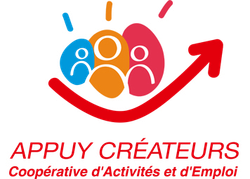


2020 - 2021

***Présenté par Philippe PERECHODOV***



*Maître de stage :*

*DUGRAIS Julien, formateur chez Apprendre.co*

*Stage effectué*

*avec Appuy-créateur*

Dossier de projet : FLORA

# Remerciements

Avant tout, je tiens à remercier mon maître de stage, Julien DUGRAIS, entrepreneur salarié, formateur chez Apprendre.co[[1]](#footnote-1), pour m’avoir accueilli, avoir partagé ses connaissances et son expérience avec moi et m’avoir fait confiance.

En effet, Julien a su m’apprendre, me conseiller, m’expliquer et me transmettre son métier toujours avec enthousiasme, bonne humeur et humour et ce, durant toute la durée du stage.

Il m’a été très agréable et enrichissant de travailler avec Julien en alternant les moments d’échange, de réflexion, de travail en autonomie et, le tout, dans une excellente ambiance et avec bienveillance.

Je remercie également mes formateurs et toute l’équipe pédagogique de l’*Access Code School* et d’*Online Formapro* pour leur accompagnement et leur disponibilité tout au long de cette formation.

Je vous souhaite une bonne lecture et espère que vous apprécierez ce dossier de projet.

Table des matières

[Remerciements 2](#_Toc61360726)

[Table des matières 3](#_Toc61360727)

[Présentation 5](#_Toc61360728)

[Résumé du projet 6](#_Toc61360729)

[Les compétences du référentiel couvertes par le projet 7](#_Toc61360730)

[ Développer la partie front-end d’une application web ou web mobile en intégrant les recommandations de sécurité 7](#_Toc61360731)

[ Maquetter une application 7](#_Toc61360732)

[ Réaliser une interface utilisateur web statique et adaptable 7](#_Toc61360733)

[ Développer une interface utilisateur web dynamique 7](#_Toc61360734)

[ Développer la partie back-end d’une application web ou web mobile en intégrant les recommandations de sécurité 7](#_Toc61360735)

[ Développer les composants d’accès aux données 7](#_Toc61360736)

[ Développer la partie back-end d’une application web ou web mobile 7](#_Toc61360737)

[Cahier des charges & spécifications fonctionnelles du projet 9](#_Toc61360738)

[ Git & GitHub 10](#_Toc61360739)

[ Environnement de développement 10](#_Toc61360740)

[ Langages utilisés 10](#_Toc61360741)

[ Utilisation de DOCKER 10](#_Toc61360742)

[ Accessibilité 12](#_Toc61360743)

[ Base de données et sa gestion 12](#_Toc61360744)

[ Exigences 13](#_Toc61360745)

[ Méthode de gestion de projet AGILE 13](#_Toc61360746)

[Spécifications techniques du projet 15](#_Toc61360747)

[Réalisations 16](#_Toc61360748)

[ Maquettage de la nouvelle interface administration 16](#_Toc61360749)

[ Amélioration de la page d’accueil du site 16](#_Toc61360750)

[ Nouvelle interface utilisateur de la tablette tactile 16](#_Toc61360751)

[ Blablabla 16](#_Toc61360752)

[ Blablabla 16](#_Toc61360753)

[Présentation d’un jeu d’essai 17](#_Toc61360754)

[ Les tests unitaires et PHPUnit 17](#_Toc61360755)

[ Le test unitaire 17](#_Toc61360756)

[ PHPUnit 17](#_Toc61360757)

[ La fonctionnalité et les tests effectués 17](#_Toc61360758)

[ La fonctionnalité développée 17](#_Toc61360759)

[ Les tests effectués 19](#_Toc61360760)

[Description d'une veille sur la sécurité & résolution d’une faille 21](#_Toc61360761)

[Recherches sur un site anglophone, extrait et traduction 22](#_Toc61360762)

[ Recherche effectuée 22](#_Toc61360763)

[ Extrait du forum anglophone 22](#_Toc61360764)

[ La question 22](#_Toc61360765)

[ La réponse 23](#_Toc61360766)

[ Traduction 23](#_Toc61360767)

[ De la question 23](#_Toc61360768)

[ De la réponse 24](#_Toc61360769)

[ Solution adoptée 24](#_Toc61360770)

[Conclusion 25](#_Toc61360771)

# Présentation

J’ai effectué mon stage uniquement avec Julien DUGRAIS, maître de stage, néanmoins, j’ai pu rencontrer plusieurs utilisateurs avec lesquels j’ai pu échanger tant sur leurs besoins que sur leurs retours d’expérience avec les fonctionnalités développées.

En effet, Julien travail seul en sa qualité d’entrepreneur-salarié de la société Appuy-Créateurs à Clermont-Ferrand et m’a accompagné durant toute la durée de mon stage en parallèle de ses nombreuses activités telle que formateur, développeur, conseil en installation, réseau et système, *community manager*, sécurisation des SI, …

Concernant le coworking de Moulins, Julien est un des membres fondateurs et animateur de l’association « CoWorkInMoulins » (Loi 1901).

De plus, il est le concepteur, réalisateur et administrateur du site internet du coworking[[2]](#footnote-2) qui est basé sur l’architecture du projet FLORA, application de gestion et d’organisation.

Le projet « FLORA », acronyme signifiant « **F**ablab **L**ibre c**O**working o**R**ganisation incub**A**teur »[[3]](#footnote-3), a été créé par Julien DUGRAIS.

Cette application est utilisée pour le développement de projet client faciliter la gestion et l’organisation de structure dans laquelle il est nécessaire d’avoir un espace utilisateur sécurisé, un espace dédié à l’administration de la structure ainsi qu’une interface utilisateur destiné au grand public et affichant les informations de la structure (présentation, localisation, tarifs, …).

Ce projet FLORA a donc été utilisé pour la gestion de l’association de coworking avec ses membres et ses différents locaux.

Mon stage s’est déroulé, dans sa quasi-totalité au coworking de Moulins (03) à l’exception de quelques jours chez Julien DUGRAIS ou en télétravail quand Julien ne pouvait être présent dans les locaux du coworking (cf. supra).

La participation au projet « FLORA » m’a offert l’opportunité de mettre en œuvre mes connaissances acquises durant la formation mais a permis de les améliorer significativement, ce qui est un des buts de ce projet. C’est ce que je vais vous résumer.

# Résumé du projet

Ma mission a été d’effectuer des mises à jour et de développer diverses fonctionnalités inexistantes à ce jour liées au front-end et back-end utilisateur.

Il s’agit d’un projet évolutif et en « open source » avec une architecture similaire à celle de Symfony mais accessible à des développeurs débutants et à des stagiaires.

Au niveau du front-end, sont utilisés le HTML, CSS, Javascript et Twig, tandis qu’au niveau back-end, le PHP natif est utilisé avec un design pattern MVC et une base de données en SQLite gérée avec Phinx.

Une 1ère partie de l’application est accessible au public et permet de présenter le coworking et ses spécificités tandis qu’une 2nde partie est destinée aux membres enregistrés et disposant de divers droits d’accès selon leur « rôles » dans l’association. Cet espace privé permet, par exemple, la consultation des heures de présence, l’achat de produits ou de services, le suivi de la comptabilité, l’émission de facture…

Il existe également une borne à l’entrée du Coworking sur laquelle les membres peuvent déclarer leur présence (entrée et sortie de l’espace physique) par un système de login et de logout.

Enfin, un écran de télévision au format portrait permet d’afficher des informations (évènements, réservations, informations…) visible de l’extérieur du coworking et qui se rafraichi toutes les 60 secondes par un script Javascript intégré directement à la page d’affichage.

J’ai d’abord dû prendre connaissance du projet et procéder à la lecture et explication du code avec mon maître de stage puis, ayant défini différentes *users story*, tant orientées front-end que back-end, j’ai commencé à réaliser les différentes tâches demandées.

J’ai pu donc réaliser des maquettes pour redesigner et optimiser l’UI et l’UX suite aux retours des utilisateurs pour l’interface administrateur et de la tablette tactile, créer une fonctionnalité « mot de passe oublié » tout en assurant la sécurité et la confidentialité des données, participer et réaliser des tests unitaires, améliorer et compléter la page principale du site internet, créer une messagerie interne aux membres, …

Ces réalisations ont été faites dans l’application déjà existante créée avec un souci de sécurité qu’il m’a fallu comprendre, utiliser, ne pas compromettre et améliorer le cas échéant.

# Les compétences du référentiel couvertes par le projet

## Développer la partie front-end d’une application web ou web mobile en intégrant les recommandations de sécurité

### Maquetter une application

J’ai réalisé une maquette de la nouvelle interface de l’espace d’administration de l’application en prenant en considération l’expérience utilisateur (*user experience* ou UX) afin d’en faciliter son accessibilité et sa prise en main. J’ai également pris en compte l’interface utilisateur (*user interface* ou IU) pour que cette interface puisse être facilement compréhensible et intuitive pour l’utilisateur, pour cela, j’ai demandé l’avis et le retour de différent utilisateur de l’application.

### Réaliser une interface utilisateur web statique et adaptable

J’ai dû réaliser une mise à jour de la page d’accueil de l’application en y implémentant de nouvelles informations (horaires, localisation, plan, menu, retour haut de page…). L’aspect *responsive* de l’application étant indispensable, ces éléments sont, certes, statiques mais s’adaptent parfaitement à tout support de visualisation, résolution ou format d’affichage, notamment sur mobile.

### Développer une interface utilisateur web dynamique

J’ai recréé tout le visuel de la tablette tactile à l’entrée du coworking permettant aux utilisateurs de déclarer leur arrivé ou leur départ de l’espace de travail. Le format et la résolution d’affichage étaient imposés, toutefois, l’interface devait être dynamique en affichant la date et l’heure en temps réelle et afficher la liste exhaustive des utilisateurs, enregistrées dans la base de données, pouvant être sélectionnées.

## Développer la partie back-end d’une application web ou web mobile en intégrant les recommandations de sécurité

### Développer les composants d’accès aux données

Il m’a été demande de créer un système de messagerie interne ne pouvant être accessible que par les utilisateurs enregistrés. L’utilisateur devait pouvoir envoyer un nouveau message à un destinataire ou en groupe de destinataire (par exemple : les administrateur). De plus, il fallait que les messages soit classés par conversation et par ordre antichronologique avec possibilité de répondre à un expéditeur.

### Développer la partie back-end d’une application web ou web mobile

Il n’y avait pas de possibilité de recréer un mot de passe en cas d’oubli du précèdent à moins d’en faire la demande directement auprès d’un administrateur.

Ainsi, j’ai créé toute la fonctionnalité sécurisée permettant à utilisateur de changer facilement et rapidement son mot de passe en cas d’oubli.

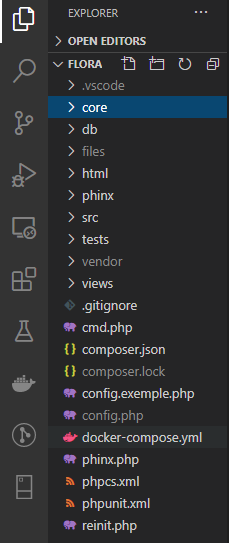
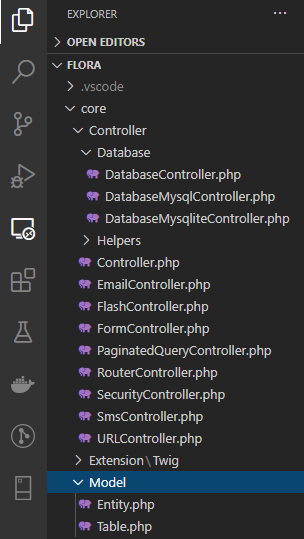
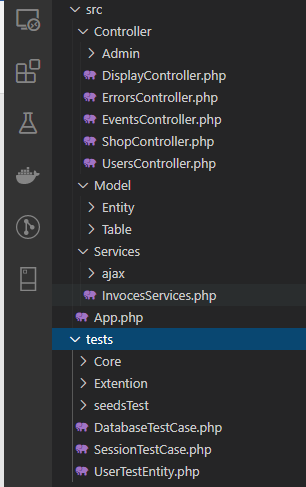
# Cahier des charges & spécifications fonctionnelles du projet

L’application sur laquelle j’ai travaillé existait déjà et était en production. Des utilisateurs étaient déjà enregistrés et utilisait l’application, notamment pour le suivi des heures de présence physique à l’espace de coworking.

Pour les responsables du coworking, l’application sert principalement à la gestion des utilisateurs, à la facturation de ceux-ci selon les heures de présence et de la formule tarifaire choisie.

Il était donc impératif de ne pas compromettre le bon fonctionnement de l’application et d’y insérer les modifications ou nouvelles fonctionnalités en toute sécurité.

Avant de commencer à développer, j'ai dû prendre en main l'architecture existante du projet afin de comprendre l’organisation et la structure de celui-ci afin de m’approprier le code et de pouvoir connaitre les langages et outils déjà en place. Cela s’est fait par une séance de lecture de code avec mon maître de stage me présentant en détail son application et son fonctionnement.

Arborescence des fichiers > partie « Core » > partie « src » et « tests »

Puis, nous avons pu mettre en place l’environnement de travail dans lequel il souhaitait que nous travaillions.

## Git & GitHub

Nous étions deux à travailler sur l’application, mon maître de stage et moi-même. Il était donc indispensable et pouvoir versionner nos avancées et de séparer nos travaux sur différentes branches de travail.

Ainsi, une branche « prod » utilisée pour l’application en production tandis que la branche « main » a été utilisée comme branche de teste en condition réelle identique à celles du serveur de l’application en production.

De plus, chacun d’entre nous disposait d’une branche de développement subdivisée en plusieurs branche pour chaque fonctionnalités (*features*) développées.

Les fusions de branche (*merge*) se faisaient par palier en fusionnant la branche mère avec la branche fille, puis, après testes et vérifications, en fusionnant la branche fille validée sur la branche mère (à l’exception de la branche « prod » uniquement gérée par mon maitre de stage).

Afin de prévenir tout conflit de fusion, j’ai utilisé les « merge à blanc » permettant de simuler un *merge* afin de voir l’existence de conflits.

Voici la commande à exécuter sur un terminal :

$ git merge --no-commit --no-ff <branch-name>

S’il est nécessaire de défaire le merge et retourner dans l’état initial, il suffit d’exécuter la commande suivante :

$ git merge --abort

Cela a permis de détecter quelques conflits en amont.

## Environnement de développement

### Langages utilisés

Le *back-end* du projet est développé principalement en PHP natif sans *framework* afin de pouvoir librement procéder à des évolutions ou des modifications et pour en faciliter la maintenance, toutefois, l’organisation du projet est similaire à celle de *Symfony.*

Au niveau du *front-end*, les langages utilisés sont le HTML, le CSS, le Twig et le Javascript.

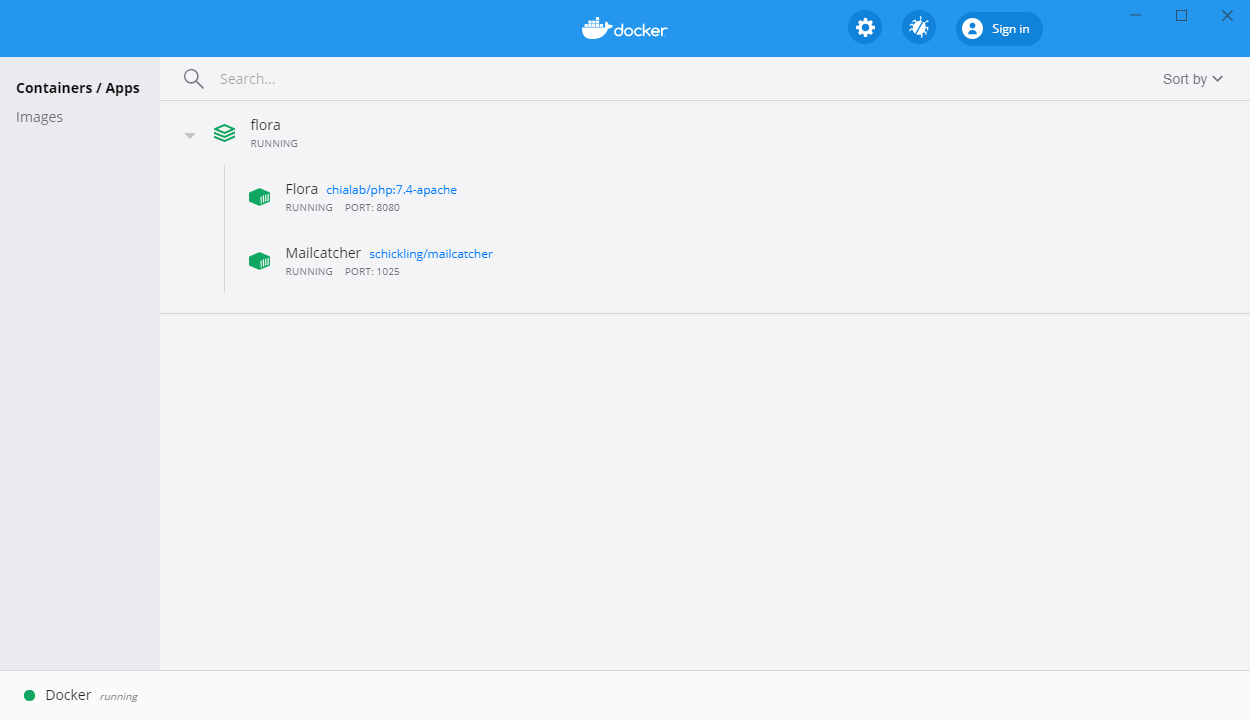
### Utilisation de DOCKER

Afin d’avoir un environnement de développement identique au serveur hébergeant l’application en production, j’ai dû utiliser DOCKER[[4]](#footnote-4). Il s’agit d’un « outil qui peut empaqueter une application et ses dépendances dans un conteneur isolé, qui pourra être exécuté sur n'importe quel serveur »[[5]](#footnote-5). Cela fonctionne avec des conteneurs logiciels appelés *container*.

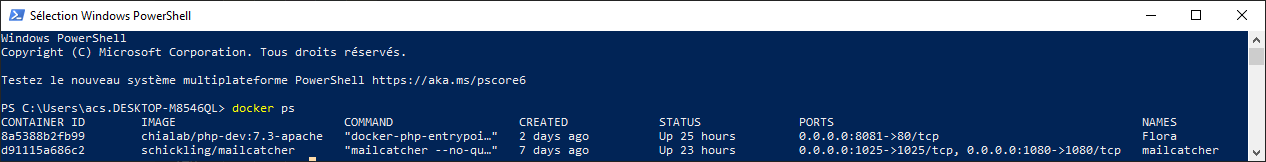
L’utilisation de DOCKER et de Docker Compose permet d’avoir un environnement stable, efficace et identique entre tous les utilisateurs simplifiant le travail en équipe en ce que chaque développeur travail sur un environnement identique aux autres.

DOCKER s’utilise par ligne de commande dans un terminal ou via interface visuelle.

Pour le projet, j’ai dû utiliser un *« mailctacher »*. L’image du conteneur du serveur « PHP 7.4 apache » et celle du *« mailctacher »* ont été empilés et utilisés avec Docker Compose et un fichier « .yml » inclus dans l’application.



Interface visuelle de DOCKER avec les deux conteneurs composant le « stack » flora.



Liste des containers affichée après validation de la commande « docker ps »

Fichier « docker\_compose.yml »

version: "3.8"

services:

  mailcatcher:

    image: schickling/mailcatcher

    ports:

      - "1025:1025"

      - "1080:1080"

    container\_name: Mailcatcher

  dev:

    image: chialab/php:7.4-apache

    volumes:

      - ./:/var/www/

    ports:

      - "8080:80"

    container\_name: Flora

    working\_dir: /var/www

networks:

  default:

    external:

      name: Flora-Network

### Accessibilité

L’accessibilité devait être favorisée le plus possible notamment en respectant les ***Règles pour l'accessibilité des contenus web*** (***Web Content Accessibility Guidelines*** ou **WCAG)** élaborée par le World Wide Web Consortium (W3C). Ainsi, je devais être attentif au contraste des couleurs, renseigner précisément les attributs *title* des balises de liens ou objet ou *alt* pour les balises d’image, indiquer visuellement les liens et leurs destinations s’ils envoyaient vers une autre application, …

### Base de données et sa gestion

La base de données était déjà créée et je n’ai pas eu besoin de créer de nouvelle table ou d’en modifier. Je devais utiliser les tables existantes qui offraient déjà les éléments qui m’étaient nécessaires.

Cette base de données a été créée avec SQLite[[6]](#footnote-6) qui est une bibliothèque et non un [système de gestion de bases de données](https://fr.wikipedia.org/wiki/Syst%C3%A8me_de_gestion_de_base_de_donn%C3%A9es) relationnelles (SGBDR) comme MySQL.

La particularité de cette base de données est qu’elle n’est pas stockée sur un serveur mais intégrée directement dans l’application via un fichier indépendant.

Pour la gestion de cette base de données, l’application utilise PHINX qui est une solution de migration et d’évolution de base de données[[7]](#footnote-7).

### Exigences

L’application devait être *responsive* (conçu et développé de façon à pouvoir s'adapter à toutes les résolutions d'écran) et pouvoir être consulté sur ordinateur, sur smartphone ou sur tablette).

Le projet étant sécurisé par un contrôleur (design pattern Modèle-Vue-Controlleur ou MVC) dédié, je devais obligatoirement utiliser les fonctionnalités de sécurités prévues à cet effet afin d’empêcher toute intrusion ou utilisation malveillante de l’application.

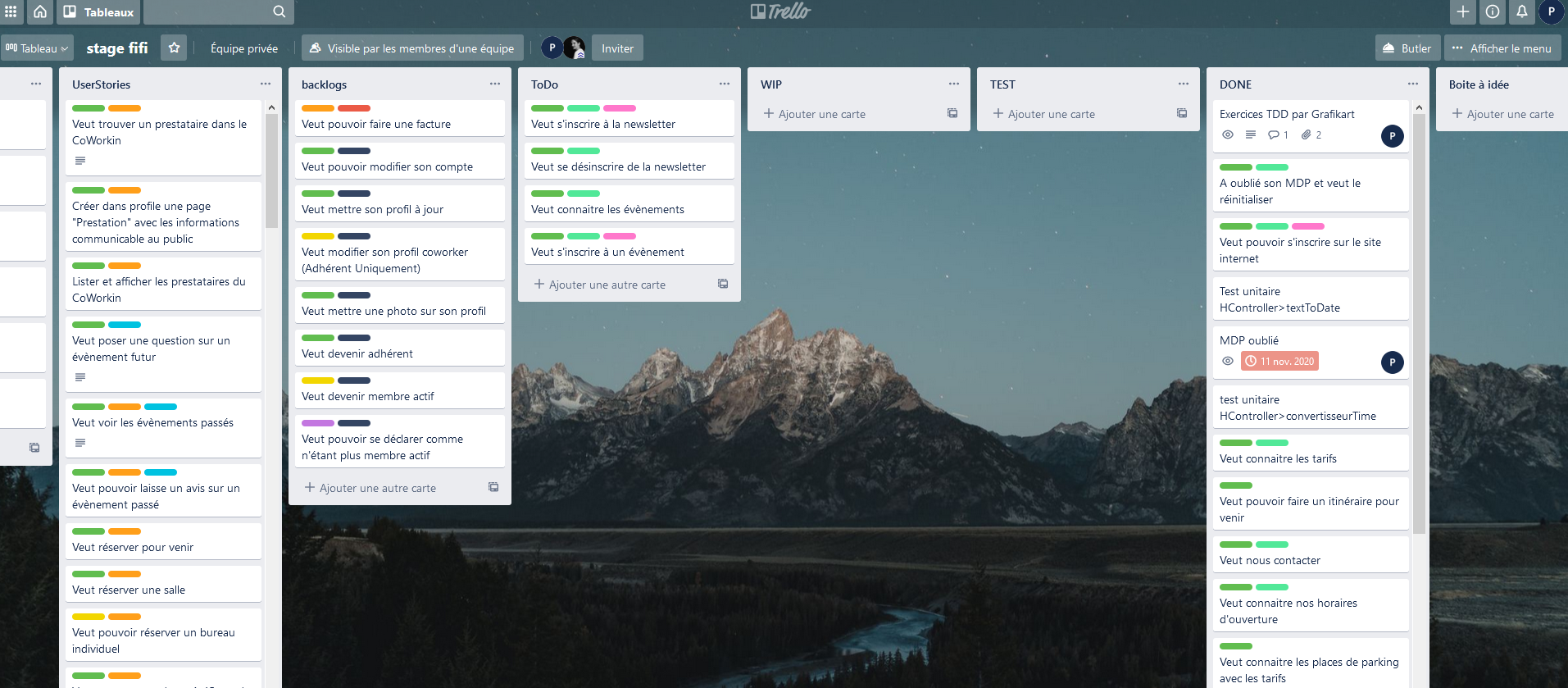
Enfin, l’application étant développée en Programmation Orienté Objet (POO ou *Object-oriented Programming*), tout devait être un objet pouvant être utilisé à l’avenir si besoin. La refactorisation du code, afin de l’alléger le plus possible, a également été utilisée.

## Méthode de gestion de projet AGILE

Durant la durée de mon stage, j’ai utilisé la méthode AGILE afin d’organiser mon travail avec mon maitre de stage et permettant de prioriser les taches, d’assurer un suivi, d’éviter de travailler sur la même fonctionnalité…

Après avoir effectuée, en dehors du stage, une formation d’initiation à la gestion de projet AGILE[[8]](#footnote-8), j’ai utilisé le TRELLO qui est « *un outil de gestion de projet en ligne et inspiré par la méthode Kanban. Il repose sur une organisation des projets en planches listant des cartes, chacune représentant des tâches* »[[9]](#footnote-9).

A la suite d’une réunion, entre des membres du coworking, mon maitre de stage et moi-même, durant laquelle des *users story* ont été déterminées, j’ai pu prendre en charge différentes tâches à réaliser (présentes dans les *backlogs)* au cours de *sprints* d’une semaine.



Capture d'écran du tableau de projet TRELLO

# Spécifications techniques du projet

blablabla

# Réalisations

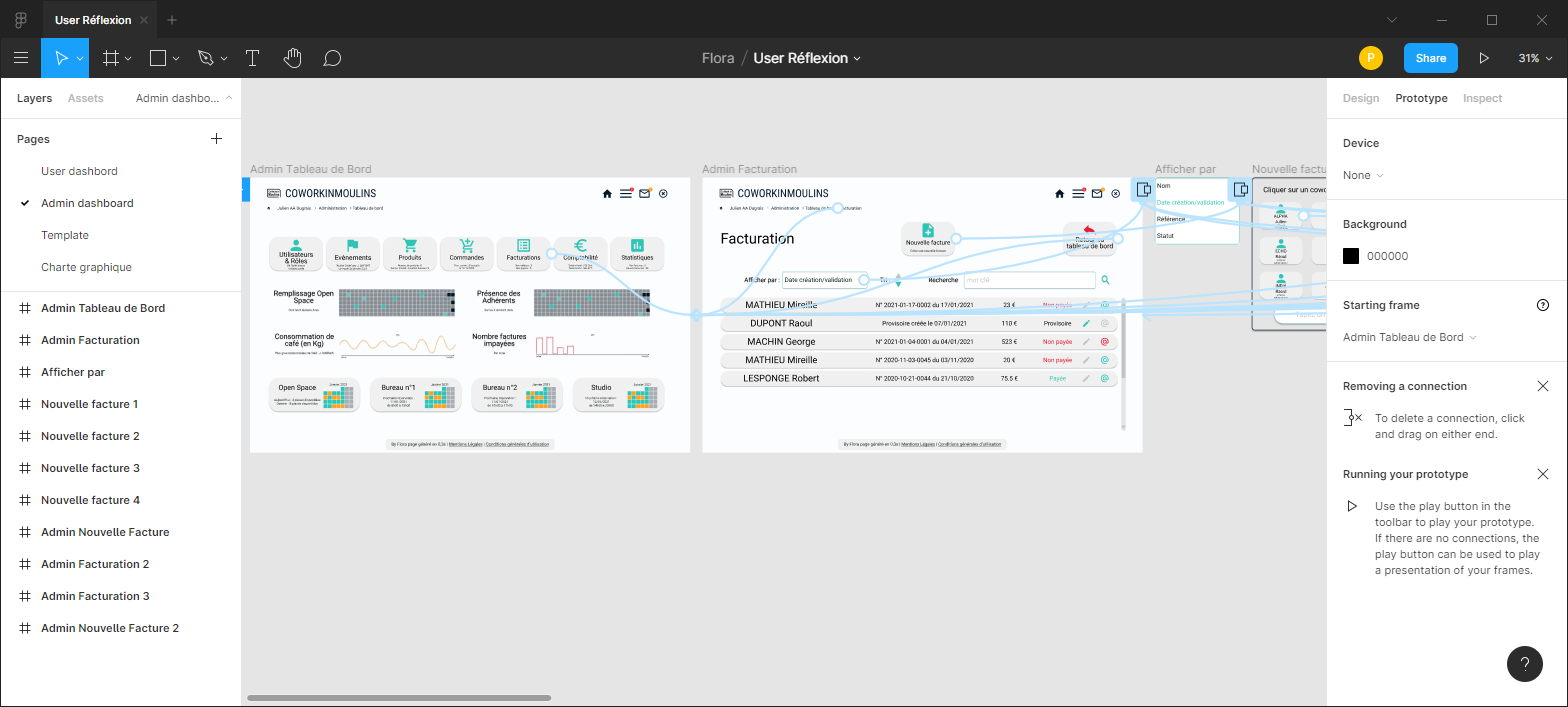
## Maquettage de la nouvelle interface administration

L’application dispose d’une interface accessible unique pour les utilisateurs disposant des droits d’administration. Cette interface a été mise en place à la création du site et est fonctionnelle mais n’a pas fait l’objet d’amélioration visuelle et des retours d’utilisateurs ont conduits à envisager une refonte graphique de cette interface.

C’est dans ce cadre qu’il m’a été demandé de proposer une nouvelle interface et de la maquetter.

Pour cela, j’ai utilisé le logicielle FIGMA[[10]](#footnote-10) qui est « *est un éditeur de graphiques vectoriels et un outil de prototypage* »[[11]](#footnote-11). Cet outil permet un travail collaboratif et ainsi de pouvoir travailler à plusieurs sur un même projet simultanément et sur des postes de travail différents.

Après une phase de « prise en main », j’ai pu établir une maquette du tableau de bord de l’interface administrateur et du visuel de la partie facturation. Figma permettant de simuler des liens, la maquette peut être interactive et permet de tester l’enchainement de différentes vues du projet.



Interface de FIGMA avec visualisation des liens

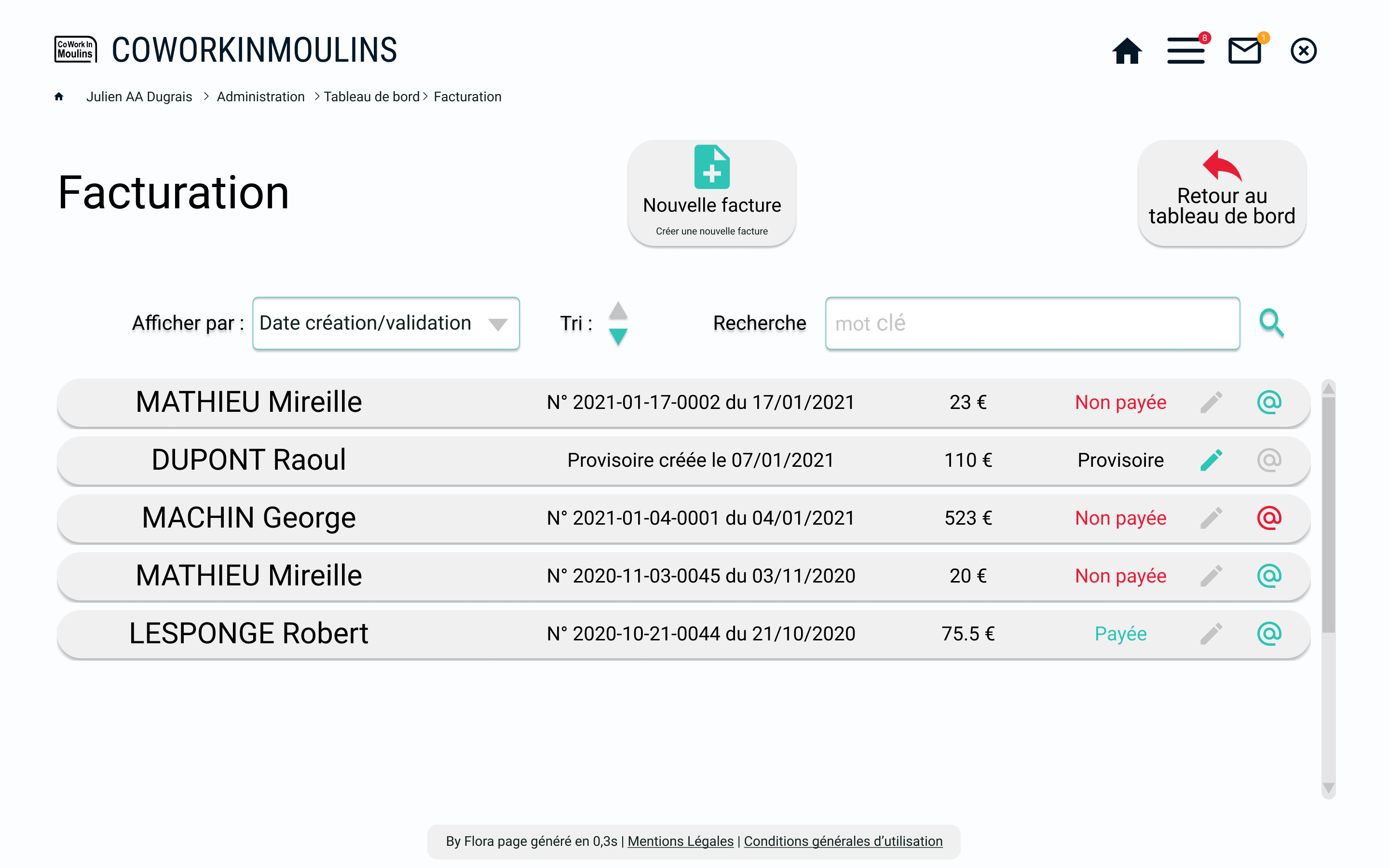
Cette maquette a pu être testée à plusieurs reprises par un administrateur du coworking afin de recueillir ses attentes et avis.

J’ai été très attentif aux actions et réactions de la personne ayant effectué les tests afin de voir si l’interface était intuitive et apportait une bonne expérience utilisateur.

Un total de 11 vignettes ont été créée afin de pourvoir matérialiser le cheminement à suivre ainsi que les différentes pages possibles et leurs évolutions.



Tableau de bord



Page « Facturation »



Page de création de facture

## Amélioration de la page d’accueil du site

Blablabla

## Nouvelle interface utilisateur de la tablette tactile

Blablabla

## Création d’une messagerie interne

Blablabla

## Mot de passe oublié

Blablabla

# Présentation d’un jeu d’essai

## Les tests unitaires et PHPUnit

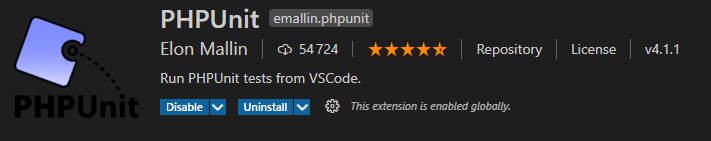
### Le test unitaire

Le test unitaire[[12]](#footnote-12) permet de vérifier que le code créé effectue bien ce pour quoi il a été fait et qu’il retourne la ou les bonne(s) valeur(s) attendu(s). La réalisation de tests unitaires est importante pour vérifier le bon fonctionnement d’un programme et permet souvent de découvrir une erreur ou de pouvoir refactoriser une partie ou encore d’être utilisé comme méthode de développement (TDD ou Test Driven Development[[13]](#footnote-13)).

J’ai effectué le développement d’une fonctionnalité avec la méthode TDD car, dans le concept de cette méthode, le code ainsi produit et déjà fonctionnel et testé, de plus, on peut vérifier que chaque nouvelle fonctionnalité est compatible avec les précédentes et ainsi éviter d’avoir à rechercher des conflits, ceux-ci apparaissent directement, par exemple, en faisant échouer un test qui était pourtant validé la nouvelle fonctionnalité.

### PHPUnit[[14]](#footnote-14)

Il s’agit d’un *framework* de tests unitaires pour le PHP. Il dispose d’une documentation et peut s’installer sur un IDE, tel que Visual Studio Code via un plugin :



PHPUnit permet d’organiser et d’effectuer des tests unitaires précis et rapides en comparant un résultat attendu avec le résultat obtenu et en ayant des informations, en cas d’échec du test, indiquant la ou les raisons de cet échec.

## La fonctionnalité et les tests effectués

### La fonctionnalité développée

J’ai dû développer une « contrainte » de formulaire, que j’ai appelé « Unique », dont le but était de vérifier dans la base de données si l’information indiquée par l’utilisateur existait déjà.

Par exemple, vérifier si le mail indiqué est déjà utilisé par un utilisateur et donc déjà présent dans la base de données.

Méthode relative à l’envoi de message depuis la page d’accueil et utilisant la contraint « unique »

public function mail(): Response

    {

        $paramUnique = TableauController::tableObjectToString("email", $this->users->all());

        $form = new FormController();

        $form->field("name", ["require"]);

        $form->field("email", ["require", "mail", "unique" => $paramUnique]);

        $form->field("message", ["require"]);

        $errors = $form->hasErrors();

        if ($errors["post"] != ["no-data"]) {

            $datas = $form->getDatas();

            $datas['id\_roles'] = 4; // 4 = Administrateur

            if (!$errors) {

                $this->messages->create($datas);

                $errors["error"] = false;

            } else {

                $errors["error"] = true;

                if (isset($errors['email'])) {

                    $errors['email'] = ["Merci d'envoyer vos messages via votre messagerie interne."];

                }

            }

        }

        return $this->jsonResponse($errors);

    }

Par exemple, vérifier si le mail indiqué est déjà utilisé par un utilisateur et donc déjà présent dans la base de données.

Cette méthode utilise les contraintes cumulatives « *require* » (le champ doit être renseigné), «*mail* » (l’information saisie doit être une adresse mail) et « *unique* » (l’information ne doit pas déjà exister en base de données). Si les contraintes sont respectées, le traitement peut avoir lieu, à défaut, un message d’erreur apparait.

Voici la méthode de la contrainte :

    private function errorUnique(string $field, array $value): void

    {

        if (!in\_array($this->postDatas[$field], $value)) {

            $this->addToDatas($field);

        } else {

            $this->newError("$field",

"La valeur indiquée dans \"{$field}\" n'est pas valide.");

        }

    }

La méthode prend deux paramètres : le champ de formulaire concerné et un tableau de valeur de référence prédéterminé (par exemple, la liste des adresses électroniques des utilisateurs présentes dans la base de données).

Cela permet de vérifier que la donnée indiquée dans le champ (*$field*) et passée en POST n’est pas présente dans un tableau de valeur (*$value*).

En cas de succès, l’information saisie par l’utilisateur est validée sur ce point.

### Les tests effectués

Sept tests ont été créés successivement selon la méthode TDD. Le 1er test est écrit mais échoue, ensuite la fonctionnalité testée est écrite et doit faire passer le 1er test « au vert ».

Le premier test

    // Tests sur "errorUnique"

    public function testErrorUniqueMailNouveauDatas(): void

    {

        $bddDatas = ["test@test.test"];

        $\_POST = ["mail" => "julien@test.test"];

        $form = new FormController();

        $form->field("mail", ["unique" => $bddDatas]);

        $datas = $form->getDatas();

        $this->assertEquals(["mail" => "julien@test.test"], $datas);

    }

Ici, je créé, un test permettant de savoir si le mail indiqué « julien@test.test » est présent dans le tableau représentant tous les mails présents dans la base de données ($bddDatas).

Si le mail n’est pas présent dans la base de données alors celui-ci est accepté est sera présent dans *$datas*.

La fonction « assertEquals » est incluse dans PHPUnit et permet de comparer deux données et ainsi voir si elles sont parfaitement identiques.

Dans l’exemple donné, le test à fonctionné mais si toutefois le mail indiqué existait déjà, le test aurait échoué et le résultat suivant serait apparu :

PHPUnit 8.5.8 by Sebastian Bergmann and contributors.

F                                                                   1 / 1 (100%)

Time: 438 ms, Memory: 4.00 MB

There was 1 failure:

1) Tests\Core\Controller\FormControllerTest::testErrorUniqueMailNouveauDatas

Failed asserting that two arrays are equal.

--- Expected

+++ Actual

@@ @@

 Array (

-    'mail' => 'julien@test.test'

 )

C:\wamp64\www\STAGE\flora\tests\Core\Controller\FormControllerTest.php:616

FAILURES!

Tests: 1, Assertions: 1, Failures: 1.

Une fois le 1er test validé, j’ai créé des tests suivants les uns après les autres au fur et à mesure qu’ils étaient valides.

En effet, la méthode TDD précise qu’un test ne doit être écrit que si tous ceux d’avant sont validés.

D’autres tests permettant de vérifier différents cas

    public function testErrorUniqueMailNouveauErrors(): void

    {

        $bddDatas = ["test@test.test"];

        $\_POST = ["mail" => "julien@test.test"];

        $form = new FormController();

        $form->field("mail", ["unique" => $bddDatas]);

        $errors = $form->hasErrors();

        $this->assertEquals([], $errors);

    }

    public function testErrorUniqueMailMultiNouveauDatas(): void

    {

        $bddDatas = ["user@test.fr", "julien@test.fr", "test@test.test"];

        $\_POST = ["mail" => "julien@test.test"];

        $form = new FormController();

        $form->field("mail", ["unique" => $bddDatas]);

        $datas = $form->getDatas();

        $this->assertEquals(["mail" => "julien@test.test"], $datas);

    }

    public function testErrorUniqueNoMailBDDDatas(): void

    {

        $bddDatas = [];

        $\_POST = ["mail" => "julien@test.test"];

        $form = new FormController();

        $form->field("mail", ["unique" => $bddDatas]);

        $datas = $form->getDatas();

        $this->assertEquals(["mail" => "julien@test.test"], $datas);

    }

    public function testErrorUniqueMailExistantBDDDatas(): void

    {

        $bddDatas = ["test@test.test"];

        $\_POST = ["mail" => "test@test.test"];

        $form = new FormController();

        $form->field("mail", ["unique" => $bddDatas]);

        $datas = $form->getDatas();

        $this->assertEquals([], $datas);

    }

    public function testErrorUniqueMailExistantBDDErrors(): void

    {

        $bddDatas = ["test@test.test"];

        $\_POST = ["mail" => "test@test.test"];

        $form = new FormController();

        $form->field("mail", ["unique" => $bddDatas]);

        $errors = $form->hasErrors();

        $this->assertEquals(["mail" => ['La valeur indiquée dans "mail" n\'est pas valide.']], $errors);

    }

# Description d'une veille sur la sécurité & résolution d’une faille

La sécurité des SI est une priorité que ce soit au niveau des réseaux que dans une application web. Afin, de m’informer de l’évolution de cette matière et pour maintenir mes connaissances à jour j’utilise principalement le réseau LinkedIn.

En effet, je suis et lis très régulièrement les publications de l’ANSSI[[15]](#footnote-15), de la CNIL[[16]](#footnote-16) et The Cyber Security Hub™[[17]](#footnote-17), lesquelles m’apportent constamment des nouvelles informations sur la thématique de la sécurité des SI.

De plus, j’ai effectué, durant mon stage, une formation[[18]](#footnote-18) concernant l’OWASP® (The Open Web Application Security Project®[[19]](#footnote-19)) et je consulte désormais régulièrement ce site afin d’actualiser mes connaissances en matière de sécurité.

Bien que je n’aie pas repéré d’information pouvant m’être utile durant le stage, j’ai découvert dans le code du projet une faille potentielle de sécurité.

En effet, durant mon travail sur la fonctionnalité (*back-end*) de la tablette tactile, je me suis aperçu d’une faille concernant l’accès aux espaces utilisateurs.

La tablette tactile présente à l’entrée des locaux sert aux utilisateurs à déclarer leur présence (arrivé et départ) afin que le coworking puisse contrôler l’utilisation des forfaits ou établir une facturation. L’utilisateur sélectionne son nom sur la tablette et tape, sur le clavier digital, son code personnel à quatre chiffres qui lui a été attribué à la création de son compte ou qu’il a modifié.

Cependant, cette fonctionnalité est aussi accessible via tout terminal en indiquant l’url de la fonctionnalité. Si cela n’est pas un problème en soit, le fait de déclarer sa présence avait pour conséquence d’ouvrir une session utilisateur sur ledit terminal et de connecter l’utilisateur à son espace privé ! Cela ouvrait donc la voie à une attaque par force brute[[20]](#footnote-20) avec seulement 10000 possibilités de code.

M’étonnant de cette fonctionnalité, j’ai immédiate interrogé mon maître de stage sur la raison de l’ouverture d’une session et lui indiquant que cela était dangereux pour la sécurité des comptes.

Mon maître de stage à immédiatement pris en charge la modification de la fonctionnalité en supprimant la création d’une session utilisateur et le code corrigé à été mis en production dans l’heure suivant ma découverte.

# Recherches sur un site anglophone, extrait et traduction

## Recherche effectuée

Lorsque j’en ai besoin, je consulte les documentations de différents langages ou *framework* utilisés comme, par exemple, la documentation de PHP[[21]](#footnote-21) que ce soit en anglais ou en français.

Également, je consulte diverses ressources disponibles comme les sites Alscreations[[22]](#footnote-22), MDN (Mozilla)[[23]](#footnote-23) stackowerflow[[24]](#footnote-24), w3school[[25]](#footnote-25)… offrant une information inépuisable et source d’inspiration.

Durant mon stage, lors de la réalisation d’une mise à jour incluant de nouvelles fonctionnalités front-end sur la page principal du site, j’ai dû intégrer une carte avec localisation du coworking.

Cette carte devait afficher un niveau de zoom initiale assez proche du lien indiqué afin de visualiser le quartier où se situe le coworking, toutefois, l’utilisateur devait pouvoir zoomer ou dézoomer de la position indiquée afin de visualiser une zone plus large ou plus étroite.

J’ai utilisé OpenStreetMap[[26]](#footnote-26) lequel permet d’inclure un code HTML directement dans l’application via une balise <iframe> en précisant différent paramètre comme notamment le niveau de zoom initial souhaité au chargement de la page.

Le zoom se positionnait comme demandé avec différents navigateurs (les tests sur Firefox et Edge étaient satisfaisants) sauf avec Chrome qui affichait une carte entière du monde ce qui correspondait à la vue la plus dézoomée possible.

J’ai donc cherché et trouvé une solution sur le forum anglophne d’OpenStreetMap dans un topic intitulé « ***Map sometimes showing as whole world in Chrome* »**[[27]](#footnote-27) **ce qui semblait correspondre parfaitement à mon problème.**

## Extrait du forum anglophone

### La question[[28]](#footnote-28)

“I've been using OSM on my website for a few years now, and the iframe embedded map, which I just have on the home page, has always shown ok.

Since December 2019 I've noticed that the map sometimes shows the whole world when loading the page in Chrome (I'm using Version 79.0.3945.117 Official Build 64-bit).

If I Ctrl / F5 the correct zoomed map shows, but if I Carriage Return on the URL it show the world or correct map randomly (about 50/50). (…)

I've tried this on 4 PCs in two locations and all have the same issue in Chrome.  Oddly IE11, Edge and Firefox are fine.

Any one else had similar issues or know what might be causing this?”

### La réponse[[29]](#footnote-29)

“If useful, instead of using **iframe**, using **object** seems to work best:

<object data="https://www.openstreetmap.org/export/embed.html?bbox=<<-- your map code -->>" border="1" width="100%"></object>

I tried **object** with IE11, Edge, Opera, Chrome, Firefox and an old Win version of Safari and all ok.

or you could try **embed**:

<embed src="https://www.openstreetmap.org/export/embed.html?bbox=<<-- your map code -->>" width="100%" onerror="alert('URL invalid !!');"></embed>

I tried **embed** with IE11, Edge, Opera, Chrome, Firefox and all ok.  The old version of Safari did some very odd things!  I haven't got a Mac to test so can't confirm if ok with latest Safari (ok on an iPhone though).  Also you cannot add a border with **embed**.

Both work for my responsive sites for all view-ports (Desktop, Tablet Portrait, Mobile Landscape and Mobile Portrait).”

## Traduction

### De la question

“J’utilise OSM sur mon site internet depuis quelques années maintenant, et la carte intégrée *iframe*, que je viens juste de mettre sur la page d’accueil, a toujours bien fonctionné.

Depuis décembre 2019 j’ai noté que la carte parfois montre le monde entier quand la page est chargée sur Chrome (j’utilise la version 79.0.3945.117 Official Build 64-bit).

Si je fais Ctrl + F5 le zoom de la carte est correct, mais si je retourne sur l’URL cela montre le monde ou la carte correcte aléatoirement (environ 50/50).

J’ai essayé sur 4 PC dans 2 endroits et tous ont le même problème avec Chrome. Bizarrement, IE11, Edge et Firefox fonctionnent bien.

Quelqu’un aurait un problème similaire ou sait ce qui pourrait causer cela ?

### De la réponse

“Utiliser **object** à la place de **iframe** semble mieux fonctionner :

<object data="https://www.openstreetmap.org/export/embed.html?bbox=<<-- your map code -->>" border="1" width="100%"></object>

J’ai essayé **object** avec IE11, Edge, Opera, Chrome, Firefox et une vieille version de Safari et tout est OK.

Ou tu peux essayer **embed :**

<embed src="https://www.openstreetmap.org/export/embed.html?bbox=<<-- your map code -->>" width="100%" onerror="alert('URL invalid !!');"></embed>

J’ai essayé **embed** avec IE11, Edge, Opera, Chrome, Firefox et tout est ok.  La vieille version de Safari a fait des choses vraiment étranges ! Je n’ai pas de Mac pour tester alors je ne peux pas confirmer que cela fonctionne avec la dernière version de Safari (ça fonctionne sur un iPhone cependant).  Également, tu ne peux pas ajouter une bordure avec **embed**.

Les deux fonctionnent sur mes sites « *responsive »* sur tous les ports d’affichage (ordinateur, tablette format portrait, mobile format paysage et portrait).”

## Solution adoptée

Effectivement, ce topic m’a permis de résoudre mon problème et j’ai finalement utilisé la balise <objetc> à la place de <iframe> en ajoutant une bordure de 1 pour que cela fonctionne sur Firefox (Firefox demande une bordure, à défaut, le même problème de dézoom sur la carte du mode se reproduit).

# Conclusion

Il est des circonstances qui méritent d’être précisées surtout quand celles-ci sont inédites.

En effet, mon stage devait être effectué dans les locaux du coworking de Moulins (03) permettant ainsi d’avoir un espace de travail propice et de pouvoir rencontrer les utilisateurs de l’application sur laquelle j’allais travailler.

Mon stage a bien débuté le vendredi 30 octobre 2020, jour du début du 2nd confinement dû à la crise sanitaire de la Covid19.

Du fait de ce nouveau confinement, le coworking ne pouvait donc plus accueillir de public.

J’ai donc commencé le stage en télétravail en échangeant fréquemment et très régulièrement avec mon maître de stage.

Puis, après avoir mis en œuvre toutes les recommandations sanitaires, le coworking a pu de nouveau ouvrir ses portes et me permettant de découvrir le lieu dans lequel j’ai pu effectuer le reste de mon stage.



L'espace de coworking de Moulins

Bien qu’il me fût possible de travailler à distance, j’ai souhaité être le plus souvent possible présent avec mon maître de stage ce qui a permis d’avoir des échanges fréquents et instructifs.

1. <https://www.linkedin.com/company/apprendre-co/about/> [↑](#footnote-ref-1)
2. <https://coworkinmoulins.fr> [↑](#footnote-ref-2)
3. <https://github.com/mastergoster/flora> [↑](#footnote-ref-3)
4. <https://www.docker.com/> [↑](#footnote-ref-4)
5. Source : <https://fr.wikipedia.org/wiki/Docker_(logiciel)> [↑](#footnote-ref-5)
6. <https://www.sqlite.org/index.html> [↑](#footnote-ref-6)
7. <https://phinx.org/> [↑](#footnote-ref-7)
8. <https://openclassrooms.com/fr/courses/4507926-initiez-vous-a-la-gestion-de-projet-agile> [↑](#footnote-ref-8)
9. Sources : <https://fr.wikipedia.org/wiki/Trello> [↑](#footnote-ref-9)
10. <https://www.figma.com> [↑](#footnote-ref-10)
11. Source : <https://fr.wikipedia.org/wiki/Figma_(logiciel)> [↑](#footnote-ref-11)
12. <https://fr.wikipedia.org/wiki/Test_unitaire> [↑](#footnote-ref-12)
13. <https://fr.wikipedia.org/wiki/Test_driven_development> [↑](#footnote-ref-13)
14. <https://phpunit.de/> [↑](#footnote-ref-14)
15. <https://www.linkedin.com/company/anssi-fr/> [↑](#footnote-ref-15)
16. <https://www.linkedin.com/company/cnil---commission-nationale-de-l'informatique-et-des-libert-s/> [↑](#footnote-ref-16)
17. <https://www.linkedin.com/company/the-cyber-security-hub/> [↑](#footnote-ref-17)
18. <https://openclassrooms.com/fr/courses/6179306-securisez-vos-applications-web-avec-lowasp> [↑](#footnote-ref-18)
19. <https://owasp.org/> [↑](#footnote-ref-19)
20. <https://fr.wikipedia.org/wiki/Attaque_par_force_brute> [↑](#footnote-ref-20)
21. <https://www.php.net/docs.php> [↑](#footnote-ref-21)
22. <https://www.alsacreations.com/> [↑](#footnote-ref-22)
23. <https://developer.mozilla.org/fr/> [↑](#footnote-ref-23)
24. <https://stackoverflow.com/> [↑](#footnote-ref-24)
25. <https://www.w3schools.com/> [↑](#footnote-ref-25)
26. <https://www.openstreetmap.org> [↑](#footnote-ref-26)
27. <https://forum.openstreetmap.org/viewtopic.php?id=68432> [↑](#footnote-ref-27)
28. Post de « VineSurgery » du 12/01/2020 à 21 :10 :49 [↑](#footnote-ref-28)
29. Post de « pnesouth » du 14/02/2020 à 17 :06 :03 [↑](#footnote-ref-29)