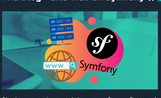
Retour

The Blog - Site Web en Symfony



C'est mon premiers projet personnel sous Framework réalisé.

Dans le cadre de ma politique Qualité, Sécurité et Environnement (QSE), PROMEO s'oriente vers la dématérialisation des documents liés à son workflow métier.

En effet, après une campagne de dématérialisation de la satisfaction client sur la partie alternance, PROMEO souhaite poursuivre sur la partie formation continue.

C'est pourquoi PROMEO a souhaité développer une application pour analyser les besoins des salariés de la division Langues.

Elle se présente sous deux aspects. Une application destinée aux clients de PROMEO. Un autre accès pour les employés de PROMEO.

Cette application vise à dématérialiser l'analyse des besoins de formation linguistique. Les fonctionnalités notables sont l'envoie d'email ainsi que la gestion de session et hashage de mots de passe. J'ai également utilisé Codacy pour suivre les erreurs et duplications de code de mon projet.

J'ai pris 2 mois pour le réaliser et cela m'a permis de comprendre l'utilisation de frameworks, le front-end et le back-end ainsi que l'orienté objet et le modèle MVC.

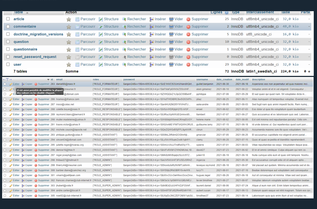
Compétences :

- ✔ Gérer le patrimoine Informatique
  - Exploiter des référentiels, normes et standards adoptés par le prestataire informatique
  - Mettre en place et vérifier les niveaux d'habilitation associés à un service
  - Gérer des sauvegardes
  - Vérifier le respect des règles d'utilisation des ressources numériques
- ✔ Répondre aux demandes d'assistance et d'évolution
  - Collecter, suivre et orienter des demandes
  - Traiter des demandes concernant les applications
- ✔ Développer la présence en ligne de l'organisation
  - Référencer les services en ligne de l'organisation et mesurer leur visibilité.
  - Participer à l'évolution d'un site Web exploitant les données de l'organisation.
- ✔ Travailler en mode projet
  - Planifier les activités
- ✔ Mettre à disposition des utilisateurs un service informatique
  - Réaliser les tests d'intégration et d'acceptation d'un service
- ✔ Organiser son développement professionnel
  - Mettre en place son environnement d'apprentissage personnel
  - Mettre en œuvre des outils et stratégies de veille informationnelle

Technologies :

- GitHub
- HTML
- CSS
- PHP
- JavaScript
- Symfony
- Composer
- PHPMyAdmin
- Mailtrap
- Codacy

Base de données

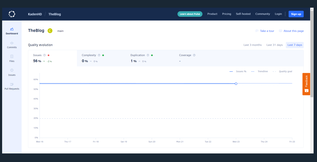


Code des fonctionnalités de l'inscription

```
    /**
     * @Route("/user/create", name="app_register")
     */
    public function register(Request $request, UserPasswordEncoderInterface $passwordEncoder): Response
    {
        if ((this->getUser().getRoles() != ["ROLE_SUPER_ADMIN"])
        && (this->getUser().getRoles() != ["ROLE_ADMIN"]))
        {
            return this->redirectToRoute('user_index', ['alert' => "userCantCreate"]);
        }

            $user = new User();
            $form = this->createForm(RegistrationFormType::class, $user);
            $form->handleRequest($request);

            if ($form->isSubmitted() && $form->isValid()) {
                $user->setDescription('');

                //select option role
                $roles = $request->request->get('roles');
                if ($roles == "formateur") { $role = ["ROLE_FORMATEUR"]; }
                elseif ($roles == "responsable") { $role = ["ROLE_RESPONSABLE"]; }
                elseif ($roles == "assistant") { $role = ["ROLE_ASSISTANT"]; }
                elseif ($roles == "admin") { $role = ["ROLE_ADMIN"]; }
                elseif ($roles == "super") { $role = ["ROLE_SUPER_ADMIN"]; }
                $user->setRoles($role);

                // encode the plain password
                $user->setPassword(
                    $passwordEncoder->encodePassword(
                        $user,
                        $form->get('plainPassword')->getData()
                    )
                );
                $user->setDateCreation(new \DateTime())
                );

                $entityManager = this->getDoctrine()->getManager();
                $entityManager->persist($user);
                $entityManager->flush();
                // do anything else you need here, like send an email

                return this->redirectToRoute('users_show', ['alert' => "userCreated"]);
            }

            return this->render('registration/register.html.twig', [
                'registrationForm' => $form->createView(),
                ]);
        }
    }
```

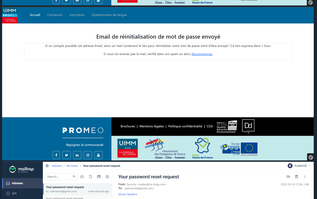
```
{% extends 'base.html.twig' %}

{% block title Utilisateur # utilisateur.id %endblock %}

{% block body %}

    {% include 'header.html.twig' %}

    {% for flashMsgTyp in app.flashes('verify_email_error') %}
        <div class="alert alert-danger" role="alert">{{ flashMsgTyp }}</div>{%endfor
    {% endif %}
    <div>
        <div class="container">
            <div class="row">
                <div class="col-12">
                    <h1 {% if form.vars.value.id % Fixed weight normal text-center">Créer un utilisateur</h1>
                    <div class="card">
                        <div class="card-center">
                            {{ form_start(registrationFormUsers) }}
                            <fieldset>
                                <legend>Rôle</legend>
                                <select name="roles" class="form-control">
                                    <option value="formateur">Formateur</option>
                                    <option value="responsable">Responsable de sections</option>
                                    <option value="assistant">Assistant</option>
                                    {% if app.user.roles == ["ROLE_SUPER_ADMIN"] %}
                                    <option value="admin">Admin</option>
                                    <option value="super">Super Administrateur
                                    {% endif %}
                                </select>
                            </fieldset>
                            {{ form_row(registrationFormUsers.username, {
                                label: 'Nom',
                                attr: {'placeholder': 'Nom de l utilisateur'}
                                }) }}
                            {{ form_row(registrationFormUsers.email, {
                                label: 'Email',
                                attr: {'placeholder': 'Email de l utilisateur'}
                                }) }}
                            {{ form_row(registrationFormUsers.plainPassword) }}
                        </div>
                            {{ form_row(registrationFormUsers.agreeTerms, {
                                label: 'Accepter la charte?'
                                }) }}
                            <br />
                            <div class="text-center">
                                <button type="submit" class="btn btn-primary" onclick="return confirm('Souhaitez-vous vraiment créer cet utilisateur ?')">Créer</button>
                            </div>
                            {{ form_end(registrationFormUsers) }}
                        </div>
                    </div>
                </div>
            </div>
        </div>
    {% include 'footer.html.twig' %}
{% endblock %}
```

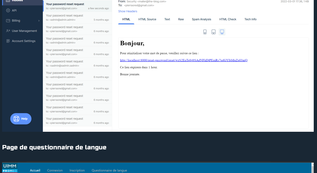
Utilisation de Codacy



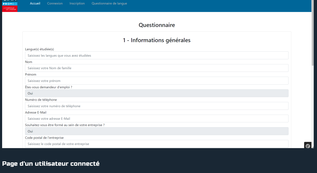
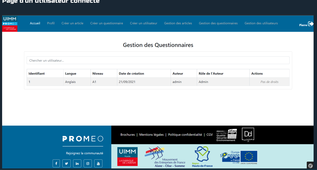
Page de connexion

Page de réinitialisation de mot de passe





Page de questionnaire de langue



Page d'un utilisateur connecté