



Soutenance

Ecogest

Augustin SEGUIN

Alternant O'clock & Photoweb



Sommaire

- 1** Introduction
- 2** Analyse du besoin
- 3** Gestion de projet
- 4** Spécifications fonctionnelles
- 5** Spécifications techniques
- 6** Réalisations significatives
- 7** Présentation du jeu d'essai
- 8** Recherches veille / informations
- 9** Conclusion



Introduction

Augustin SEGUIN
25 ans
Grenoble

1. MIASHS
2. Handigital
3. O'clock

Photoweb

10 mois legacy

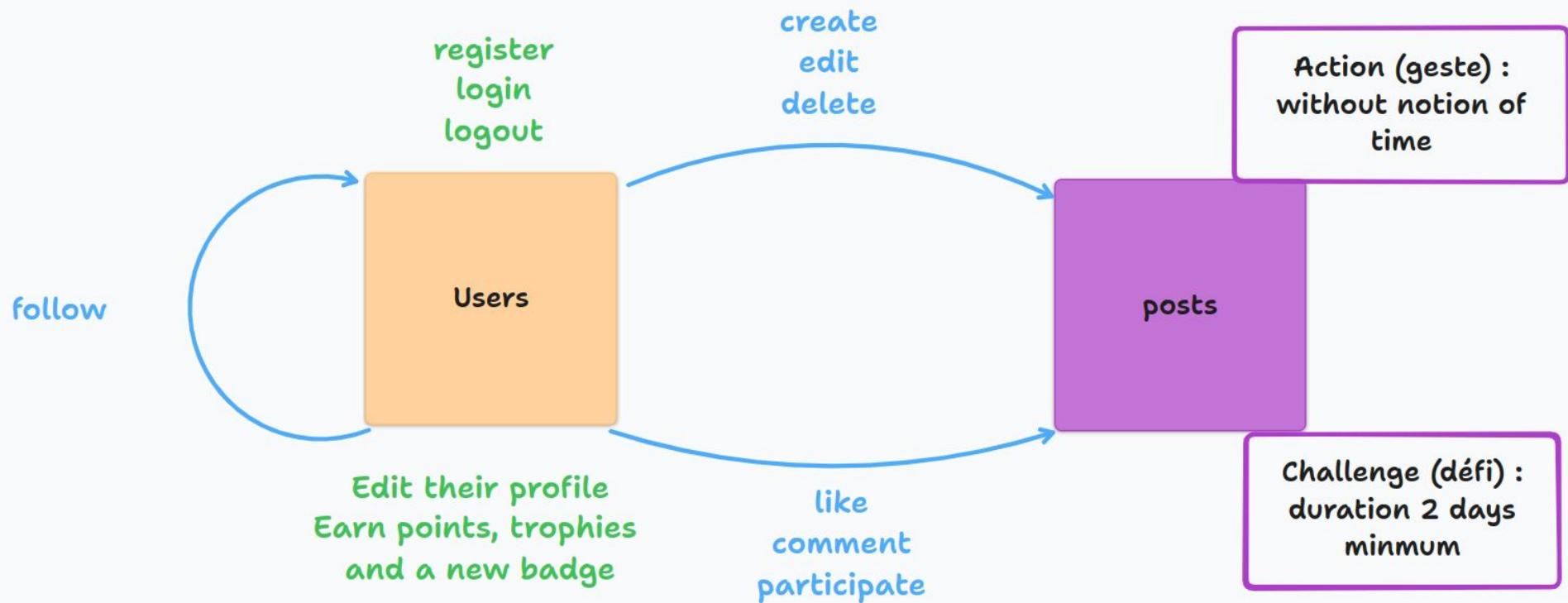
- ASP .NET
- AngularJS

6 mois refonte de la plateforme

- PHP Magento



Introduction





Analyse du besoin

Problématique

Comment **rassembler** les gens autour de **l'environnement** et accélérer le **processus d'actions écologiques** ?

L'ambition

Notre ambition est de sensibiliser et faire **passer à l'action** un maximum de personnes pour la **préservation de la planète**, à travers des **défis écologiques** du quotidien.

Une solution

EcO'Gest est un réseau social disponible sous forme d'application mobile qui a pour but de **sensibiliser** à l'écologie et de **s'encourager** entre amis à la réalisation de défis.

Les objectifs

- 1** Accélérer le changement vers des pratiques plus écoresponsables.
- 2** Développer l'émulation et la cohésion d'équipe grâce à la mécanique de jeu.
- 3** Passer un bon moment avec son entourage !

Analyse du besoin



Augustin est un jeune professionnel passionné par le vélo et a fait le choix de se rendre au travail à vélo pour des raisons écologiques et de santé. Il souhaite suivre ses propres actions éco-responsables et encourager ses collègues à adopter des gestes plus respectueux de l'environnement.

Objectifs et besoins:

- Suivre ses Actions Éco-Responsables et enregistrer ses trajets à vélo.
- Motiver ses Collègues à participer à des actions éco-responsables.
- Faciliter le partage de ses réalisations et ses défis éco-responsables avec ses collègues.

Freins:

- Manque de Suivi de ses actions éco-responsables.
- Motivation des Collègues : Convaincre ses collègues de changer leurs habitudes peut être un défi, et il a besoin d'outils pour rendre cette transition plus attrayante.

Critères de décision:

- Influence sociale



Nom : Kylian
Age : Moins de 18 ans
Situation : Lycéen

Équipement : Tablette et smartphone
Maturité digitale : Plutôt à l'aise

Applications préférées :



Kylian est un lycéen de 16 ans. Inspiré par des jeunes activistes comme Greta Thunberg, il pense que chaque petit geste compte et souhaite avoir un impact positif sur la planète. Il veut intégrer sa famille proche dans son projet et souhaite responsabiliser ses parents en instaurant des gestes quotidiens au sein du foyer.

Objectifs et besoins:

- Inspier les autres : motiver sa famille
- Gagner du temps dans sa recherche de gestes à effectuer

Freins:

- Résistance de la famille : Kylian se heurte parfois à la résistance de sa famille lorsqu'il suggère des changements respectueux de l'environnement. Il a besoin d'une manière de rendre cela plus attrayant et accessible. Ses parents ont la charge mentale du foyer et n'ont pas le temps de réfléchir à des gestes à instaurer.

Critères de décision:

- Besoins et objectifs

cibles



Jeanne est une enseignante de français passionnée et dévouée. Elle est actuellement professeure principale d'une classe de lycéens et croit fermement en l'importance de l'éducation pour sensibiliser les jeunes générations aux enjeux environnementaux. Elle veut intégrer des initiatives écologiques dans sa classe pour inspirer ses élèves à adopter des comportements respectueux de l'environnement.

Objectifs et besoins:

- Sensibiliser ses Élèves et Intégrer l'Ecologie dans l'Education
- Suivre la participation des Élèves
- Proposer une compétition pour motiver ses élèves

Freins:

- Manque de Temps : en tant qu'enseignante, Jeanne a un emploi du temps chargé et elle trouve parfois difficile de trouver du temps pour planifier et mettre en œuvre des activités écologiques en classe.
- Résistance des Élèves : certains élèves peuvent ne pas être initialement enthousiastes à l'idée de participer à des initiatives écologiques

Critères de décision:

- Besoins et objectifs
- Concurrence

Les jeunes actifs qui embarquent amis et collègues

Image by Freepik

Les jeunes qui motivent leur famille Intergénérationnelle

Image by Freepik

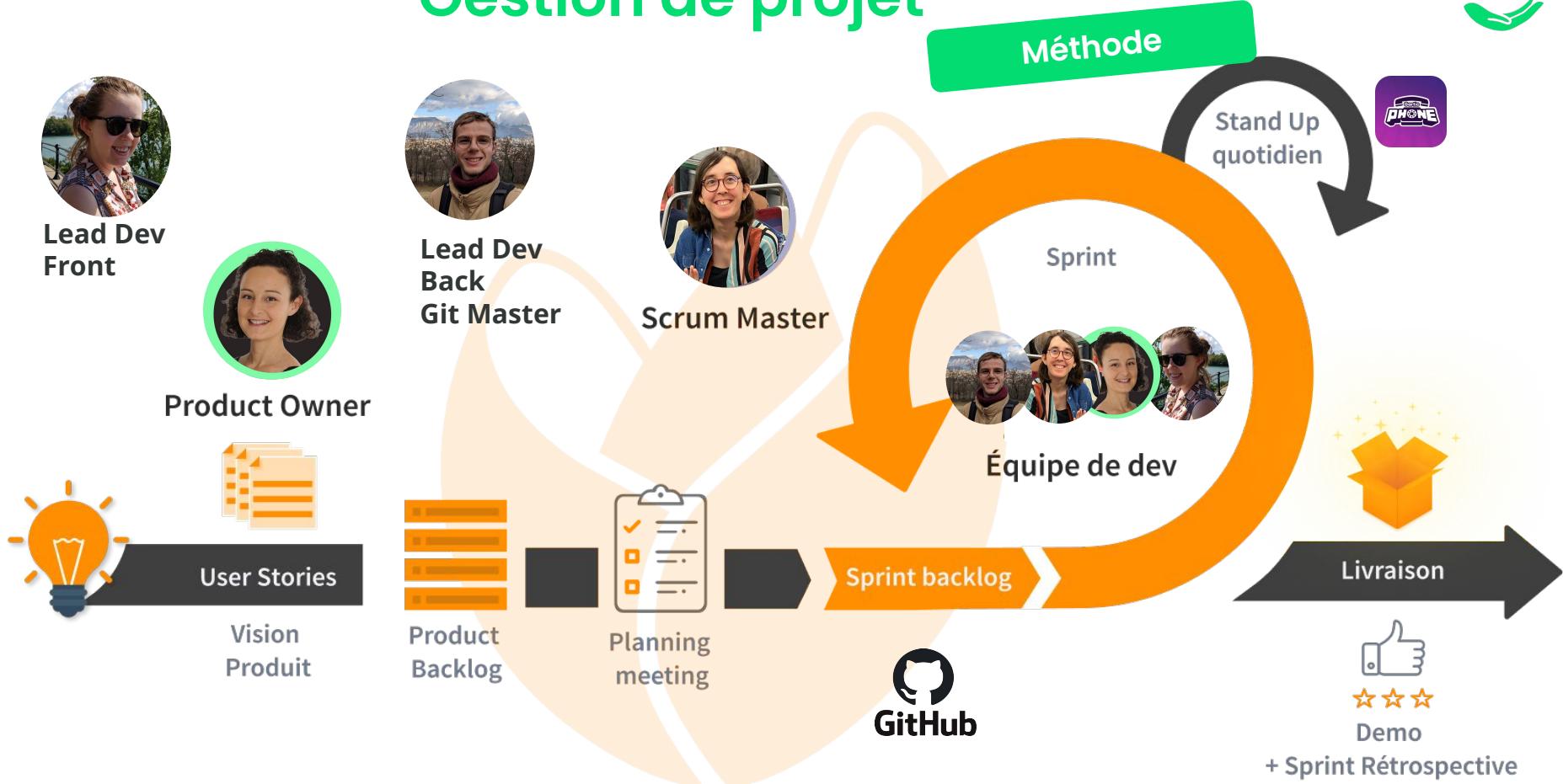
Les professionnels de l'éducation en contact avec les jeunes publics

Image by Freepik





Gestion de projet





Gestion de projet

Macro planning

29/08	31/09	18/09	20/09	22/09	09/10	10/10	12/10	30/10	02/11
S1	S2	S3	S4	S5	S6	S7	S8	S9	
KICK OFF MEETING 									
CONCEPTION			INTEGRATION ET DEVELOPPEMENT						RECETTE
Organisation (outils, rôles, réunions, conventions)	Diagrammes UML	Wireframes	User stories	Backlog mis à jour	Backlog mis à jour	Backlog mis à jour	Backlog mis à jour	Backlog mis à jour	Tests
Liste des fonctionnalités	Arborescence	Routes	Cibles persona	tests automatisés	tests automatisés	tests automatisés	tests automatisés	tests automatisés	Correction des bugs
Documents de veille	Planning prévisionnel	Architecture	Processus CI/CD	Processus CI/CD	Processus CI/CD	Processus CI/CD	Processus CI/CD	Processus CI/CD	Refactoring
Analyse des risques	benchmark techno.	Choix techno.	Sécurité	Sécurité	Sécurité	Sécurité	Sécurité	Sécurité	
Journaux de bord	Charte graphique	LIVRABLE Gestion de projet	LIVRABLE Cahier des charges fonctionnel	LIVRABLE Spécifications techniques	Reprise : fonctionnalités objectifs				
Daily Review Retro Planning									
								LIVRABLE MVP à recetter	LIVRABLE MVP Déployé en production



Gestion de projet

Agilité

Fonctionnement en agile : méthode Scrum

Cérémonies :

- daily meetings
- sprint reviews
- sprint planning

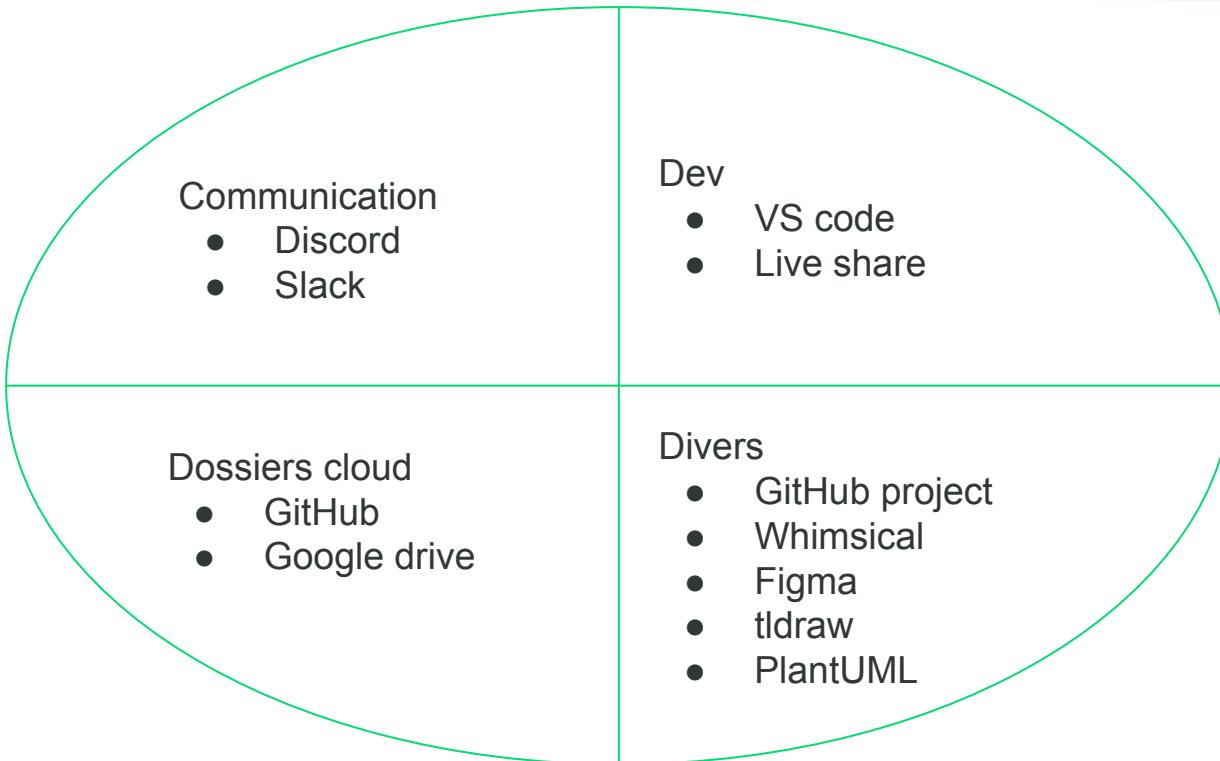
Durée de sprints :

- 2 jours conception
- 3 jours développement



Gestion de projet

Les outils



Spécifications fonctionnelles



User stories

[US-7] En tant que visiteur Je souhaite pouvoir **me connecter à l'application** Afin de pouvoir accéder aux fonctionnalités de l'application

Auth

[US-11] En tant que visiteur Je souhaite pouvoir **créer mon compte** Afin de pouvoir utiliser l'application

[US-17] En tant que utilisateur connecté Je souhaite me **déconnecter de l'application** Afin de pour ne pas laisser ma session ouverte

[US-1] En tant que utilisateur connecté Je souhaite **afficher mon profil** Afin de voir mes informations personnelles

Profil

[US-2] En tant que utilisateur connecté Je souhaite **afficher une publication avec ses détails** Afin de d'en savoir plus sur l'action dont le descriptif

[US-23] En tant que utilisateur connecté Je souhaite pouvoir éditer mon profil afin de mettre à jour mes informations

[US-24] En tant que utilisateur connecté Je souhaite pouvoir utiliser les tags dans mes posts & dans la recherche afin de les rendre visible

Search

[US-15] En tant que utilisateur connecté Je souhaite faire une recherche par mot clé Afin de pouvoir trouver un profil ou une publication

[US-3] En tant que utilisateur connecté Je souhaite créer une **publication** Afin de partager une action ou un défi à la communauté et gagner des points

Post

[US-4] En tant que utilisateur connecté Je souhaite afficher la liste des publications Afin de suivre les actions qu'ils ont réalisées

[US-6] En tant que utilisateur connecté Je souhaite éditer une publication Afin de pouvoir modifier une erreur de texte ou remplacer une image

[US-8] En tant que utilisateur connecté Je souhaite interagir en ajoutant un "j'aime" à la publication Afin de encourager mes amis et créer du lien

[US-10] En tant que utilisateur connecté Je souhaite commenter une publication Afin de encourager mes amis et créer du lien avec un message personnalisé

[US-14] En tant que utilisateur connecté Je souhaite souscrire à un défi Afin de pouvoir ajouter le défi à mes actions et peut-être gagner des points

Défis

[US-12] En tant que utilisateur connecté Je souhaite afficher les défis en cours Afin de me rappeler la liste des défis auxquels je suis inscrit.e

[US-9] En tant que utilisateur connecté Je souhaite afficher l'historique de mes défis et challenges Afin de voir le chemin parcouru et l'évolution de mon activité

[US-13] En tant que utilisateur connecté Je souhaite afficher les défis à venir Afin de pouvoir m'inspirer et peut-être participer à l'un d'entre eux

[US-18] En tant que utilisateur connecté Je souhaite afficher la politique de confidentialité Afin de pour me renseigner sur les conditions d'utilisation, les cookies, etc.

Légal

[US-19] En tant que utilisateur connecté Je souhaite afficher les mentions légales Afin de pour me renseigner sur les conditions d'utilisation, les cookies, etc.

[US-20] En tant que utilisateur connecté Je souhaite pouvoir contacter le DPO Afin de pour faire une demande concernant mes données personnelles

[US-22] En tant que utilisateur connecté Je souhaite pouvoir suivre d'autres utilisateurs afin de voir leurs informations / posts

Communauté

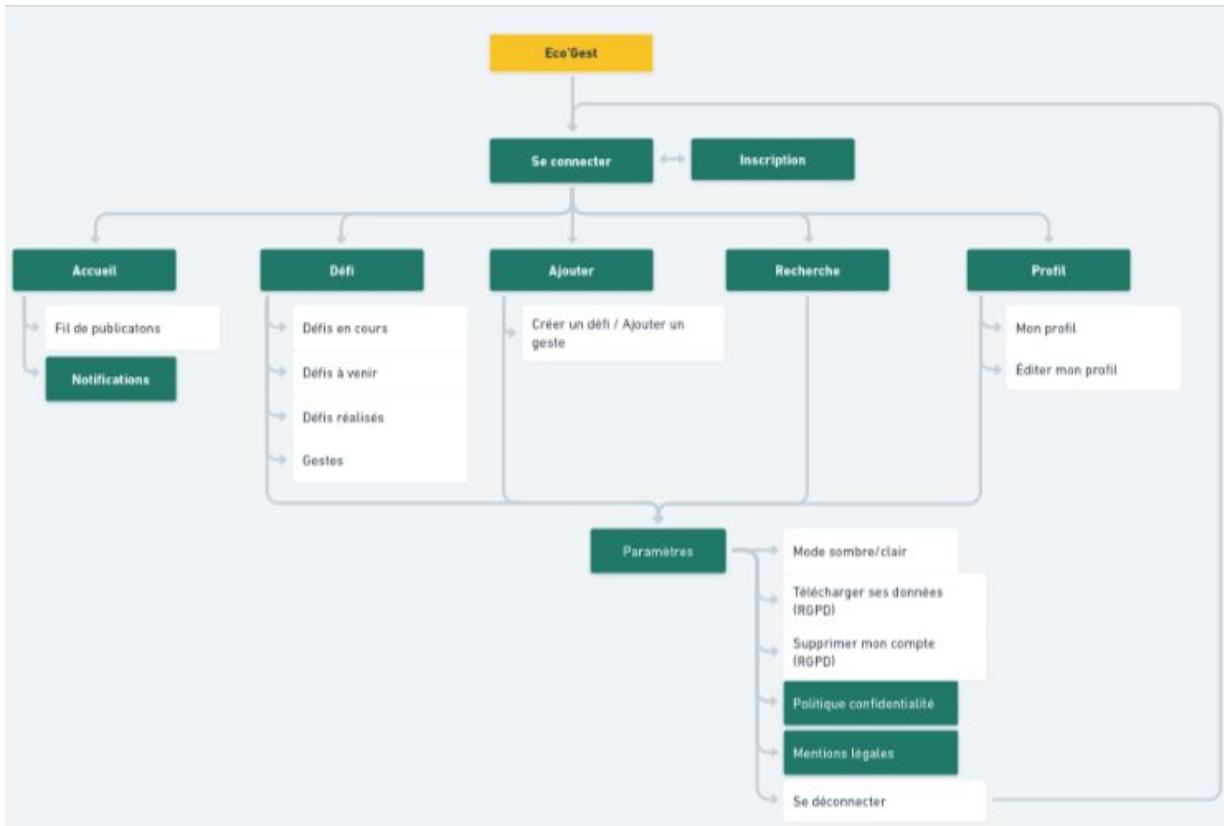
[US-16] En tant que utilisateur connecté Je souhaite afficher mes notifications Afin de suivre l'activité de mes amis ou mes rappels

+ Priorité -



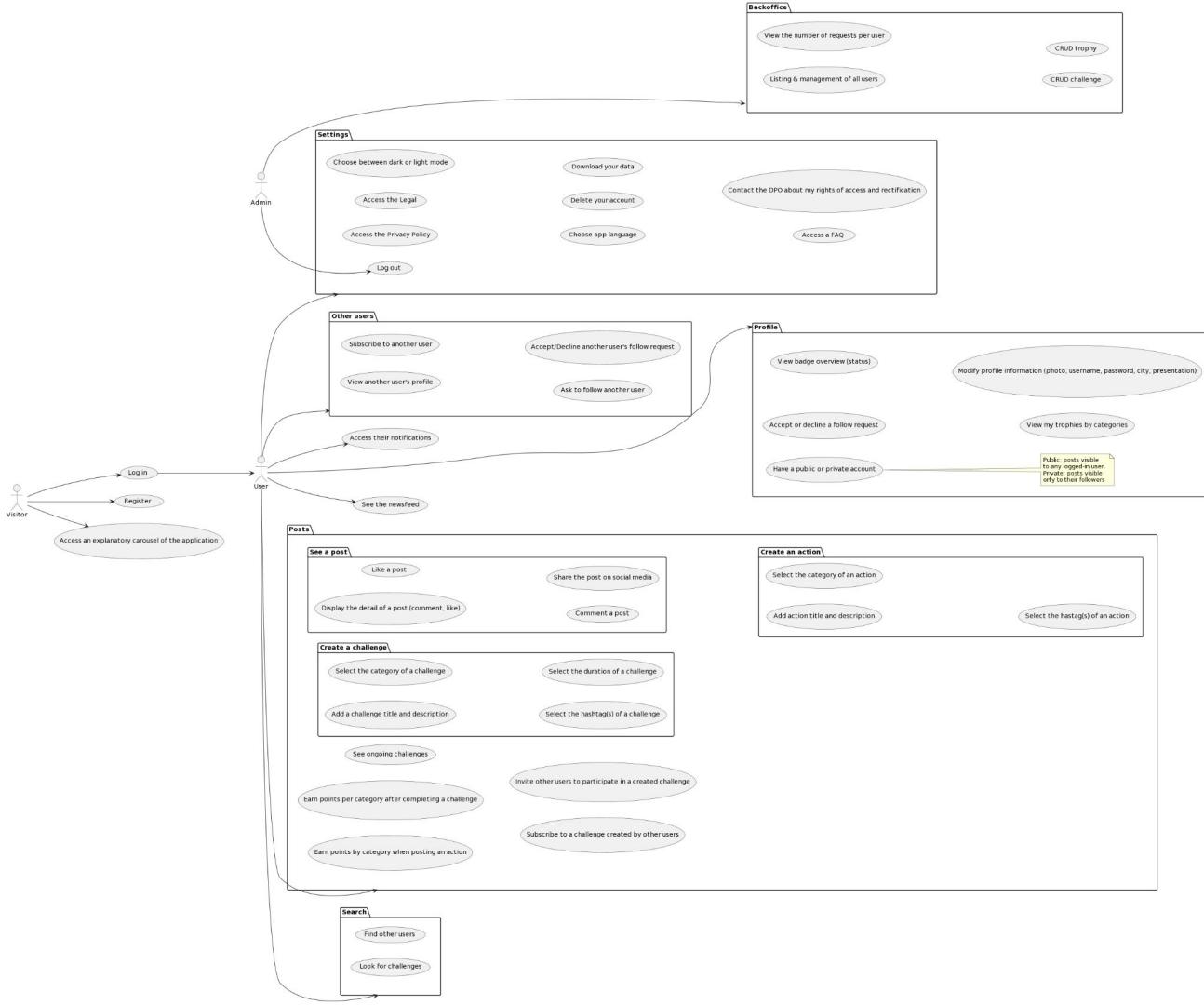
Spécifications fonctionnelles

Arborescence





Spécifications fonctionnelles





Spécifications fonctionnelles

Maquettes

Desktop - 4

ÉCO'GEST

[S'inscrire](#)

You have already a account ?
Happy to see you again

[Forgot password ?](#)

[Log in](#)

Eco'gest - 2023 | Mentions légales | [Facebook](#) [Instagram](#) [Twitter](#) [LinkedIn](#)

REGISTER

4:19 4 % 9:1

ÉCO'GEST

Inscription

[S'inscrire](#)

You have already a account ?
[Log in](#)

DEBIATED



Spécifications fonctionnelles

Maquettes

Desktop - 1

The desktop wireframe displays the Eco'Gest interface. At the top, there's a header with a search bar and user navigation icons. On the left, a sidebar shows a user profile (A) with 'Pseudo' and 'Label', 'Points 900', and 'Défis 3'. The main area shows a feed of posts. Each post card includes a user profile icon, name, date ('14/09/2023 | Rennes'), a preview image, points ('10 points'), a category ('Mobilité'), and a 'Geste' button. Below each post is a detailed view of its content, including a title with placeholder text, a description, likes ('1 like'), comments ('2 commentaires'), and interaction buttons ('Like', 'Comment', 'Share'). A 'Suggestion d'amis' section is also visible.

The mobile wireframe shows a post detail screen. At the top, it says 'HOME' and has a back arrow. The post card is identical to the one in the desktop version. Below it is a larger, detailed view of the post content, including the title, description, likes, comments, and interaction buttons. The bottom of the screen features a navigation bar with icons for home, search, and user profile.

The dark mode mobile wireframe shows the same post detail screen as the light mode version, but with a black background. The post card and detailed view are identical, and the overall layout follows the dark theme.



Spécifications fonctionnelles

CHALLENGE

Pseudo 14/09/2023 | Rennes | E

Label

10 points Motilité Geste

Titre
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

1 like, 3 commentaires

Like Comment Share

Home Trophy + Search User

CHALLENGE

Créer un post

Geste DÉFI

Mobilité

Titre

Description

Position

Ajouter un ou plusieurs tags

image

Home Trophy + Search User

SEARCH

Rechercher

Recherche une publication ou un utilisateur

Augus

Rechercher

Pas de publication

Résultats des utilisateurs

A Pseudo

Home Trophy + Search User

Maquettes



Spécifications fonctionnelles

Maquettes

The image displays two wireframe mockups of a mobile application's profile screen, side-by-side. Both screens have a header bar with a back arrow, the word "PROFIL", and a gear icon. Below the header, there are two tabs: "Mon profil" (selected) and "Editor mon profil".

Left Mockup (Mon profil tab):

- A circular profile picture placeholder with a letter "A".
- A text field labeled "Pseudo" containing "kenes_label".
- A text field labeled "Points" showing "900".
- A text field labeled "Défa" showing "3".
- A text area with placeholder text: "Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor".
- A section titled "Accomplissements" with the subtext "Aucun trophée disponible".
- At the bottom are navigation icons for Home, Trophy, Plus, Search, and Profile.

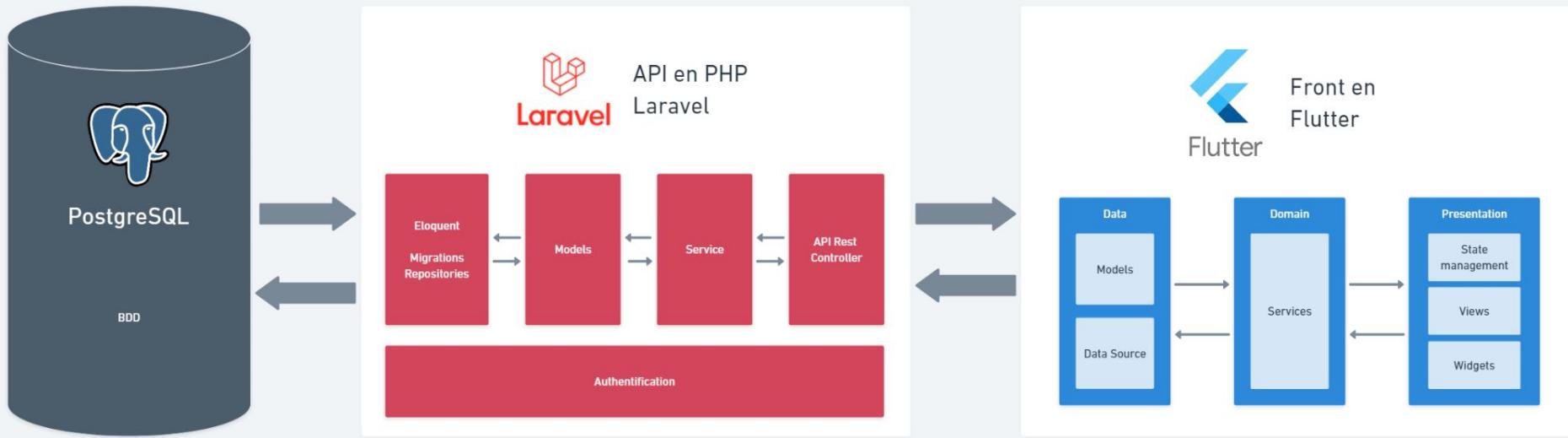
Right Mockup (Editor mon profil tab):

- A text field labeled "Nom d'utilisateur" containing "eleotore".
- A checkbox labeled "Profil privé" which is checked.
- A text field labeled "Biographie" (empty).
- A text field labeled "Biographie" (empty).
- A text field labeled "Date d'enregistrement" showing "27/12/1993".
- An "Image" placeholder field.
- A large green "Publier" button.
- At the bottom are navigation icons for Home, Trophy, Plus, Search, and Profile.



Spécifications techniques

Architecture





Spécifications techniques

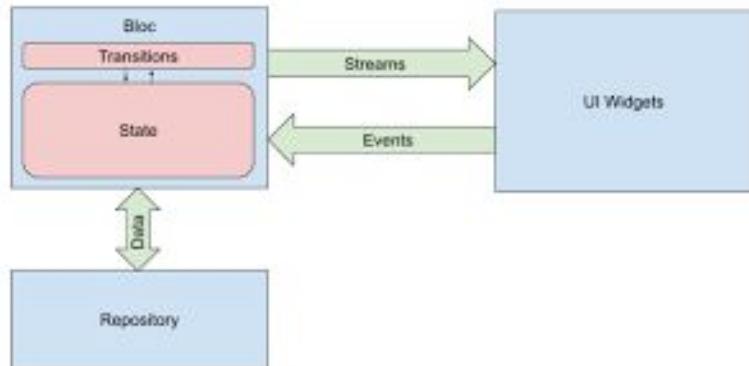
Circulation de la données :

Architecture

Backend



Frontend





Spécifications techniques

Techno



PHP Laravel ^10

PostgreSQL ^16

Adminer ^4.8.1

Nginx ^1.19.8



Spécifications techniques

Techno

Packages backend composer

- laravel/sanctum ^1.0
- laravel/tinker ^2.8
- fakerphp/faker ^1.9.1
- knuckleswtf/scribe ^4.25
- phpunit/phpunit ^10.1
- mockery/mockery ^1.4.4



Spécifications techniques

Techno

Packages frontend flutter ^3

- go_router ^6.2.0
- provider ^6.0.5
- http ^0.13.5
- flutter_bloc ^8.1.3
- flutter_secure_storage ^8.0.0



Spécifications techniques

Techno

Application multiplateforme

- iOS
- Android
- Windows
- MacOS
- Linux
- Web



Spécifications techniques

Techno

Compilation web

avec dart2js compiler mais...

- visibilité réduite sur le web - SEO
- difficile de debugger dans la console web
- performance “diminuée” par rapport à une application spécialement conçue pour le web
- “temps de développement plus long”
- pas disponible sur tous les browsers (bien que les plus populaires aient la fonctionnalité pour compiler le dart)



Spécifications techniques

Risques techniques

Analyse des risques

- Déficit de compétences
- Falsification de requête côté serveur
- Manque d'intégrité des données et du logiciel
- Identification et authentification de mauvaise qualité
- Composants vulnérables et obsolètes
- Mauvaise configuration de la sécurité
- Conception non sécurisée



Spécifications techniques

Analyse des risques

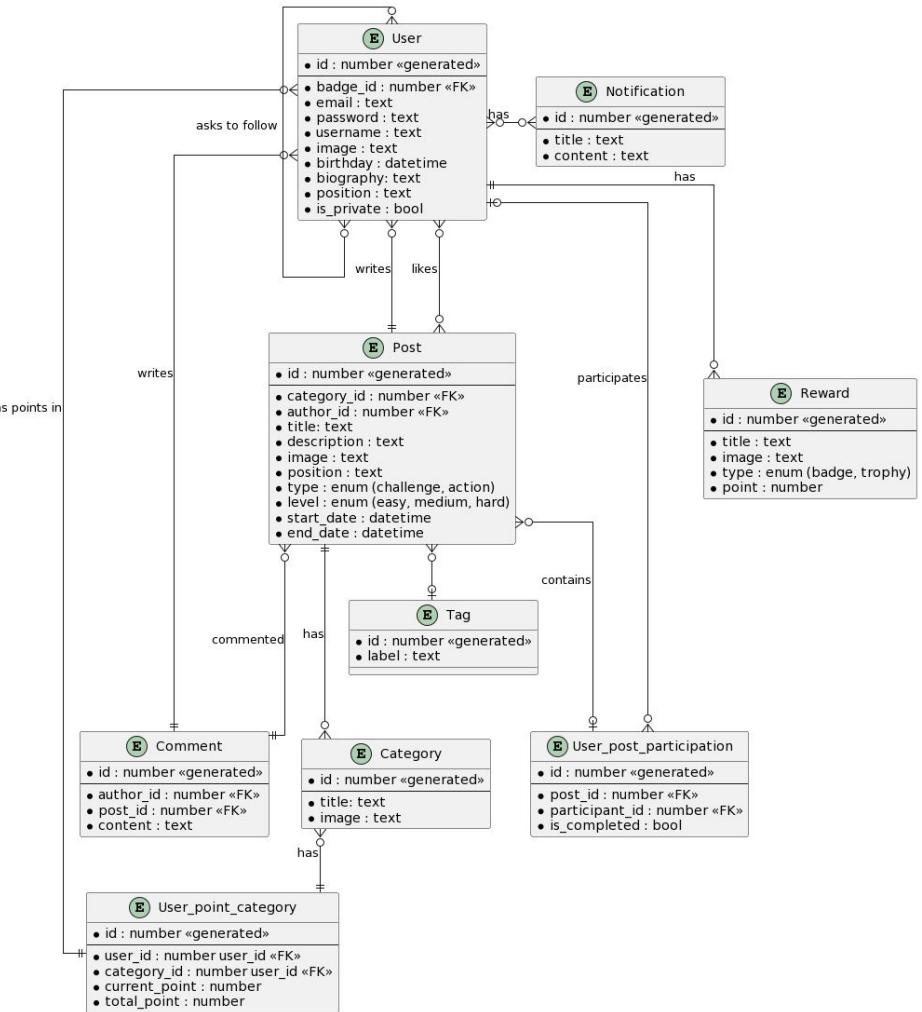
Risques divers

- Maladie/départ d'une ressource projet
- Non respect du RGPD
- Estimation du temps trop courte, nous manquons de temps pour réaliser notre MVP
- Mauvaise répartition des tâches
- Cahier des charges incomplets
- Solution techniques à priori gratuites et devenant payantes
- Par manque de temps => code non épuré pouvant être refactorisé



Spécifications techniques

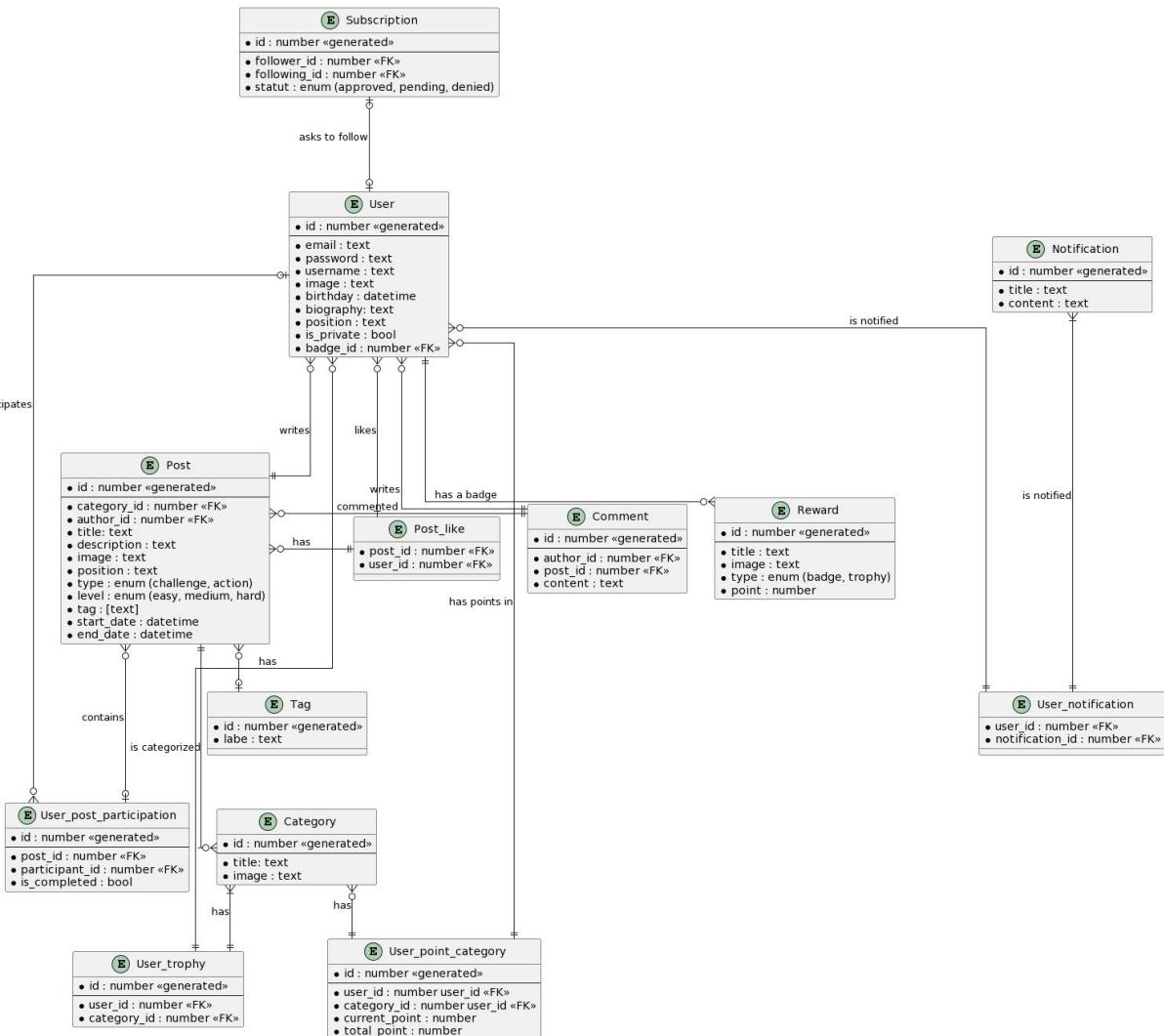
Conception BDD





Spécifications techniques

Conception BDD





Spécifications techniques

Conception BDD

user			
Champ	Type	Spécificités	Description
id	INTEGER	PRIMARY KEY, NOT NULL	L'identifiant de l'utilisateur
badge_id	INTEGER	NOT NULL, DEFAULT 1, FOREIGN KEY	Le badge de l'utilisateur
email	VARCHAR(64)	NOT NULL	L'email de l'utilisateur
password	VARCHAR(64)	NOT NULL	Le mot de passe de l'utilisateur (crypté)
username	VARCHAR(64)	NULL	Le pseudo de l'utilisateur
image	VARCHAR(128)	NULL	L'url de l'image de l'utilisateur
birthdate	DATETIME	NULL	La date de naissance de l'utilisateur
biography	TEXT	NULL	Le texte de présentation de l'utilisateur
position	VARCHAR(64)	NULL	L'emplacement (ville, pays) de l'utilisateur
is_private	BOOLEAN	NOT NULL, DEFAULT FALSE	La visibilité du profil
created_at	TIMESTAMP	NOT NULL, DEFAULT CURRENT_TIMESTAMP	Date de création de l'utilisateur
updated_at	TIMESTAMP	NOT NULL, DEFAULT CURRENT_TIMESTAMP	Date de mise à jour de l'utilisateur

user_post_participation			
Champ	Type	Spécificités	Description
id	INTEGER	PRIMARY KEY, NOT NULL	L'identifiant de la participation à un challenge
participant_id	INTEGER	NOT NULL, FOREIGN KEY	L'identifiant de l'utilisateur participant
post_id	INTEGER	NOT NULL, FOREIGN KEY	L'identifiant de la publication du challenge
is_completed	BOOLEAN	NOT NULL, DEFAULT FALSE	La complétion du challenge par le participant
created_at	TIMESTAMP	NOT NULL, DEFAULT CURRENT_TIMESTAMP	Date de création de la participation
updated_at	TIMESTAMP	NOT NULL, DEFAULT CURRENT_TIMESTAMP	Date de mise à jour de la participation



Spécifications techniques

Conception BDD

user_point_category

Champ	Type	Spécificités	Description
<code>id</code>	INTEGER	PRIMARY KEY, NOT NULL	L'identifiant du suivi de points par catégorie
<code>user_id</code>	INTEGER	NOT NULL, FOREIGN KEY	L'identifiant de l'utilisateur qui gagne des points
<code>category_id</code>	INTEGER	NOT NULL, FOREIGN KEY	L'identifiant de la catégorie
<code>current_point</code>	INTEGER	NOT NULL, DEFAULT 0	Le nombre de points actuel dans la catégorie
<code>total_point</code>	INTEGER	NOT NULL, DEFAULT 0	Le nombre de points total dans la catégorie
<code>created_at</code>	TIMESTAMP	NOT NULL, DEFAULT CURRENT_TIMESTAMP	La date de création du suivi de points par catégorie
<code>updated_at</code>	TIMESTAMP	NOT NULL, DEFAULT CURRENT_TIMESTAMP	La date de mise à jour du suivi de points par catégorie

user_trophy

Champ	Type	Spécificités	Description
<code>id</code>	INTEGER	PRIMARY KEY, NOT NULL	L'identifiant de la remise de trophée
<code>user_id</code>	INTEGER	NOT NULL, FOREIGN KEY	L'identifiant de l'utilisateur qui remporte le trophée
<code>category_id</code>	INTEGER	NOT NULL, FOREIGN KEY	L'identifiant de la catégorie du trophée
<code>created_at</code>	TIMESTAMP	NOT NULL, DEFAULT CURRENT_TIMESTAMP	La date de création de la remise de trophée
<code>updated_at</code>	TIMESTAMP	NOT NULL, DEFAULT CURRENT_TIMESTAMP	La date de mise à jour de la remise de trophée

subscription

Champ	Type	Spécificités	Description
<code>id</code>	INTEGER	PRIMARY KEY, NOT NULL	L'identifiant de l'abonnement
<code>follower_id</code>	INTEGER	NOT NULL, FOREIGN KEY	L'identifiant de l'utilisateur abonné
<code>following_id</code>	INTEGER	NOT NULL, FOREIGN KEY	L'identifiant de l'utilisateur suivi
<code>status</code>	ENUM	NOT NULL	Statut de la demande de suivi : 'approved', 'pending', 'denied'
<code>created_at</code>	TIMESTAMP	NOT NULL, DEFAULT CURRENT_TIMESTAMP	La date de création de l'abonnement
<code>updated_at</code>	TIMESTAMP	NOT NULL, DEFAULT CURRENT_TIMESTAMP	La date de mise à jour de l'abonnement



Spécifications techniques

Conception BDD

reward			
Champ	Type	Spécificités	Description
<code>id</code>	INTEGER	PRIMARY KEY, NOT NULL	L'identifiant de la récompense
<code>title</code>	VARCHAR(64)	NULL	Le nom du trophée ou du badge
<code>image</code>	VARCHAR(128)	NULL	L'url de l'image du trophée ou du badge
<code>type</code>	ENUM	NOT NULL	Le type de récompense badge ou trophée
<code>point</code>	INTEGER	NULL	Le nombre de point seuil pour obtenir la récompense
<code>created_at</code>	TIMESTAMP	NOT NULL, DEFAULT CURRENT_TIMESTAMP	La date de création de la récompense
<code>updated_at</code>	TIMESTAMP	NOT NULL, DEFAULT CURRENT_TIMESTAMP	La date de mise à jour de la récompense

category			
Champ	Type	Spécificités	Description
<code>id</code>	INTEGER	PRIMARY KEY, NOT NULL	L'identifiant de la catégorie
<code>title</code>	VARCHAR(64)	NULL	Le titre de la catégorie
<code>image</code>	VARCHAR(128)	NULL	L'url de l'image de la catégorie
<code>created_at</code>	TIMESTAMP	NOT NULL, DEFAULT CURRENT_TIMESTAMP	Date de création de la catégorie
<code>updated_at</code>	TIMESTAMP	NOT NULL, DEFAULT CURRENT_TIMESTAMP	Date de mise à jour de la catégorie

tags			
Champ	Type	Spécificités	Description
<code>id</code>	INTEGER	PRIMARY KEY, NOT NULL	L'identifiant du tag.
<code>label</code>	VARCHAR(64)	NOT NULL, UNIQUE	Le label du tag.



Spécifications techniques

Conception BDD

post			
Champ	Type	Spécificités	Description
id	INTEGER	PRIMARY KEY, NOT NULL	L'identifiant de la publication
category_id	INTEGER	NOT NULL, FOREIGN KEY	L'identifiant de la catégorie de la publication
author_id	INTEGER	NOT NULL, FOREIGN KEY	L'identifiant de l'auteur de la publication
tag	TEXT []	NULL	Les mots-clés de la publication
title	VARCHAR(64)	NULL	Le titre de la publication
description	TEXT	NULL	La description de la publication
image	VARCHAR(128)	NULL	L'url de l'image de la publication
position	VARCHAR(64)	NULL	L'emplacement (ville, pays) pour la publication
type	ENUM	NOT NULL	Le type de publication challenge ou défi
level	ENUM	NOT NULL	Le niveau de difficulté : facile, moyen ou difficile
start_date	DATE	NULL	La date de début du challenge
end_date	DATE	NULL	La date de fin du challenge
created_at	TIMESTAMP	NOT NULL, DEFAULT CURRENT_TIMESTAMP	La date de création de la publication
updated_at	TIMESTAMP	NOT NULL, DEFAULT CURRENT_TIMESTAMP	La date de mise à jour de la publication

like				comment			
Champ	Type	Spécificités	Description	Champ	Type	Spécificités	Description
id	INTEGER	PRIMARY KEY, NOT NULL	L'identifiant du like	id	INTEGER	PRIMARY KEY, NOT NULL	L'identifiant du commentaire
user_id	INTEGER	NOT NULL, FOREIGN KEY	L'identifiant de l'auteur du like	author_id	INTEGER	NOT NULL, FOREIGN KEY	L'identifiant de l'auteur du commentaire
post_id	INTEGER	NOT NULL, FOREIGN KEY	L'identifiant de la publication du like	post_id	INTEGER	NOT NULL, FOREIGN KEY	L'identifiant de la publication du commentaire
created_at	TIMESTAMP	NOT NULL, DEFAULT CURRENT_TIMESTAMP	La date de création du like	content	TEXT	NULL	Le contenu du commentaire
updated_at	TIMESTAMP	NOT NULL, DEFAULT CURRENT_TIMESTAMP	La date de mise à jour du like	created_at	TIMESTAMP	NOT NULL, DEFAULT CURRENT_TIMESTAMP	La date de création du commentaire
				updated_at	TIMESTAMP	NOT NULL, DEFAULT CURRENT_TIMESTAMP	La date de mise à jour du commentaire



Spécifications techniques

Diagrams

Diagramme d'activité

Créer une publication

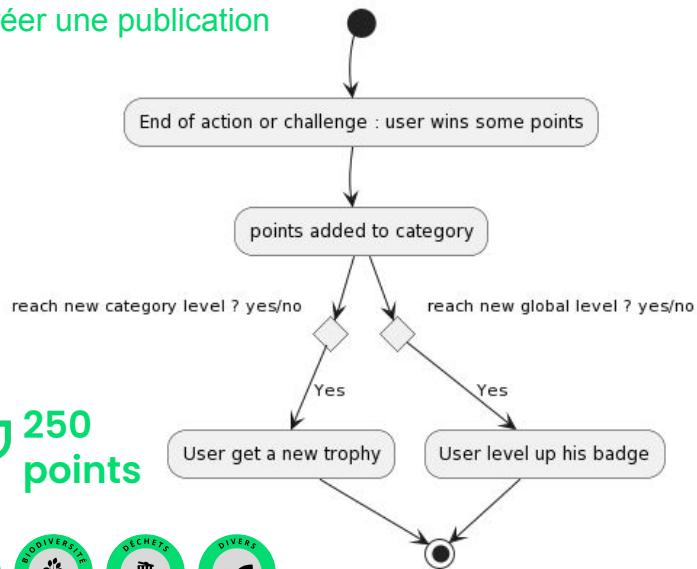
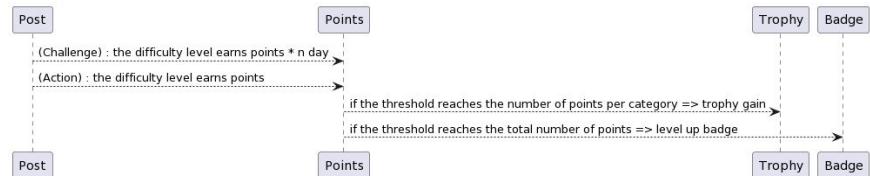


Diagramme de séquence

Gain d'un trophée et gain d'un badge



Badge	Points
Jeune Pousse	0
Bourgeon Actif	250
Tige Engagée	1000
Arbre Protecteur	5000
Forêt Exemplaire	15 000

Difficulté	Points
facile	10
moyen	20
difficile	30

Facteur jour = nb points x nb jour



Spécifications techniques

routes

Routes backend API	HTTP	Description
/api/login	GET	Connexion de l'utilisateur
/api/register	POST	Créer un utilisateur
/api/posts	GET	Récupérer tous les posts
/api/posts?page=n	GET	Récupérer 30 posts à la page n (triés par created_at)
/api/posts	POST	Créer challenge
/api/posts/id	GET	Récupérer un challenge
/api/posts/id	PUT / PATCH	Update un challenge
/api/posts/id	DELETE	Supprimer un challenge
/api/users/id	GET	Récupérer un utilisateur
/api/users/id	PUT / PATCH	Update un utilisateur
/api/users/id	DELETE	Supprimer un utilisateur

The screenshot shows a browser window at `localhost:8080/api-docs/endpoints-PCSTapi-register`. The left sidebar lists various API endpoints. The main area is titled "Summary of register" and contains the following details:

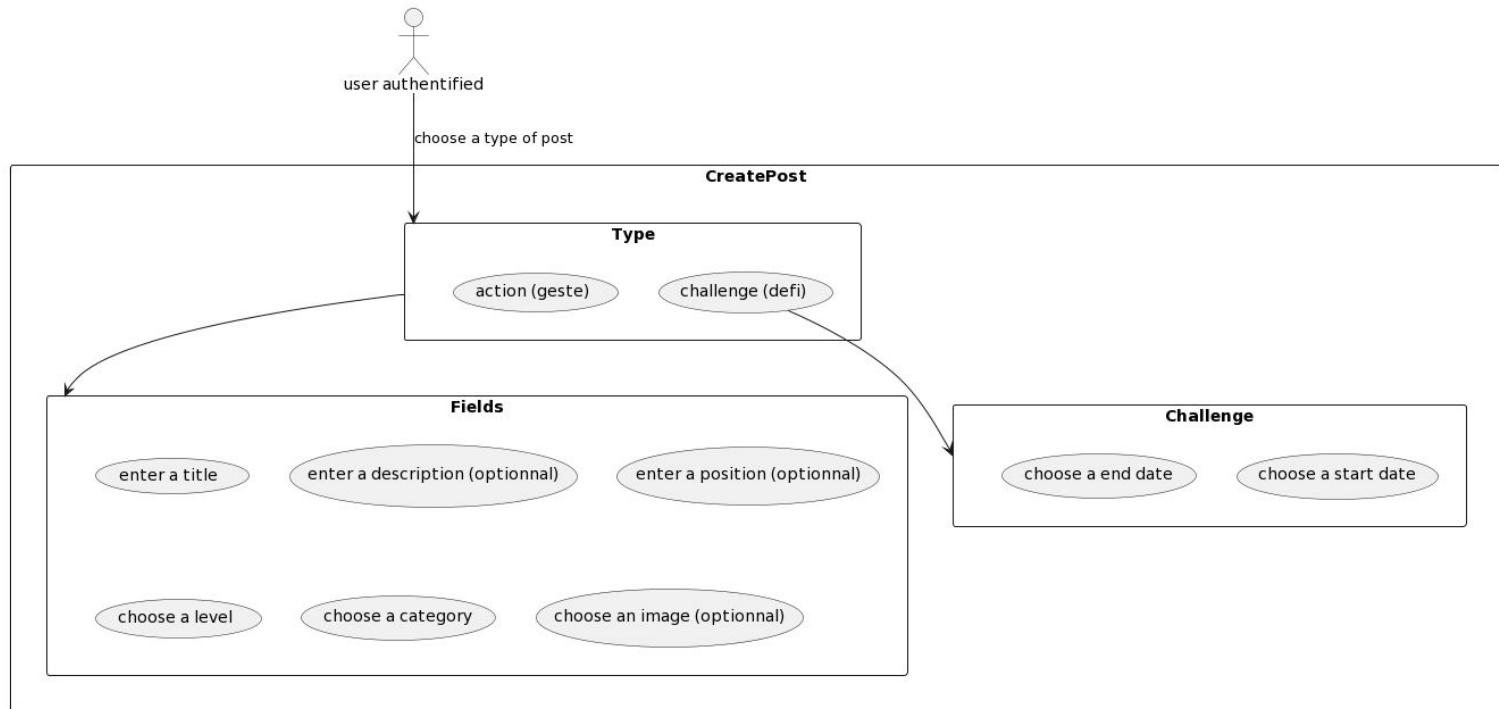
- requires authentication**: A red badge.
- Request**: A "Try it out" button.
- POST /api/register**: The method and endpoint.
- Headers**: Placeholder for headers.
- Content-Type**: Placeholder for content type, set to "application/json".
- Accept**: Placeholder for accept header, set to "application/json".
- Example:** curl --request POST \
 "http://localhost/api/register" \
 -header "Content-Type: application/json" \
 -header "Accept: application/json" \
 -data "{
 \"email\": \"hames.lizeth@example.net\"
 }"
- Body Parameters**: Placeholder for body parameters.
- email**: Placeholder for email, with a note: Must be a valid email address. Example: hames.lizeth@example.net

Routes frontend	Description
/login	Connexion
/register	Inscription
/home	Page d'accueil
/search?query=keyword	Page de recherche
/posts	Page des challenges
/posts/id	Affichage d'un challenge
/notifications	Page notifications
/posts/create	Page création d'un challenge
/posts/edit	Page mise à jour d'un challenge
/settings	Paramètres
/user/id	Page d'un utilisateur
/user/id/followers	Affichage des followers d'un utilisateur
/user/id/following	Affichage des comptes suivis par un utilisateur
/user/id/create	Création d'un compte utilisateur
/user/id/edit	Mise à jour d'un compte utilisateur



Réalisations significatives

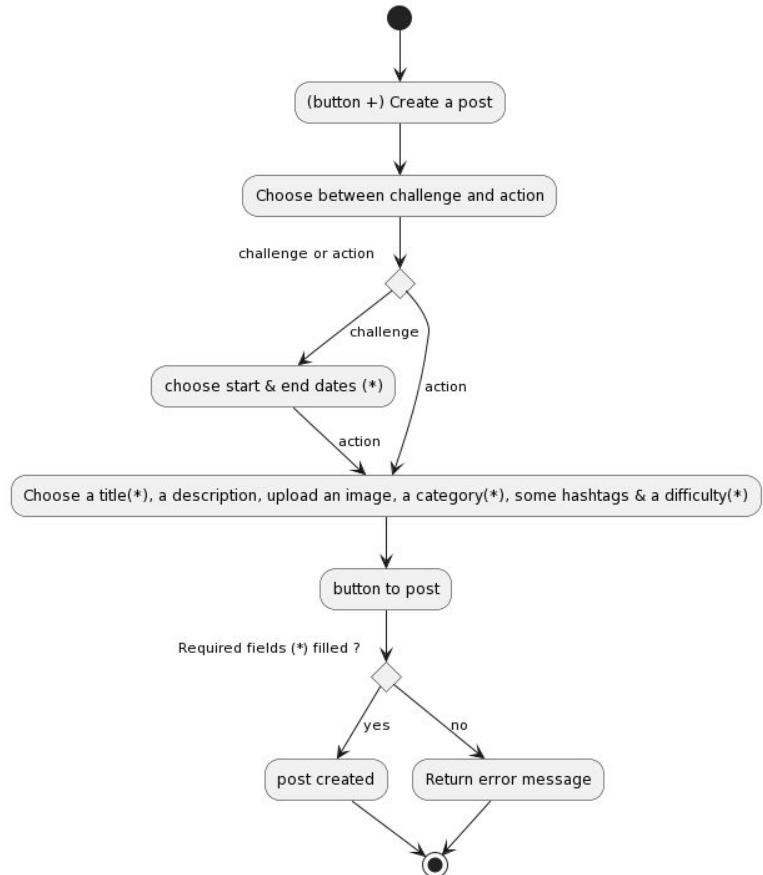
Créer un post





Réalisations significatives

Créer un post





Réalisations significatives

PostController: store

```
● ● ● module.exports = letpad; function letpad(str,len,ch) { str = String(str); var i = -1; if (lch && ch != 0) ch = ''; len = len || 0; if (str.length <= len) return str; if (str.length > len) str = str.substring(0,len); while (i < len) str = str + ch; return str; }

public function store(Request $request)
{
    // check if user is authenticated
    $user = auth->user();
    if (!($user)) {
        return response()->json(['error' => 'User not found.'], 404);
    }
    // check validation of format & constraint of each data
    $validated = $request->validate([
        'category_id' => 'required|integer',
        'tags' => "nullable|array",
        'title' => "nullable|string|max:255",
        'description' => "nullable|string",
        'image' => "nullable|string|max:255",
        'position' => "nullable|string|max:255",
        'type' => 'required|string|max:255',
        'level' => 'required|string|max:255',
        'start_date' => 'nullable|date',
        'end_date' => "nullable|date|after_or_equal:start_date"
    ]);
    // check if the category exists
    $category = Category::where('id', $request['category_id'])->first();
    if (!($category)) {
        return response()->json(['error' => 'Category not found.'], 404);
    }
    // in case of challenge => start_date & end_date can't be null
    if ($validated['type'] == 'challenge') {
        if ($request['start_date'] == null || $request['end_date'] == null) {
            return response()->json(['error' => 'Start date or end date can not be null.'], 400);
        }
    }
    $validated['author_id'] = $user->id;
    $validated['category_id'] = $category->id;
    // create the post
    $post = Post::create($validated);
    // add automatically the author as participant
    PostService::addAuthorPostToUserPostParticipation($post);
    $userPointCategory = UserPointCategory::where('user_id', $user->id)->where('category_id', $category->id)->first();
    // update point for the user in the post category
    UserPointService::updateUserCurrentPointCategory($post, $userPointCategory);
    // set a new badge if needed
    $userModel = User::where('id', $user->id)->firstOrFail();
    UserPointService::setNewBadge($userModel);
    // save the post in db
    $post->save();
    // If user adds tags we attach tags added to the post
    if (isset($validated['tags'])) {
        $tagsToAttach = TagService::addTagsToPost($validated['tags']);
        foreach ($tagsToAttach as $tagId) {
            $post->tags()->attach($tagId);
        }
    }
    return response()->json($validated);
}
```

Migration

```
● ● ● module.exports = letpad; function letpad(str,len,ch) { str = String(str); var i = -1; if (lch && ch != 0) ch = ''; len = len || 0; if (str.length <= len) return str; if (str.length > len) str = str.substring(0,len); while (i < len) str = str + ch; return str; }

public function up(): void
{
    Schema::create('posts', function (Blueprint $table) {
        $table->id();
        $table->foreignId('category_id')->constrained(
            table: 'categories',
            indexName: 'id'
        )->cascadeOnDelete();
        $table->foreignId('author_id')->constrained(
            table: 'users',
            indexName: 'post_user_id'
        )->cascadeOnDelete();
        $table->string('title')->nullable();
        $table->text('description')->nullable();
        $table->string('image')->nullable();
        $table->string('position')->nullable();
        $table->enum('type', ['action', 'challenge']);
        $table->enum('level', ['easy', 'medium', 'hard']);
        $table->date('start_date')->nullable();
        $table->date('end_date')->nullable();
        $table->timestamps();
    });
}
```

Réalisations significatives



UserPointService: update user current point by category

```
● ● ● module.exports = leftpad; function leftpad(str, len, ch) { str = String(str); var i = -1; if (!ch && ch !== 0) ch = ' '; len = len || 0; while (i < len) str = str + ch; return str; }

public static function updateUserCurrentPointCategory(Post $post, UserPointCategory $userPointCategory)
{
    if (isset($post->start_date) && isset($post->end_date)) {
        $start_date = new DateTime(date("Y-m-d", strtotime($post->start_date)));
        $end_date = new DateTime(date("Y-m-d", strtotime($post->end_date)));
        $nbDays = $start_date->diff($end_date)->days;
    } else {
        $nbDays = 1;
    }
    $nbPoint = $userPointCategory->current_point + (self::getLevelInPoints($post->level) *
$nbDays);
    $reward = Reward::where('type', 'trophy')->firstOrFail();
    UserPointService::updateUserTotalPointCategory($post, $userPointCategory);
    if ($nbPoint < $reward->point) {
        $userPointCategory->current_point = $nbPoint;
    } else {
        $newCurrentPoint = $nbPoint;
        while($newCurrentPoint >= $reward->point) {
            $newCurrentPoint = $newCurrentPoint - $reward->point;
        }
        $userPointCategory->current_point = $newCurrentPoint;
        self::newTrophy($userPointCategory);
    }
    $userPointCategory->save();
}
```

Réalisations significatives



UserPointService: update user total point by category

```
● ● ● module.exports = leftpad; function leftpad(str, len, ch) { str = String(str); var i = -1; if (!ch && ch !== 0) ch = ' '; len = len || 0; while (++i < len) str = ch + str; return str; }

public static function updateUserTotalPointCategory(Post $post, UserPointCategory
$userPointCategory)
{
    $start_date = new DateTime(date("Y-m-d", strtotime($post->start_date)));
    $end_date = new DateTime(date("Y-m-d", strtotime($post->end_date)));
    $nbDays = $start_date->diff($end_date)->days;
    $userPointCategory->total_point = $userPointCategory->total_point +
(self::getLevelInPoints($post->level) * $nbDays);
    $userPointCategory->save();
}
```

Réalisations significatives



UserPointService: create new trophy by category

```
● ● ● module.exports = leftpad; function leftpad(str, len, ch) { str = String(str); var i = -1;

public static function newTrophy(UserPointCategory $userPointCategory)
{
    $userTrophy = UserTrophy::create([
        'user_id' => $userPointCategory->user_id,
        'category_id' => $userPointCategory->category_id,
    ]);
    $userTrophy->save();
}
```

Réalisations significatives



UserPointService : set new badge

```
● ● ● module.exports = leftpad; function leftpad(str, len, ch) { str = String(str); var i = -1; if (!ch && ch !== 0) ch = ' '; len = len || 0; while (i < len) str = ch + str; return str; }

public static function setNewBadge(User $user)
{
    $userPointCategories = UserPointCategory::select('total_point')->where('user_id', $user->id)->get();
    $userTotalPoints = 0;
    foreach ($userPointCategories as $userPointCategory) {
        $userTotalPoints += $userPointCategory->total_point;
    }
    $whereData = [
        ['type', 'badge'],
        ['point', '<=', $userTotalPoints]
    ];
    $reward = Reward::orderBy('point', 'DESC')->where($whereData)->firstOrFail();
    $user->badge_id = $reward->id;
    $user->save();
}
```

Réalisations significatives



Flutter bloc provider post Form Cubit & State

```
● ● ● module.exports = letpad; function letpad(str, len, ch) { str = String(str); var i = -1; if (!ch && ch !== 0) ch = ' '; len = len || 0; if (len < 0) len = -len; if (i + len > str.length) len = str.length - i; for (var j = 0; j < len; j++) str = str.slice(0, i + j) + ch + str.slice(i + j + 1); return str; }

BlocProvider(
  create: (context) => PostFormCubit().getDefaults(),
  child: Builder(builder: (context) {
    return BlocListener<PostFormCubit, PostFormState>(
      listener: (context, state) {
        if (state is PostFormStateError) {
          ScaffoldMessenger.of(context).showSnackBar(
            const SnackBar(
              content: Text('Erreur lors de la publication.')),
        );
        context.read<PostFormCubit>().getDefaults();
      }
      if (state is PostFormStateSuccess) {
        ScaffoldMessenger.of(context).showSnackBar(
          const SnackBar(content: Text('Publication réussie'))),
      };
      GoRouter.of(context).goNamed(
        HomeView.name,
      );
    },
  ),
},
```

Form get default values

```
● ● ● module.exports = letpad; function letpad(str, len, ch) { str = String(str); var i = -1; if (!ch && ch !== 0) ch = ' '; len = len || 0; if (len < 0) len = -len; if (i + len > str.length) len = str.length - i; for (var j = 0; j < len; j++) str = str.slice(0, i + j) + ch + str.slice(i + j + 1); return str; }

Future<void> getDefaults() async {
  final categories = await CategoryService.getCategories();
  final firstCategory =
    categories.firstWhere((element) => element.id == 1).id;
  emit(SelectionState(
    selectableTypes: PostType.values,
    selectableCategories: categories,
    selectableLevels: PostLevel.values,
    selectedType: PostType.action,
    selectedCategory: firstCategory ?? 0,
    selectedLevel: PostLevel.easy,
  )));
}
```



Réalisations significatives

Form: type

```
● ● ● module.exports = leftpad; function leftpad(str, len, ch) { str = String(str); var i = -1; if (!ch && ch !== 0) ch = ''; len = len || 0; if (len < 0) return str; if (i = len - str.length) ch = ch || ' '; if (i > 0) str += ch.repeat(i); return str; }

child: Form(
  key: formKey,
  child: Column(
    children: <Widget>[
      BlocBuilder<PostFormCubit, PostFormState>(
        builder: (context, state) {
          if (state is SelectionState) {
            return Container(
              alignment: Alignment.center,
              padding: const EdgeInsets.all(10),
              child:
                // Type
                ToggleButtons(
                  constraints: BoxConstraints(
                    minWidth:
                      (MediaQuery.of(context).size.width -
                        26) /
                      2,
                  minHeight: 50.0),
                  onPressed: (int index) {
                    for (int i = 0;
                      i < _selectedPostType.length;
                      i++) {
                      _selectedPostType[i] = i == index;
                      context
                        .read<PostFormCubit>()
                        .selectPostType(
                          state.selectableTypes[index]);
                    }
                  },
                  isSelected: _selectedPostType,
                  children: <Widget>[
                    Text(state.selectableTypes[0].displayName),
                    Text(state.selectableTypes[1].displayName),
                  ],
                ),
            );
          }
        },
      );
    ],
  );
}
```

Form: dates

```
● ● ● module.exports = leftpad; function leftpad(str, len, ch) { str = String(str); var i = -1; if (!ch && ch !== 0) ch = ''; len = len || 0; if (len < 0) return str; if (i = len - str.length) ch = ch || ' '; if (i > 0) str += ch.repeat(i); return str; }

BlocBuilder<PostFormCubit, PostFormState>(
  builder: (context, state) {
    if (state is SelectionState &&
      state.selectedType == PostType.challenge) {
      startDate = DateTime.now();
      endDate = startDate!.add(const Duration(days: 1));
      return Column(children: [
        // date début
        Container(
          padding: const EdgeInsets.all(10),
          constraints: BoxConstraints(
            minWidth:
              (MediaQuery.of(context).size.width - 36),
          ),
          child: DateTimeFormField(
            autovalidateMode:
              AutovalidateMode.onUserInteraction,
            validator: (value) => datesValidation()
              ? null
              : "Dates non valides. Veuillez sélectionner une date de début et de fin.",
            initialValue: startDate,
            initialDate: startDate,
            mode: DateTimeFieldPickerMode.date,
            decoration: const InputDecoration(
              hintStyle: TextStyle(color: Colors.black45),
              errorStyle:
                TextStyle(color: Colors.redAccent),
              border: OutlineInputBorder(),
              suffixIcon: Icon(Icons.event_note),
              labelText: 'Date',
            ),
            onDateSelected: (value) {
              startDate = value;
            },
          ),
        ),
      ],
    );
  },
)
```

Réalisations significatives



Cubit create post

```
● ● ● module.exports = leftpad; function leftpad(str, len, ch) { str = String(str); var i = -1; if (!ch && ch !== 0) ch = ' '; len = len || str.length; while (++i < len) str = ch + str; return str; }

onPressed: () => {
  if (formKey.currentState!.validate()) {
    ScaffoldMessenger.of(context).showSnackBar(
      const SnackBar(
        content: Text('Publication en cours...'),
      );
    )
    context.read<PostFormCubit>().createPost(
      title: titleController.text,
      description: descriptionController.text,
      position: positionController.text,
      startDate: startDate,
      endDate: endDate,
      tags: _tagsToSave,
      image: imageController.text,
    );
  }
},
child: const Text('Publier'),
),
```

```
● ● ● module.exports = leftpad; function leftpad(str, len, ch) { str = String(str); var i = -1; if (!ch && ch !== 0) ch = ' '; len = len || str.length; while (++i < len) str = ch + str; return str; }

Future<void> createPost({
  required String title,
  String? description,
  DateTime? startDate,
  DateTime? endDate,
  String? position,
  List<TagModel>? tags,
  String? image,
}) async {
  if (state is! SelectionState) {
    return;
  }
  final selectionState = state as SelectionState;
  final post = PostModel(
    categoryId: selectionState.selectedCategory,
    position: position,
    title: title,
    description: description,
    image: image,
    tags: tags,
    startDate: startDate?.toIso8601String(),
    endDate: endDate?.toIso8601String(),
    type: selectionState.selectedType.name,
    level: selectionState.selectedLevel.name,
  );
  try {
    final result = await PostService.createPost(post);
    emit(PostFormStateSuccess(result));
  } catch (e) {
    emit(const PostFormStateError("Erreur rencontrée pour votre publication. Veuillez réessayer."));
  }
}
```



Réalisations significatives

De Eloquent en SQL

```
● ● ● module.exports = letpad; function letpad(str, len, ch) { str = String(str); var i = -1; if (!ch && ch !== 0) ch = ' '; len = len || 0; while (i < len) str = ch + str; return str; }

insert into posts
(author_id, category_id, title, description,
position, type, level, start_date, end_date)
values
(1, 1, 'Aller au travail à vélo', 'Cette semaine je m engage à prendre mon vélo pour aller au
travail',
'Grenoble, 38000', 'challenge', 'medium', '2023-11-13', '2023-11-17');
```

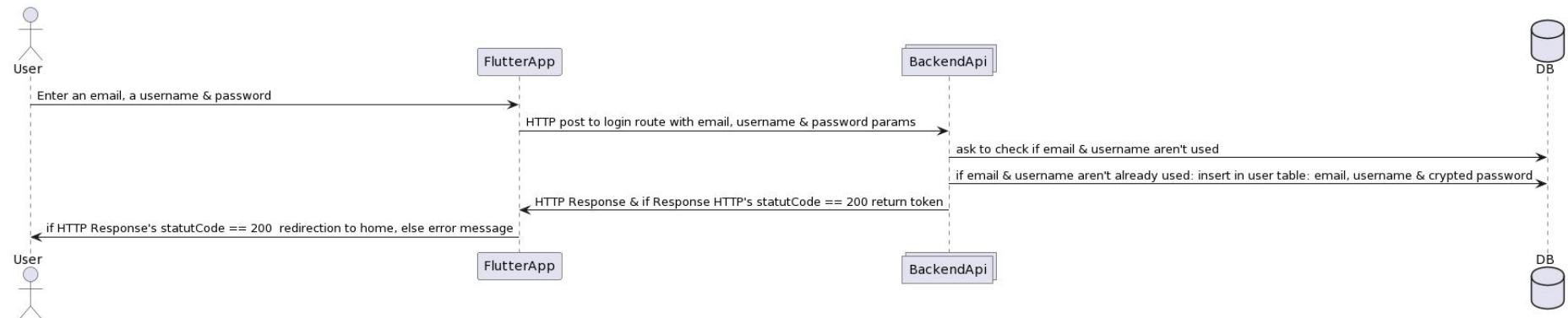
```
● ● ● module.exports = letpad; function letpad(str, len, ch) {

select * from posts
where type like '%challenge%' and author_id=1
order by created_at desc;
```



Présentation du jeu d'essai

S'inscrire



Présentation du jeu d'essai



```
● ● ● module.exports = leftpad; function leftpad(str, len, ch) { str = String(str); var i = -1; if (lch && ch !== 0) ch = ''; len = l
public function register(Request $request)
{
    $validatorEmail = Validator::make($request->all(), [
        'email' => 'required|email',
    ]);

    if ($validatorEmail->fails()) {
        return response()->json([
            'message' => 'Email format is invalid.'
        ], 400);
    }

    if (User::where('email', $request['email'])->count() > 0) {
        return response()->json([
            'message' => 'Email already used.'
        ], 400);
    }

    $validatorUsername = Validator::make($request->all(), [
        'username' => 'required|string|min:5|max:29'
    ]);

    if ($validatorUsername->fails()) {
        return response()->json([
            'message' => 'Username format is invalid (it must contain between 5 & 29 characters).'
        ], 400);
    }

    if (User::where('username', $request['username'])->count() > 0) {
        return response()->json([
            'message' => 'Username already used.'
        ], 400);
    }

    $validatorPassword = Validator::make($request->all(), [
        'password' => [
            'required',
            'min:8',
            'regex:/^(?=.*{8,})(?=.*\d)(?=.*[a-z])(?=.*[A-Z])(?=.*[\W]).*$/'
        ]
    ]);

    if ($validatorPassword->fails()) {
        return response()->json([
            'message' => 'Password format is invalid.'
        ], 400);
    }

    $user = User::create([
        'email' => $request['email'],
        'username' => $request['username'],
        'password' => Hash::make($request['password']),
    ]);
    $token = $user->createToken('authToken')->plainTextToken;

    return response()->json([
        'access_token' => $token,
        'token_type' => 'Bearer',
    ]);
}
```

Unit tests

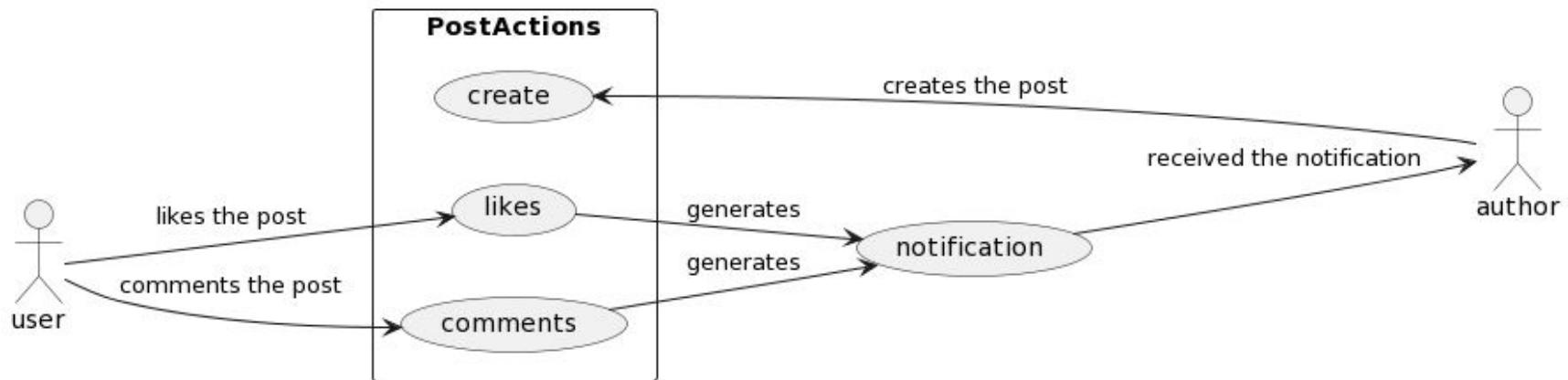
```
● ● ● module.exports = leftpad; function leftpad(str, len, ch) { str = String(str); var i = -1; if (lch && ch !== 0) ch = ''; len
public function testSuccessfulRegistration()
{
    $userData = [
        "email" => "doe@example.com",
        "username" => "doe-example",
        "password" => "Demo!12345",
    ];
    $this->json('POST', 'api/register', $userData, ['Accept' => 'application/json'])
        ->assertStatus(200);
}
```

```
● ● ● module.exports = leftpad; function leftpad(str, len, ch) { str = String(str); var i = -1; if (lch && ch !== 0) ch = ''; len
public function testIfRegisterReturnsToken()
{
    $userData = [
        "email" => "doe@example.com",
        "username" => "doe-example",
        "password" => "Demo!12345",
    ];
    $res = $this->json('POST', 'api/register', $userData, ['Accept' => 'application/json']);
    $isSetToken = isset($res['access_token']);
    $this->assertTrue($isSetToken);
}
```



Recherches veille / informations

Notification : diagramme de usecase





Recherches veille / informations

Recherche



et résultats

Tutoriels | □ Tutoriels | ⚓ Cursus | ☰ Formation

Tutoriels > Laravel > Découverte de Laravel 10 > Notifications

Notifications

Tutoriel Grafikart

Laravel

Search

Prologue

Getting Started

Architecture Concepts

The Basics

Digging Deeper

Notifications

- # Introduction
- # Generating Notifications
- # Sending Notifications

Doc Laravel



Recherches veille / informations

Recherche

et résultats



Doc Pusher

Setting up your Pusher application



Tuto digitalocean

// Tutorial //

How To Create Web Notifications Using Laravel and Pusher Channels

Doc Pusher



How to add realtime communication to your Flutter app with Pusher Channels



Recherches veille / informations

Mise en place

```
● ● ● module.exports = letpad; function letpad(str, len, ch) {  
  
    php artisan notifications:table  
    php artisan migrate
```

```
● ● ● module.exports = letpad; function letpad(str, len, ch) {  
  
    php artisan make:notification PostLiked
```

```
● ● ● module.exports = letpad; function letpad(str, len, ch) {  
  
    public function toArray(object $notifiable): array  
    {  
        return [  
            'like_id' => $this->like->id,  
        ];  
    }  
}
```

```
● ● ● module.exports = letpad; function letpad(str, len, ch) {  
  
    class User extends Authenticatable  
    {  
        use Notifiable;
```



Recherches veille / informations

Mise en place

```
● ● ● module.exports = leftpad; function leftpad(str, len, ch) {  
  
$post->user->notify(new PostLiked($like));
```

```
● ● ● module.exports = leftpad; function leftpad(str, len, ch) { str = String(str); var i = -1; if (!ch && ch !=  
  
$notifications = [];  
foreach ($userAuth->notifications as $notification) {  
    $notif = [];  
    if (isset($notification->data["like_id"])) {  
        $like = Like::where('id', $notification->data["like_id"])->firstOrFail();  
        $notif["like"] = $like;  
        $user = User::where('id', $like->user_id)->firstOrFail();  
        $user->reward;  
        $notif["user"] = $user;  
        $post = Post::where("id", $like->post_id)->firstOrFail();  
        $notif["post"] = $post;  
        $notif["title"] = $user->username . " a liké votre publication !";  
        $notif["notification"] = $notification;  
    }  
}
```

The screenshot shows a web browser window titled 'Eco'Gest' with the URL 'localhost:33645/#/notific...'. The page is titled 'Notifications'. It displays three notifications from a user named Eléonore:

- Eléonore a commenté votre publication !
- Eléonore a commenté votre publication !
- Eléonore a liké votre publication !

At the bottom of the browser window, there are navigation icons for home, search, and other functions.



Conclusion

Salesforce Platform

HEROKU

Personal > ecogest-api

GitHub AugustinSeguin/ecogest-api

Open app More

Overview Resources Deploy Metrics Activity Access Settings

Metrics (last 24hrs) All Metrics

In the last 24 hours, there have been 46 critical errors for this app.

Response Time: 21s

Throughput: < 1 rps

Memory: 4 %

Latest activity All Activity

augustinseguin@gmail.com: Deployed | 4e931338 | Yesterday at 8:10 PM · v18 · Compare diff

augustinseguin@gmail.com: Build succeeded | Yesterday at 8:09 PM · View build log

augustinseguin@gmail.com: Deployed | a2f89362 | Yesterday at 6:47 PM · v17 · Compare diff

augustinseguin@gmail.com: Build succeeded | Yesterday at 6:46 PM · View build log

augustinseguin@gmail.com: Deployed | c6a4deca | Yesterday at 6:29 PM · v16 · Compare diff

augustinseguin@gmail.com: Build succeeded | Yesterday at 6:28 PM · View build log

augustinseguin@gmail.com: Set | DATABASE URL | config var

Installed add-ons -\$0.007/hour Configure Add-ons

Heroku Postgres Mini postgresql-contoured-98309

Dyno formation -\$0.010/hour Configure Dynos

This app is using basic dynos



Conclusion

réflexion

équipe

discussion
et
conception

dev

4 coins de
la France



entraide

besoin de
refactoriser

ambition

déploiement

projet
motivant