

Titre professionnel Développeur web et web mobile

Soutenance

NADEZHDA THOUVENIN - TodoList

Qui suis-je ?

- Expérience professionnelle en France (2023 - 2025)
Auto-formation en développement web (en parallèle)
- Formation DWWM - 2025 / 2026

SOMMAIRE

I.
Présentation
du projet

II.
Conception

III.
Développement

IV.
Réalisations
personnelles

V.
Démonstration
du projet

VI.
Exemple de
recherche

VII.
Conclusion

TodoList - Genèse

CONSTAT

Il est difficile d'organiser ses tâches lorsqu'elles sont dispersées, sans outil simple permettant une visualisation claire et la gestion des priorités.

AUDIENCE

- Utilisateurs individuels
- Étudiants
- Actifs souhaitant organiser leurs tâches et priorités
- Personnes qui ont besoin de rappels pour ne pas oublier leurs tâches

Organisation du travail

- Projet réalisé dans le cadre d'une mise en situation professionnelle
- **Durée** : 4 semaines
- Organisation du travail en autonomie
- Daily meetings avec le formateur
- Journées de formation de 9h à 17h



- User stories
 - Wireframes
 - Réflexion sur le MVP
 - Etc...
- Corrections de bugs
 - Ajout de petites fonctionnalités

CONCEPTION



User stories

2 types d'utilisateurs

:

- Visiteur
- Utilisateur connecté

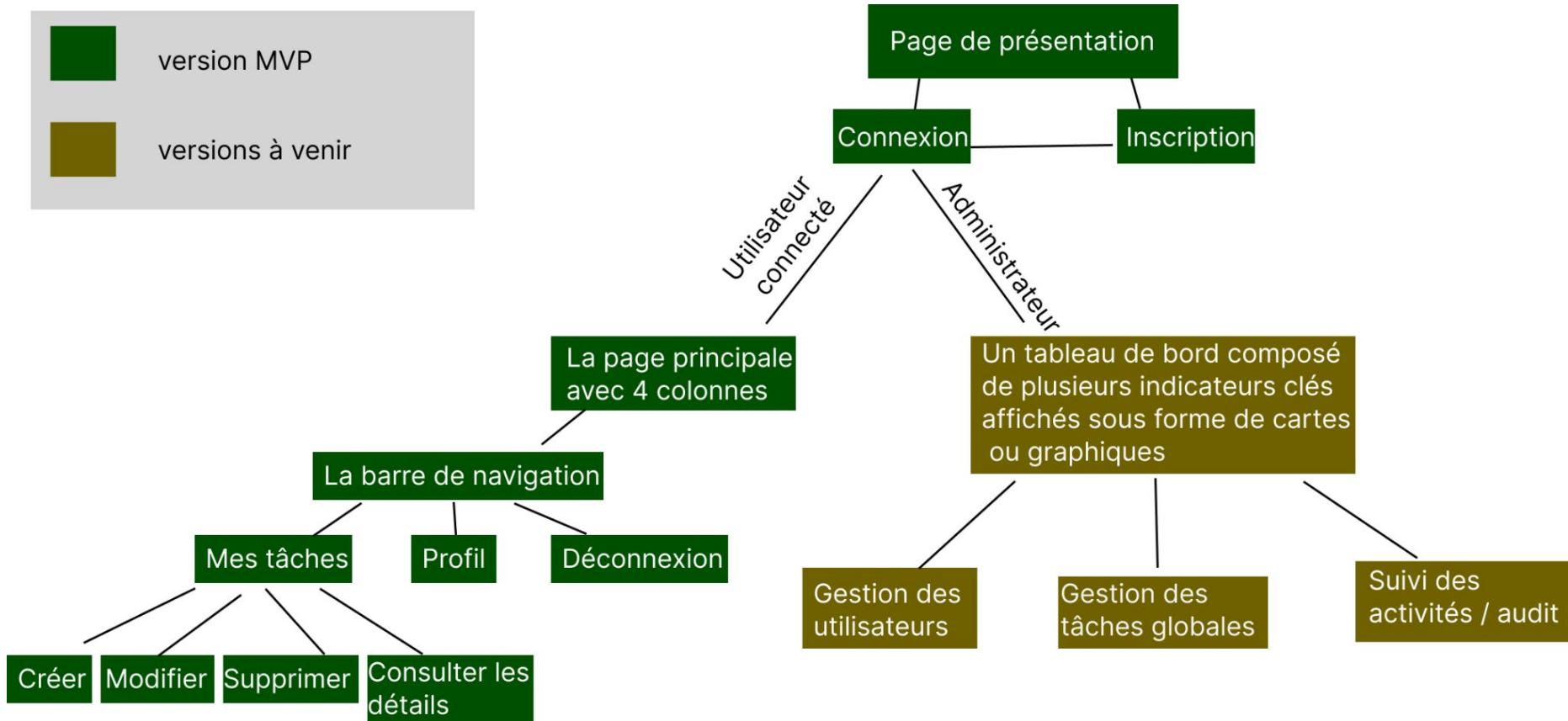
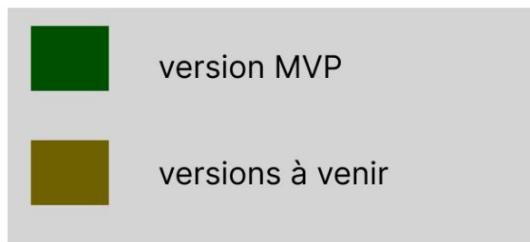
En vert : MVP

En jaune : évolutions futures (UX/UI)

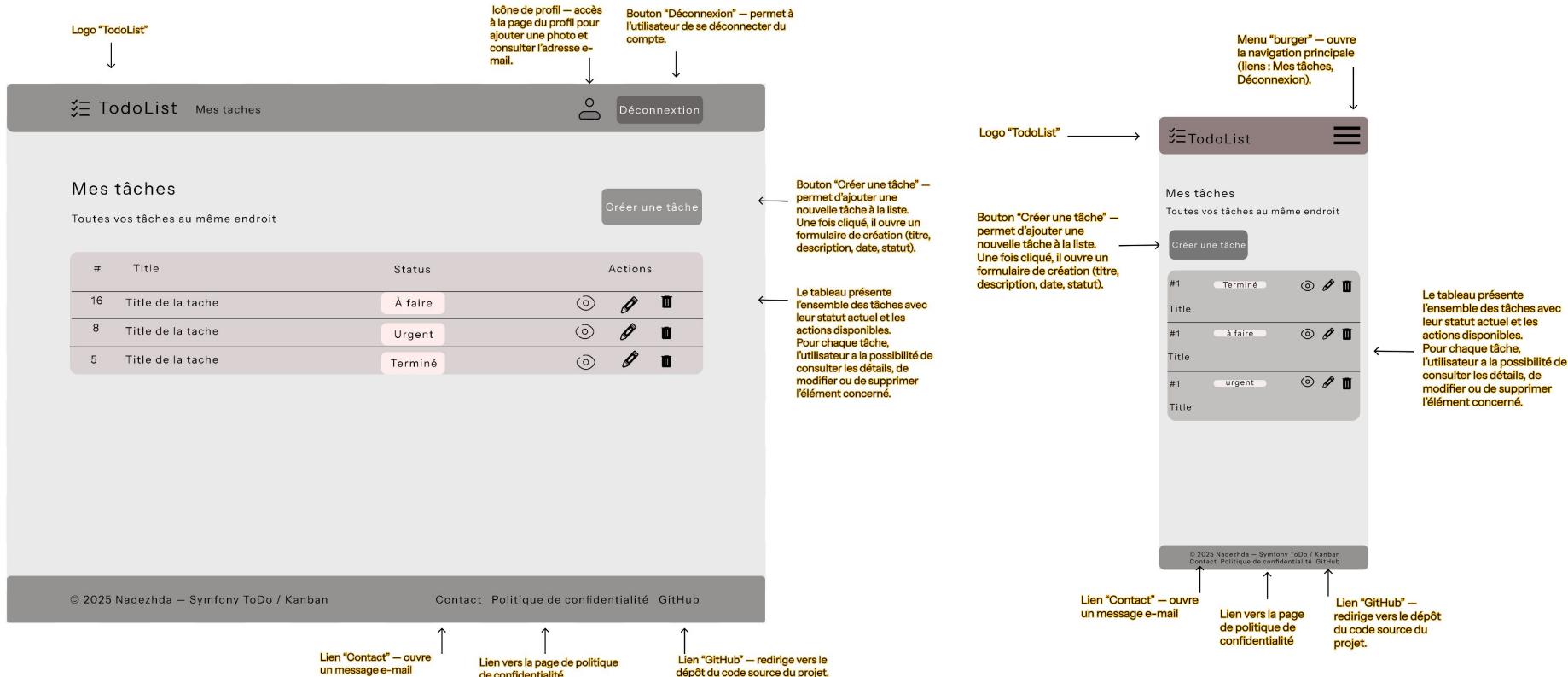
En tant que...	Je souhaite...	Afin de...
Visiteur	Me créer un compte	Accéder à mon tableau de tâches
Utilisateur connecté	Me connecter à l'application	Retrouver mes tâches et continuer mon organisation
Utilisateur connecté	Créer une tâche	Noter ce que je dois faire
Utilisateur connecté	Modifier une tâche	Corriger ou mettre à jour mes informations
Utilisateur connecté	Supprimer une tâche	Enlever ce qui n'est plus nécessaire
Utilisateur connecté	Déplacer mes tâches dans le tableau (drag & drop)	Organiser visuellement mon travail
Utilisateur connecté	Recevoir un e-mail quand une tâche est en retard	Être informé même si je ne suis pas connecté
Utilisateur connecté	Ajouter un avatar à mon profil	Personnaliser mon espace
Utilisateur connecté	Supprimer mon compte	Contrôler mes données personnelles (RGPD)

En tant que...	Je souhaite...	Afin de...
	Activer un mode sombre	Améliorer mon confort visuel
Utilisateur connecté	Filtrer ou trier mes tâches	Retrouver rapidement ce dont j'ai besoin
Nouvel utilisateur connecté	Suivre une aide interactive	Comprendre rapidement comment utiliser l'app
Utilisateur connecté	Bénéficier d'animations plus fluides lors du glisser-déposer	Rendre l'organisation de mes tâches plus agréable

ARBORESCENCE



WIREFRAMES – validation des parcours utilisateurs



Maquettes

The wireframe shows a mobile application interface for managing tasks. At the top, there is a header bar with the logo "TodoList" and the text "Mes tâches". On the right side of the header is a user icon and the text "Déconnexion". Below the header, the title "Mes tâches" is displayed, followed by the subtitle "Toutes vos tâches au même endroit". A blue button labeled "Créer une tâche" is located in the top right corner. The main content area is a table with three columns: "#", "Title", and "Actions". The "#" column contains task identifiers (16, 8, 5). The "Title" column lists "Title de la tâche" for each task. The "Status" column uses color-coded boxes to indicate task status: blue for "À faire", red for "Urgent", and green for "Terminé". Each row has a set of three icons in the "Actions" column: a circle with a dot, a pencil, and a trash can.

#	Title	Status	Actions
16	Title de la tâche	À faire	(circle) (pencil) (trash)
8	Title de la tâche	Urgent	(circle) (pencil) (trash)
5	Title de la tâche	Terminé	(circle) (pencil) (trash)

© 2025 Nadezhda — Symfony ToDo / Kanban
Contact Politique de confidentialité GitHub

This wireframe shows a mobile application interface for managing tasks. At the top, there is a header bar with the logo "TodoList" and a menu icon. Below the header, the title "Mes tâches" is displayed, followed by the subtitle "Toutes vos tâches au même endroit". A blue button labeled "Créer une tâche" is located in the top right corner. The main content area displays three task cards. Each card includes a task identifier (#1), a status indicator (green for TERMINÉ, blue for A FAIRE, red for URGENT), and a set of three icons: a circle with a dot, a pencil, and a trash can. The cards also have a "Title" placeholder below them. At the bottom of the screen, there is a footer with the text "© 2025 Nadezhda — Symfony ToDo / Kanban Contact Politique de confidentialité GitHub".

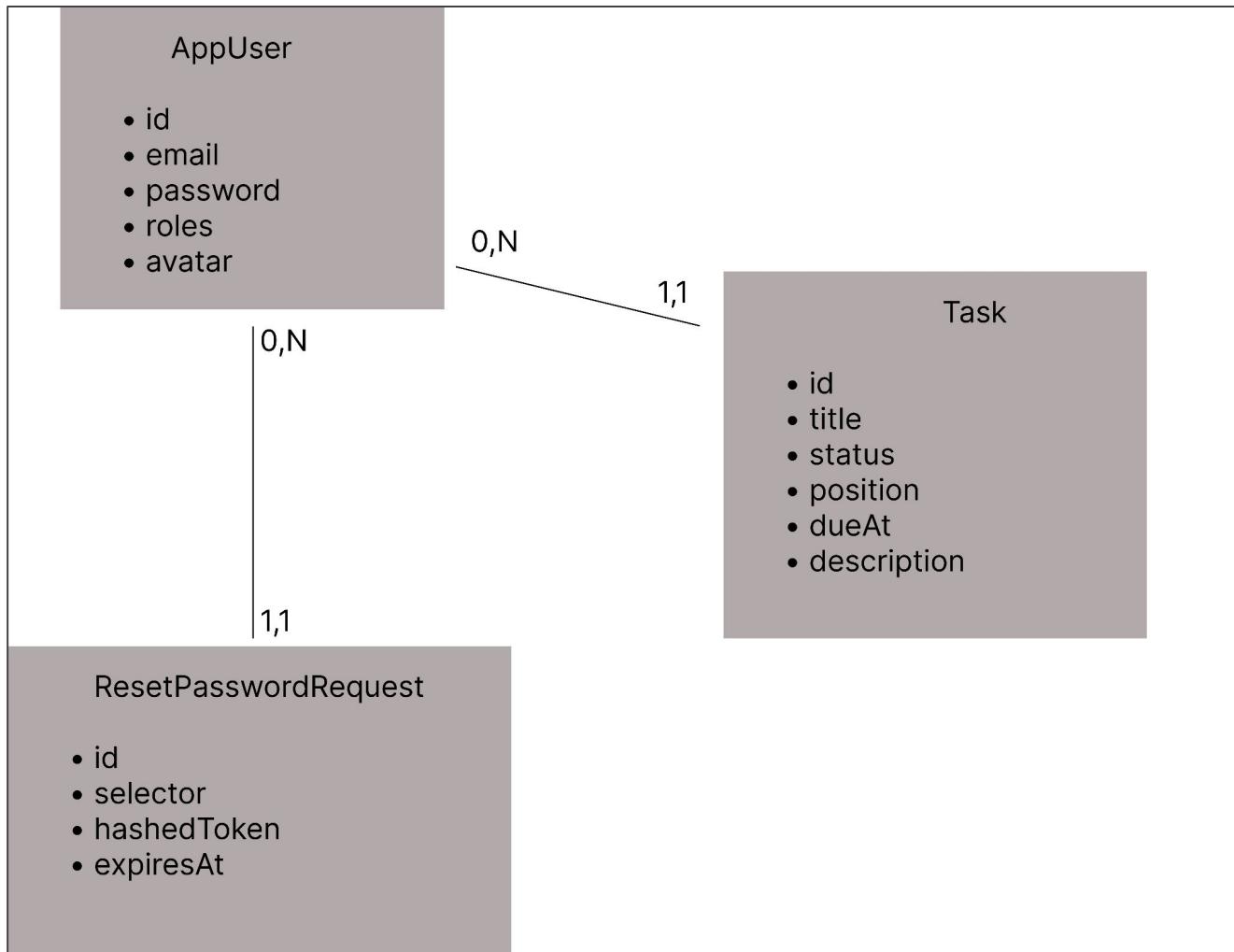
#1 TERMINÉ (circle) (pencil) (trash)
Title

#1 A FAIRE (circle) (pencil) (trash)
Title

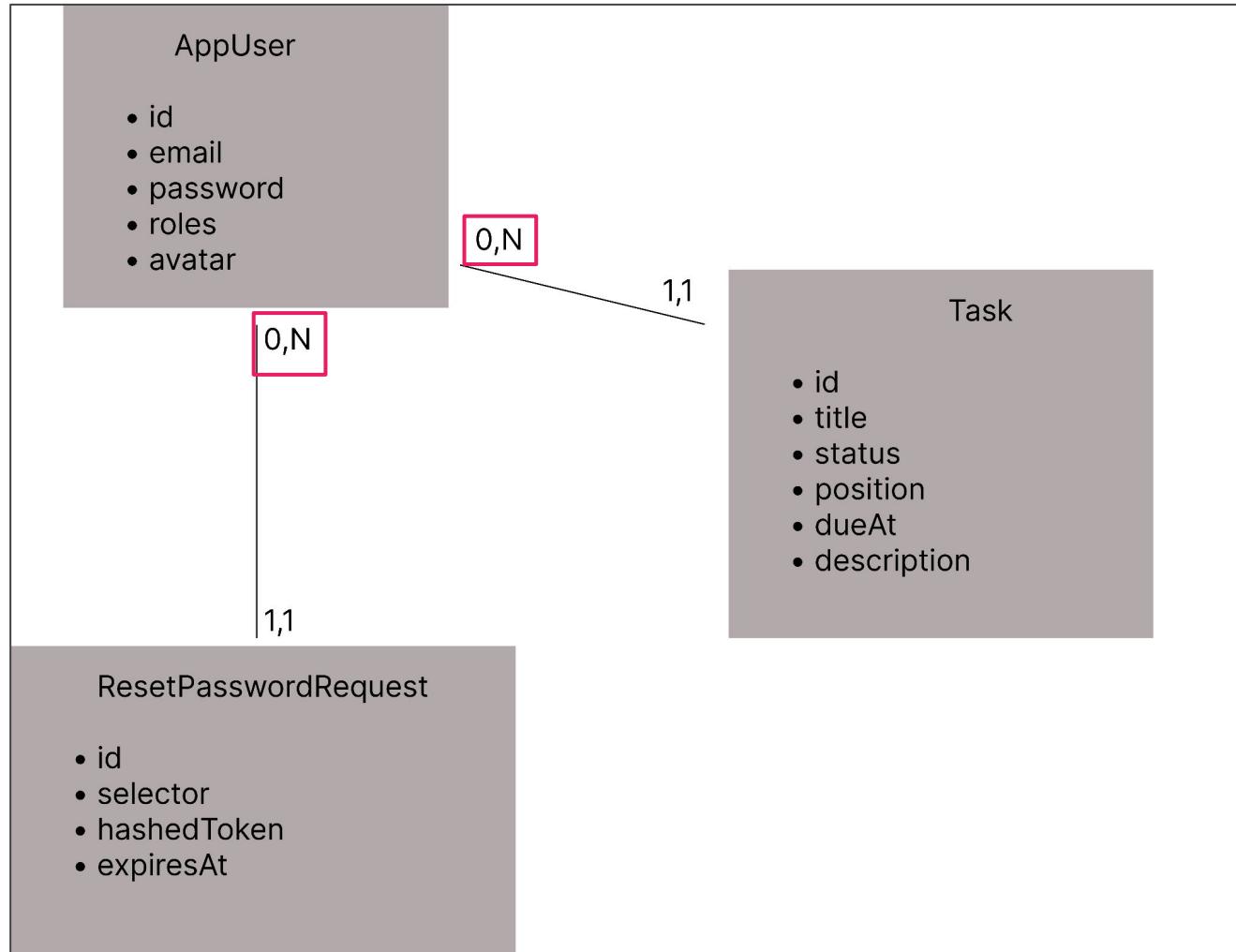
#1 URGENT (circle) (pencil) (trash)
Title

© 2025 Nadezhda — Symfony ToDo / Kanban
Contact Politique de confidentialité GitHub

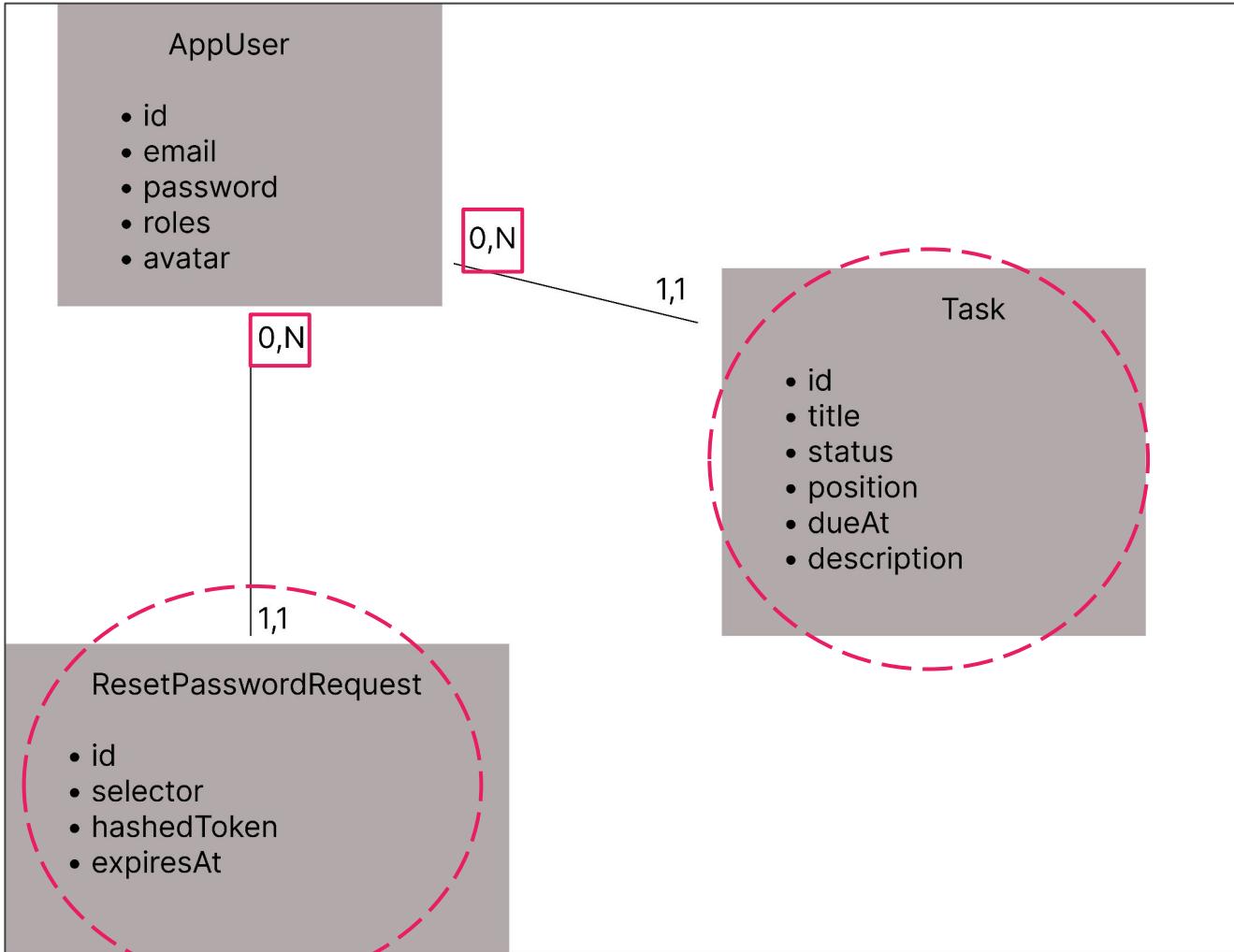
MCD



MCD



MCD



MLD

APP_USER (id, email, password, roles, avatar)

TASK (id, title, status, position, due_at, description, #owner_id)

RESET_PASSWORD_REQUEST (id, selector, hashed_token, expires_at, #user_id)

MLD

APP_USER (id, email, password, roles, avatar)

TASK (id, title, status, position, due_at, description, #owner_id)

RESET_PASSWORD_REQUEST (id, selector, hashed_token, expires_at, #user_id)

DICTIONNAIRE DES DONNÉES

Table app_user

FIELD	TYPE	DETAILS	DESCRIPTION
id	int	PK, NOT NULL, GENERATED AS IDENTITY	Identifiant unique de l'utilisateur
email	varchar(180)	UNIQUE, NOT NULL	Adresse e-mail de l'utilisateur
roles	json	NOT NULL	Rôles stockés au format JSON
password	varchar(255)	NOT NULL	Mot de passe chiffré
avatar	varchar(255)	DEFAULT NULL	Nom de fichier de l'avatar téléchargé

Table task

FIELD	TYPE	DETAILS	DESCRIPTION
id	int	PK, NOT NULL, GENERATED AS IDENTITY	Identifiant unique de la tâche
title	varchar(120)	NOT NULL	Titre de la tâche
status	varchar(20)	NOT NULL	Statut de la tâche (todo, doing, done, urgent)
position	int	NOT NULL, DEFAULT 0	Position d'affichage (ordre dans la colonne)
due_at	TIMESTAMP WITHOUT TIME ZONE	DEFAULT NULL	Date d'échéance de la tâche
description	text	DEFAULT NULL	Description optionnelle de la tâche
owner_id	int	NOT NULL, FK → app_user(id)	Référence à l'utilisateur propriétaire

Table reset_password_request

FIELD	TYPE	DETAILS	DESCRIPTION
id	int	PK, NOT NULL, GENERATED AS IDENTITY	Identifiant unique de la demande
user_id	int	NOT NULL, FK → app_user(id)	Utilisateur ayant demandé la réinitialisation
selector	varchar(20)	NOT NULL	Identifiant public du lien de réinitialisation
hashed_token	varchar(100)	NOT NULL	Jeton privé haché (stocké côté serveur)
requested_at	TIMESTAMP WITHOUT TIME ZONE	NOT NULL	Date et heure de la demande
expires_at	TIMESTAMP WITHOUT TIME ZONE	NOT NULL	Date d'expiration du lien

STACK TECHNIQUE

Front

- Twig
- Bootstrap 5
- CSS
- JavaScript

Back

- Symfony
- Doctrine ORM
- PostgreSQL
- Symfony Mailer
- ResetPasswordBundle

DEVELOPER EXPERIENCE

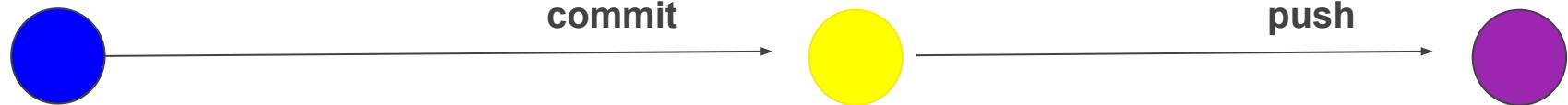
- Git / GitHub
- VS Code

Gestion du code – Git / GitHub

Travail local
(développement
des fonctionnalités)

Git – branche main

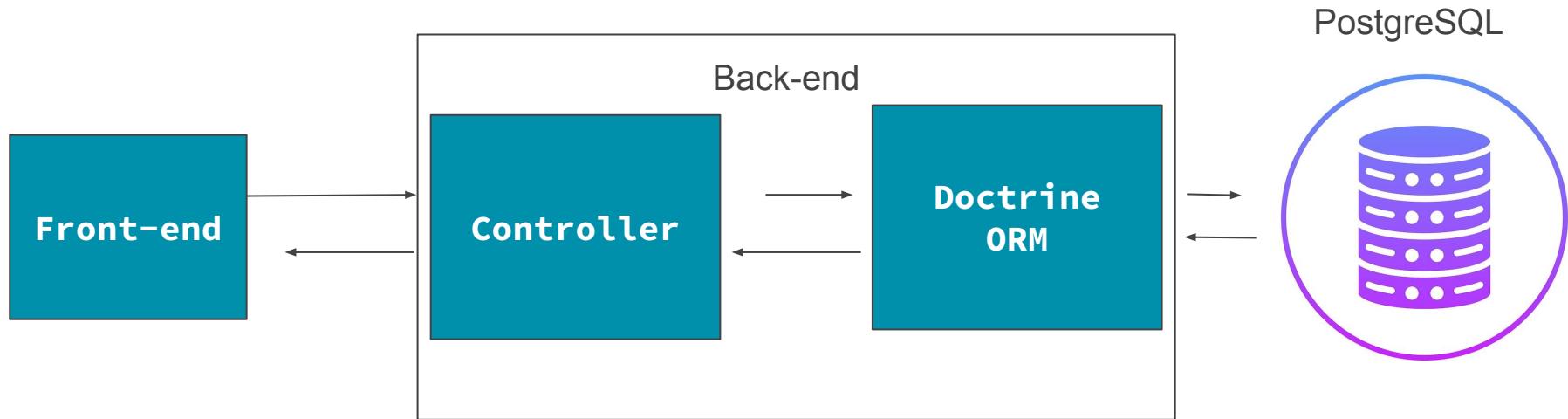
GitHub –
branche main



DEVELOPPEMENT



ARCHITECTURE – MVC (Symfony)



Création et évolution de la base de données

```
1 doctrine:  
2     dbal:  
3         url: "%env(resolve:DATABASE_URL)%"  
4
```

Extrait du fichier doctrine.yaml

```
9 #[ORM\Entity(repositoryClass: TaskRepository::class)]  
10 #[ORM\Table(name: 'task')]  
11 class Task  
12 {  
13     #[ORM\Id]  
14     #[ORM\GeneratedValue]  
15     #[ORM\Column]  
16     private ?int $id = null;  
17  
18     #[ORM\Column(length: 120)]  
19     private ?string $title = null;  
20  
21     #[ORM\Column(length: 20)]  
22     private ?string $status = null;  
23  
24     #[ORM\Column(type: 'integer', options: ['default' => 0])]  
25     private int $position = 0;  
26  
27     #[ORM\Column(type: 'datetime_immutable', nullable: true)]  
28     private ?\DateTimeImmutable $dueAt = null;  
29  
30     #[ORM\ManyToMany(targetEntity: AppUser::class)]  
31     #[ORM\JoinColumn(nullable: false, onDelete: 'CASCADE')]  
32     private ?AppUser $owner = null;  
33  
34     #[ORM\Column(type: Types::TEXT, nullable: true)]  
35     private ?string $description = null;
```

migrations
Version20250901093203.php
Version20250903083715.php
Version20250905130550.php
Version20250917134518.php
Version20250918140107.php
Version20250919133508.php
Version20250919151950.php
Version20251002145350.php
Version20251113144836.php
Version20251113172755.php

Création de la table Task avec ses champs.

```
$this->addSql('CREATE TABLE task (id SERIAL NOT NULL, title VARCHAR(120) NOT NULL,  
| | | | status VARCHAR(20) NOT NULL, PRIMARY KEY(id))');
```

Ajout de owner_id sur task

```
$this->addSql('ALTER TABLE task ADD owner_id INT NOT NULL');
```

Clé étrangère vers app_user

```
$this->addSql('ALTER TABLE task ADD CONSTRAINT FK_527EDB257E3C61F9 FOREIGN KEY (owner_id)  
| | | | REFERENCES app_user (id) NOT DEFERRABLE INITIALLY IMMEDIATE');
```

Index pour améliorer les performances

```
$this->addSql('CREATE INDEX IDX_527EDB257E3C61F9 ON task (owner_id)');
```

php bin/console make:migration

php bin/console doctrine:migrations:migrate

Accès aux données

Lire (SELECT) ➔ Repository

Créer (INSERT) ➔ EntityManager (persist + flush)

Modifier (UPDATE) ➔ EntityManager (flush)

Supprimer (DELETE) ➔ EntityManager (remove + flush)

Le repository sert à lire les données.

L'EntityManager sert à créer, modifier et supprimer les données.

Lecture des tâches (READ)

```
$tasks = $taskRepository->findBy(['owner' => $this->getUser()], ['id' => 'ASC']);
```

Association de la tâche à l'utilisateur

```
$user = $this->getUser();  
$task->setOwner($user);
```

Création d'une tâche (CREATE)

```
$entityManager->persist($task);  
$entityManager->flush();
```

Suppression d'une tâche (DELETE)

```
$entityManager->remove($task);  
$entityManager->flush();
```

}

Réalisation du Back-end



```
firewalls:
    main:
        lazy: true
        provider: app_user_provider
        form_login:
            login_path: app_login
            check_path: app_login
            enable_csrf: true

            default_target_path: app_kanban
            username_parameter: email
            password_parameter: password

        logout:
            path: app_logout
            target: app_home
```

Le firewall vérifie la session et l'utilisateur connecté.

Extrait du fichier security.yaml

Après le firewall, la requête est traitée dans le contrôleur.

Le contrôleur gère la création, la validation et l'enregistrement de la tâche.

```
$task = new Task();           ← Création d'un nouvel objet Task
$form = $this->createForm(TaskType::class, $task);   ← Création du formulaire lié à l'objet Task
$form->handleRequest($request);    ← Récupération des données envoyées par le formulaire

if ($form->isSubmitted() && $form->isValid()) {    ← Vérification de l'envoi et validation des données

    /** @var \App\Entity\AppUser $user */

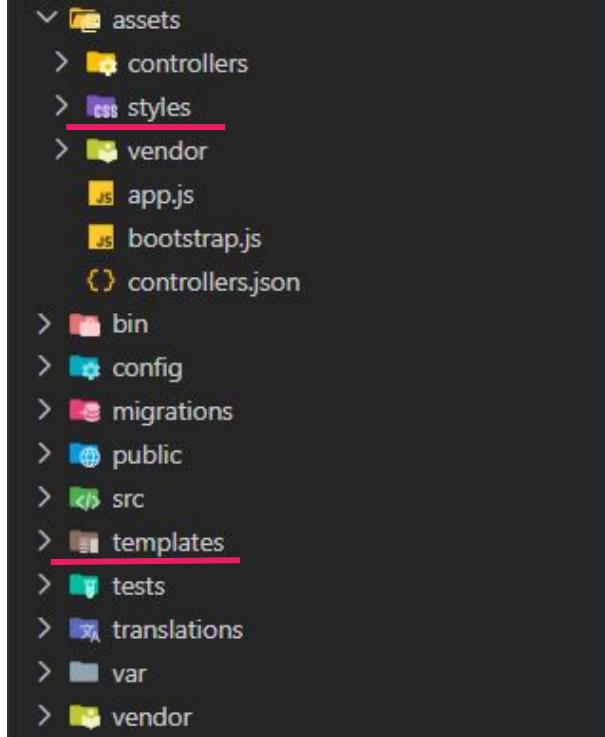
    $user = $this->getUser();
    $task->setOwner($user);    ← Association de la tâche à l'utilisateur connecté

    $entityManager->persist($task);
    $entityManager->flush();    ← Enregistrement de la tâche en base de données
```

Extrait du fichier TaskController.php

Réalisation du Front-end

partie statique



Structure du projet

- Application découpée en pages Twig
- Les routes sont gérées par Symfony

```
34 <section class="kanban kanban-glass shadow-lg rounded-4 p-3 p-md-2"
35   || | role="region" aria-label=" Tableau principal à quatre colonnes et des tâches à l'intérieur">
36
37   <div class="row flexnowrap g-3 overflow-auto kanban-columns">
38
39     {% for key in ['urgent','todo','doing','done'] %}
40       {% set items = columns[key] ?? [] %}
41
42       <div class="col-10 col-sm-8 col-md-6 col-lg-3">
43         <div class="kanban-column h-100 rounded-3 overflow-hidden"
44           || | role="region"
45           || | aria-labelledby="col-title-{{ key }}">
46
47           <div class="kanban-column-header d-flex align-items-center justify-content-between {{ HEADER_CLASS[key] }} px-3 py-2">
48             <span class="fw-semibold text-uppercase text-white small"
49               || | id="col-title-{{ key }}">{{ LABELS[key] }}</span>
50
51             <span id="count-{{ key }}"
52               || | class="badge text-bg-light text-dark"
53               || | aria-live="polite" aria-atomic="true">{{ items|length }}</span>
54           </div>
55
56           <div class="kanban-cards p-2" id="col-{{ key }}">
57             || | role="list" aria-label="Cartes : {{ LABELS[key] }}">
58             {% for task in items %}
59               <div class="card kanban-card mb-2 shadow-sm border-0"
60                 || | data-id="{{ task.id }}"
61                 || | role="listitem"
62                 || | tabindex="0"
63                 || | aria-label="Tâche #{{ task.id }} - {{ task.title }}">
64                 <div class="card-body p-2">
65                   <div class="d-flex align-items-start justify-content-between">
66                     <div>
67                       <div class="kanban-meta small text-muted">#{{ task.id }}</div>
68                       <div class="small fw-semibold text-break">{{ task.title }}</div>
69                     </div>
70                     <span class="dot {{ DOT_COLOR[key] }}" aria-hidden="true"></span>
71                   </div>
72                 </div>
73             {% else %}>
74               <div class="text-muted small px-2"></div>
75             {% endif %}>
76           </div>
```

Extrait du fichier Kanban/index.html.twig

Réalisation du Front-end

Style

- Bootstrap pour l'adaptation du site à tous les écrans
- Fichier CSS personnel pour des styles supplémentaires

```
41
42     <div class="col-10 col-sm-8 col-md-6 col-lg-3">
43         <div class="kanban-column h-100 rounded-3 overflow-hidden"
44             | role="region"
45             | aria-labelledby="col-title-{{ key }}">
46
47             <div class="kanban-column-header d-flex align-items-center justify-content-between {{ HEADER_CLASS[key] }} px-3 py-2">
48                 <span class="fw-semibold text-uppercase text-white small"
49                     | id="col-title-{{ key }}">{{ LABELS[key] }}</span>
50
51                 <span id="count-{{ key }}"
52                     | class="badge text-bg-light text-dark"
53                     | aria-live="polite" aria-atomic="true">{{ items|length }}</span>
54             </div>
55
56             <div class="kanban-cards p-2" id="col-{{ key }}"
57                 | role="list" aria-label="Cartes : {{ LABELS[key] }}">
58                 {% for task in items %}
59                     <div class="card kanban-card mb-2 shadow-sm border-0"
60                         | data-id="{{ task.id }}"
61                         | role="listitem"
62                         | tabindex="0"
63                         | aria-label="Tâche #{{ task.id }} - {{ task.title }}">
64                         <div class="card-body p-2">
65                             <div class="d-flex align-items-start justify-content-between">
66                                 <div>
67                                     <div class="kanban-meta small text-muted">#{{ task.id }}</div>
68                                     <div class="small fw-semibold text-break">{{ task.title }}</div>
69                                 </div>
70                                 <span class="dot {{ DOT_COLOR[key] }}" aria-hidden="true"></span>
71                             </div>
72                     </div>
73             </div>
```

Extrait du fichier Kanban/index.html.twig

```
49     .kanban-column {
50         background: var(--kanban-col-bg);
51         display: flex;
52         flex-direction: column;
53         min-height: 360px;
54     }
55
56     .kanban-column-header {
57         font-size: 0.85rem;
58     }
59
60     .kanban-cards {
61         flex: 1;
62         overflow: auto;
63         padding-bottom: 0.25rem;
64     }
65
66     .kanban-card .card-body {
67         line-height: 1.2;
68     }
69     .kanban-meta {
70         opacity: 0.8;
71     }
72
73     .dot {
74         display: inline-block;
75         width: 0.65rem;
76         height: 0.65rem;
77         border-radius: 50%;
78         flex: 0 0 auto;
79         margin-left: 0.5rem;
80     }
```

Extrait du fichier Kanban.css

Front dynamique (SSR avec Twig)

- Routes Symfony
- Contrôleurs
- Accès aux données avec Doctrine
- Rendu dynamique avec Twig (Server-Side Rendering)

La requête arrive via une route Symfony

```
#[Route('/task')]
final class TaskController extends AbstractController
{
    #[Route(name: 'app_task_index', methods: ['GET'])]
    public function index(TaskRepository $taskRepository): Response
    {
        Récupération des tâches de l'utilisateur connecté
        $tasks = $taskRepository->findBy(['owner' => $this->getUser()], ['id' => 'ASC']);

        Transmission des données au template Twig
        return $this->render('task/index.html.twig', [
            'tasks' => $tasks
        ]);
    }
}
```

Extrait du fichier TaskController.php

Affichage dynamique des tâches dans la page

```
<div class="fw-semibold mb-2 text-break">{{ task.title }}</div>
```

Extrait du fichier Kanban/index.html.twig

Sécurité

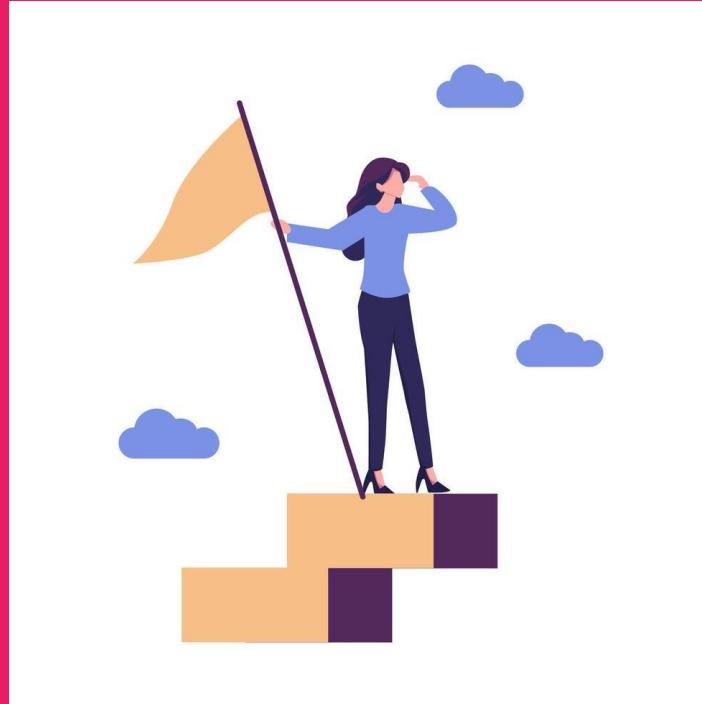
FAIT

- ★ **Validation** des données (Symfony Validator)
- ★ **Protection CSRF** (formulaires Symfony)
- ★ **Mots de passe hashés** (PasswordHasher)
- ★ **Protection XSS** par défaut avec Twig
- ★ **Protection** contre les injections **SQL** avec Doctrine
- ★ **Authentification** via Firewall Symfony

A FAIRE

- ★ **Rate limiting** (limitation du nombre de requêtes)
- ★ **Règles CORS** plus strictes

RÉALISATIONS PERSONNELLES



Front – Drag & Drop

- • Événement drop : mise à jour de l'UI (couleur + compteurs)
- • Création des données (format JSON) (todo/doing/done/urgent)
- • Protection CSRF : token lu depuis <meta name="csrf-kanban">
- • Requête POST vers /kanban/save-order

```
Déclenchement quand une carte est déposée dans une colonne
col.addEventListener('drop', (e) => {
  e.preventDefault(); Empêche le comportement par défaut
  e.stopPropagation();
  Récupération de la clé de la colonne (todo / doing / done / urgent)
  const colKey = (col.id || '').replace('col-', '');
  if (dragged) applyDotColor(dragged, colKey); Mise à jour des couleurs
  updateCounters(); Mise à jour des compteurs
  saveOrder(); On sauvegarde l'ordre.
});
```

```
Fonction qui envoie le nouvel ordre des tâches au backend
function saveOrder() {
  const csrf = document.querySelector('meta[name="csrf-kanban"]')?.content || '';
  Préparation des données : IDs des tâches et Token CSRF
  const data = {
    _token: csrf,
    todo: Array.from(document.querySelectorAll('#col-todo .kanban-card'))
      .map(c => c.dataset.id),
    doing: Array.from(document.querySelectorAll('#col-doing .kanban-card'))
      .map(c => c.dataset.id),
    done: Array.from(document.querySelectorAll('#col-done .kanban-card'))
      .map(c => c.dataset.id),
    urgent: Array.from(document.querySelectorAll('#col-urgent .kanban-card'))
      .map(c => c.dataset.id),
  };
  Envoi de la requête POST vers le backend
  fetch('/kanban/save-order', {
    method: 'POST',
    headers: { 'Content-Type': 'application/json' }, Envoi de données JSON
    body: JSON.stringify(data) Conversion des données en JSON
  });
}
```

Back

- • Route Symfony POST : /kanban/save-order
- • Lecture du JSON envoyé par le front (Request->getContent())
- • Sécurité CSRF : validation du token côté serveur
- • Contrôle d'accès : tâche appartenant à l'utilisateur connecté
- • Mise à jour & persistance : status + position, puis flush()
- • Réponse JSON : { ok: true }

```
##[Route('/kanban/save-order', name: 'kanban_save_order', methods: ['POST'])]
public function saveOrder(
    Request $req,
    TaskRepository $repo,
    EntityManagerInterface $em,
    CsrfTokenManagerInterface $csrf
): JsonResponse {
    On récupère le JSON envoyé par le front
    $data = json_decode($req->getContent(), true) ?? [];
    $token = (string)($data['_token'] ?? '');

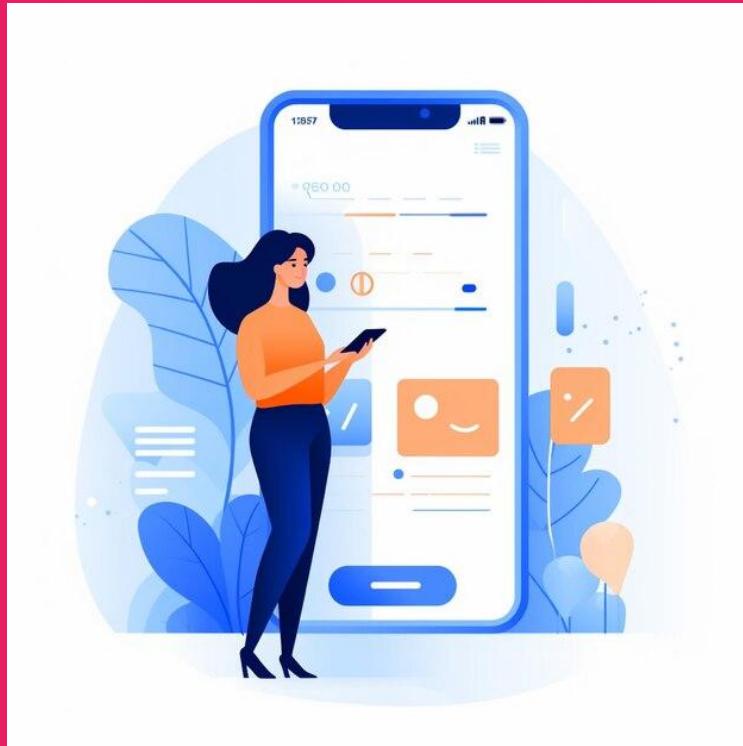
    Sécurité : vérification du token CSRF
    if (!$csrf->isTokenValid(new CsrfToken('kanban_order', $token))) {
        return new JsonResponse(['ok' => false, 'error' => 'bad_csrf'], 400);
    }

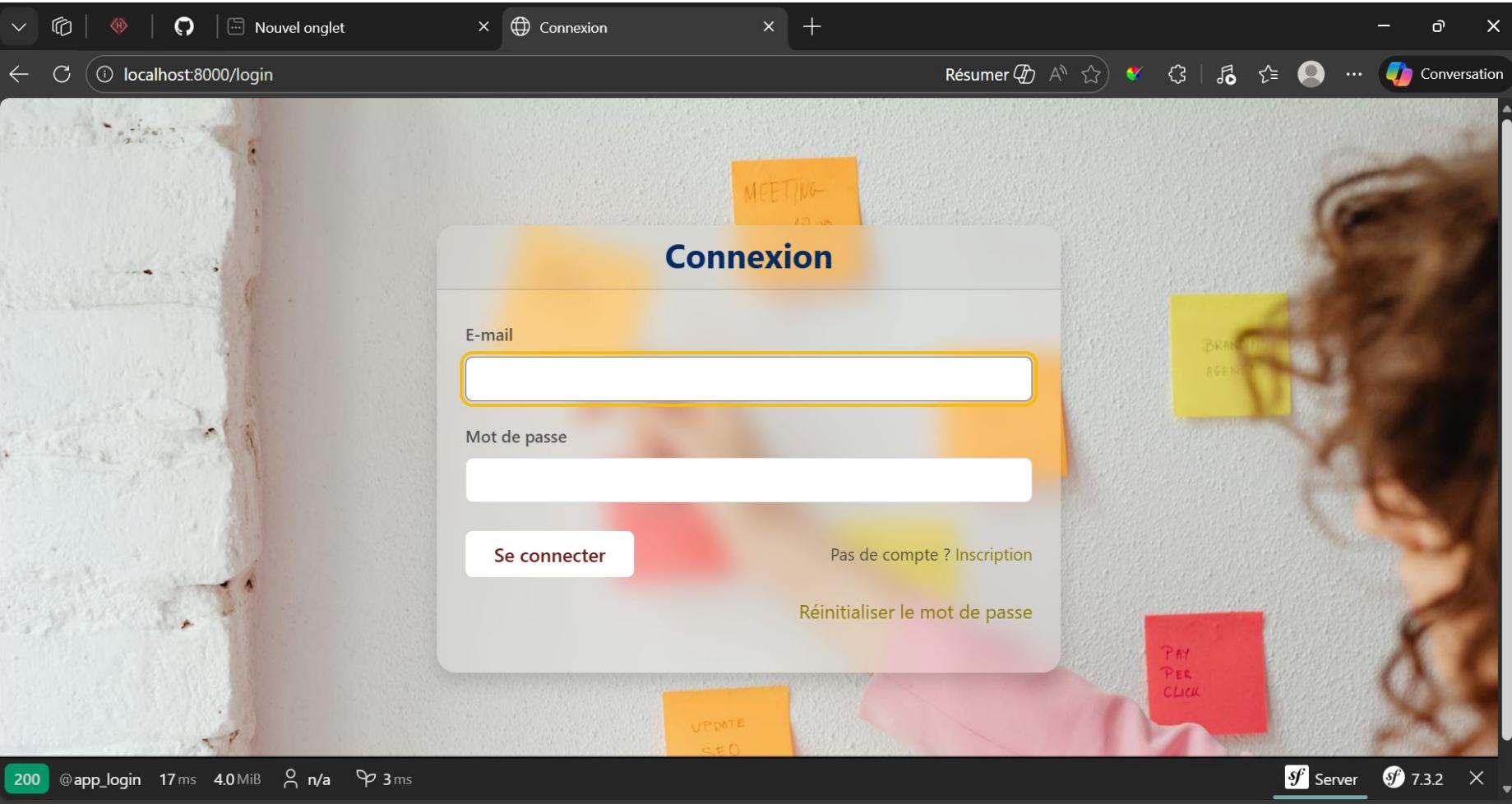
    Mise à jour des colonnes et des positions
    $columns = ['todo', 'doing', 'done', 'urgent'];
    foreach ($columns as $col) {
        $data[$col] contient la liste des ID dans cette colonne
        $ids = array_map('intval', (array)($data[$col] ?? []));
        foreach ($ids as $i => $id) {
            Vérification de l'existence de la tâche et de son appartenance à l'utilisateur courant
            if ($task = $repo->findOneBy(['id' => $id, 'owner' => $this->getUser(),])) {
                $task->setStatus($col);
                $task->setPosition($i);
            }
        }
    }

    On enregistre toutes les modifications en base PostgreSQL
    $em->flush();

    On renvoie une réponse JSON
    return new JsonResponse(['ok' => true]);
}
```

DÉMO DU PROJET





Nouvel onglet Kanban Board

localhost:8000/kanban

TodoList Mes tâches

Déconnexion

URGENT	A FAIRE	EN COURS	TERMINÉ
#27 Réserver une table au restaurant pour l'anniversaire de mariage	#28 Faire les courses pour le dîner (poulet, légumes, pain)	#25 Préparer les documents pour la réunion de parents	#24 Payer la facture d'électricité avant le 30
#23 Prendre rendez-vous chez le dentiste pour Max la semaine prochaine			

200 @app_kanban 63 ms 4.0 MiB ↓↑ 2 nadia1@nadia1.com 9ms 2 in 4.23 ms

Server 7.3.2

Nouvel onglet

Mes tâches

Résumer A ⚡

localhost:8000/task

Conversation

TodoList Mes tâches

Déconnexion

Mes tâches

Toutes vos tâches au même endroit

Créer une tâche

#	Titre	Statut	Actions
#23	Prendre rendez-vous chez le dentiste pour Max la semaine prochaine	Urgent	
#24	Payer la facture d'électricité avant le 30	Terminé	
#25	Préparer les documents pour la réunion de parents	En cours	
#27	Réserver une table au restaurant pour l'anniversaire de mariage	Urgent	
#28	Faire les courses pour le dîner (poulet, légumes, pain)	À faire	

200 @app_task_index 225 ms 4.0 MiB nadi1@nadi1.com 5 ms 2 in 27.48 ms

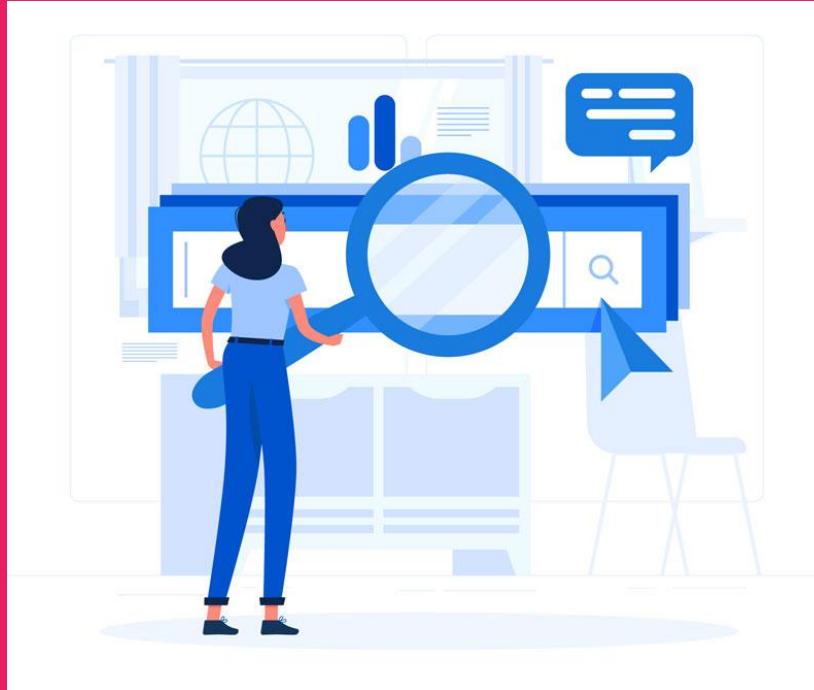
sf Server sf 7.3.2

Créer une tâche

Renseignez les informations de votre tâche

[← Retour](#)[Enregistrer](#)**Titre****Statut****Description****Date à laquelle vous souhaitez être alerté**

EXEMPLE DE RECHERCHE



Présentation d'une recherche

- Problématique technique précise
- Recherche en anglais
- Mots-clés simples et ciblés

The screenshot shows a search results page with the following details:

- Search Query:** symfony csrf form
- Results Count:** Environ 65 400 résultats
- Top Result:** **CSRF Token in Symfony**
 - Description: A CSRF token (Cross-Site Request Forgery token) is a security mechanism used in Symfony to protect web applications from unauthorized actions performed by malicious actors. It ensures that requests to the server originate from trusted sources, preventing attackers from exploiting authenticated sessions.
 - Section: How CSRF Tokens Work in Symfony
 - Description: Symfony automatically integrates CSRF protection into its forms. A unique token is generated and embedded as a hidden field in forms. When the form is submitted, Symfony validates the token to confirm the request's authenticity. This prevents unauthorized actions like form submissions from external websites.
 - Section: Enabling CSRF Protection
 - Description: CSRF protection is enabled by default in Symfony. You can configure it in the `framework.yaml` file:

```
framework:
  csrf_protection:
    enabled: true
```
 - Link: [Symfony](https://symfony.com/doc/current/security/csrf.html) (Traduire ce résultat)
 - Section: How to Implement CSRF Protection (Symfony Docs)
 - Description: Symfony Forms include CSRF tokens by default and Symfony also checks them automatically for you. So, when using Symfony Forms, you don't have to do anything to be protected against ...
 - Section: SymfonyConnect
 - Description: SymfonyConnect is a developer social ...
 - Section: How to Implement CSRF Prot...
 - Description: Although Symfony Forms provide ...
 - Section: Symfony1 Legacy
 - Description: symfony 1.x legacy website The symfony ...
 - Section: ESI Fragment
 - Description: Fortunately, Symfony provides a solution ...
 - Section: How to Configure Symfony t...
 - Description: If your Symfony application runs behind a ...
 - Section: Symfony Blog
 - Description: Official Symfony Blog: all about Symfony ...
 - Section: Afficher uniquement les résultats de symfony.com
- Second Result:** Sling Academy
 - Description: https://www.slingacademy.com/article/csrf-in-... (Traduire ce résultat)
 - Section: CSRF in Symfony: A Practical Guide (with Examples)
 - Description: 14 janv. 2024 - In this tutorial, we will look at how to handle CSRF protection in Symfony, one of the most popular PHP frameworks. Symfony provides an easy-to-use CSRF token service out of ...
- Third Result:** DavFi
 - Description: https://davfi.fr/formulaire-web-comment-generer-un-jeton-csrftoken-valide...
 - Section: Formulaire web : comment générer un jeton CSRF valide en PHP / ...
 - Description: 4 juin 2025 - Comment générer un jeton CSRF sécurisé dans Symfony étape par étape ? Pour ...

Présentation d'une recherche

Critères de sélection des sources :

- Documentation officielle
- Sources techniques fiables
- Vérification de la date et de mise à jour

The screenshot shows a search bar at the top with the query "symfony csrf form". Below the search bar is a navigation menu with links for "TOUT", "RECHERCHER", "IMAGES", "VIDÉOS", "CARTES", "ACTUALITÉS", and "COPILOT". The main content area displays a search result titled "Environ 65 400 résultats". The first result is a summary for "CSRF Token in Symfony" with two numbered links. The summary text explains that a CSRF token is a security mechanism used in Symfony to protect web applications from unauthorized actions. It ensures that requests to the server originate from trusted sources, preventing attackers from exploiting authenticated sessions.

CSRF Token in Symfony 1 | 2

A **CSRF token** (Cross-Site Request Forgery token) is a security mechanism used in Symfony to protect web applications from unauthorized actions performed by malicious actors. It ensures that requests to the server originate from trusted sources, preventing attackers from exploiting authenticated sessions.

How CSRF Tokens Work in Symfony

Symfony automatically integrates CSRF protection into its forms. A unique token is generated and embedded as a hidden field in forms. When the form is submitted, Symfony validates the token to confirm the request's authenticity. This prevents unauthorized actions like form submissions from external websites.

Enabling CSRF Protection

CSRF protection is enabled by default in Symfony. You can configure it in the `framework.yaml` file:

```
framework:  
    csrf_protection:  
        enabled: true
```

 [Symfony](https://symfony.com/doc/current/security/csrf.html) [Traduire ce résultat](https://symfony.com/doc/current/security/csrf.html)

How to Implement CSRF Protection (Symfony Docs)

Symfony Forms include CSRF tokens by default and Symfony also checks them automatically for you. So, when using Symfony Forms, you don't have to do anything to be protected against ...

SymfonyConnect

SymfonyConnect is a developer social ...

How to Implement CSRF Prot...

Although Symfony Forms provide ...

Symfony1 Legacy

symfony 1.x legacy website The symfony ...

[Afficher uniquement les résultats de symfony.com](#)

ESI Fragment

Fortunately, Symfony provides a solution ...

How to Configure Symfony t...

If your Symfony application runs behind a ...

Symfony Blog

Official Symfony Blog: all about Symfony ...

 [Sling Academy](https://www.slingacademy.com/article/csrf-in-symfony/) [Traduire ce résultat](https://www.slingacademy.com/article/csrf-in-symfony/)

CSRF in Symfony: A Practical Guide (with Examples)

14 janv. 2024 - In this tutorial, we will look at how to handle CSRF protection in Symfony, one of the most popular PHP frameworks. Symfony provides an easy-to-use CSRF token service out of ...

 [DavFI](https://davfi.fr/formulaire-web-comment-generer-un-jeton-csrf-valide...) [Traduire ce résultat](https://davfi.fr/formulaire-web-comment-generer-un-jeton-csrf-valide...)

Formulaire web : comment générer un jeton CSRF valide en PHP / ...

4 juin 2025 - Comment générer un jeton CSRF sécurisé dans Symfony étape par étape ? Pour

Présentation d'une recherche

Pour valider ma compréhension, j'utilise ChatGPT comme outil d'aide à la compréhension approfondie.

Generating and Checking CSRF Tokens Manually

Although Symfony Forms provide automatic CSRF protection by default, you may need to generate and check CSRF tokens manually for example when using regular HTML forms not managed by the Symfony Form component.

Consider a HTML form created to allow deleting items. First, use the [csrf_token\(\) Twig function](#) to generate a CSRF token in the template and store it as a hidden form field:

```
<form action="{{ url('admin_post_delete', { id: post.id }) }}" method="post" ↴
    {# the argument of csrf_token() is the ID of this token #}
    <input type="hidden" name="token" value="{{ csrf_token('delete-item') }}">

    <button type="submit">Delete item</button>
</form>
```

Then, get the value of the CSRF token in the controller action and use the [isCsrfTokenValid\(\)](#) method to check its validity, passing the same token ID used in the template:

Conclusion

- Satisfaction du travail réalisé
- Projet mené en autonomie
- Respect des objectifs fixés
- Perspectives d'évolution du projet

A vos questions !