- FIN DE TRONC COMMUN -

- PROJET YOWL -
“URL-LAND : Le voyage par le partage”

par
Anaïs Letellier - Runyi Wang - Clémence Vasseur

PRESENTATION DU PROJET

BUT : Création d'un site communautaire et participatif inspiré du fonctionnement de REDDIT.

reddit

PROBLEMATIQUES :

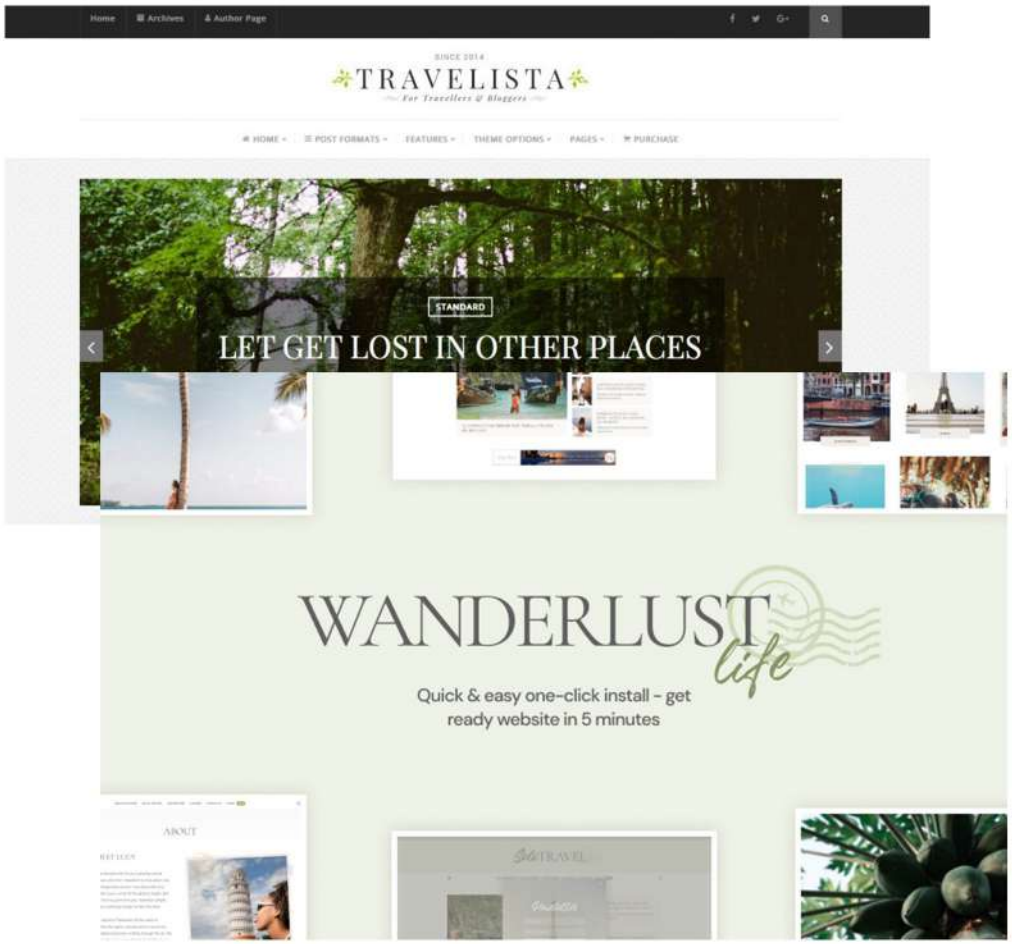
- Gérer le CRUD des Users, des Posts, des Commentaires de posts et de la partie Admin.

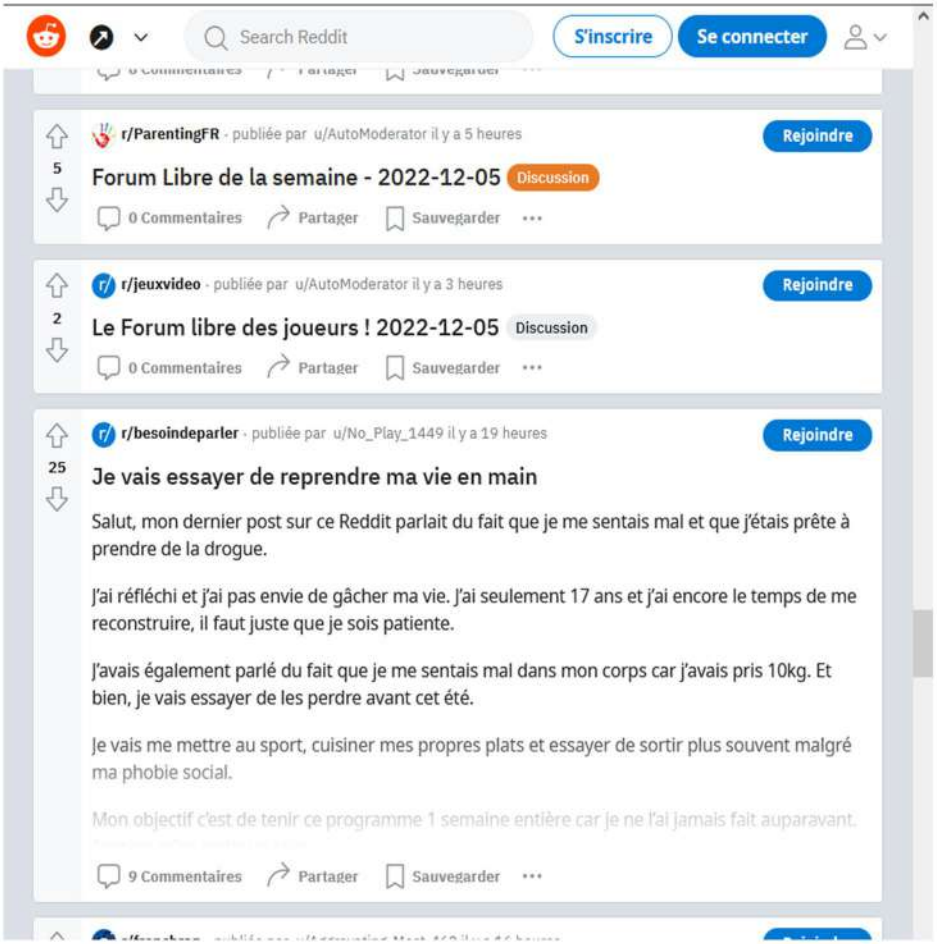
- Les users étant humains - et parfois en proie aux émotions négatives-, l'Administrateur se voit octroyé des droits de Modération en ayant son propre CRUD des Users, Posts et Commentaires.

- La partie Admin contient également une partie STAT permettant la gestion du flux et une veille concernant le site.

PRESENTATION DU PROJET

Nous sommes partis d'une volonté d'avoir un site type blog de voyage, tout en gardant la vue globale et centrale de Reddit :





Nous avions le souhait de rendre agréable la consultation et la participation au site communautaire. Reddit étant assez rébarbatif, voir archaïque, nous voulions changer cela.

PRESENTATION DU PROJET

Gamme de couleur nature/neutre

Réutilisation de la combinaison d'un Titre imposant et minimaliste avec un sous-titre fluide et léger.

Une vue centrale rassemblant les posts, les commentaires et les filtres.

Intégration de photo sur la page d'accueil

Boutons permettant la redirection vers une page LOGIN/REGISTER.

Affichage d'un post en cliquant simplement sur la zone "post".

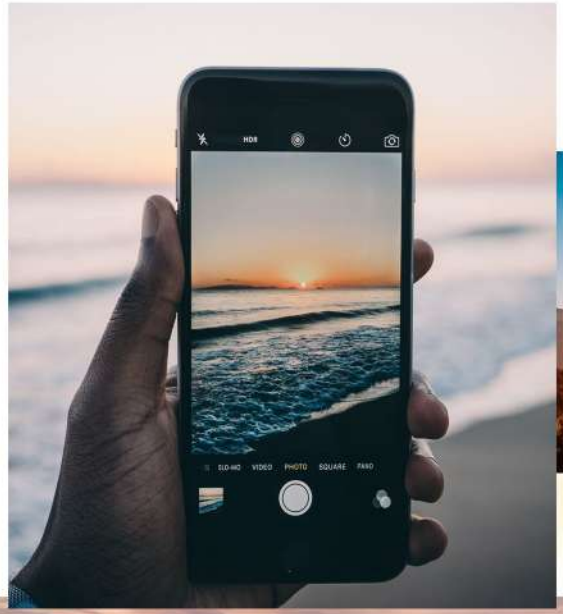
MOODBOARD & CHARTE GRAPHIQUE

Pour choisir notre ambiance de site, nous avons définis des mots clefs suivants... :

- NATURE
- VOYAGE/ESCAPADE
- LIBERTE
- REVE
- PARTAGE
- CONNEXION
- RESEAU SOCIAL

... nous donnant accès aux images inspirantes suivantes :

et encore... ce n'est qu'une maigre séléction...





MOODBOARD & CHARTE GRAPHIQUE

Notre choix s'est porté sur l'Irlande pour ses couleurs, ses paysages, l'ambiance...

et la Guinness mais ça... c'est surtout Klaym...

MOODBOARD & CHARTE GRAPHIQUE

Notre en avons tiré un titre, un slogan et une charte graphique, inspiré de l'Irlande, des excurtions et de ce côté apaisant et hors du temps qu'offre le voyage.

IRLANDE

IRL-LAND (jeu de mot entre In Real Life et Irlande mais c'était trop orienté spécial Irlande...)

URL-LAND a finalement été choisi...

Et comme nous sommes sur un site de communauté et de partage (des tips de voyage, des idées d'excurtions, des bons plans etc.)... nous avons retenu le slogan "le voyage par le partage".

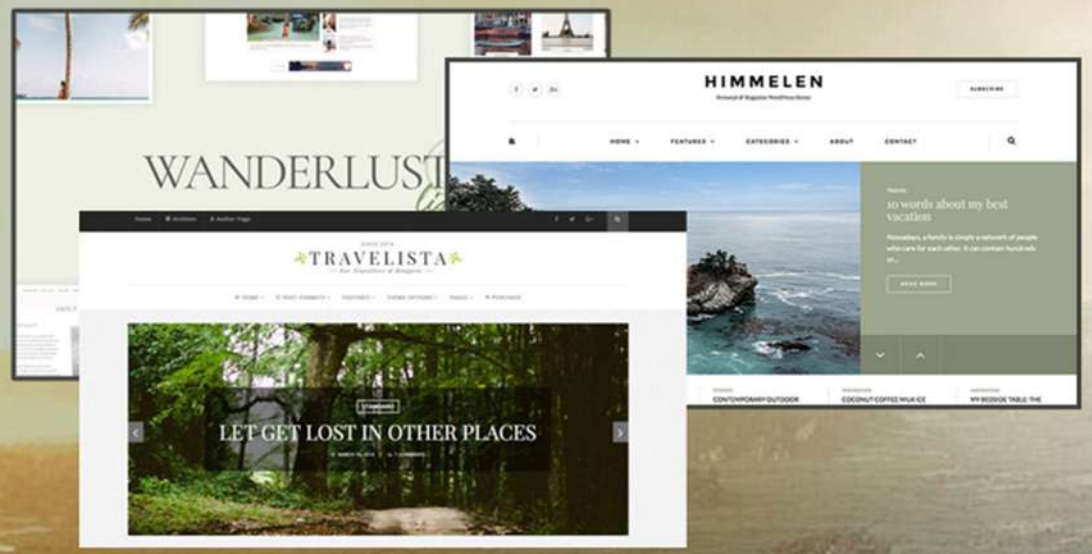
URL-LAND

le Voyage par le Partage...

MOODBOARD & CHARTE GRAPHIQUE

Ce qui nous donne...

Moodboard - URL-LAND



inspiration site
+ blog type "voyage"

Retour d'expérience, astuces de voyage, partage de bons plans...

MOODBOARD & CHARTE GRAPHIQUE

URL-LAND

Le partage du voyage...

POLICES :

google font - titre

ANTON

google font - sous titre

La Belle Aurore

google font - corps

Roboto Condensed

#FFFEF6

#B8CBD0

#A47E16

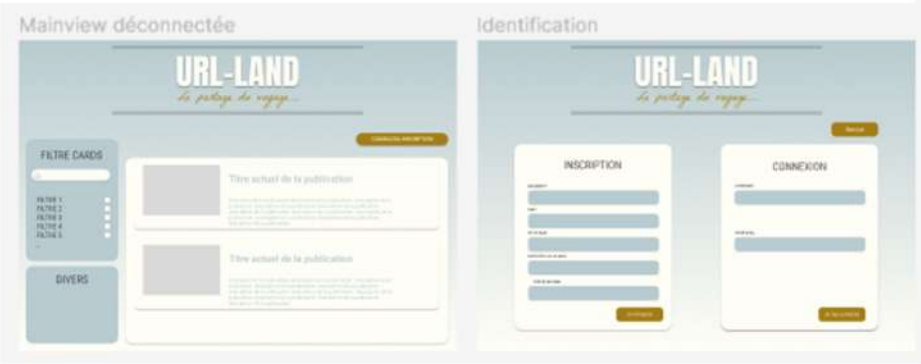
#B0B9A8

#606C5A

#424340

FIGMA & LA MISE EN PLACE GRAPHIQUE DU SITE

Réalisation du FIGMA :



FIGMA & LA MISE EN PLACE DU SITE - COTE ADMIN

Ceci est le côté ADMIN du site.

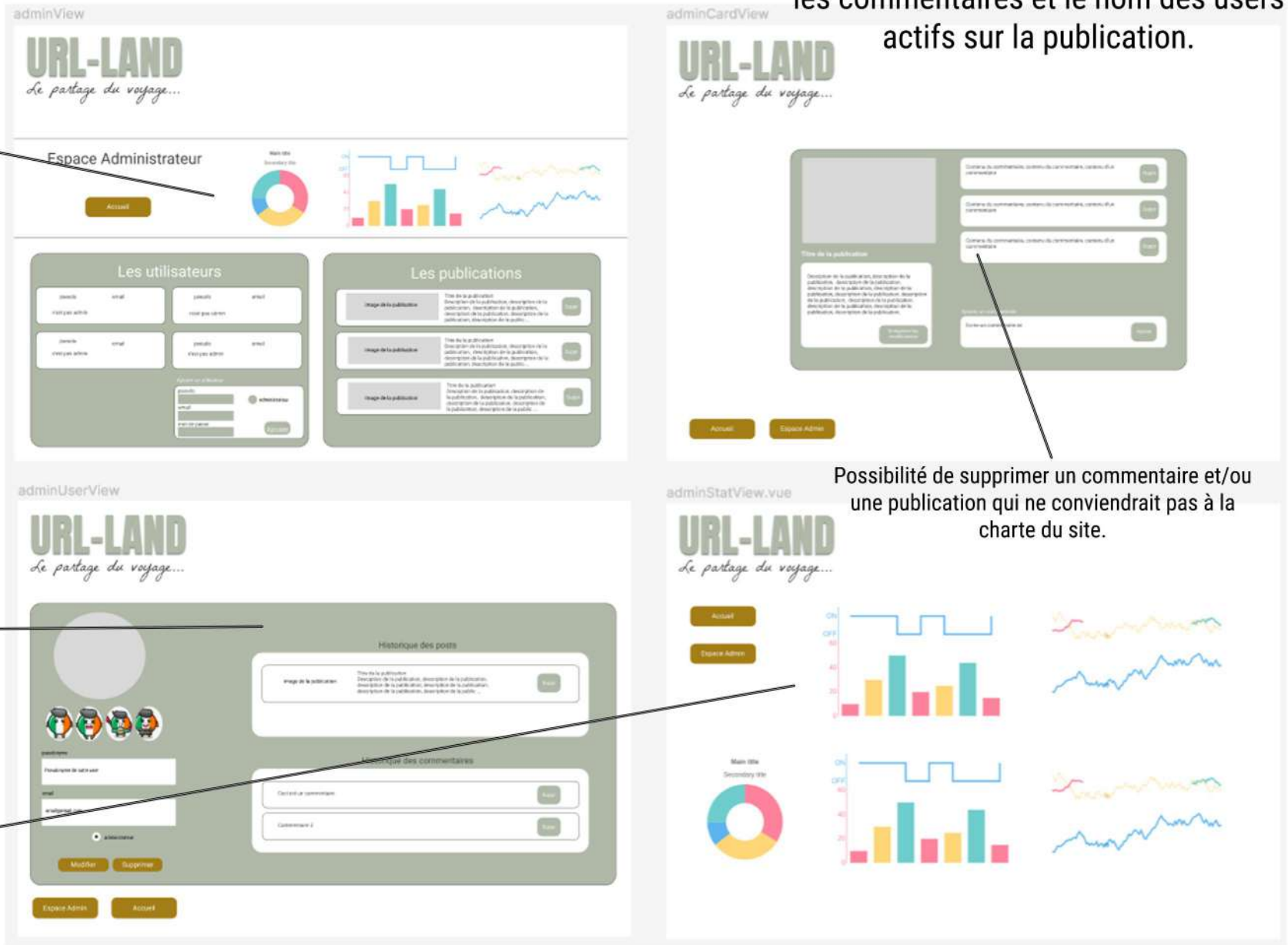
Page d'accueil rassemblant toutes les informations du site (Stats, users, publications etc.)

Vue de chaque publication avec les commentaires et le nom des users actifs sur la publication.

Possibilité de supprimer un commentaire et/ou une publication qui ne conviendrait pas à la charte du site.

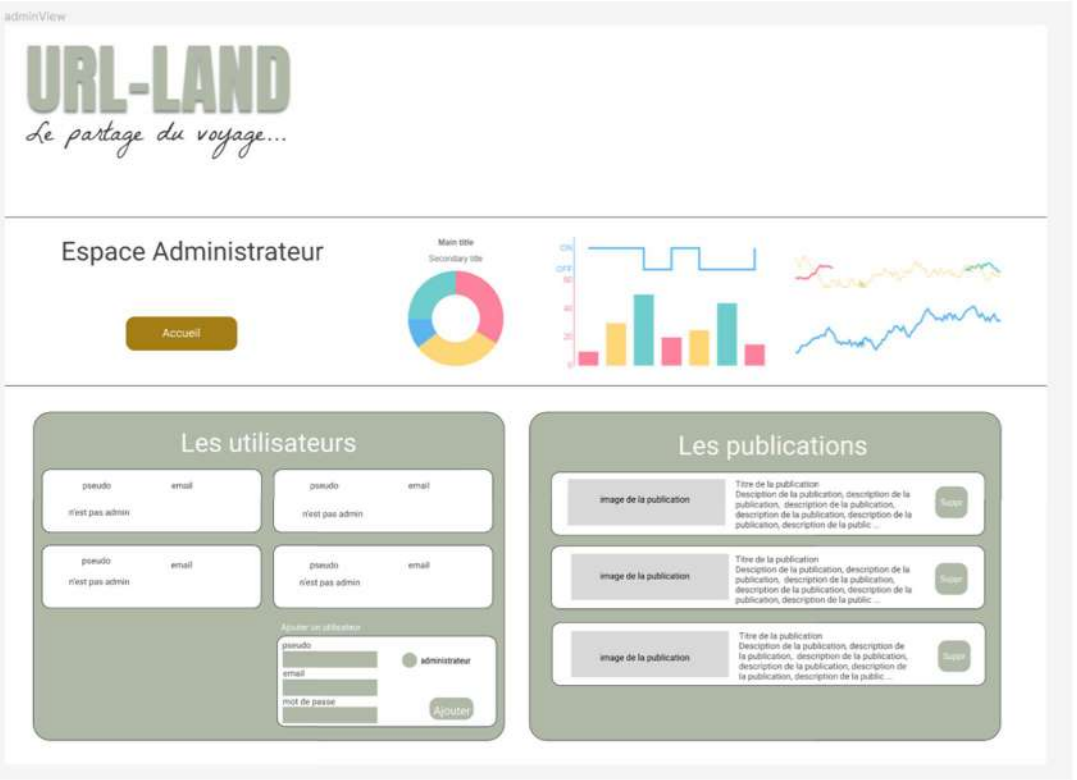
Profil de chaque user + historique

Gestion des Stats



FIGMA & LA MISE EN PLACE GRAPHIQUE DU SITE

Différence entre MainView Client et Admin

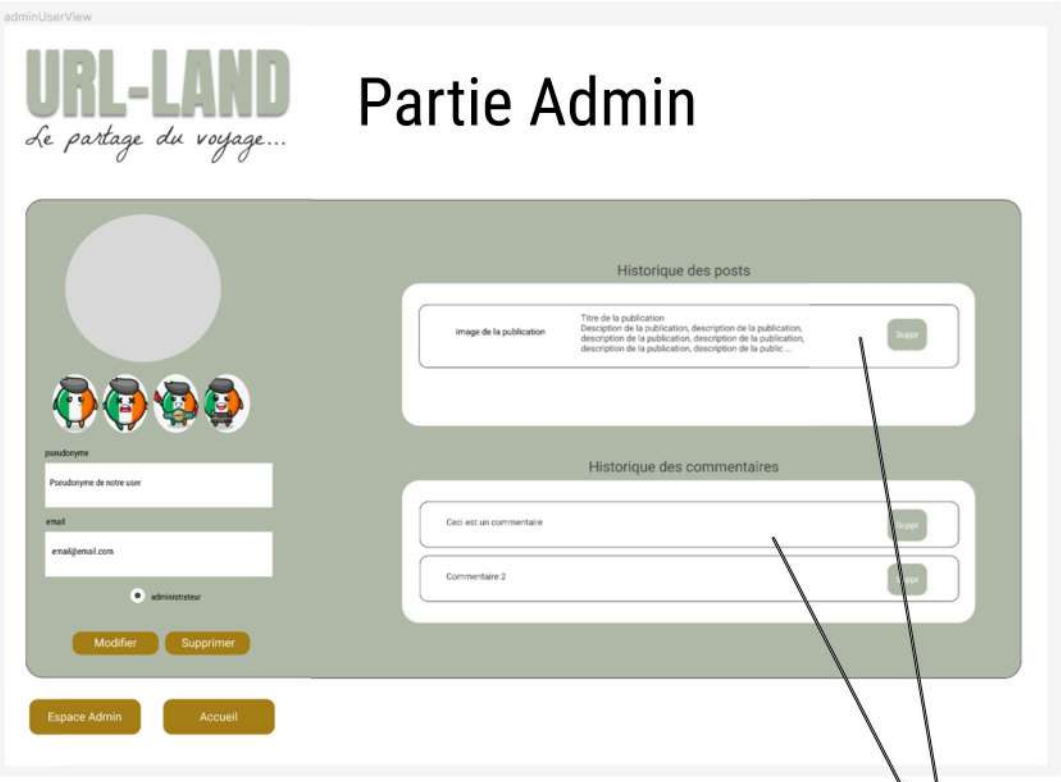


Partie Client

Partie Admin

FIGMA & LA MISE EN PLACE GRAPHIQUE DU SITE

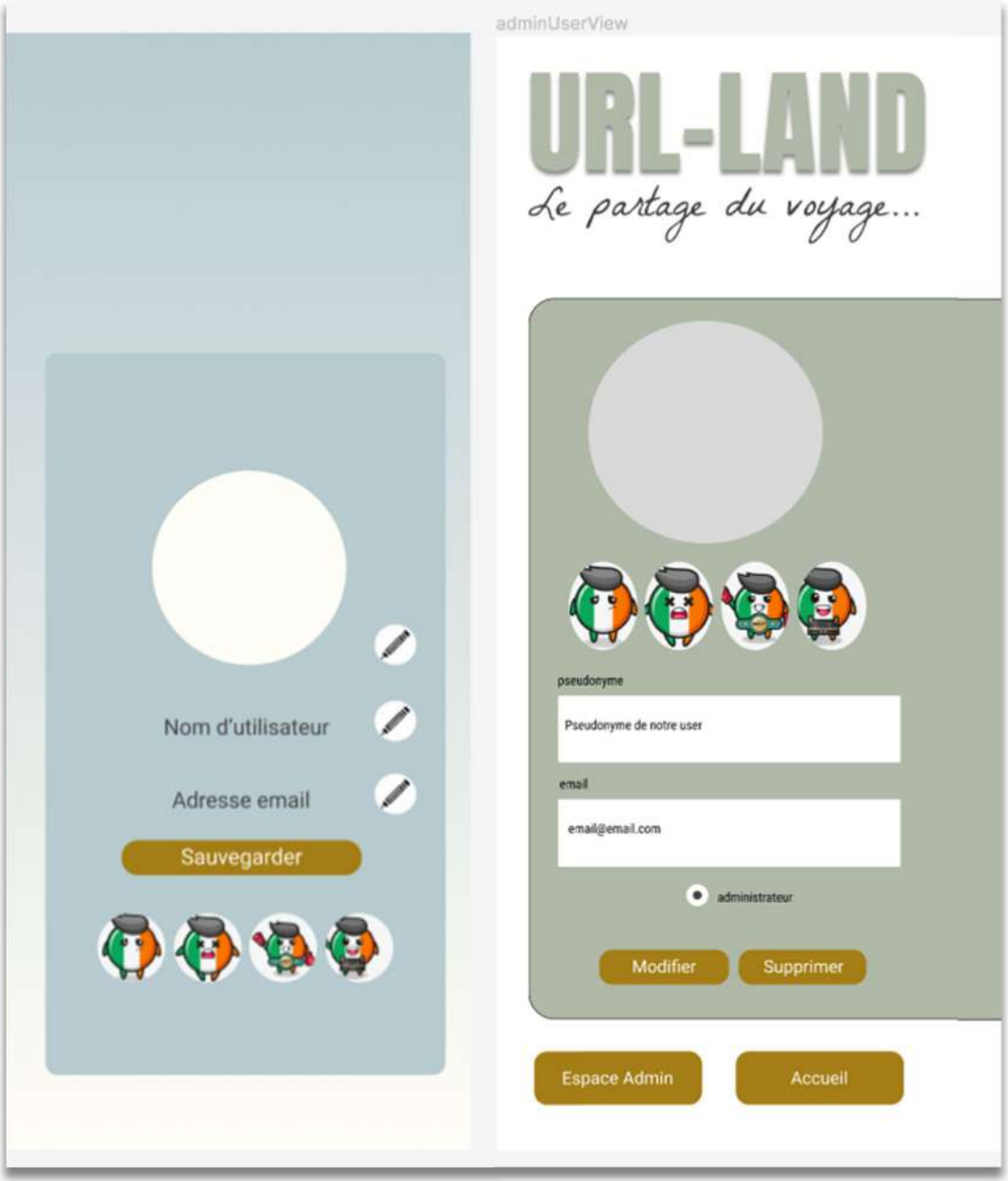
Gestion des pages de profil utilisateur, côté Client et côté Admin :



Historique des publications et des commentaires laissés par le.a user connecté.e à son profil, pour une recherche plus rapide du flux d'échange et de conversation.

l'Admin, en tant que Moderateur a également accès à l'historique des publications et des commentaires d'un.e user.

FIGMA & LA MISE EN PLACE GRAPHIQUE DU SITE

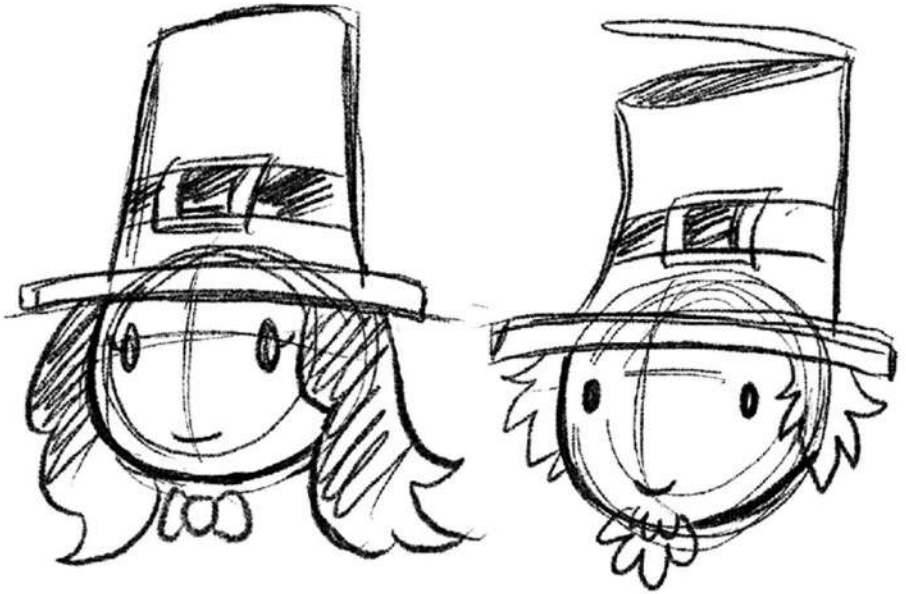


Nous voulions une fonction qui permette à l'utilisateur.rice de choisir parmis une sélection d'avatars pré-créés.

Cela permet une personnalisation de son profil tout en restant dans une charte cohérente et en gérant nous même ce qui est dans notre DB.
Cela permet également de mettre tout le monde sur un pied d'égalité technique et sociétal en ayant toustes la même base d'avatar.

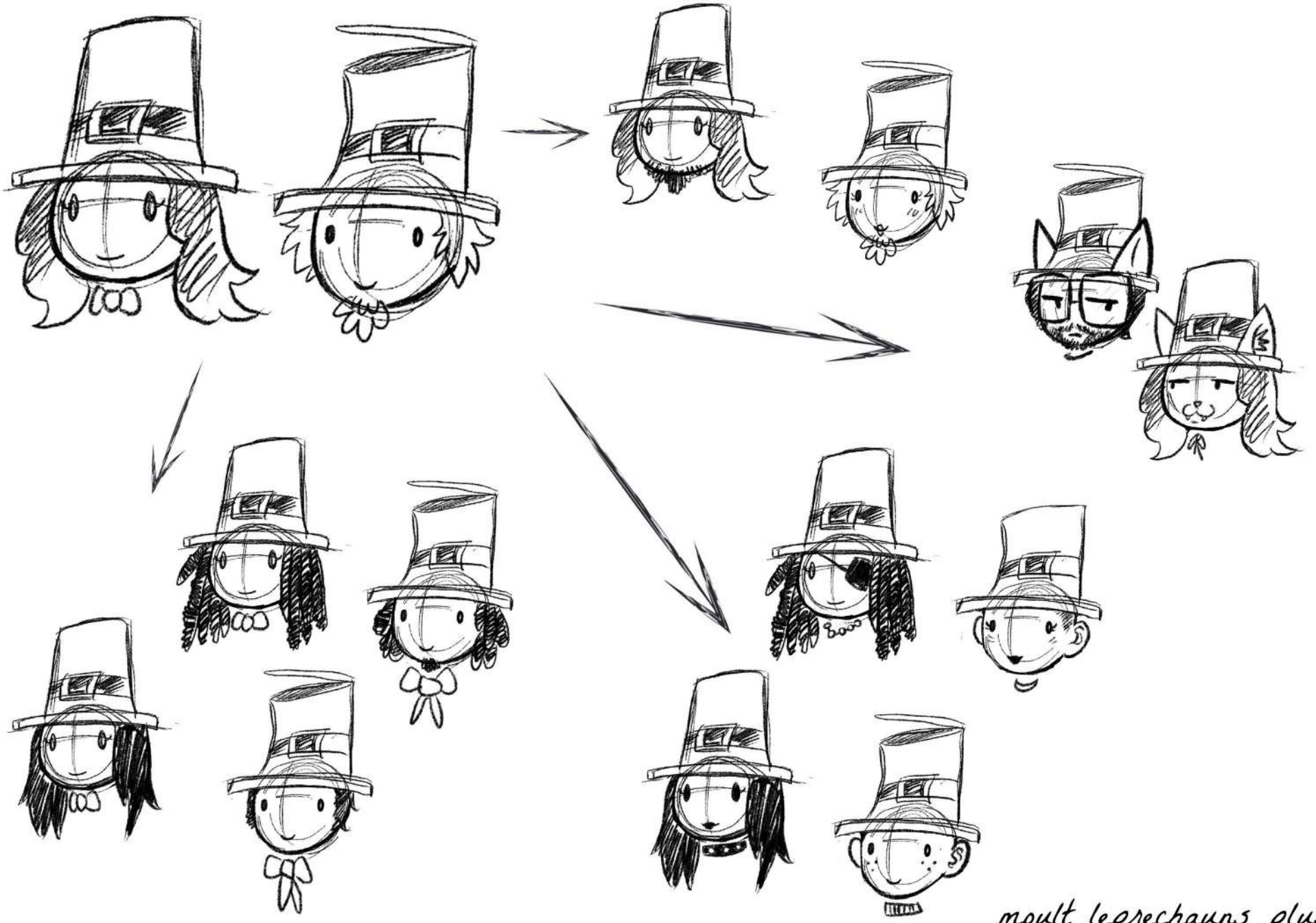
Pour créer des avatars uniques, nous sommes partis du folklore irlandais et de la légende des Leprechauns...

FIGMA & LA MISE EN PLACE GRAPHIQUE DU SITE



premières idées d'avatars

FIGMA & LA MISE EN PLACE GRAPHIQUE DU SITE

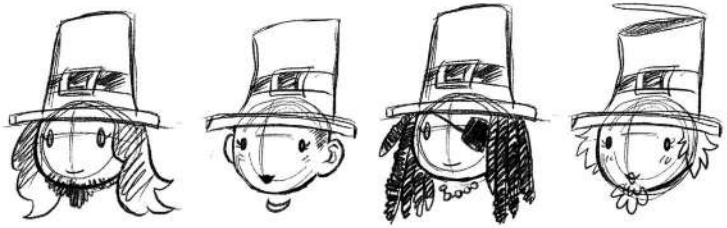


moult leprechauns plus tard...

FIGMA & LA MISE EN PLACE GRAPHIQUE DU SITE

Premiers avatars finaux
(représentant les trois membres
du groupe + notre cher référent).

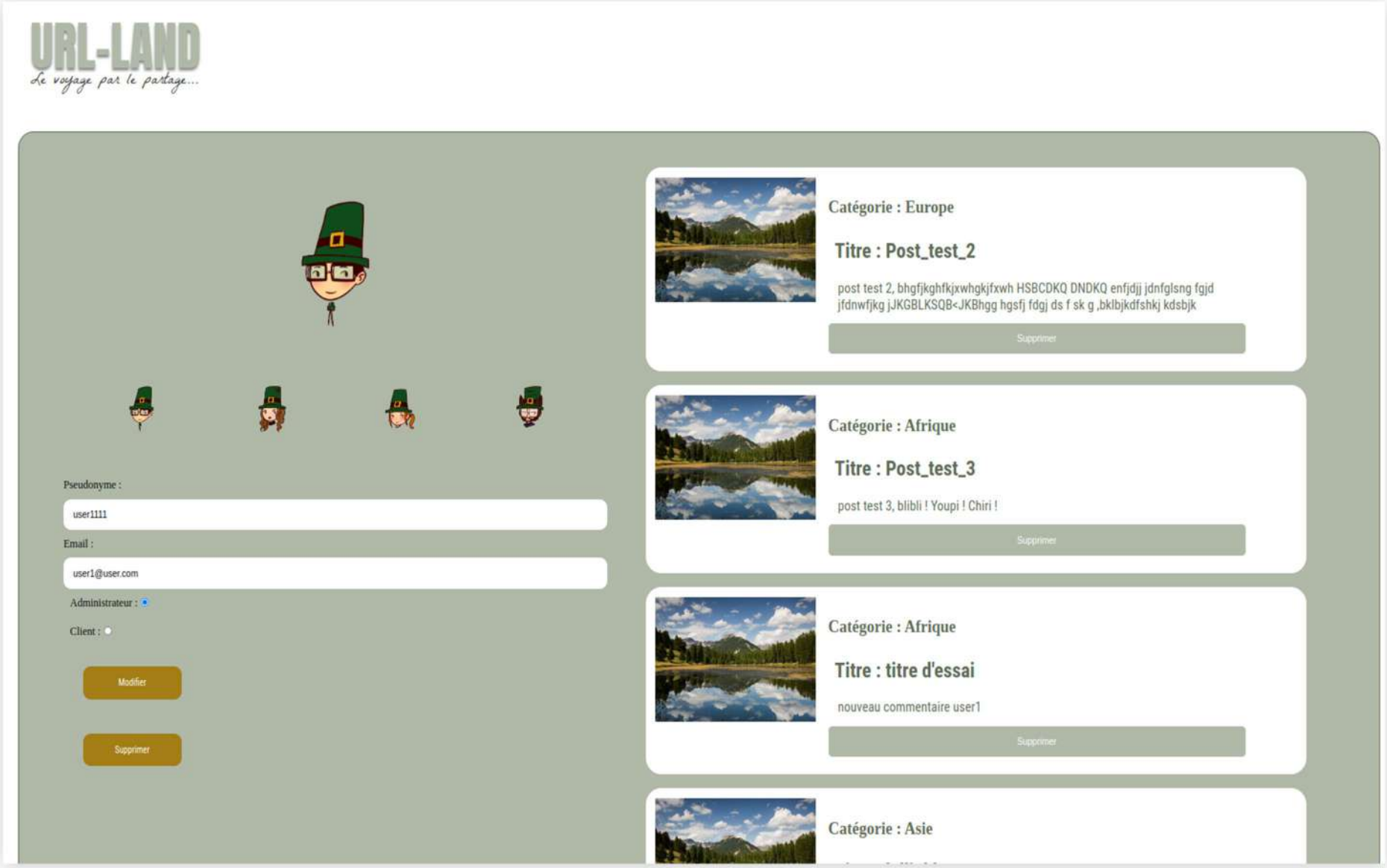
non... ce n'est même pas pour
avoir 15 crédits supplémentaires...



...Le but étant, à terme, d'avoir des avatars inclusifs représentant le plus de diversité possible.

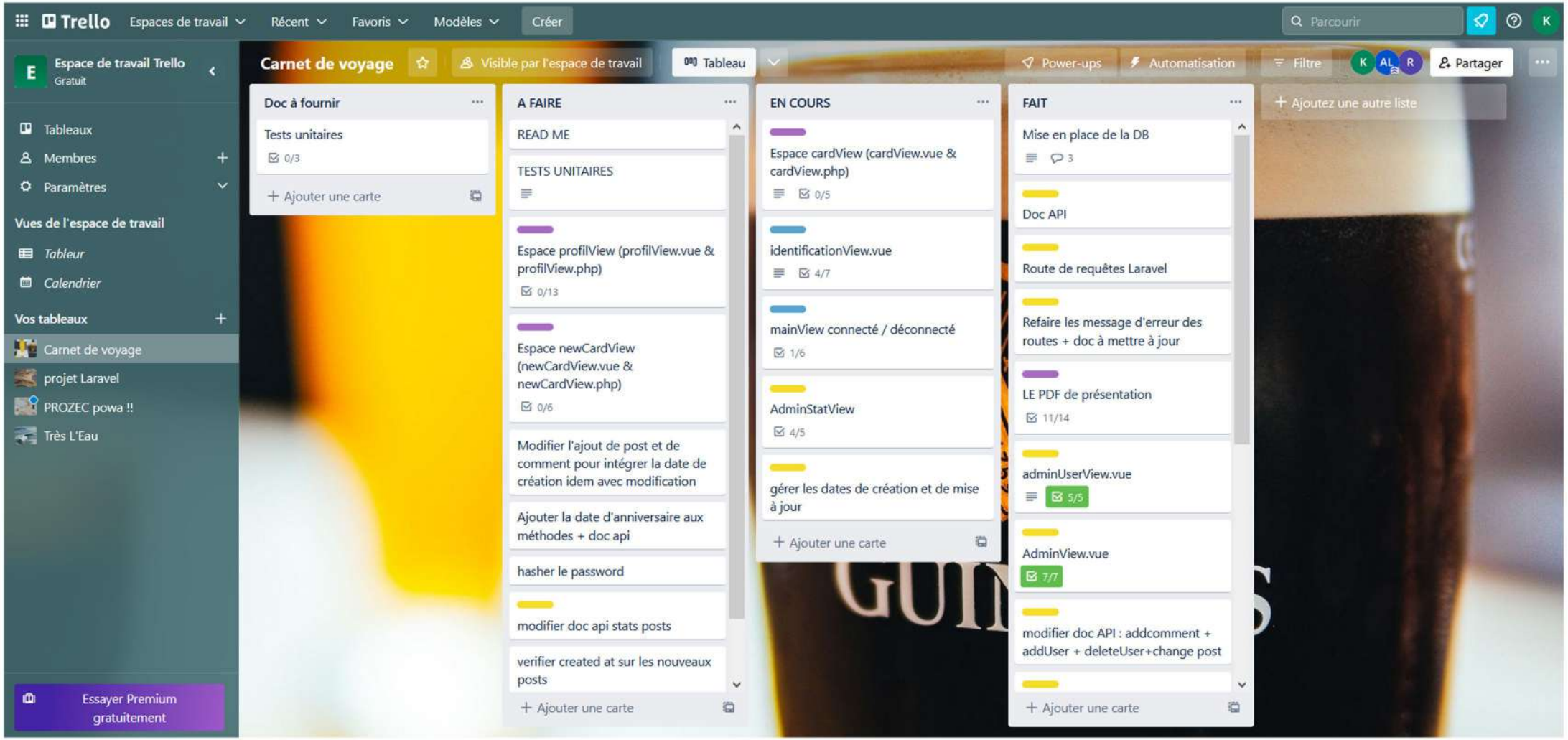
FIGMA & LA MISE EN PLACE GRAPHIQUE DU SITE

Exemple d'un visuel de profil utilisateur-rice
- côté ADMIN -



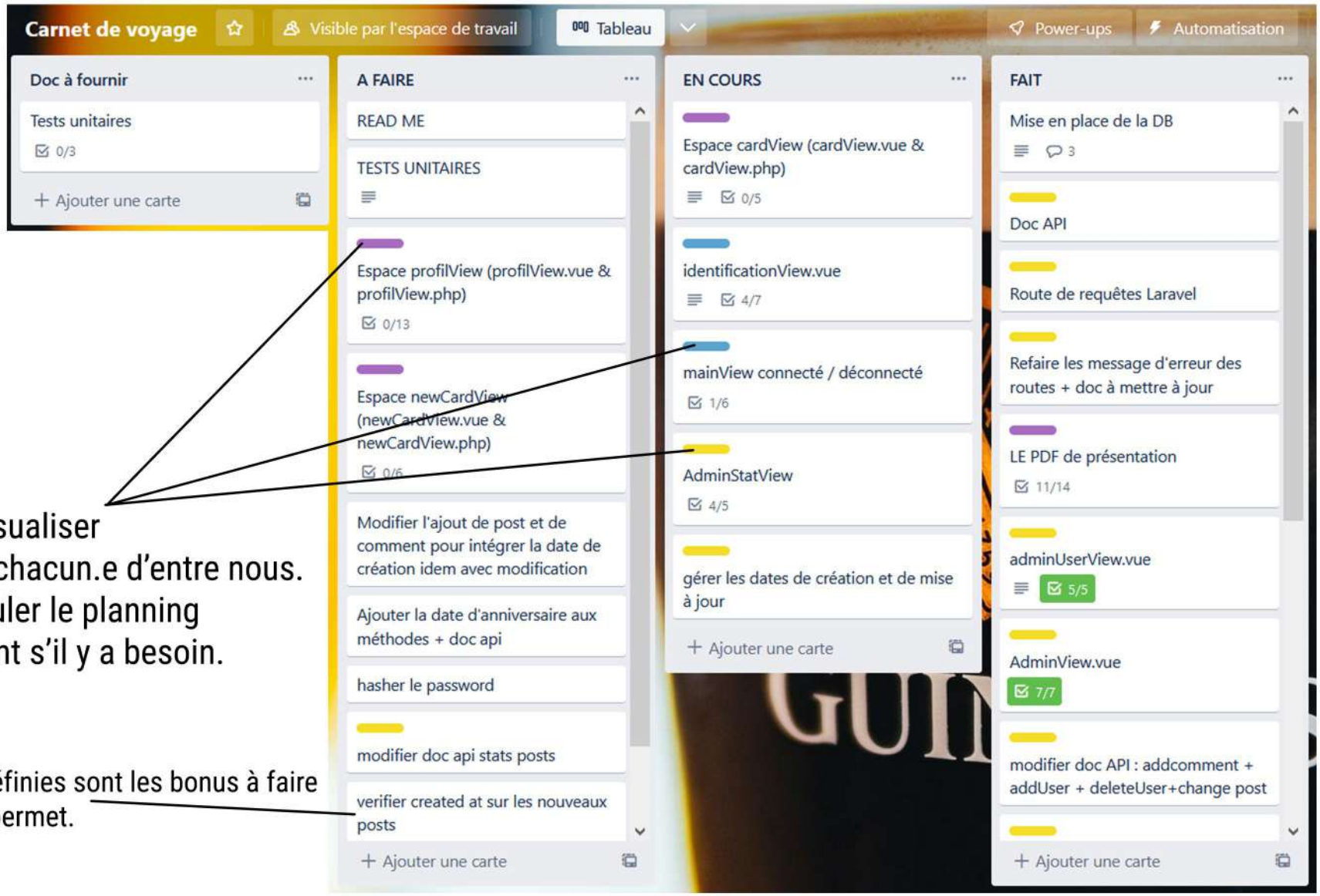
TRELLO

Une fois la charte graphique et le moodboard réalisé.e.s, nous avons utilisé TRELLO
afin de s'organiser et se répartir les différentes taches suivant les volontés de chacun.e.



TRELLO

Les étapes sont définies par module au début du projet et sont parfois redefinies suivant le temps imparti et les difficultés rencontrées.



Les couleurs servent à visualiser
qui fait quoi et où en est chacun.e d'entre nous.
Ainsi nous pouvons moduler le planning
et/ou s'aider mutuellement s'il y a besoin.

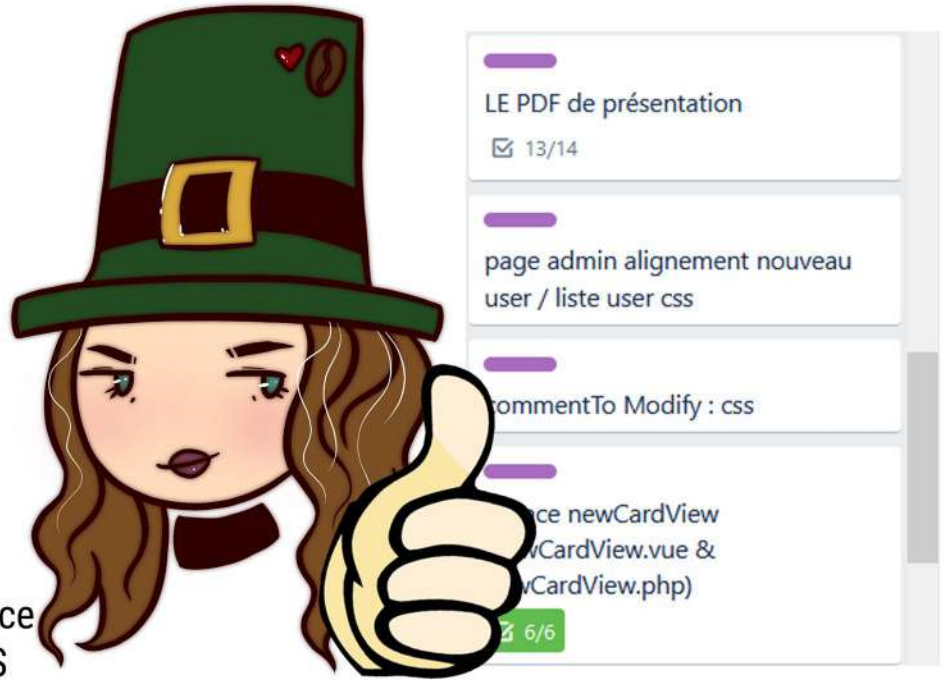
Les cartes indéfinies sont les bonus à faire
si le temps le permet.

QUI FAIT QUOI ?

Nous avons défini 4 poles - plutôt évidents ;
- Le Pole BACK
- Le Pole FRONT
- La DOC TECHNIQUE
- Les VIEWS VUE.JS

Le BACK, la majeure partie des fonctions du site ainsi que la doc API ont été géré.e.s par Anaïs.

Runyi s'est occupé de la DOC utilisateur ainsi que de fonctions ponctuelles dans les VIEWS.
Il a également fait toutes les recherches de de photos pour la mise en situation du site et la mise en place du CSS.



La partie graphique, la mise en place de la DB ainsi que certaines VIEWS ont été créées par Klaym.

ROUTES, CONTROLLERS etc.

api.php

laravel_urlland > routes > api.php

```php
<?php

use Illuminate\Http\Request;
use Illuminate\Support\Facades\Route;
use App\Http\Controllers\PostsController;
use App\Http\Controllers\UsersController;
use App\Http\Controllers\CommentsController;


/* ---------------------------------------- Route des posts ---------------------------------------- */

Route::get("/posts", [PostsController::class, 'displayPosts']); //OK
Route::get("/filter/posts", [PostsController::class, 'filterResults']); //OK
Route::get("/user/{id}/posts", [PostsController::class, 'userPosts']); //OK

Route::get("/post/{id}", [PostsController::class, 'displayPost']); //OK
Route::get("/select/posts", [PostsController::class, 'selectPosts']); //OK

Route::get("/stats/postsByCategory", [PostsController::class, 'getPostsByCategory']);
Route::get("/stats/postsByUser", [PostsController::class, 'getPostsByUser']);
Route::get("/stats/numberOf", [PostsController::class, 'getNumberOf']);


Route::put("/post/{id}", [PostsController::class, 'modifyPost']); //OK
Route::post("/post", [PostsController::class, 'addPost']); //OK
Route::delete("/post/{id}", [PostsController::class, 'deletePost']); //OK

/* ---------------------------------------- Route des users ---------------------------------------- */

Route::get("/users", [UsersController::class, 'displayUsers']); //OK
Route::get("/user/{id}", [UsersController::class, 'displayUser']); //OK
Route::get("/deconnexion", [UsersController::class, 'deconnectUser']); //NON TESTEE
Route::get("/user/isadmin", [UsersController::class, 'verifyIsAdmin']); //OK

Route::put("/user/{id}", [UsersController::class, 'modifyProfilUser']); //OK
Route::put("/admin/user", [UsersController::class, 'modifyUser']); //OK
Route::post("/connexion", [UsersController::class, 'connectUser']); //OK

Route::post("/admin/user", [UsersController::class, 'addUser']); //OK
Route::post("/register/user", [UsersController::class, 'registerUser']); //OK
Route::delete("/user/{id}", [UsersController::class, 'deleteUser']); //OK

/* ---------------------------------------- Route des comments ---------------------------------------- */

Route::get("/user/{id}/comments", [CommentsController::class, 'displayCommentsByUser']); //OK manque orderBy
Route::get("/post/{id}/comments", [CommentsController::class, 'displayCommentsByPost']); //OK manque orderBy

Route::put("/comment", [CommentsController::class, 'modifyComment']); //OK
Route::post("/comment", [CommentsController::class, 'addComment']); //OK
Route::delete("/comment/{id}", [CommentsController::class, 'deleteComment']); //OK

```

PostsController.php

laravel_urlland > app > Http > Controllers > PostsController.php > ...

```php
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use App\Models\Users;
use App\Models\Posts;
use App\Models\Comments;
use Session;

class PostsController extends Controller
{
    // Affichage de tous les posts
    public function displayPosts()
    {
        $posts = Posts::orderBy('created_at', 'desc')->get();
        return $posts;
    }

    // Affichage d'un post en fonction de son id
    public function displayPost($id_post)
    {
        $post = Posts::find($id_post);
        // return json_encode($post);
        return $post;
    }

    // Modification d'un post à partir des données du formulaire
    public function modifyPost(Request $request, $id)
    {
        $post = Posts::find($id);
        $post->title = $request->input('title');
        $post->content = $request->input('content');
        if($request->input('category') != null) {
            $post->category = $request->input('category');
        }
        if($request->input('picture') != null) {
            $post->picture = $request->input('picture');
        }
        $post->save();
        $post = Posts::find($id);
        return response($post, 200);
    }
```

Exemple des routes Laravel et des controllers des “articles” POSTS...

postComponent.vue

vue_urlland > src > components > postComponent.vue > Vetur > script

```vue
<script setup>
    defineProps({
        id: {
            type: String,
            required: true,
        },
        title: {
            type: String,
            required: true,
        },
        content: {
            type: String,
            required: true,
        },
        picture: {
            type: String,
            required: true,
        },
        category: {
            type: String,
            required: true,
        },
        nb_like: {
            type: Number,
            required: true,
        },
        created_at: {
            type: Date,
            required: true,
        }
    });
</script>

<template>

    <div class="post">
        <div class="post_picture">
            <div class="rectangle">... </div>
        </div>
        <div class="title_content">
            <h2 class="post_title" :title="id">{{ title }}</h2>
            <div class="post_content">{{ content }}</div>
            <div class="post_picture">{{ picture }}</div>
            <div class="post_category">{{ category }}</div>
            <div class="post_nb_like">{{ nb_like }}</div>
            <div class="post_created_at">{{ created_at }}</div>
            <!-- <h2 class="post_title" :title="id">TEST TITLE</h2>
            <div class="post_content"> Proin convallis erat in lectus
                Mauris efficitur neque ut dolor pharetra varius. Mauris
                Nunc turpis erat, lobortis non massa sed, viverra feugi
                Etiam et dui in diam vestibulum lobortis nec in lacus.
                Aenean iaculis, turpis vel condimentum consectetur, lor
                Cras nisi mi, mollis nec justo ut, dignissim elementum
                Pellentesque sit amet mi justo. Praesent et arcu mauris
                Nullam a ex eu massa aliquet pulvinar eget id justo. Mo
                Donec quis porttitor augue, et tempus ex.
                Phasellus posuere volutpat velit, non fringilla orci.
                Orci varius natoque penatibus et magnis dis parturient
            </div> -->
        </div>
    </div>
</template>

<style>
.post {
    display: flex;
    flex-direction: row;
    border-radius: 24px;
    background-color: #FFFEF8;
    margin-left: 150px;
    margin-right: 150px;
    padding: 15px;
}
.title_content{
    display: flex;
    flex-direction: column;
```

etc.

... suivis par le component POST de VUE.JS.

Et ce n'est qu'une petite partie de la globalité des Controllers et components...

TESTS UNITAIRES !

Exemples de nos tests unitaires côté BACK :

Test pour afficher tous les users du site

Test qui retourne le nombre de posts par utilisateur pour chaque utilisateur.

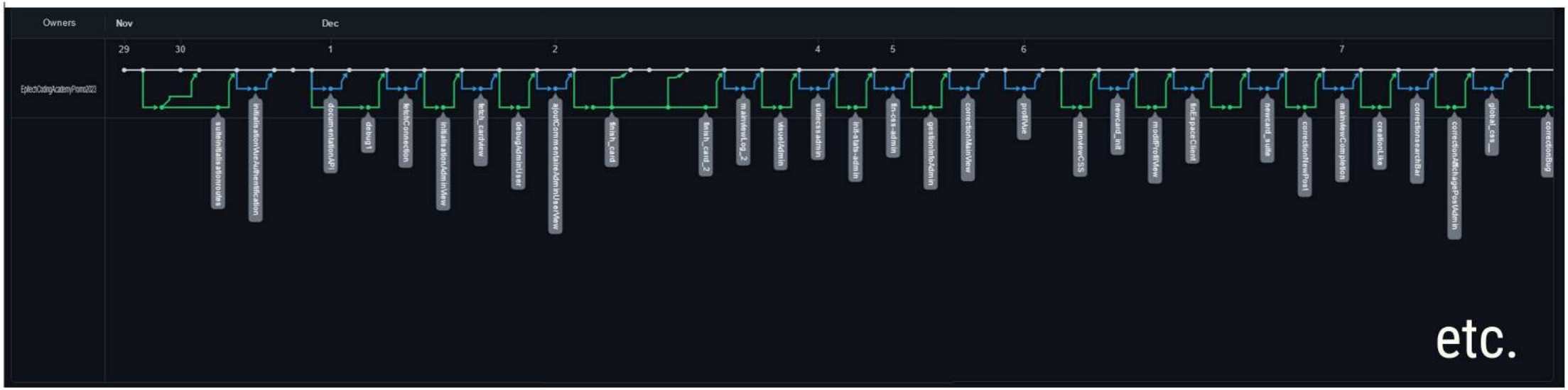
Test de la requête qui ajoute un utilisateur

ExampleTest.php

laravel_urlland > tests > Feature > ExampleTest.php > ...

```php
<?php

namespace Tests\Feature;

// use Illuminate\Foundation\Testing\RefreshDatabase;
use Tests\TestCase;

class ExampleTest extends TestCase
{
    /**
     * A basic test example.
     *
     * @return void
     */
    public function test_the_application_returns_a_successful_list_of_users()
    {
        $response = $this->get('api/users');

        $response->assertStatus(200);
    }

    public function test_the_application_returns_a_successful_list_of_posts()
    {
        $response = $this->get('api/posts');

        $response->assertStatus(200);
    }

    public function test_the_application_returns_a_successful_number_of_posts_by_category(
    {
        $response = $this->get('api/stats/postsByCategory');

        $response->assertStatus(200);
    }

    public function test_the_application_returns_a_successful_number_of_posts_by_user()
    {
        $response = $this->get('api/stats/postsByUser');

        $response->assertStatus(200);
    }

    public function test_the_application_returns_a_successful_list_of_comments_of_a_post()
    {
        $response = $this->get('api/post/12/comments');

        $response->assertStatus(200);
    }

    public function test_the_application_returns_a_successful_input_a_user()
    {
        $response = $this->postJson('/api/admin/user', ['nickname' => 'TestMan', 'email' =

        $response->assertStatus(200);

    }
}
```

GIT HUB & LES BRANCHES...



Environs 40 Branches différentes, plus de 40 commits, 73 fichiers ont changé avec la suppression de 621 élements et l’ajout d’environs 4800 éléments...

Difficultés :

Ce n'est clairement pas la partie que nous préférons...

- La mise en place du couple Laravel et VueJS
- Jongler entre les fonctions back et front des deux projets
- Comprendre les routes API
- Comprendre le fonctionnement des tests unitaires
- Gérer les conflits de CSS entre les VIEWS
- Gestion des SESSIONS ?? Cela reste un mystère à ce jour...

LIMITES DU PROJET

Clairement pas assez de temps pour mettre en place tout ce qu'on voulait faire.
Entre répartir les taches, faire le plan du site et des taches à realiser par rapport à celui ci, gérer les à côtés comme ce magnifique PDF de présentation etc.
Tout cela prendre beaucoup de temps.
C'est un peu frustrant mais il a fallu faire des choix et se dire que ce site pourra continuer d'évoluer à l'avenir...
ou pas !
:D