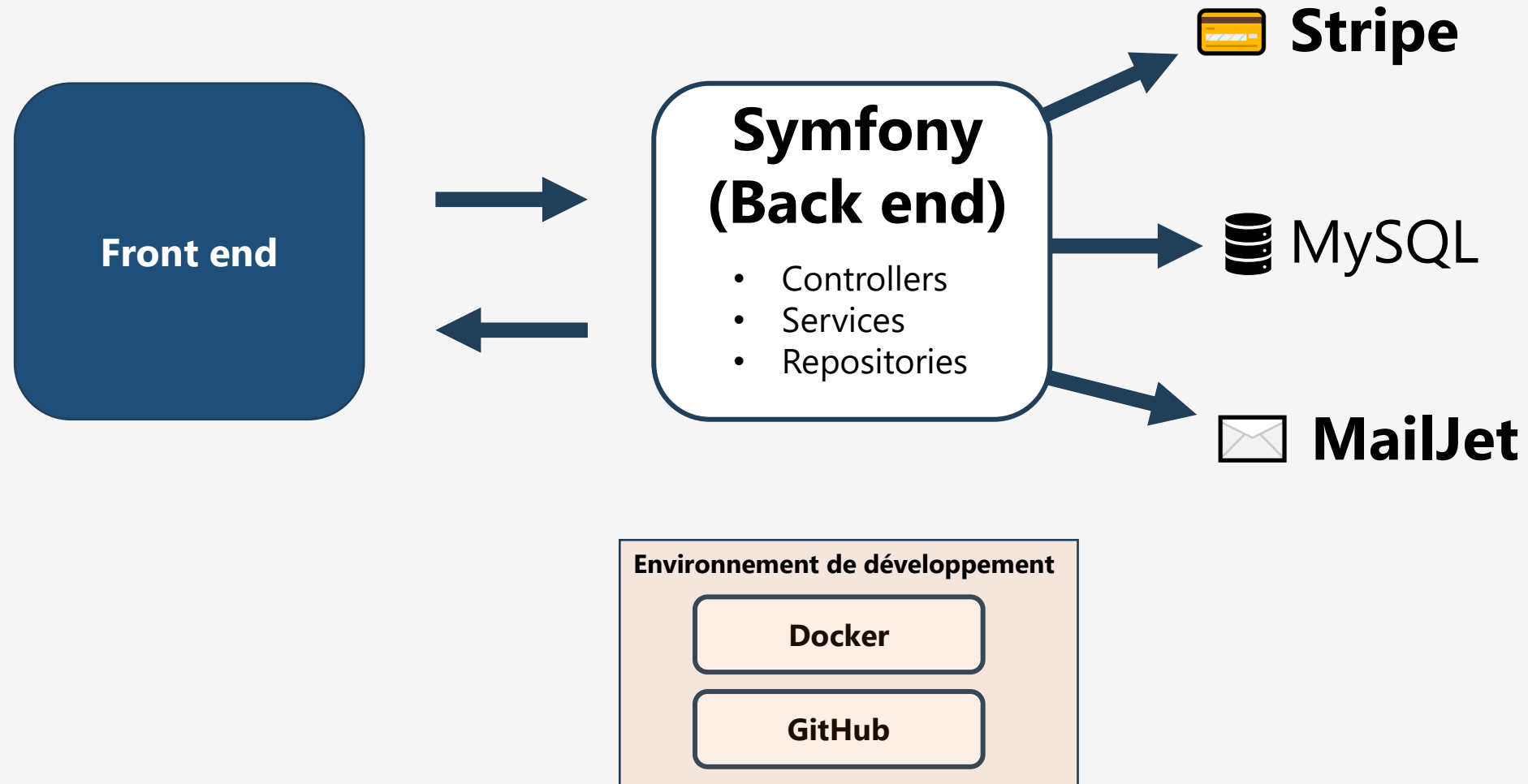
- 1 - Branche **updates**: contient les modifications sur lesquelles travaillent les membres de l'équipe.
- 2 - Branche **test**: fusionne les branches Updates pour effectuer des essais.
- 3 - Branche **main**: fusionne les branches validées, version destinée à la production

Le merge se fait uniquement via une **Pull Request**, suivie d'une revue de code par le Responsable Git Master « David »

Développement

[illegible]

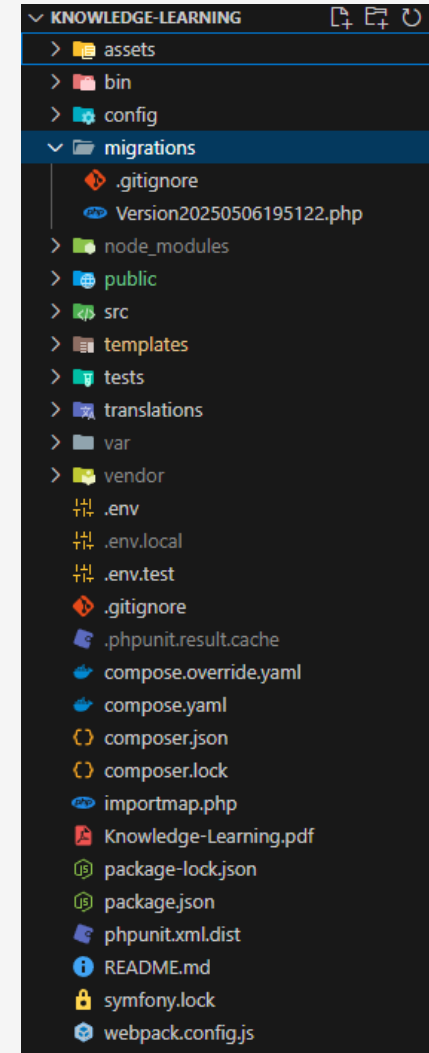
Architecture



Création de la base de données

```
1 <?php
2
3 namespace App\Entity;
4
5 use App\Entity\Order;
6 use App\Entity\Certification;
7 use App\Repository\UserRepository;
8 use Doctrine\Common\Collections\ArrayCollection;
9 use Doctrine\Common\Collections\Collection;
10 use Symfony\Bridge\Doctrine\Validator\Constraints\UniqueEntity;
11 use Doctrine\ORM\Mapping as ORM;
12 use Doctrine\ORM\Mapping\EntityListeners;
13 use Symfony\Component\Security\Core\User\PasswordAuthenticatedUserInterface;
14 use Symfony\Component\Security\Core\User\UserInterface;
15
16 #[ORM\Entity(repositoryClass: UserRepository::class)]
17 #[ORM\EntityListeners(['App\EventListener\UserListener'])]
18 #[ORM\Table(name: 'user')]
19 #[UniqueEntity(fields: ['username'], message: 'Cet identifiant existe déjà.')]
20 #[UniqueEntity(fields: ['email'], message: 'Cette adresse email est déjà utilisée.')]
21 class User implements UserInterface, PasswordAuthenticatedUserInterface
22 {
23     #[ORM\Id]
24     #[ORM\GeneratedValue]
25     #[ORM\Column]
26     private ?int $id = null;
27
28     #[ORM\Column(length: 255, unique: true)]
29     private ?string $email = null;
30
```

Extrait du fichier user.php



Extrait du dossier migrations

* À chaque modification de l'entité: **php bin/console doctrine:migrations:diff** et **php bin/console doctrine:migrations:migrate ***

Création de la base de données

```
1 <?php
2
3 namespace App\DataFixtures;
4
5 use App\Entity\Theme;
6 use App\Entity\Cursus;
7 use App\Entity\Lesson;
8 use App\Entity\User;
9 use Doctrine\Bundle\FixturesBundle\Fixture;
10 use Doctrine\Persistence\ObjectManager;
11 use Symfony\Component\PasswordHasher\Hasher\UserPasswordHasherInterface;
12
13 class AppFixtures extends Fixture
14 {
15     private UserPasswordHasherInterface $passwordHasher;
16
17     public function __construct(UserPasswordHasherInterface $passwordHasher)
18     {
19         $this->passwordHasher = $passwordHasher;
20     }
21
22     public function load(ObjectManager $manager): void
23     {
24         // 1. Themes with associated metadata and curricula
25         $themesMeta = [
26             'Musique' => [
27                 'text' => 'Apprenez à maîtriser un instrument grâce à nos cours experts.',
28                 'image' => 'music.webp',
29                 'icon' => 'fa-music',
30                 'cursus' => [
31                     ['Cursus d'initiation à la guitare', 50, [
32                         ['Découverte de l'instrument', 26],
33                         ['Les accords et les gammes', 26],
34                     ]],
35                     ['Cursus d'initiation au piano', 50, [
36                         ['Découverte de l'instrument', 26],
37                         ['Les accords et les gammes', 26],
38                     ]],
39                 ],
40             ],
```

Extrait du fichier AppFixtures.php

```
PS C:\Users\tyson\OneDrive\Bureau\knowledge-learning> php bin/phpunit tests/Controller/SecurityControllerTest.php
PHPUnit 9.6.21 by Sebastian Bergmann and contributors.

Testing App\Tests\Controller\SecurityControllerTest
..
2 / 2 (100%)

Time: 00:00.907, Memory: 34.00 MB

OK (2 tests, 5 assertions)
PS C:\Users\tyson\OneDrive\Bureau\knowledge-learning> php bin/phpunit tests/Controller/StudentControllerTest.php
PHPUnit 9.6.21 by Sebastian Bergmann and contributors.

Testing App\Tests\Controller\StudentControllerTest
.....
5 / 5 (100%)

Time: 00:00.797, Memory: 40.00 MB

OK (5 tests, 13 assertions)
```

Extrait d'une validation d'insertion de données de test (fixture)

* Pour insérer les données fixtures : **php bin/console doctrine:fixtures:load ***

Accès aux données

- Lecture (Read) via Repository
- Récupération des données filtrées
- Doctrine ORM

```
1 // 1) Retrieves all user commands
2 $orders = $orderRepository->findBy(['user' => $user]);
```

- Création (Create) via setters
- Hydratation via setters
- Insertion avec persist() + flush()

```
1 // Create users
2 $student = new User();
3 $student->setEmail('student@example.com')
4         ->setUsername('student')
5         ->setRoles(['ROLE_USER'])
6         ->setPassword($this->passwordHasher->hashPassword($student, 'password'))
7         ->setActivated(true);
8 $manager->persist($student);
9
10 $manager->flush();
```

- Suppression (Delete) d'un utilisateur
- Doctrine ORM
- remove() + flush()

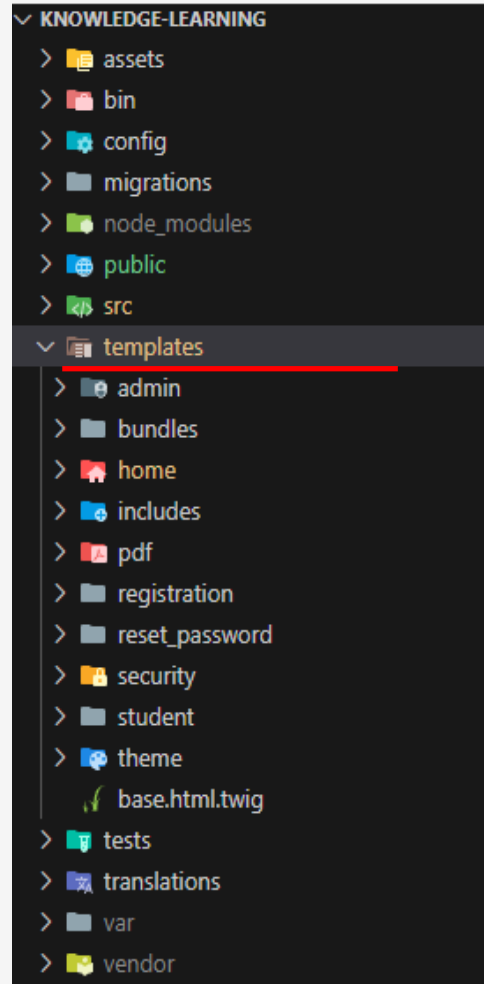
```
1 $em->remove($user);
2 $em->flush();
```

* Les entités Doctrine font office de modèles. Elles sont automatiquement liées à la base via Doctrine ORM, ce qui permet les opérations CRUD sans SQL manuel *

Réalisation du front-end

Partie dynamique générée via Twig

- Templates organisés par sections (admin, student, etc.) dans templates/
- Application structurée avec des templates Twig
- Routage géré par les annotations dans les contrôleurs Symfony



Structure du projet

```
1
2 <title>
3     {% block title %}Knowledge-Learning
4     {% endblock %}
5 </title>
6
7
8 {% block stylesheets %}
9     {{ encore_entry_link_tags('style') }}
10 {% endblock %}
11
12 <main>
13     {% block body %}{% endblock %}
14 </main>
15
16 {% block javascripts %}
17     {{ encore_entry_script_tags('app') }}
18 {% endblock %}
```

Extrait d'un template

Réalisation du front-end

Le front-end repose sur Twig pour générer des pages dynamiques, stylées avec SCSS modulaire.

```
1 {% for theme in themes %}
2     <div class="card2">
3         
4         <h3 class="{{ theme.title|lower|replace({' ': '-'}) }}-title">
5             <i class="fas {{ theme.icon }}"></i>
6             {{ theme.title }}
7         </h3>
8         <p>{{ theme.text }}</p>
9         <a href="{{ path('visitor_themes_filter', { 'filter': theme.id }) }}" class="btn-custom">Je Découvre</a>
10    </div>
11 {% endfor %}
```

Extrait de la page **Home.html.twig**

```
1 @import "../admin.scss"; /* Import scss admin */
2 @import "../mobile.scss"; /* Import scss mobile */
3 @import "../tablet.scss"; /* Import scss tablet */
4
5 /* Colors Template*/
6 $color-1: #f1f8fc;
7 $color-2: #0074c7;
8 $color-3: #00497c;
9 $color-4: #384050;
10 $color-5: #cd2c2e;
11 $color-6: #82b864;
12 $color-7: #fff;
13
14 html,
15 body {
16     margin: 0;
17     padding: 0;
18     font-family: "Comic Sans MS", cursive, sans-serif;
19     height: 100%;
20     background-color: $color-1;
21 }
22
23 main {
24     padding: 0;
25     width: 100%;
26 }
27
28 header {
29     background-color: $color-7;
30     padding: 10px 0;
31     border-bottom: 4px solid $color-3;
32 }
```

Extrait du fichier **assets/css/style.scss**

Réalisation du front-end

La fonction `{% extends %}` permet à un fichier Twig d'hériter d'un autre fichier, généralement un template de base.

```
KNOWLEDGE-LEARNING
├── assets
├── bin
├── config
├── migrations
├── node_modules
├── public
├── src
└── templates
    ├── admin
    ├── bundles
    ├── home
    ├── includes
    ├── pdf
    ├── registration
    ├── reset_password
    ├── security
    ├── student
    └── theme
        └── base.html.twig
```

```
1 <!DOCTYPE html>
2 <html lang="fr">
3   <head>
4     <meta charset="UTF-8">
5     <meta name="viewport" content="width=device-width, initial-scale=1.0">
6     <meta name="description" content="La boutique en ligne dédiée à la mode avec caractère">
7     <link rel="icon" type="image/x-icon" href="{{ asset('favicon.ico') }}">
8     <meta name="robots" content="index, follow">
9     <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.5.1/c
10
11   <title>
12     {% block title %}Knowledge-Learning
13     {% endblock %}
14   </title>
15
16   {% block stylesheets %}
17     {{ encore_entry_link_tags('style') }}
18   {% endblock %}
19 </head>
20
21 <body>
22   <main>
23     {% block body %}{% endblock %}
24   </main>
25
26   {% block javascripts %}
27     {{ encore_entry_script_tags('app') }}
28   {% endblock %}
29
30 </body>
31
32 </html>
```

Extrait de la page **base.html.twig**

```
1 {% extends 'base.html.twig' %}
2
3 {% block title %}Accueil
4 {% endblock %}
5
6 {% block body %}
7   <!-- Inclusion of header -->
8   {% include 'includes/header.html.twig' %}
9
10   <section class="heading">
11     <p class="heading_title_description">
12       Avec
13       <strong>Knowledge</strong>, formez-
14     </p>
15     <a href="{{ path('app_register') }}" cl
16       <i class="fas fa-star"></i>
17       Je me forme dès maintenant
18     </a>
19   </section>
```

Extrait de la page **home.html.twig**

Réalisation du back-end



Exemple du fichier: src/Controller/StudentCoursesController.php

```
1  foreach ($orders as $order) {
2      // only processes pack orders
3      if (!$c = $order->getCursus()) {
4          continue;
5      }
6
7      $totalLessons = count($c->getLessons());
8
9      // count lessons validated for this course
10     $validatedCount = 0;
11     foreach ($allValidations as $val) {
12         if ($val->getLesson()->getCursus()->getId() === $c->getId()) {
13             $validatedCount++;
14         }
15     }
16
17     $pct = $totalLessons > 0
18         ? (int) round($validatedCount / $totalLessons * 100)
19         : 0;
20
21     $progress[$c->getId()] = $pct;
22 }
```

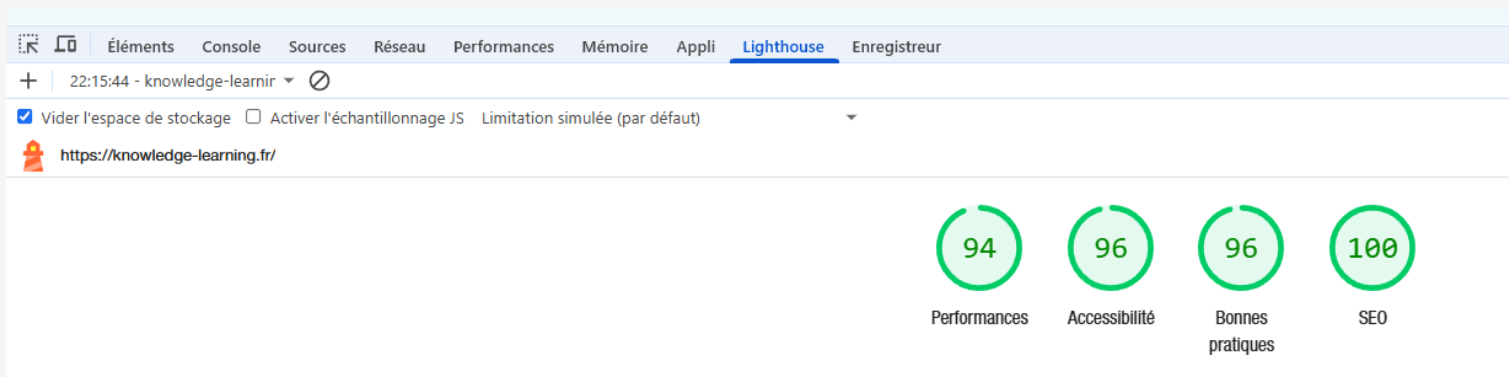

Sécurité

FAIT	À FAIRE
<ul style="list-style-type: none">➤ Hachage des mots de passe➤ Protection CSRF➤ Validation côté client (JavaScript)➤ Validation des données côté serveur➤ Filtrage des entrées utilisateurs➤ Gestion des rôles et des accès➤ Encodage automatique contre XSS → via Twig	<ul style="list-style-type: none">➤ Limitation de requêtes (Rate limiting)➤ Politique de sécurité HTTP stricte (CSP, HSTS)➤ Vérification + nettoyage de fichiers uploadés➤ Logs d'activités utilisateurs / tentatives de connexion

Accessibilité & Performance

Les optimisations réalisées visent à améliorer l'accessibilité, les performances et la conformité aux standards web.

Ces résultats assurent une meilleure expérience utilisateur, une conformité aux standards d'accessibilité et un impact SEO renforcé.

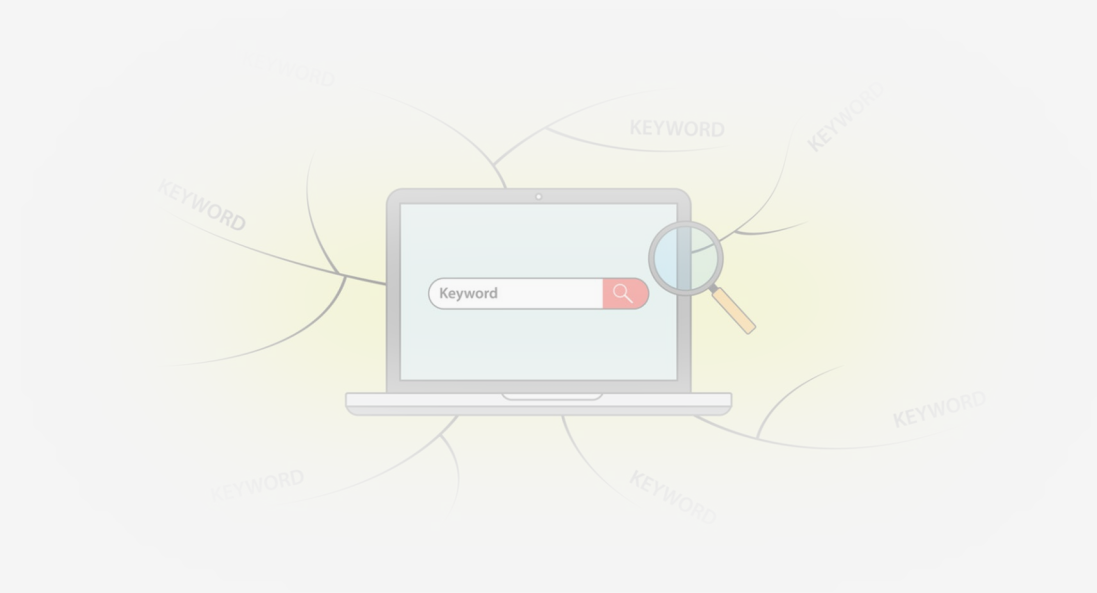


Résultats Lighthouse (Performances & Accessibilité)



Validation du code CSS selon la norme W3C (niveau 3 + SVG)

Exemple de recherche



Présentation de plusieurs recherches de sécurité sur Symfony

Pour les bonnes pratiques dans Symfony :

Symfony security best practices


All Videos Short videos Images Forums Web News More ▾

Past year ▾ All results ▾ Advanced Search Clear

Comprehensive Guide to Symfony Security Best Practices

- Exploiting the Symfony Profiler. ...
- Securing Access Control on Symfony Applications. ...
- Protecting Against Cross-Site Scripting (XSS) Vulnerabilities. ...
- Countering SSTI (Server-Side Template Injection) Vulnerabilities. ...
- Preventing Host Header Poisoning Attacks.

[More items...](#) • 16 Jan 2025

 Vaadata
<https://www.vaadata.com> › blog › symfony-security-best...
Symfony Security Best Practices, Vulnerabilities and Attacks

Pour identifier les failles liées à l'authentification et aux autorisations

Symfony authentication and authorization vulnerabilities


All Videos Images Short videos Forums Web News More ▾

Past year ▾ All results ▾ Advanced Search Clear

Why Symfony APIs Are Vulnerable

- Exposing sensitive routes.
- Improper input validation.
- CSRF and CORS misconfigurations.
- Broken authentication logic.
- Insecure serialization/deserialization.

4 days ago



 DEV Community
<https://dev.to> › pentest_testing_corp › api-vulnerabilite...
API Vulnerabilities in Symfony: Real-World Examples


Pour montrer comment Symfony gère les risques listés par OWASP (Injection, XSS, CSRF)


Symfony OWASP top 10 protection

All Videos Images Short videos Forums Web News More ▾

Past year ▾ All results ▾ Advanced Search Clear

 Medium · Tihomir Manushev
1 like
Fixing OWASP API1:2023 — Broken Object Level ...
One of the most common vulnerabilities in APIs is Broken Object Level Authorization (API1:2023) from the OWASP API Security Top 10, where attackers can access ...

 Axopen
<https://www.axopen.com> › Blog
OWASP : Top 10 des failles API en 2024
8 Nov 2024 — Dans cet article, nous allons explorer les 10 failles API les plus communes identifiées par l'OWASP, en analysant comment elles fonctionnent, quelles en ...

 brosseau.ovh
<https://cours.brosseau.ovh> › owasp · [Translate this page](#)
La méthode OWASP - Cours
10 Aug 2024 — Le nouveau TOP 10 est très intéressant, car il met en lumière le croisement entre les failles et les risques. Mais il est plus complexe à mémoriser. Il est donc ...

Présentation d'une recherche sur la sécurité web

Classement:

1. Toujours filtrer les entrées avec Validator
2. Protéger les formulaires avec des tokens CSRF
3. Restreindre les accès par rôles (IsGranted)
4. Éviter d'exposer les messages d'erreur sensibles

Why Symfony?

On this page, you'll learn about the motivations for choosing Symfony and how it compares to other PHP frameworks like Laravel or Slim.

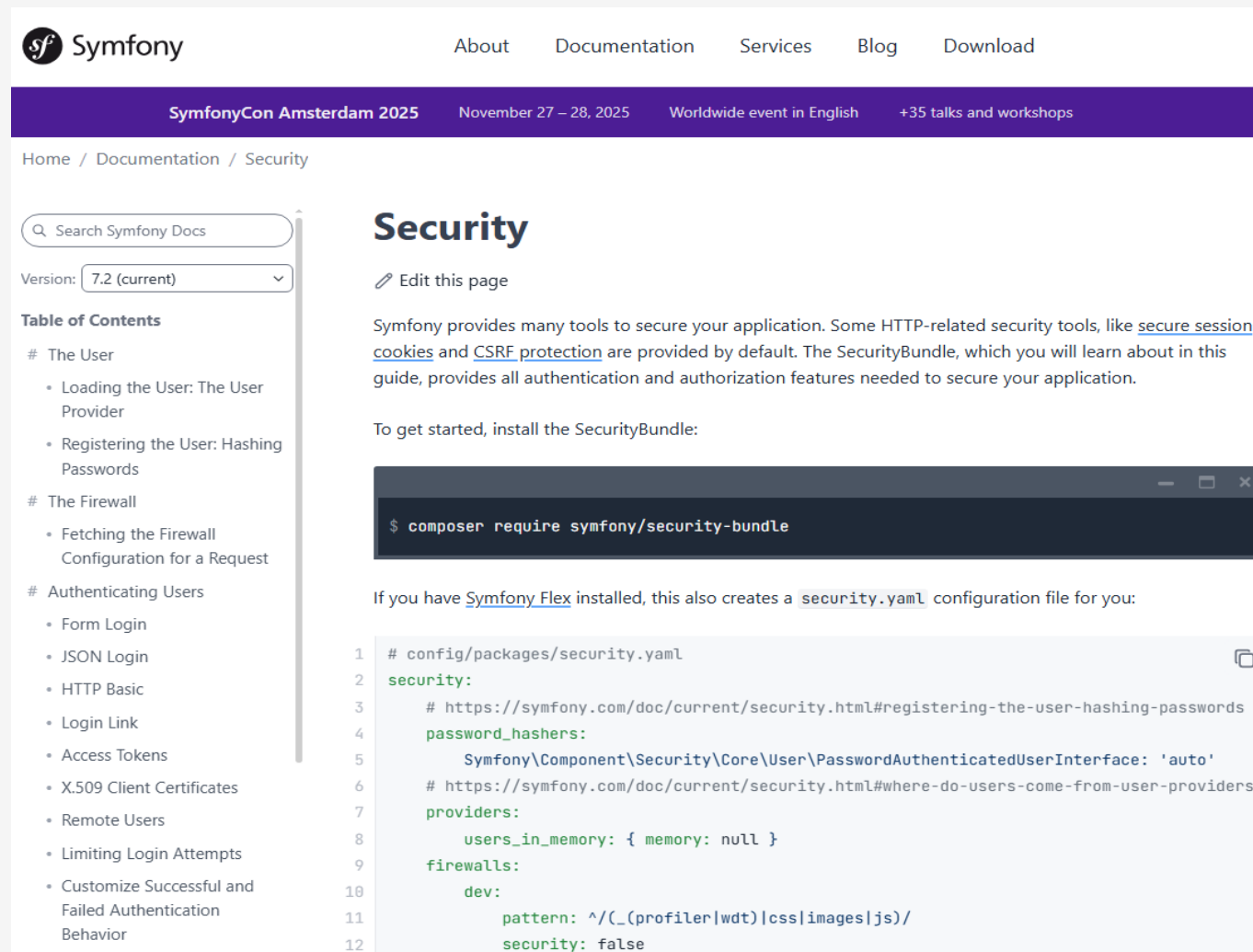
Symfony is designed for large-scale and complex web applications. Its modular component system, solid architecture, and extensive documentation make it a reliable choice for long-term projects.

Security, performance, and scalability are central to Symfony's ecosystem. With built-in tools such as routing, service containers, security firewalls, and Doctrine ORM, developers gain fine control over every aspect of the application.

Symfony is widely adopted in enterprise environments and follows best practices by default, making it ideal for professional-grade applications.

TLDR

Symfony provides a robust, scalable, and secure framework with a strong community and long-term support. Its flexibility and adherence to standards make it a trusted choice for developing maintainable and enterprise-level web applications.



The screenshot shows the Symfony documentation page for Security. The page has a purple header with navigation links: About, Documentation, Services, Blog, and Download. Below the header, there's a banner for 'SymfonyCon Amsterdam 2025' with dates 'November 27 - 28, 2025' and details 'Worldwide event in English' and '+35 talks and workshops'. The breadcrumb trail is 'Home / Documentation / Security'. A search bar and a version dropdown (set to '7.2 (current)') are present. A 'Table of Contents' sidebar lists sections like 'The User', 'The Firewall', and 'Authenticating Users'. The main content area is titled 'Security' and includes an 'Edit this page' link. The text explains that Symfony provides tools for security, mentioning 'secure session cookies' and 'CSRF protection'. It instructs users to install 'SecurityBundle' using the command: `$ composer require symfony/security-bundle`. It also notes that if 'Symfony Flex' is installed, a 'security.yaml' file is created. A code block shows the configuration for 'security.yaml', including password hashers, providers, and firewalls. The 'security' key is highlighted with a red underline.

Security

Edit this page

Symfony provides many tools to secure your application. Some HTTP-related security tools, like [secure session cookies](#) and [CSRF protection](#) are provided by default. The SecurityBundle, which you will learn about in this guide, provides all authentication and authorization features needed to secure your application.

To get started, install the SecurityBundle:

```
$ composer require symfony/security-bundle
```

If you have [Symfony Flex](#) installed, this also creates a `security.yaml` configuration file for you:

```
1 # config/packages/security.yaml
2 security:
3     # https://symfony.com/doc/current/security.html#registering-the-user-hashing-passwords
4     password_hashers:
5         Symfony\Component\Security\Core\User\PasswordAuthenticatedUserInterface: 'auto'
6     # https://symfony.com/doc/current/security.html#where-do-users-come-from-user-providers
7     providers:
8         users_in_memory: { memory: null }
9     firewalls:
10         dev:
11             pattern: ^/(_(profiler|wdt)|css|images|js)/
12             security: false
```

Exemple d'une page officielle: <https://symfony.com/doc/current/security.html>

Conclusion

- Fierté du travail accompli en autonomie
- Renforcement des compétences en développement web
- Apprentissage approfondi de Symfony et des bonnes pratiques
- Envie de continuer à faire évoluer le projet (ajout de fonctionnalités, tests, sécurité...)

À vos questions !