



MINISTÈRE CHARGÉ
DE L'EMPLOI

DOSSIER-PROJET

Nom de naissance

► Ura

Nom d'usage

► Ura

Prénom

► Mersad

Adresse

► 52 Avenue Raymond Bergougnan Clermont-Ferrand

Titre professionnel visé

Concepteur Développeur d'Applications

Sommaire

Remerciements	4
Résumé du projet en anglais	5
Liste des compétences du référentiel	6
Cahier des charges ou expressions des besoins de l'application à développer	7
Spécifications fonctionnelles	11
Spécifications techniques	11
Réalisations	13
Conclusions	54

REMERCIEMENTS

Je tiens d'abord à remercier l'établissement Simplon qui m'a ouvert de nouvelles portes et aidé à tracer mon avenir. Je tenais à remercier nos formateurs David Rigaudie, Yann qui savent très bien transmettre leur savoir, leur conseil et leur professionnalisme. Je tiens à remercier également pour l'accueil particulièrement chaleureux qui m'a été accordé par Ileaa. M'a aidé à comprendre plus le monde du travail et le savoir être dans le milieu du travail. M'a accueilli et donné un projet à taille réelle qui m'a énormément fait monter en compétences. Merci!

RÉSUMÉ DU PROJET EN ANGLAIS

The goal of this project is to create a platform where users can get their profile photo evaluated. The way I got this idea is when I wanted to get a proper profile photo for LinkedIn and I didn't know which photo to pick.

I think my project can help other people with the same problem.

Users can pick for what type they want their photo voted for. Like Dating Business and Social profile photos. Users can vote for others' photos.

I started everything from scratch.

I started by putting my idea in a paper meaning doing the wireframe in paper. Created the prototype in Figma. After some days working on my prototype I got the results I wanted and started working on the back-end. I used the framework Symfony because I think it has a great user role gestion. I used MySQL for my Database.

For the front end I used Bootstrap, native CSS, JS,

jQuery. The pros and cons :

+ It was a challenging project because I have never done it before and I liked every moment of it which made coding it more fun.

+ I worked from scratch and full-stack meaning that it helped me grow in skills, front, back, project gestion etc.

- It was tough making time for the project when I already had a big (and very interesting) one for my internship with many new technologies like Python Django.

In the end, I managed to deliver a working and pleasing to look at website for both desktop and mobile users, ready for production.

RÉSUMÉ DU PROJET EN FRANCAIS

L'objectif de ce projet est de créer une plate-forme où les utilisateurs peuvent faire évaluer leurs photos de profil.

La façon dont j'ai eu cette idée, c'est quand je voulais obtenir une photo de profil approprié pour LinkedIn et je ne savais pas quelle photo choisir .

Je pense que mon projet peut aider d'autres personnes avec le même problème .

L'utilisateur peut choisir pour quel type ils veulent qu'ils photo voté. Comme business et dating social

. L'utilisateur peut voter les photos d'autres utilisateurs .J'ai commencé à partir de zéro.

J'ai commencé par mettre mon idée dans un papier signifiant faire le wireframe dans le papier. Création du prototype en figma. Après quelques jours de travail dans mon prototype, j'ai obtenu lesrésultats que je voulais et j'ai commencé à travailler sur le back-end. J'ai utilisé le framework Symfony parce que je pense qu'il a une grande gestion de rôle utilisateur. J'ai utilisé MySql pour ma base de données .

Pour le front j'ai utilisé BootStrap, css, jQuery. Les avantages et les inconvénients :

+ C'était un projet difficile parce que je ne l'ai jamais fait avant et j'ai aimé chaque instant de celui-ci qui a rendu le codage plus amusant .

+J'ai travaillé à partir de zéro et full-stack ce qui signifie qu'il m'a aidé à grandir dans les compétences

, Front,Back , gestion de projet, etc .

- C'était difficile de prendre du temps pour le projet quand j'en avais déjà un grand (et très intéressant) projet pour mon stage avec de nombreuses nouvelles technologies comme Python Django Wagtail .

En fin de compte, j'ai réussi à fournir un travail fonctionnel et agréable à regarder le site web pour les utilisateurs de bureau et mobiles

LISTE DES COMPÉTENCES DU RÉFÉRENTIEL

1. - Maquetter une application
2. - Développer une interface utilisateur de type desktop
3. - Développer des composants d'accès aux données
4. - Développer la partie front-end d'une interface utilisateur Web
5. - Développer la partie back-end d'une interface utilisateur Web
6. - Concevoir une base de données
7. - Mettre en place une base de données
8. - Développer des composants dans le langage d'une base de données
9. - Collaborer à la gestion d'un projet informatique et à l'organisation de l'environnement de développement
- 10.- Concevoir une application
- 11.- Développer des composants métier
- 12.- Construire une application organisée en couches
- 13.- Développer une application mobile
- 14.- Préparer et exécuter les plans de tests d'une application
- 15.- Préparer et exécuter le déploiement d'une application

CAHIER DES CHARGES OU EXPRESSIONS DES BESOINS DE L'APPLICATION À DÉVELOPPER

DESCRIPTION	
Wireframe	Mersad Ura
Maquette	Mersad Ura
MCD	Mersad Ura
Back-end	Mersad Ura
Front-end	Mersad Ura

INTRODUCTION

L'AUDIENCE VISÉE

Nous visons un public national pour le moment.

LES OBJECTIFS DU SITE:

Il s'agit de réaliser un site web où l'utilisateur peut se faire évaluer ces photo des profils pour choisir lequel il peut poster sur les réseau sociaux.

DESCRIPTION

PERSPECTIVE DU PRODUIT

Ce site est réalisé dans le contexte de la formation Concepteur Développeur Web 2021/2022 de Simplon.co Clermont, il comprend les fonctionnalités suivantes : gestion de comptes utilisateurs, gestion de photo, moyens de paiement sécurisés, Création de compte avec Gmail, Facebook.

CARACTÉRISTIQUES

Gestion d'utilisateurs:

Les utilisateurs peuvent créer/supprimer un compte, modifier leurs informations et visionner le tout dans leur espace personnel.

Gestion de photo :

Les photos peuvent être ajoutées, visionnées, supprimées et peuvent être mis disponible d'être voté par les autres utilisateurs dans une interface sécurisée dédiée aux utilisateurs du site.

Les utilisateurs peuvent choisir les catégories auxquelles ils veulent leur photo soit voter. Ils peuvent voter les photos des autres utilisateurs. Ils peuvent choisir d'acheter des crédits. De crédits leur donner l'opportunité d'évaluer leurs photos plus vite.

Creation de compte avec gmail :

Les utilisateurs peuvent créer leur compte avec Gmail.

Creation de compte avec facebook:

Les utilisateurs peuvent créer leur compte avec Facebook

DOCUMENTATION

Le site sera livré sur un environnement local et restera sur le repository Gitlab sur lequel il est actuellement en privé jusqu'à ce que j'ai terminé toutes les fonctionnalités et je décide de passer en production.

Pages:**Utilisateur non connecté:****MENUS****---Menu Principal---**

- Accueil
- FAQ
- Blog
- A PROPOS

-Register

-Login

Accueil :

L'utilisateur peut avoir des exemples et informations sur comment ça marche le site .FAQ:

Questions et réponses pour mieux comprendre le site .Blog

:

Un petit blog avec la thème de photo Register:

Pour que ils peuvent créer une compteLogin : Se

connecter s' ils ont unecompte déjà

Utilisateur connecter:

MENUS

---Menu Principal---

- Accueil
- Mes photos
- Nouvelle photo
- Vote
- A PROPOS
- Crédits

- icon

---Sous-Menu Vote--

- Business
- Dating
- Social

---Sous-Menu Icon--

- Blog
- Faq
- Logout

Accueil :

L'utilisateur peut avoir des exemples et informations sur comment ça marche le site . Mes

photos :

Les photos d'utilisateur .

Nouvelle photo : Pour ajouter une nouvelle

photoVote:

Utilisateur ils peuvent voter sur les photo des autres utilisateurs et choisir sur quel catégorie ils veulent voter .

Crédits: Utilisateur peut acheter des credit

Titre Professionnel Développeur.se logiciel

SPÉCIFICATIONS FONCTIONNELLES

- Utiliser peut créer un compte , ajouter des photo le quel seront voter des autres utilisateurs .
- Une utilisateur ne peut pas voter ces photos et ne peut pas voter 2 fois la même photo.
- L'administrateur doit pouvoir afficher, ajouter, modifier et supprimer les catégories, traits, articles du blog dans le back office .
- Les articles seront dirigés avec WYSIWYG .
- Utilisateur, il peut faire évaluer ces photos avec des crédits ou avec énergie . l'énergie se fait gagner en votant sur les photos des autres utilisateurs .L'énergie permet à des utilisateurs de faire voter leur photos gratuitement.
- Toutes les données utilisateurs sont stockés en base de données et des backups doivent être faits régulièrement
- Les données sensibles (données bancaires, mots de passe) doivent être encodées avant d'être rentrées en base de données

SPÉCIFICATIONS TECHNIQUES

- Toutes les données utilisateurs sont stockées en base de données et des backups doivent être faits régulièrement
- Les données sensibles (données bancaires, mots de passe) doivent être encodées avant d'être rentrées en base de données

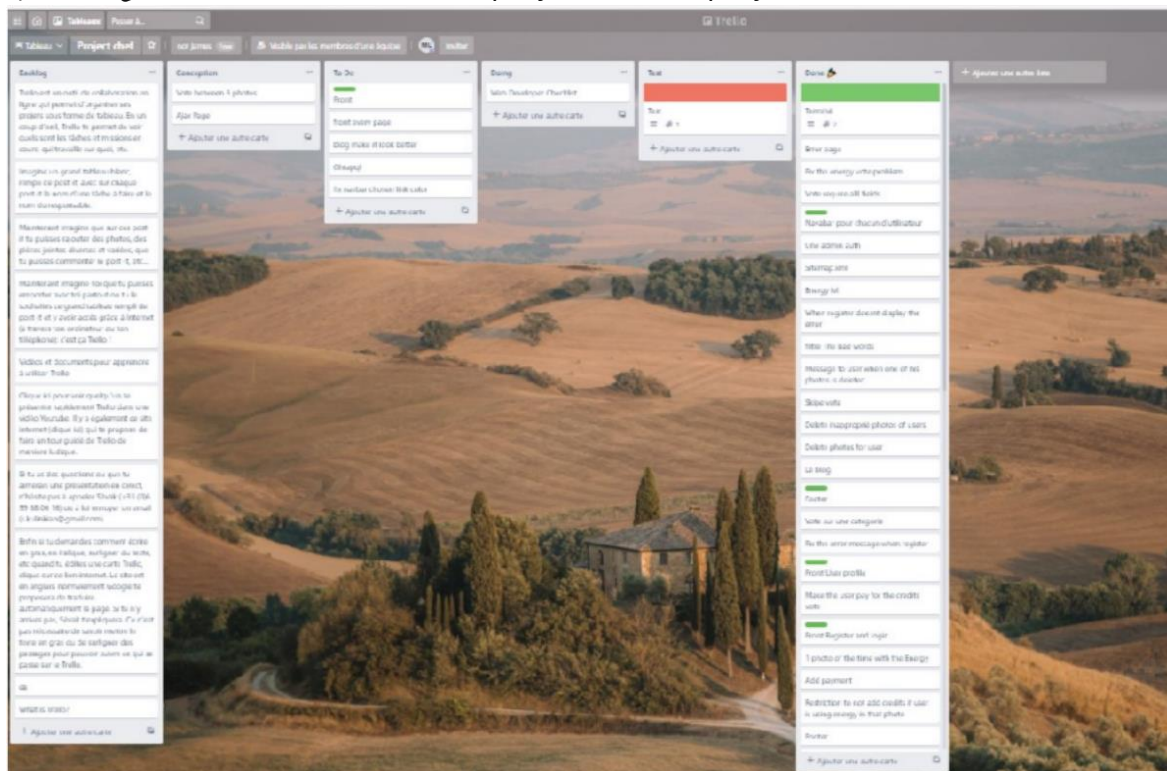
RÉALISATIONS

Gestion de projet:

Trello est un outil de collaboration en ligne qui permet d'organiser ses projets sous forme de tableau. En un coup d'œil, Trello te permet de voir quelles sont les tâches et missions en cours, qui travaille sur quoi, etc.

Pour mon projet j'ai fait 6 étapes dans trello .

- 1) Backlog : Dans backlog j'ai écrit une liste de fonctionnalités ou de tâches, jugées nécessaires et **suffisantes** pour la réalisation satisfaisante du projet.
- 2) Conception : C'est le manier auquel je pense à réussir intègre les fonctionnalités .
- 3) To Do : C'est toutes les tâches que j'ai dois faire .
- 4) Doing : C'est toutes les tâches que je suis en train de faire .
- 5) Test: C'est pour tester si la tâche que je viens de finir marche bien sur toutes utilisations possibles.
- 6) Done 🍀 : C'est toutes les tâches que j'ai faites et que j'ai testé .



Prototype:**Wireframe ;**

Wireframe est un schéma utilisé lors de la conception d'une interface utilisateur pour définir les zones et composants qu'elle doit contenir. À partir d'un wireframe peut être réalisée l'interface proprement dite par un graphiste.

J'ai réalisé mon wireframe sur des papiers .

Maquette

Une maquette est une représentation partielle ou complète d'un système ou d'un projet afin d'en tester et valider certains aspects et/ou le comportement , ou simplement à des fins ludiques ou informatives).

La maquette peut être réalisée en deux ou trois dimensions, à une échelle donnée, le plus souvent réduite ou agrandie pour en faciliter la visualisation ou la manipulation. Elle peut être statique ou dynamique, et dans ce dernier cas on parlera alors de modélisme.

J'ai réalisé la maquette avec Figma:

Figma est une application qui permet de créer des interfaces d'applications web & mobiles et qui offre en plus une solution de prototypage qui permet de présenter les interactions au clients où aux développeurs. Une solution entièrement en ligne La plus grande particularité de Figma est son fonctionnement entièrement en ligne.

Exemple de la maquette de mon projet



Les Technos utiliser:

Symfony :

Symfony est un Framework PHP, qui permet d'accélérer la création et la maintenance d'applications Web et à remplacer les tâches de codage récurrentes.

Un Framework est un ensemble de programmes universels et réutilisables qui accomplissent des tâches particulières pour faciliter le développement d'applications logicielles.

Métaphoriquement, il s'agit d'un squelette, d'une carcasse ou d'un sous-sol pour une application Web ou le développement d'un site Web. De la même manière que vous achetez une maison vide et la remplissez de meubles, les développeurs prennent donc des modules et remplissent donc le site ou l'application par des modules en y ajoutant leur couche de personnalisation.

Doctrine:

Doctrine est un ORM (couche d'abstraction à la base de données) pour PHP. Il s'agit d'un logiciel libre sous licence GNU LGPL. Doctrine est l'ORM par défaut du Framework Symfony (depuis la version 1.3 de ce Framework). Cependant son utilisation dans le cadre d'un projet développé avec Symfony est optionnelle.

HTML:

Le HTML est un langage informatique qui permet de mettre en forme du contenu à l'aide de balises. Ce langage est très utilisé sur l'internet, un tel fichier s'ouvre à l'aide d'un navigateur web. Ce langage est très utilisé sur l'internet, un tel fichier s'ouvre à l'aide d'un navigateur web.

CSS

CSS désigne **Cascading Style Sheets** (pour Feuilles de style en cascade). Il s'agit d'un langage de style dont la syntaxe est extrêmement simple mais son rendement est remarquable.

Bootstrap:

Bootstrap est une collection d'outils utiles à la création du design (graphisme, animation et interactions avec la page dans le navigateur, etc.) de sites et d'applications web. C'est un ensemble qui contient des codes HTML et CSS, des formulaires, boutons, outils de navigation et autres éléments interactifs, ainsi que des extensions JavaScript en option. C'est l'un des projets les plus populaires sur la plate-forme de gestion de développement GitHub.

Twig :

Twig est un moteur de templates pour le langage de programmation PHP. Il est principalement utilisé pour la sortie HTML, mais peut également être utilisé pour générer tout autre format texte. C'est un composant autonome qui peut être facilement intégré dans tout projet PHP. Il offre de nombreuses fonctionnalités excellentes:

jQuery:

jQuery est **une bibliothèque JavaScript** libre et multiplateforme créée pour faciliter l'écriture de scripts côté client dans le code HTML des pages web

Une bibliothèque est un ensemble de fonctions et de routines utilisées par d'autres programmes et vous contrôlez totalement si vous appelez une méthode à partir d'une bibliothèque

Javascript:

JavaScript est un langage de programmation qui permet d'implémenter des mécanismes complexes sur une page web. À chaque fois qu'une page web fait plus que simplement **afficher** du contenu statique — **afficher** du contenu mis à jour à des temps déterminés, des cartes interactives, des animations 2D/3D, des menus vidéo défilants, etc.

GitHub-Gitlab:

GitHub est un service web d'hébergement et de gestion de développement de logiciels, utilisant le logiciel de gestion de versions Git.. Le site assure également un contrôle d'accès et des fonctionnalités destinées à la collaboration comme le suivi des bugs, les demandes de fonctionnalités, la gestion de tâches et un wiki pour chaque projet. Pour ajouter les nouvelles versions que j'ai faites sur git j'entrai les commandes suivantes : git add . après git commit -m 'ICI JE MIS LE MESSAGES AVEC LES CHANGEMENT QUE JE FAIS' après un git push pour le mettre sur git . GitHub ca m'as beaucoup aider parce que une ou deux fois j'ai fais des erreur sur le projet et je me suis retourné me suis retourné sur un commit ou tout marche .

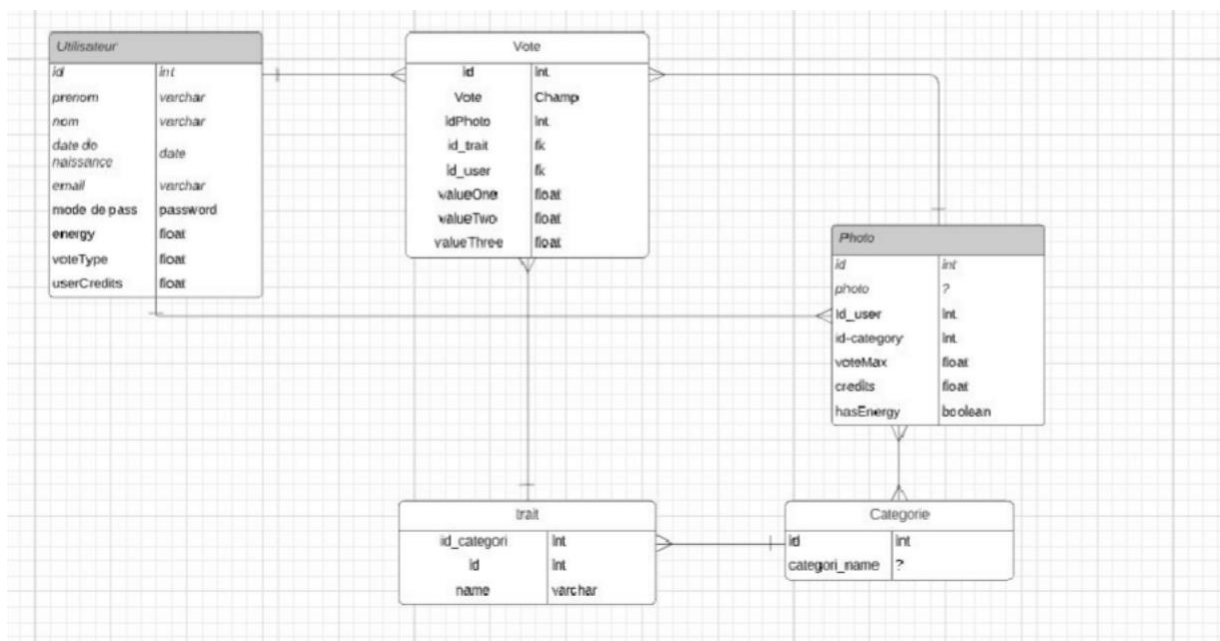
Base de données :

Toujours pendant l'étape de conception, j'ai conçu un schéma de base de données. Il est composé des différentes tables qui représentent mes modèles. À l'intérieur de celles-ci, on trouve les colonnes, qui sont en quelques sortes les propriétés de mes modèles. La plupart des tables comportent un champ "id" qui est une clef primaire et qui s'auto-incrémente. Ainsi chaque instance de mon modèle est atteignable via son "id". Dans ce diagramme, on peut aussi voir les relations entre les tables. Dans un modèle de base de données dit

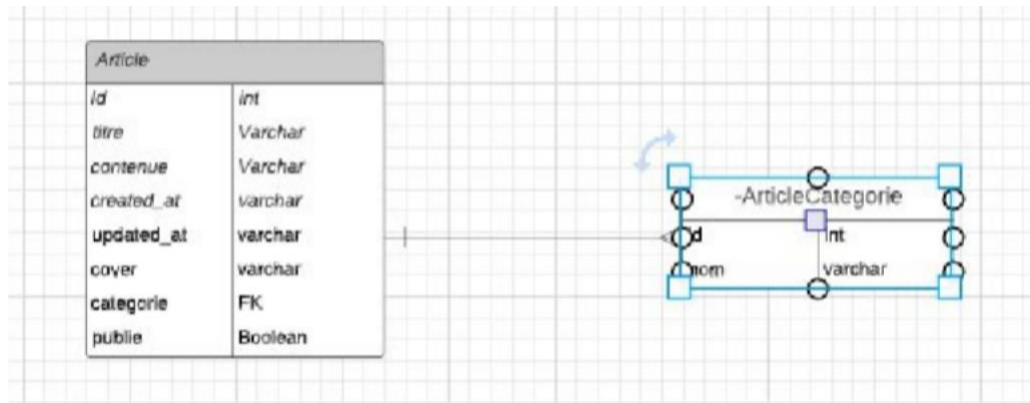
“relationnel” (tel que j'utilise), chaque colonne peut avoir une relation avec une colonne d'une autre table. Il existe plusieurs types de relations : - One-To-One : A peut avoir un seul B - One-To-Many : A peut avoir plusieurs B - Many-To-One : Plusieurs A peuvent avoir un B - Many-To-Many : Plusieurs A peuvent avoir plusieurs B Dans ce dernier cas, on utilise une table relationnel, c'est-à-dire une table intermédiaire qui comportera dans la plupart des cas deux colonnes : A_id et B_id. Les relations se font à l'aide de “Foreign Key” aussi appelées clés étrangères qui permettent d'avoir des relations propres et non redondantes notamment grâce à des contraintes. J'ai favorisé le modèle relationnel car je le trouve plus approprié pour le web et que j'ai besoin de beaucoup de relations entre les tables dans ce cas précis.

La base de données est composée de 7 tables .

- Utilisateur pour garde tout les info sur les utilisateur
- Photo , lie avec catégorie
- catégorie
- traits d'un categorie
- vote pour gérer le vote que un photo prends et qu' une utilisateur donne



- Article pour gérer les article
- ArticleCategorie pour le categorie d'article



Le virtual host :

Pour commencer , j'ai utilisé mon serveur local (Xampp) puis j'ai téléchargé Composer, un gestionnaire de dépendances pour PHP. Il permet de télécharger des librairies qui vont venir compléter mon projet ,la gestion de ces dépendances est disponible dans le fichier [composer.json](#).

```
PS C:\wamp64\www\projet_chef\myProjectVote> composer create-project symfony/website-skeleton myProjectVote
```

Cette commande va installer un projet Symfony avec des dépendances déjà incluses.

J'ai modifié ces 2 fichiers pour créer mon virtualhost sur le port 80. Je peux ainsi travailler sur mon serveur local avec une url propre et adaptée à mon projet.

Je lui spécifie qu' il doit aller chercher le projet pour le faire afficher sur lenavigateur .Je dois lui donner la route de dossier public de mon projet sinon ça affichera une erreur .

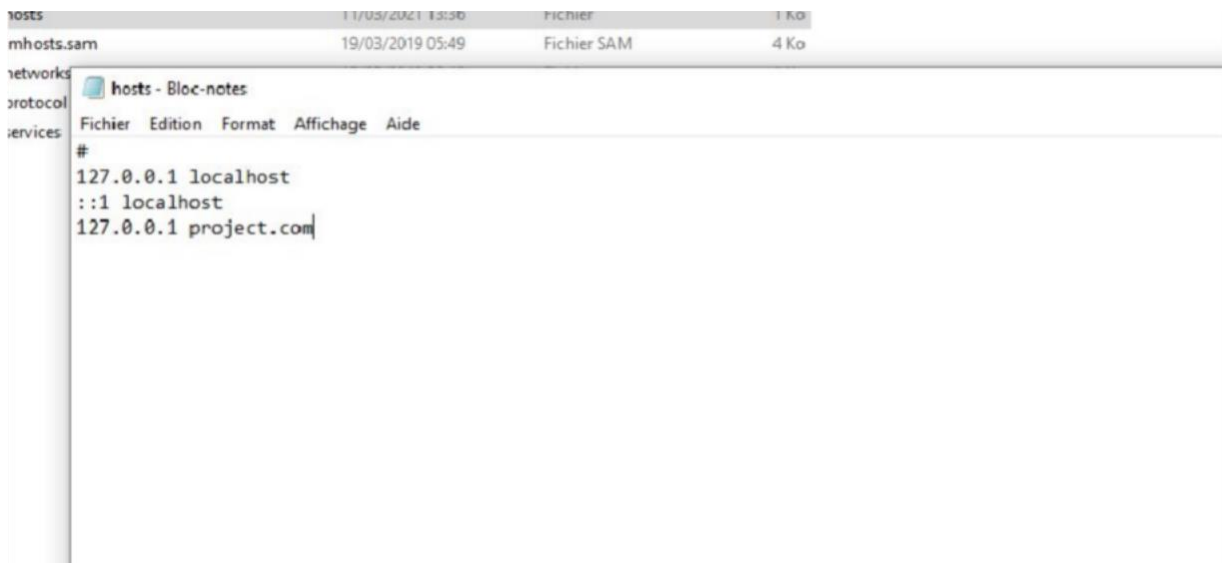
```

VirtualHost *:80>
    ServerAdmin localhost
    DocumentRoot "${INSTALL_DIR}/www/projet_chef/myProjectVote/public"
    ServerName project.com
    ServerAlias project.com
    <Directory "${INSTALL_DIR}/www/projet_chef/myProjectVote/public">
    Options +Indexes +Includes +FollowSymLinks +MultiViews
    AllowOverride All
    Require local
    </Directory>
    ErrorLog "logs/project.com-error.log"
    CustomLog "logs/project.com-access.log" common
./VirtualHost>

```

Ensuite je lui ai donné un nom de virtual host . Quand je taper projet.com sur le navigateur il le sais que je parle de mon projet .

Après avoir fait mon maquet , mon virtualhost , mon schéma de base dedonnées , j'ai commencer a coder sur mon projet.



Création d'utilisateurs :

J'ai commencé par lie la base de données avec mon projet . Avec cett commande **php bin/console doctrine:database:create** et j'ai changer ca sur le fichier .env

```
# DATABASE_URL="sqlite:///kernel.project_dir%/var/data.db"  
DATABASE_URL="mysql://root:@127.0.0.1:3306/symfonyVoteProjet?10.4.14-MariaDB"  
# DATABASE_URL="postgresql://db_user:db_password@127.0.0.1:5432/db_name?serverVersion=13&charset="
```


J'ai ensuite créé mes controllers avec la commande **php bin/console make:controller**. Ceux-ci ont pour rôles de permettre aux navigateurs qui appellent une route (ou adresse) de leur fournir une réponse HTTP pour afficher un rendu sur ce même navigateur.

Je peux désormais me lancer dans la création des tables de ma base de données.

Pour cela Symfony utilise un ORM (Object-Relational Mapping) qui est Doctrine. Doctrine sert de lien entre mon application et ma base de données et va me permettre de gérer mes tables et même les lignes de ces dernières. Avec Doctrine je vais créer des classes appelées entités qui représentent les tables. J'ai créé les entités avec la commande **php bin/console make:entity**. Avec cette commande je pouvais écrire tout le nom des 'attributs' dont j'avais besoin. Par exemple, c'est mon Entités d'images. Il contient les attributs de l'image comme id, name etc et les attributs que l'on le lie avec les autres tables. Par exemple, la \$catégory le lie avec la table de category.

```

* @ORM\Entity(repositoryClass=ImagesRepository::class)
*/
class Images
{
    /**
     * @ORM\Id
     * @ORM\GeneratedValue
     * @ORM\Column(type="integer")
     */
    private $id;

    /**
     * @ORM\Column(type="string", length=255)
     */
    private $name;

    /**
     * @ORM\ManyToOne(targetEntity=User::class, inversedBy="images")
     * @ORM\JoinColumn(nullable=false)
     */
    private $user;

    /**
     * @ORM\OneToMany(targetEntity=Vote::class, mappedBy="images")
     */
    private $votes;

    /**
     * @ORM\ManyToOne(targetEntity=Category::class, inversedBy="images")
     * @ORM\JoinColumn(nullable=true)
     */
    private $category;

    /**
     * @ORM\Column(type="float", nullable=true)
     */
    private $voteMax;

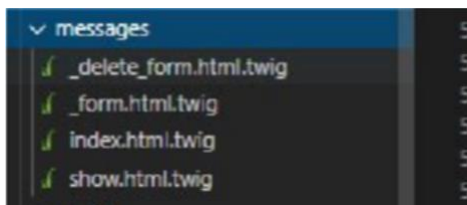
    /**
     * @ORM\Column(type="boolean", nullable=true)
     */
    private $hasEnergy;

    /**
     * @ORM\Column(type="float", nullable=true)
     */
    private $credits;

    /**
     * @ORM\Column(type="float", nullable=true)
     */
    private $voteCredits;
}

```

Ensuite j'ai créé le CRUD (L'acronyme informatique anglais **CRUD** (pour *create, read, update, delete*) désigne les quatre opérations de base pour la persistance des données, en particulier le stockage d'informations en base de données.) , avec la commande **php bin/console make:crud <le nom de entites que je veux faire lecrud>** .Une fois cet commande entre symfony crée des fichier twig qui permet de faire le crud des entités que j'ai choisir.



Pour faire les entités des user symfony utiliser un comander différent **php bin/console make:user**.

Symfony a une très bonne gestion d'utilisateurs. C'est très facile de faire avec symfony le gestion d'utilisateur avec symfony . J'ai pu contrôler ça sur le fichier security.yml . Je donne

```

30      # target: app_any_route
31
32      # activate different ways to authenticate
33      # https://symfony.com/doc/current/security.html#firewalls-authen
34      # https://symfony.com/doc/current/security/impersonating_user.h
35      # switch_user: true
36
37      # Easy way to control access for large sections of your site
38      # Note: Only the *first* access control that matches will be used
39      access_control:
40          - { path: ^/admin, roles: ROLE_ADMIN }
41          - { path: ^/profile, roles: ROLE_USER }
42          - { path: ^/efconnect, role: ROLE_ADMIN }
43          - { path: ^/elfinder, role: ROLE_ADMIN }
44

```

accès par rapport aux rôles sur quelques pages

Je lui dis au symfony que tous les contrôleurs qui commencent avec un /admin et ilssont pas admin , de ne pas lui donner le droit de voir cette page .

Par exemple la création des article :

```

/**
 * @Route("admin/article")
 */
class ArticleController extends AbstractController
{
    /**

```

Les page que les utilisateur peuvent voir par rapport a leurs rôle c'est :

Utilisateur non connecter:

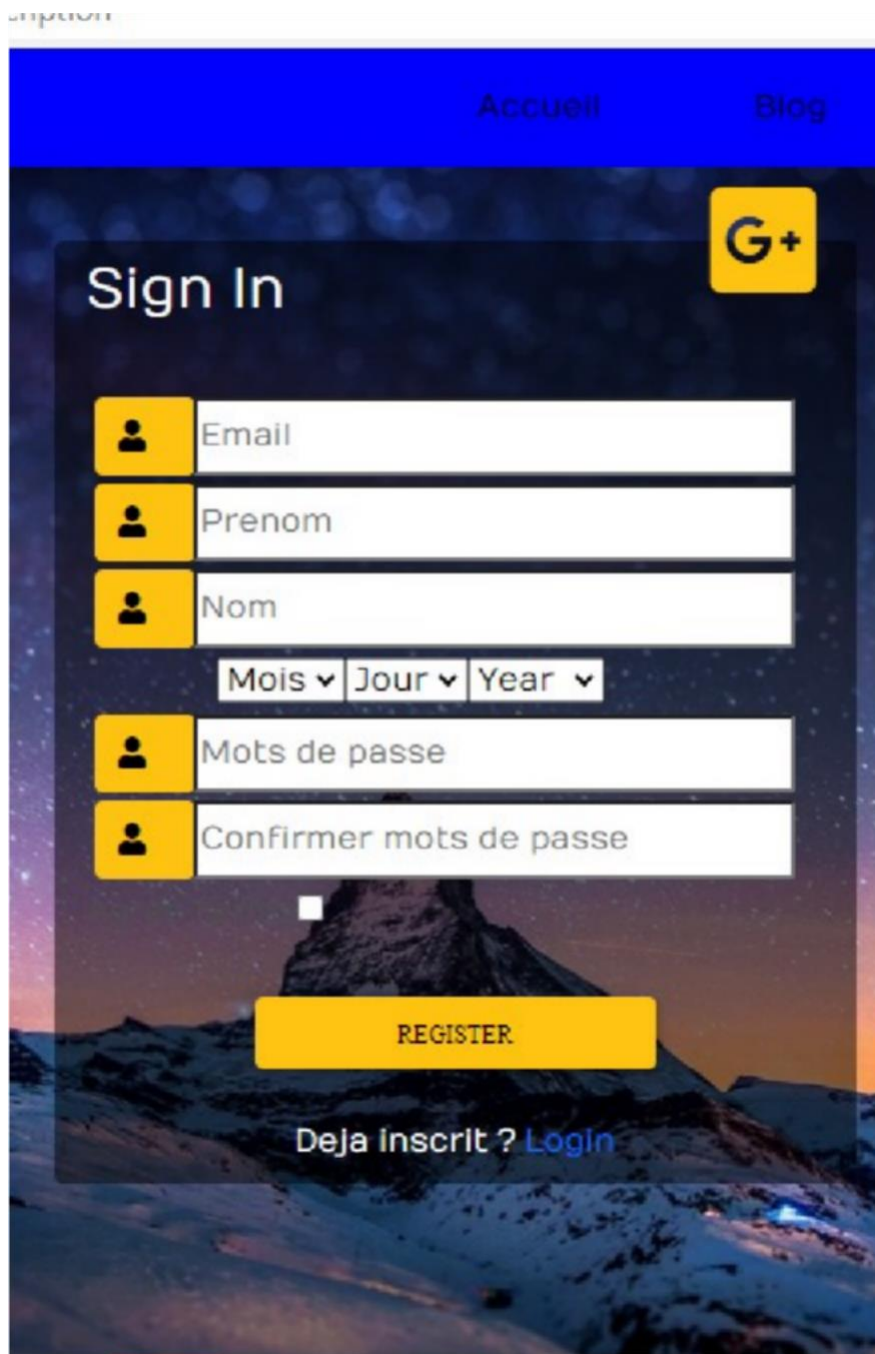
L'utilisateur non connecté peut aller sur la page d'accueil pour regarder comment marche le site.C'est une **page static**.(Une page web statique est une page web dont le contenu ne varie

pas en fonction des caractéristiques de la demande, c'est-à-dire qu'à un moment donné tous les internautes qui demandent la page reçoivent le même contenu.).

Il peut aller sur la page **a propos** pour prendre plus d'information sur le site . C'est une page static aussi.Ensuite il peut s'inscrire sur la page inscription.

Pour s'inscrire l'utilisateur doit fournir d'un email de son prénom, nom , anniversaire et mot de passe .Le mot de passe doit contenir 8 caractères minimum et doit être le même que sur le confirmer le mot de passe sinon un message d'erreur apparaît . Le message d'erreur explique l'erreur .

Un lien vers la page de connexion pour se connecter si l'utilisateur a déjà un compte .



Sign In

Email

Prenom

Nom

Mois ▼ Jour ▼ Year ▼

Mots de passe

Confirmer mots de passe

REGISTER

Deja inscrit ? [Login](#)

Après l'inscription symfony encode le mot de passe pour sécuriser le donnerd'utilisateur .
On envoie un email pour confirmer l'inscription . On peut modifier le message que l'utilisateur reçoit sur le fichier confirmation_email.html.twig.

```

/**
 * @Route("/inscription", name="app_register")
 */
public function register(Request $request, UserPasswordEncoderInterface $passwordEncoder, GuardAuthenticatorInterface $guardAuthenticator)
{
    $user = new User();
    $form = $this->createForm(RegistrationFormType::class, $user);
    $form->handleRequest($request);

    if ($form->isSubmitted() && $form->isValid()) {
        // encode the plain password
        $user->setPassword(
            $passwordEncoder->encodePassword(
                $user,
                $form->get('plainPassword')->getData()
            )
        );

        $entityManager = $this->getDoctrine()->getManager();
        $entityManager->persist($user);
        $entityManager->flush();

        // generate a signed url and email it to the user
        $this->emailVerifier->sendEmailConfirmation('app_verify_email', $user,
            (new TemplatedEmail())
                ->from(new Address('uramersad97@gmail.com', 'services Mail'))
                ->to($user->getEmail())
                ->subject('Please Confirm your Email')
                ->htmlTemplate('registration/confirmation_email.html.twig')
        );
    }
}

```

L'envoi d'email est possible avec Mailjet . Je prends la clé sur mailjet après avoir créer un compte ensuite je mets sur le fichier .env de mon projet .

MAILER_DSN=mailjet+smtp:

Quand l'utilisateur il confirme la création de compte

```
/**
 * @Route("/verify/email", name="app_verify_email")
 */
public function verifyUserEmail(Request $request): Response
{
    $this->denyAccessUnlessGranted('IS_AUTHENTICATED_FULLY');

    // validate email confirmation link, sets User::isVerified=true and persists
    try {
        $this->emailVerifier->handleEmailConfirmation($request, $this->getUser());
    } catch (VerifyEmailExceptionInterface $exception) {
        $this->addFlash('verify_email_error', $exception->getReason());

        return $this->redirectToRoute('app_register');
    }

    // @TODO Change the redirect on success and handle or remove the flash message in your templates
    $this->addFlash('success', 'Your email address has been verified.');
```

GMail :

Les utilisateurs peuvent se connecter ou s'inscrire avec GMail aussi :

C'était possible avec Oauth .

J'ai configuré sur knp_oauth2_client.yaml pour connecter les utilisateurs avec google. Le cle secret j'ai pris sur google api platform . Cela permet d'intégrer gmail sur le site .

J'ai après créé une Google Authenticator.php pour prendre le Acces token et si l'utilisateur n'existe pas on le crée .

```
//Specify how to start an Authentication
public function start(Request $request, ?AuthenticationException
$authException = null)
{
    return new
RedirectResponse($this->router->generate('app_login'));
}
//See if whe are in the good route
public function supports(Request $request)
{
    return 'connect_google_check' ===
$request->attributes->get('_route');
}
public function getCredentials(Request $request)
```



```

    {
        return $this->fetchAccessToken($this->getClient());
    }
    /**
     * Undocumented function
     *
     * @param AccessToken $credentials
     */
    public function getUser($credentials, UserProviderInterface
$userProvider)
    {
        $googleUser =
$this->getClient()->fetchUserFromToken($credentials);
        return
$this->userRepository->findOrCreateFromOauth($googleUser);
    }
    public function onAuthenticationFailure(Request $request,
AuthenticationException $exception)
    {
    }
    public function onAuthenticationSuccess(Request $request,
TokenInterface $token, string $providerKey)
    {
        return new RedirectResponse($this->router->generate('profile'));
    }
    public function getClient(): GoogleClient
    {
        return $this->clientRegistry->getClient('google');
    }
}

```

Le fonction findOrCreateFromOauth mets dans la base de données le nouveau utilisateur
On auto remplir les information depuis Gmail sur le profile de L'utilisateur comme le Email,

```

public function findOrCreateFromOauth(ResourceOwnerInterface $owner):
User
    {
        $user = $this->createQueryBuilder('u')

```

Titre

prénom , nom

```
->where('u.googleId = :googleId')
->orWhere('u.email = :email')
->setParameters([
    'email' => $owner->getEmail(),
    'googleId' => $owner->getId()
])
->getQuery()
//get Singel result
->getOneOrNullResult();
if ($user) {
    if ($user->getGoogleId() === null) {
        $user->setGoogleId($owner->getId());
        $this->getEntityManager()->flush();
    }
    return $user;
}
//Takes data from google and add to the account
$user = (new User())
    ->setRoles(['ROLE_USER'])
    ->setGoogleId($owner->getId())
    ->setEmail($owner->getEmail())
    ->setPrenom($owner->getFirstName())
    ->setNom($owner->getLastName());
$em = $this->getEntityManager();
$em->persist($user);
$em->flush();
return $user;
}
```

Facebook :

Les utilisateurs peuvent se connecter ou s'inscrire avec Facebook aussi :

J'ai configuré sur hwi.oauth.yaml pour connecter les utilisateurs avec facebook.

```
hwioauth:
# list of names of the firewalls in which this bundle is active, this setting MUST be set
firewall_names: |main|

# https://github.com/hwi/HwIOAuthBundle/blob/master/Resources/doc/2-configuring-resource-owners
resource_owners:
  facebook:
    type: facebook
    client_id: '%env(FB_ID)%'
    client_secret: '%env(FB_SECRET)%'
    scope: "email"
    options:
      display: popup
      csrf: true
```

Pour le FB_ID et FB_SECRET j'ai créé un app sur le meta developers pour tester la connexion avec facebook et dedans j'ai pris les l'id et le cle secret.



Qui va me permettre de me connecter avec un compte facebook.

Le tableau de bord pour configurer la connexion avec Facebook.

Tableau de bord

Paramètres

Général

Avancé

Rôles

Alertes

Produits [Ajouter un produit](#)

Facebook Login

API Marketing

Journal des activités

Journal des activités

Vous êtes en train de modifier une version test de **vota**.

Identifiant de l'application

Clé secrète

Afficher

Norm à l'écran

vota - Test2

Namespace

Domaines de l'app

localhost X

URL de la Politique de confidentialité

Politique de confidentialité pour la boîte de dialogue de connexion...

URL des conditions de service

Conditions de service pour la boîte de dialogue de connexion et e...

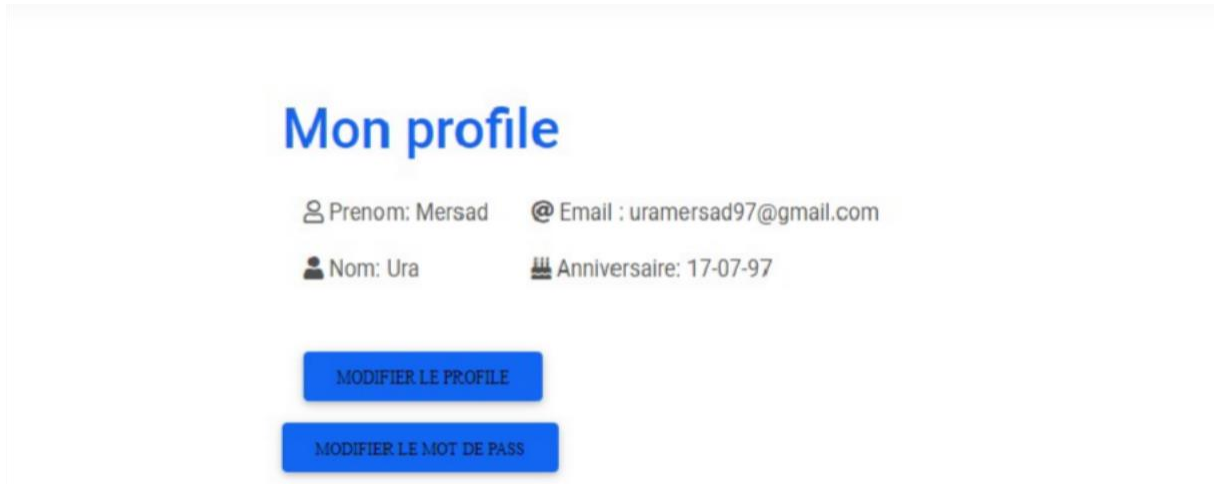
icone de l'app (1024 x 1024)

1024 x 1024

Page Profil:

Sur cette page l'utilisateur peut aller voir ces informations de profil et les changer s' il le souhaite

.



Mon profile

👤 Prenom: Mersad @ Email : uramersad97@gmail.com

👤 Nom: Ura 🎂 Anniversaire: 17-07-97

MODIFIER LE PROFILE

MODIFIER LE MOT DE PASS

Ajouter une photo :

Cet route nous permet d'ajouter une photo

Il prend l'image télécharger et ca le sauvegarde dans un folder et dans base de données avec un nom unique comme ca les photos ont des nom différent .

Cet commande permet de mettre un nom unique aux photos

```
$fichier = md5(uniqid()) . '.' . $image->guessExtension();
```

Après les photos sont liées avec l'utilisateur lequel il l'a téléchargé .

```
* @Route("/new", name="images_new", methods={"GET", "POST"})
*/
public function new(Request $request): Response
{
    $image = new Images();
    $form = $this->createForm(ImagesType::class, $image);
    $form->handleRequest($request);
    if ($form->isSubmitted() && $form->isValid()) {
        // Get the image
        $images = $form->get('name')->getData();
        // Generate a new file name
        foreach ($images as $image) {
            $fichier = md5(uniqid()) . '.' . $image->guessExtension();
            // Copy the file in the folder
            $image->move(
                $this->getParameter('images_directory'),
                $fichier
            );
            // Create image in DB
            $user = $this->get('security.token_storage')->getToken()->getUser();
            $img = new Images();
            $img->setName($fichier);
            $img->setUser($user);
        }
        $entityManager = $this->getDoctrine()->getManager();
        $entityManager->persist($img);
        $entityManager->flush();
        return $this->redirectToRoute('images_index');
    }
    return $this->render('images/new.html.twig', [
        'image' => $image,
        'form' => $form->createView(),
    ]);
}
```

Après, l'utilisateur choisit la catégorie duquel il veut que la photo soit voter .

Une fois la catégorie choisie, cette catégorie serait liée avec l'image que l'utilisateur a choisie.

Il y a trois catégories que l'utilisateur peut choisir, chacun de catégorie a 3 traits aussi .


```

// Add the category to the image
/**
 * @Route("/{id}/edit", name="images_edit", methods={"GET","POST"})
 */
public function edit(Request $request, Images $image): Response
{
    $idImage = $request->get('id');
    $form = $this->createForm(ImagesCategoryType::class, $image);
    $form->handleRequest($request);
    if ($form->isSubmitted() && $form->isValid()) {
        $this->getDoctrine()->getManager()->flush();
        return $this->redirectToRoute('images_voteMax', array('id' => $idImage));
    }
    return $this->render('images/edit.html.twig', [
        'image' => $image,
        'form' => $form->createView(),
    ]);
}

```

Après l'utilisateur choisir avec quoi il veut que la photo soit voter , l'énergie ou crédits

-Avec le setEnergyAttribute fait que soit ajouté l'énergie ou photo

```

//Sets the energy attribut to the image
public function setEnergyAttribute($value, $user)
{
    return $this->createQueryBuilder('i')
        ->where('i.id != :identifiant')
        ->setParameter('identifiant', $value)
        ->andWhere('i.user = :identifiant')
        ->setParameter('identifiant', $user)

        ->getQuery()
        ->getResult();
}

```

-Le boucle for qui fait que si il choisi voter avec l'énergie alors l'autre photo qui a l'attribut énergie soit enlever l'attribut et soit mis à cette nouvelle photo. Parce que il y a que une photo qui peut être voter avec l'énergie.

-Si l'utilisateur veut choisir les crédits et il y a pas suffisant des crédits alors on affiche une

erreur

```

* @Route("/{id}/edit/voteMax", name="images_voteMax", methods={"GET","POST"})
*/
public function voteMax(Request $request, ImagesRepository $imagesRepository, Images $image)
{
    $user = $this->get('security.token_storage')->getToken()->getUser();
    $form = $this->createForm(VoteMaxType::class, $image);
    $form->handleRequest($request);
    if ($form->isSubmitted() && $form->isValid()) {
        $imageVoteMax = $image->getVoteMax();
        $userEnergy = $user->getEnergy();
        $userCredits = $user->getUserCredits();
        //When user picks energy for his voteMax
        if ($imageVoteMax == $userEnergy) {
            //Set the energy to the image used picked energy for .
            $imageId = $image->getId();
            $hasEnergyToFalse = $imagesRepository->setEnergyAttribute($imageId, $user);
            //Sets all the other image that use energy to false,
            // cuz only one image can have energy .
            foreach ($hasEnergyToFalse as $hasEnergyToFalse) {
                if ($hasEnergyToFalse->getHasEnergy() == 1) {
                    $hasEnergyToFalse->setVoteMax(0);
                }
                $hasEnergyToFalse->setHasEnergy(0);
            };
            $image->setHasEnergy(1);
            $this->getDoctrine()->getManager()->flush();
            return $this->redirectToRoute('categoryImage_index');
            //If user want to use credits and has insuffisant credits
        } else if (($userCredits < $imageVoteMax) && ($userCredits = $imageVoteMax)) {
            $this->addFlash('msg', "Vous n'avez pas assez de credits!");
        } else {

```

-Si l'utilisateur veut choisir les crédits et il y a suffisant des crédits alors on ajoute le numéro qu' il veut sur le photo et on enlève le numéro de crédits sur son compte.

```

        } else {
            //If user want to use credits for this image
            //set has energy to false
            $image->setHasEnergy(0);
            //Remove the credits used of the userCredits account
            $newUserCredits = $userCredits - $imageVoteMax;
            $image->setCredits($imageVoteMax);
            $user->setUserCredits($newUserCredits);
            $this->getDoctrine()->getManager()->flush();
            return $this->redirectToRoute('categoryImage_index');
        }
    }
}

```

Si l'utilisateur souhaite ajouter des votes sur une photo

-Il peut pas ajouter des votes si il y a pas des crédits ou le photo utilise l'énergie

```
// To add more votes to a specific image
/**
 * @Route("/imagesCategory/{id}/addVote", name="add_vote", methods={"GET", "POST"})
 */
public function addVotes(Request $request, ImagesRepository $imagesRepository, Images $image): Response
{
    $user = $this->get('security.token_storage')->getToken()->getUser();
    $form = $this->createForm(AddVoteType::class, $image);
    $form->handleRequest($request);
    $idImage = $request->get('id');
    if ($form->isSubmitted() && $form->isValid()) {
        $userCredits = $user->getUserCredits();
        $voteCredits = $image->getVoteCredits();
        //Show msg If user hasn't enough credits
        if (($userCredits < $voteCredits) && (($userCredits = $voteCredits))) {
            $this->addFlash('msg', "Vous n'avez pas assez de credits!");
            //Show msg if Image is using energy
        } elseif ($image->getHasEnergy() == true) {
            $this->addFlash('msg', "Vous pouvez pas ajouter si cet image c'est avec energie!");
        } else {
            //If user has credits and image hasn't energy then add the desired votes to image
            $imageVoteMax = $image->getVoteMax();
            $newImageVoteMax = $imageVoteMax + $voteCredits;
            $image->setVoteMax($newImageVoteMax);

            $newUserCredits = $userCredits - $voteCredits;
            $image->setCredits($newImageVoteMax);
            $user->setUserCredits($newUserCredits);
            $this->getDoctrine()->getManager()->flush();
            return $this->redirectToRoute('single_category_image', array('id' => $idImage));
        }
    }
    $user = $this->get('security.token_storage')->getToken()->getUser();
    return $this->render('images/addVote.html.twig', [
        'images' => $imagesRepository->findBy(['user' => $user, 'id' => $idImage]),
        'form' => $form->createView(),
    ]);
}
```

Si l'utilisateur veut changer donner l'attribue énergie à une autres photo

-Il peut pas donner l'attribue énergie si la photo as des credits, ca vais afficher un message d'erreur

```

/**
 * @Route("/imagesCategory/{id}/{energy}", name="toggle_energy", methods={"GET", "POST"})
 */
public function toggleEnergy(Request $request, ImagesRepository $imagesRepository): Response
{
    $user = $this->get('security.token_storage')->getToken()->getUser();
    $idImage = $request->get('id');
    $energy = $request->get('energy');
    $image = $imagesRepository->find(['id' => $idImage]);
    //If image hasn't energy and user request to have energy added to image .
    if ($energy == 1) {
        //If image has credits then it doesn't add energy
        if ($image->getCredits() > 0) {
            $this->addFlash('msg', "Veilleur attendre que les credits soit finis ");
        } else {
            //Add has energy to image
            $hasEnergyToFalse = $imagesRepository->setEnergyAttribute($idImage, $user);
            foreach ($hasEnergyToFalse as $hasEnergyToFalse) {
                //Set the image that had energy to false and their voteMax to 0
                if ($hasEnergyToFalse->getHasEnergy() == 1) {
                    $hasEnergyToFalse->setVoteMax(0);
                }
                //Set all other image with nerenrgy to false
                $energyVoteNumber = $user->getEnergy();
                $image->setVoteMax($energyVoteNumber);
                $hasEnergyToFalse->setHasEnergy(0);
                //Set selected image energy to true
                $image->setHasEnergy(1);
            }
            $this->getDoctrine()->getManager()->flush();
        }
        //If image has energy and user request not to have it anymore
    } else if ($energy == 0) {
        $image->setVoteMax(0);
        $image->setHasEnergy(0);
        $this->getDoctrine()->getManager()->flush();
    };

    $user = $this->get('security.token_storage')->getToken()->getUser();
    return $this->render('images/singleCategoryImage.html.twig', [
        'images' => $imagesRepository->findBy(['user' => $user, 'id' => $idImage]),
    ]);
}

```

Ces Route pour afficher tous les photo d'utilisateur et aussi le photo avec plus d'informationsur le data des photos


```
// Show all the images of the the user
/**
 * @Route("/imagesCategory", name="categoryImage_index", methods={"GET"})
 */
public function showCategoryImage(ImagesRepository $imagesRepository): Response
{
    $user = $this->get('security.token_storage')->getToken()->getUser();

    return $this->render('images/imagesCategory.html.twig', [
        'images' => $imagesRepository->findImagesOfUserWithCategory($user),
    ]);
}

//Show a single image with his category and votes the image got
/**
 * @Route("/imagesCategory/{id}", name="single_category_image", methods={"GET","POST"})
 */
public function showSingleCategoryImage(Request $request, ImagesRepository $imagesRepository): Response
{
    $idImage = $request->get('id');
    $user = $this->get('security.token_storage')->getToken()->getUser();
    return $this->render('images/singleCategoryImage.html.twig', [
        'images' => $imagesRepository->findBy(['user' => $user, 'id' => $idImage]),
    ]);
}
```

Voter:

Les utilisateur peuvent vote sur le categorie qu' ils veulent

Le button vote il peut pas être clicker si tous les votes sont pas choisi

Voter

Est ce que cette personne a l'air ...?

Business



Hello good sir	Hello bad sir	Nasty
<input type="button" value="BEACUP"/>	<input type="button" value="BEACUP"/>	<input type="button" value="BEACUP"/>
<input type="button" value="OUI"/>	<input type="button" value="OUI"/>	<input type="button" value="OUI"/>
<input type="button" value="UN PEU"/>	<input type="button" value="UN PEU"/>	<input type="button" value="UN PEU"/>
<input type="button" value="NON"/>	<input type="button" value="NON"/>	<input type="button" value="NON"/>

Trop beau/belle
Mauvais photo
Joie photo
Mauvaise visibilité
Joie sourire
Trop sérieux/ese

JQuery pour mettre le valeur des button dans le form de symfony et pour le commentaire automatique.

```
// Disable the vote button until the user has pick 3 traits
$('#voteButton').prop("disabled", true);
//First vote , get value
$("#trait_1 button").click(function () {
    var id = $(this);
    $(".trait_1").removeClass("trait_1");
    $(id).addClass("trait_1");
    $("#voteOne").attr("value", $(id).val());
});
//Second vote , get value
$("#trait_2 button").click(function () {
    var id = $(this);
    $(".trait_2").removeClass("trait_2");
    $(id).addClass("trait_2");
    $("#voteTwo").attr("value", $(id).val());
});
//Third vote , get value
$("#trait_3 button").click(function () {
    var id = $(this);
    $(".trait_3").removeClass("trait_3");
    $(id).addClass("trait_3");
    $("#voteThree").attr("value", $(id).val());
});
$("#container").click(function() {
    //Enable the vote only when user has pick 3 traits
    if ( $("#trait_1 button").hasClass("trait_1")==true && $("#trait_2 button").hasClass("trait_2")==true && $("#trait_3 button").hasClass("trait_3")==true ) {
        $('#voteButton').prop("disabled", false);
    }
});
//Add the auto comment to textarea
$("#autoCommentUI li").click(function() {
    var id = $(this);
    autoComment=$(id).text()+ ' '
    $("#commentArea").append( autoComment);
});
```

Le controller de cet page est celui ci qui fait que on vote sur les image avec le categorie choisi et que

les utilisateurs votent pas sur leurs photo .Les utilisateurs votent 3 fois plus sur les photo avec crédits que les photo avec énergie .

```
//To vote by category
/**
 * @Route("/categorie/{name}", name="vote_by_categorie", methods={"GET","POST"})
 */
public function social(Request $request, ImagesRepository $imagesRepository, VoteRepository $voteRepository): Response
{
    $vote = new Vote();
    $user = $this->getUser();
    $voteType = $user->getVoteType();
    $categorie = $request->get('name');
    $form = $this->createForm(VoteType::class, $vote);
    $imageCredits = $imagesRepository->findByCredits($user, $categorie);

    // Chose 3 more times the credits image than energy type vote
    if (($voteType > 0)
        and ($imageCredits != null))
    {
        $newVoteType = $voteType - 1;
        $user->setVoteType($newVoteType);
        return $this->render('vote/categorie.html.twig', [
            'categorienom' => $categorie,
            'images' => $imagesRepository->findByCredits($user, $categorie),
            'vote' => $vote,
            'form' => $form->createView(),
        ]);
    } else {
        return $this->render('vote/categorie.html.twig', [
            'categorienom' => $categorie,
            'images' => $imagesRepository->findByRandom($user, $categorie),
            'vote' => $vote,
            'form' => $form->createView(),
        ]);
    }
}
```

Fonction findByRandom fait qu' on récupère une photo au hasard dans la base de données que l'utilisateur ait pas voter déjà et qu' il y a le même categorie que sur ce que l'utilisateur est en train de voter

```

//Finds random image so the user can vote ,
/**
 * @return Images[] Returns an array of Images objects
 */
public function findByRandom($value, $categorie)
{
    return $this->createQueryBuilder('i')
        ->orderBy('RAND()')
        ->where('i.user != :identifiant')
        ->andWhere('i.voteMax != 0')
        ->setParameter('identifiant', $value)
        ->leftJoin('i.category', 'c')
        ->andWhere('c.name = :name')
        ->setParameter('name', $categorie)
        ->leftJoin('i.votes', 'v')
        ->leftJoin('v.user', 'u', 'WITH', 'u.id = :user')
        ->setParameter('user', $value)
        ->andWhere('u.id IS NULL')
        ->setMaxResults(1)
        ->getQuery()
        ->getResult();
}

```

Fonction findByCredits fait qu' on récupère une photo au hasard dans la base de données que l'utilisateur ait pas voter déjà et qu' il y a le même categorie que sur ce que l'utilisateur est en train de voter et que la photo as l'attribut crédits

```

//find random image with credits,
/**
 * @return Images[] Returns an array of Images objectsRandom
 */
public function findByCredits($value, $categorie)
{
    return $this->createQueryBuilder('i')
        ->orderBy('RAND()')
        ->where('i.user != :identifiant')
        ->andWhere('i.voteMax != 0')
        ->andWhere('i.credits != 0')
        ->setParameter('identifiant', $value)
        ->leftJoin('i.category', 'c')
        ->andWhere('c.name = :name')
        ->setParameter('name', $categorie)
        ->leftJoin('i.votes', 'v')
        ->leftJoin('v.user', 'u', 'WITH', 'u.id = :user')
        ->setParameter('user', $value)
        ->andWhere('u.id IS NULL')
        ->setMaxResults(1)
        ->getQuery()
        ->getResult();
}

```

Après que les utilisateur ont voté il passe plusieurs choses

- On ajoute des points d'énergie au Utilisateur qui a voté avec une max de 28 point
- On enlève des points au utilisateur auquel la photo appartient , on regarde si il utilise l'énergie pour cet image alors on enlève 3 point sinon on enlève un crédit
- On les redirige sur la Route précédente avec le meme categorie mais une autre photo

```
//After user votes
/**
 * @Route("/addVote/{name}/{id}", name="vote_new_categorie", methods={"POST"})
 */
public function addVote(Request $request, Image $image): Response
{
    $form = $this->createForm(VoteType::class, $vote);
    $form->handleRequest($request);
    $user = $this->getUser();
    //Add Energy lvl to user
    $userEnergy = $user->getEnergy();
    $newUserEnergy = $userEnergy + 1;
    $user->setEnergy($newUserEnergy);
    if ($user->getEnergy() == 28) {
        $user->setEnergy(27);
    }
    $vote->setUser($user);
    $vote->setImages($image);
    //Reduce vote point when voted;
    $voteMax = $image->getVoteMax();
    $credits = $image->getCredits();
    //If the user is using energy then lower his energy too
    if ($image->getHasEnergy() == 'true') {
        $newVoteMax = $voteMax - 3;
        $userImage = $image->getUser();
        $userOfImage = $userImage->getEnergy();
        //Remove 3 points of energy the user of the image
        $newUsersImageEnergy = $userOfImage - 3;
        $userImage->setEnergy($newUsersImageEnergy);
    } else {
        //Remove 1 vote on the image
        $newVoteMax = $voteMax - 1;
    }
}
```

```
    } else {  
        //Remove 1 vote on the image  
        $newVoteMax = $voteMax - 1;  
        $newCredits = $credits - 1;  
        $image->setVoteMax($newVoteMax);  
        $image->setCredits($newCredits);  
    }  
    //Vote type for showing 3 more times credits image then energy lv image  
    $voteType = $user->getVoteType();  
    if ($voteType > 0) {  
        $newVoteType = $voteType - 1;  
        $user->setVoteType($newVoteType);  
    } else {  
        $user->setVoteType(3);  
    }  
  
    $entityManager = $this->getDoctrine()->getManager();  
    $entityManager->persist($vote);  
    $entityManager->flush();  
    $categorie = $request->get('name');  
    return $this->redirectToRoute('vote_by_categorie', array('name' => $categorie));  
}
```

Paie ment avec stripe:

J'ai choisi stripe (Stripe est un processus de paiement en ligne, similaire à Paypal. Ils proposent leur service dans 25 pays à travers le monde. Des sociétés telles que Unicef, Lyft et Deliveroo utilisent Stripe pour leur service de paiement.) pour les paiements sur mon site. Les utilisateurs peuvent acheter des crédits pour faire voter les photos plus rapidement.

Ils peuvent choisir sur 4 variantes des prix .



Chacun des ces variable est composé de ce manier ou dans l'intérieur ont une form stripe avec une script stripe .Le input type='hidden' , l'utilisateur ne peut pas le voir mais il est la pour envoyer les valeur ou controller .

```

<!-- column for price -->
<div class="col-lg-3 col-md-5">
  <div class="card text-center card-shadow on-hover border-0 mb-4">
    <div class="card-body font-14">
      <span class="badge badge-inverse p-2 position-absolute price-badge font-weight-normal">Populaire</span>
      <h5 class="mt-3 mb-1 font-weight-medium">INTERMEDIATE</h5>
      <h6 class="subtitle font weight normal">Pour 30 Credits</h6>

      <div class="pricing my-3">
        <span class="monthly display-5">16</span>
        <sup>
          <i class="fas fa-euro-sign"></i>
        </sup>
      </div>
    </div>
    {# Stripe form #}
    <form action="/create-checkout-session" method="POST">
      <input type="hidden" name="credits" value="30">
      <input type="hidden" name="price" value="1600">
      <div class="button-btn">
        {# Stripe payment script #}
        <script src="https://checkout.stripe.com/checkout.js" class="stripe-button" data-key="{{stripe_p">
        </script>
        // Hide default stripe button, be careful there if you
        // have more than 1 button of that class
        document.getElementsByClassName("stripe-button-el")[1].style.display = 'none';
        </script>

        <button class="btn btn-danger-gradient btn-md text-white btn-block" type="submit">
          Acheter
        </button>
      </div>
    </form>
  </div>

```

Cet form envoie les donner au controller StripeController dans la route

/create-checkout-session

Cette route envoie les données dans stripe api pour le paiement .Une fois le paiement effectué, les crédits sont ajoutés dans le compte de l'utilisateur .

```
/**
 * @Route("/create-checkout-session", name="checkout")
 */
public function checkout(Request $request)
{
    $user = $this->getUser();

    if ($request->isMethod('POST')) {
        $token = $request->get('stripeToken');

        \Stripe\Stripe::setApiKey('sk_test_51HZCESIFy5wkBuVsgrzr16
        \Stripe\Charge::create(array(
            "amount" => $request->get('price'),
            "currency" => "eur",
            "source" => $token,
        ));
        //Add credits to user account
        $oldCredits = $user->getUserCredits();
        $credits = $request->get('credits');
        $newCredits = $oldCredits + $credits;
        $user->setUserCredits($newCredits);
        $this->getDoctrine()->getManager()->flush();
    }
    return $this->render('credits/stripeTest.html.twig');
}
```

Partie blog :

J'ai décidé d'ajouter un parti blog sur mon site pour que les utilisateurs puissent regarder des Tips comment prendre des photos .

J'utilise le ckeditor pour gérer le création d'articles

```
twig:
  form_themes:
    - "@FOSCKEditor/Form/ckeditor_widget.html.twig"
fos_ck_editor:
  configs:
    main_config:
      toolbar:
        - (
          name: "styles",
          items:
            "Bold",
            "Italic",
            "Underline",
            "Strike",
            "Blockquote",
            "-",
            "Link",
```

J'utilise elfinder pour mettre des images dans les articles quand je suis en train de créer un article

```
config > packages > ! fm_elfinder.yaml
1  fm_elfinder:
2    assets_path: /public/fm_elfinder # chemin des fichiers JS
3    instances:
4      default:
5        locale: fr # Langue
6        editor: ckeditor # Editeur utilisé
7        fullscreen: true # Taille d'affichage
8        theme: smoothness # Thème à utiliser
9
10     connector:
11       debug: false # Désactive le debug
12       roots:
13         uploads:
14           show_hidden: false # Masque les fichiers cachés
15           driver: LocalFileSystem # Pilote des fichiers
16           path: uploads/images # Chemin d'upload
17           upload_allow: ["image/png", "image/jpg", "image/jpeg"] # Fichiers autorisés
18           upload_deny: ["all"] # Fichiers interdits
19           upload_max_size: 2M # Taille maximum
20
```

Admin:

L'admin peut créer des articles , ajouter, supprimer, modifier les catégories, les traits et aussi il peut supprimer une photo inappropriée .

Accueil Blog BlogCategory Trails Category Delete Images Logout						
Article index						
id	Title	Content	CreatedAt	UpdatedAt	Published	action

Après que le photo est supprimé l'utilisateur reçoit un message que ca photo et supprimer .

```

/**
 * @Route("/images/delete/{images}", name="admin_delete_image",
methods={"GET","POST"})
 */
public function deleteImage(Request $request, VoteRepository
$voteRepository, ImagesRepository $imagesRepository): Response
{
    $imageId=$request->get('images');
    $message = new Messages();

    $image=$imagesRepository->findOneBy(['id' => $imageId]);
    $user = $image->getUser();
    if ($image->getHasEnergy() == 0) {
        $userCredits = $user->getUserCredits();
        $credits=$image->getCredits();
        $newUserCredits = $userCredits + $credits;
        $user->setUserCredits($newUserCredits);
    }
    $voteRepository->deleteByImages($imageId);
    $imageName= $image->getName();
    $message->setPhotoname($imageName)
        ->setMessage("Votre photo étai supprimer pour des
raisonne que cette photo est inapproprié ")
        ->setUser($user);

    $imagesRepository->deleteByImages($imageId);

    $entityManager = $this->getDoctrine()->getManager();
    $entityManager->persist($message);
    $entityManager->flush();
    return $this->redirectToRoute('admin_render_images' );
}

```

Docker:

Docker est une plateforme permettant de lancer certaines applications dans des conteneurs logiciels.

Sur docker-compose j'ai créé un conteneur pour le nginx, comme image. NGINX est un logiciel open source pour le service Web, le proxy inverse, la mise en cache, l'équilibrage de charge. Sur le port 8083. J'ai configuré la conf nginx sur le fichier conf.d.

```
nginx-service:
  image: nginx:stable-alpine
  container_name: nginx-container
  ports:
    - "8083:80"
  volumes:
    - ./:/var/www/project
    - ./nginx/conf.d:/etc/nginx/conf.d/
  depends_on:
    - www
    - db
  networks:
    - dev
```

Ensuite le conteneur pour mon projet symfony.

```
www:
  build: php
  container_name: www_docker_symfony
  volumes:
    - ./:/var/www/project
  restart: always
  ports:
    - 9000:9000
  networks:
    - dev
```

Je lui dis de build le DockerFile qui se trouve sur le folder php et que le volume de projet se trouve à la racine.

Ensuite le conteneur de database et phpmyadmin pour que je puisse me connecter avec la base de donner.

```
db:
  image: mysql
  container_name: db_docker_symfony
  restart: always
  volumes:
    - db-data:/var/lib/mysql
  environment:
    MYSQL_ALLOW_EMPTY_PASSWORD: password
  networks:
    - dev

phpmyadmin:
  image: phpmyadmin
  container_name: phpmyadmin_docker_symfony
  restart: always
  depends_on:
    - db
  ports:
    - 8023:80
  environment:
    PMA_HOST: db
  networks:
    - dev
```

Ci-Cd:

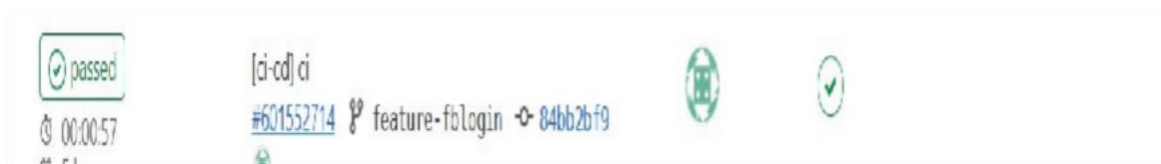
J'import image jakzal/phpqa :php7.4 qui va me permettre de deploie sur un enviroment de php7.4 Sur le before script j'install composer ensuite je passe des tests avec phpunit.

```
gitlab-ci (ci.json)
image: jakzal/phpqa:php7.4
before_script:
  - composer install

cache:
  paths:
    - vendor/

stages:
  - test

test_basic:
  stage: test
  script:
    - vendor/bin/phpunit --testdox
```



Tests Unitaire :

Je teste le connexion avec la base de donner pour les entite.

```
class UserUnitTest extends TestCase
{
    public function testUserExists(): void
    {
        $user = new User;
        $user->setEmail("test@test.com")
        ->setPrenom('prenom')
        ->setNom('nom')
        ->setPassword('password');
        $this->assertTrue($user->getEmail() === "test@test.com");
        $this->assertTrue($user->getPrenom() === "prenom");
        $this->assertTrue($user->getNom() === "nom");
        $this->assertTrue($user->getPassword() === "password");
    }
    public function testUserFalse(): void
    {
        $user = new User;
        $user->setEmail("test@test.com")
        ->setPrenom('prenom')
        ->setNom('nom')
        ->setPassword('password');

        $this->assertFalse($user->getEmail() === "false@test.com");
        $this->assertFalse($user->getPrenom() === "false");
        $this->assertFalse($user->getNom() === "prenom");
        $this->assertFalse($user->getPassword() === "pass");
    }
    public function testUserEmpty(): void
    {
        $user = new User();
        $this->assertEmpty($user->getEmail());
        $this->assertEmpty($user->getPrenom());
        $this->assertEmpty($user->getNom());
        $this->assertEmpty($user->getPassword());
    }
}
```

Je test si l'utilisateur existe. Si l'utilisateur est false et si l'utilisateur est vide.

```
46 Warning: Your XML configuration validates against a deprecated schema.
47 Suggestion: Migrate your XML configuration using "--migrate-configuration"!
48 Testing
49 Category Unit (App\Tests\CategoryUnit)
50 ✓ Category exists
51 ✓ Category false
52 ✓ Category empty
53 Default Controller (App\Tests\ControllerTests\DefaultController)
54 ✓ Default controller
55 ✓ Default controller about
56 Images Unit (App\Tests\ImagesUnit)
57 ✓ Image exists
58 ✓ Image false
59 ✓ Image empty
60 Unit (App\Tests\Unit)
61 ✓ Images
62 User Unit (App\Tests\UserUnit)
63 ✓ User exists
64 ✓ User false
65 ✓ User empty
66 Vote Unit (App\Tests\VoteUnit)
67 ✓ Vote exists
68 ✓ Vote false
69 ✓ Vote empty
70 Time: 00:00.988, Memory: 60.50 MB
71 OK (15 tests, 55 assertions)
73 Saving cache for successful job
74 Creating cache default-non protected...
```

Toutes les tests qui passent avec succes.

Application Desktop :

Sur une des mes briefs j'utilise electron pour déployer sur desktop une app qui nous permette de jouer un jeu pour trouver en 5 coups le numéro entre 1-50. Sur main.js je lui dis de loadFile index.html où se trouve le jeu aussi de switch le dark-mode dans le jeu. , app.whenReady va nous permettre d'appeler createWindow qui nous permet de créer un Windows desktop.

```
const { app, BrowserWindow, ipcMain, nativeTheme } = require('electron');
const path = require('path');
require('electron-reload')(__dirname);

const createWindow = () => {
  const win = new BrowserWindow({
    width: 800,
    height: 600,
    webPreferences: {
      preload: path.join(__dirname, 'preload.js')
    }
  })

  win.loadFile('index.html')

  ipcMain.handle('dark mode:toggle', () => {
    if (nativeTheme.shouldUseDarkColors) {
      nativeTheme.themeSource = 'light'
    } else {
      nativeTheme.themeSource = 'dark'
    }
    return nativeTheme.shouldUseDarkColors
  })

  ipcMain.handle('dark-mode:system', () => {
    nativeTheme.themeSource = 'system'
  })
}

app.whenReady().then(() => {
  createWindow()

  //Open a window if none are open on macos
  app.on('activate', () => {
    if (BrowserWindow.getAllWindows().length === 0) createWindow()
  })
})

app.on('window-all-closed', () => {
  if (process.platform !== 'darwin') app.quit()
})
```


CONCLUSIONS

J'ai beaucoup aimé travailler sur ce projet qui me tient à cœur.

Le projet peut encore être amélioré, étant from scratch, mais propose une interface fluide et intuitive qui permet à l'utilisateur de faire ce qu'il a à faire, à savoir visualiser des photo acheter des crédits ,créer un compte, renseigner des informations personnelles, les modifier,L'administrateur peut changer les catégorie , les traits , gere les articles . Symfony c'était nouveau pour moi et j'ai dû lire beaucoup de la doc pour reussire faire ce projet . Je suis fière de moi d' avoir réalisé ce projet parce que ca ma permis de gagner beaucoup de compétences en symfony, js , jquery aussi de compétences en debug . Il y a rien de plus beau que de reussire un fonctionnalité que j'ai passé des heures à coder (Le cas de Gmail) . Je tiens encore à remercier tout le monde qui ont fait partie de cet incroyable voyage au sein de Simplon et en dehors .