Dossier de synthèse – développeur logiciel

Valérie Baron

Simplon.co Cannes Frayère

10/10/2018

Table des matières

Introduction 2

Compétences du référentiel 3

I Présentation du projet « Guider » 4

1.1 Origine du projet et structure d'accueil 4

1.2 Maquettage du site 5

1.3 Techniques mises en œuvre et planification des activités 11

1.4 Déroulement du stage et rapports avec le Sictiam 13

1.5 Résultats obtenus et avancement du projet 14

1.6 Problèmes rencontrés et évaluation personnelle 17

II Conception/programmation 19

1.1 Mise en place de la base de données 19

1.2 Connexion à la base de données 21

1.3 Développement du site 22

1.3.1 La page d'accueil 22

1.3.2 Un arrière-pays 24

1.3.3 Fiche guide et partie réservation 28

1.3.4 Partie administrateur 30

1.3.5 Partie User 32

1.3.5 Les fonctionnalités à venir 39

1.4 Responsivité 39

1.4 Déploiement du site 41

INTRODUCTION

ABSTRACT

The project I'm going to present you is born from a third prize won at the Hackathon for Tourism contest that took place in April 2018 at Sophia-Antipolis.

The main objectives were to create and enhance tourism in Alpes Maritimes's backyards.

I imagined with one colleague a website that could create links between local guides, wishing to tell the story of their own village and reveal the hidden heritage, and people who want to run from the beaches and the crowded coast.

The objective is also to attract new tourism toward local farms, local producers (as beekeepers).

Sictiam, the company who gave us this third prize, proposed us an internship in order to develop our idea.

First of all, this website makes a small presentation of the seven countries that divide our department, emphasizing geographical characteristics and local sights. Each country proposes a selection of local guides available for discovery, promenade or tasting experience. Each guide is bookable online. Each guide can manage his appointments.

The administrators of the website (us) keep the control on it. They recruit themselves the guides and create the personal file thanks to a panel control (CRUD) which is available after login.

The user can create a personal account, log in and eventually modify his personal data like email address, phone. He can also cancel the latest reservation.

COMPETENCES DU REFERENTIEL

Maquetter une application

Le projet Guider étant né lors d'un Hackathon, l'idée du projet a dû être développée de A à Z avec mon collègue. Les wireframes ont été créés à l'aide de Balsamiq 3.

Concevoir une base de données

Les fonctionnalités du site ont nécessité la mise en place d'une base de données contenant 6 tables.

Mettre en place une base de données

Utilisation de MySQLl et phpMyAdmin pour l'intégration de la base de données.

Développer une interface utilisateur

Interface utilisateur développée à partir du langage PHP 5, utilisation de la programmation orientée objet pour créer la classe User construite à partir du diagramme d'activité.

Développer des composants d'accès aux données

Utilisation de l'extension PDO pour accéder à la base de données depuis PHP.
Elaboration de 2 CRUD destinés à l'administrateur pour gérer les données (CRUD user et CRUD guides)

Développer des pages web en lien avec une base de données

A partir du wireframe et des spécifications fonctionnelles et techniques, j'ai mis en place le front-end du site web en développant des pages dynamiques.

Développer une application simple de mobilité numérique

Construction du site effectuée avec le framework Bootstrap afin de proposer à l'utilisateur une mobilité numérique.

Utiliser l'anglais dans son activité professionnelle

Introduction effectuée en anglais, documentation du code effectuée en anglais.

I. PRESENTATION GENERALE DU PROJET « GUIDER »

1.1 Origine du projet et structure d'accueil

Le projet a été imaginé lors du **Hackathon** « Hack for tourism », organisé par Telecom Valley et la CCI de Nice, dont l'objectif est de créer des idées nouvelles pour le développement du tourisme dans l'arrière-pays du département des Alpes Maritimes et le tourisme frontalier.

Notre arrière-pays est riche en villages typiques, une nature étonnante et variée que peu de gens connaissent. Les personnes qui sont à même de livrer l'histoire locale, le patrimoine et des anecdotes sur leur village sont les gens qui y vivent toute l'année.

Définition du projet

Avec Guider, nous proposons une nouvelle forme de tourisme afin de valoriser le patrimoine : un « guide local », pour chaque village incontournable, qui propose des visites. Chaque visite lui apportera un complément de revenu, il pourra créer des partenariats avec des fermes, des mielleries, des fromageries et d'autres petites exploitations.

➡ Le projet est donc un site de présentation des 7 arrière-pays qui composent le département et un moyen de réserver des guides locaux sélectionnés par village pour des activités.

Le **Sictiam** (Syndicat d'Ingénierie pour les Collectivités et Territoires Innovants des Alpes et de la Méditerranée), **partenaire** du Hackathon et spécialiste en aménagement numérique du territoire, nous a reçu en stage dans ses locaux, mon collègue Jean-Louis Bailet et moi-même, pendant un mois afin de nous aider à démarrer dans la création du projet.

1.2 Maquettage du site

- La page de présentation de Guider donne accès à des informations sur les 7 arrière-pays qui composent le département 06. Chaque arrière-pays y est détaillé et des guides locaux sont proposés à l'utilisateur.
- L'utilisateur peut créer un compte et se connecter afin de réserver un guide ou plusieurs.
- Des fonctionnalités pour les guides sont également prévues
- L'administrateur du site (c'est-à-dire nous, si nous lançons le projet) a la gestion des guides, et également la gestion des utilisateurs.

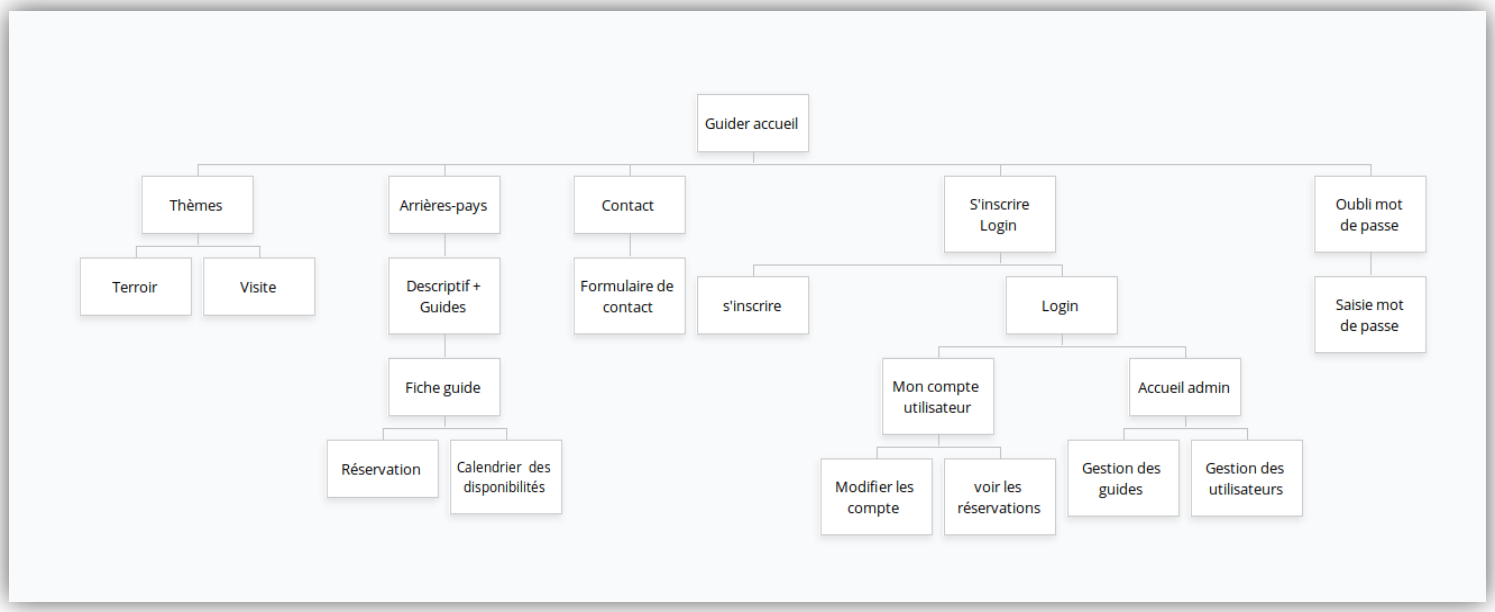
a) Fonctionnalités à développer

Les fonctionnalités pour l'utilisateur
Accès à la page d'accueil
Accès aux pages arrière-pays/thèmes **soit par le slideshow soit par le menu pour plus d'ergonomie**
Accès au formulaire de contact
Création d'un compte utilisateur/connexion au compte utilisateur
Accès aux fiches des guides
Visualisation de la localisation des guides
Réservation d'un guide
Consultation/Modification des données personnelles
Consultation des 3 dernières réservations/Annulation dernière réservation

Les fonctionnalités pour les guides
Accès à la page d'accueil
Accès aux pages arrière-pays/thèmes
Accès au formulaire de contact
Connexion compte guide
Consultation des réservations
Annulation des réservations

Les fonctionnalités pour l'administrateur
Accès à la page d'accueil
Accès aux pages arrière-pays/thèmes
Connexion en tant qu'administrateur
Accès au panneau de gestion des guides
Accès au panneau de gestion des utilisateurs

b) Architecture du site et maquettes



Après avoir déterminé les fonctionnalités et l'architecture du site, j'ai élaboré la charte graphique et le logo :

Guider – Charte graphique et Logo

#4f7a52

#FF5733

#c36a40

#2D5986

Les couleurs sont tirées de l'image principale présente sur la page d'accueil du site. Ce sont des couleurs provençales en rapport avec le thème.

Le logo Guider

- Le logo créé dans la couleur #2D5986 peut être ombré ou bordé.
- La police choisie pour le site est Roboto, sans sérif. Le texte sera de couleur blanche ou noire.

(https://fonts.google.com/specimen/Roboto)

J'ai réalisé avec mon collègue les **wireframes** avec l'outil **Balsamiq** 3 et ai obtenu les maquettes suivantes:

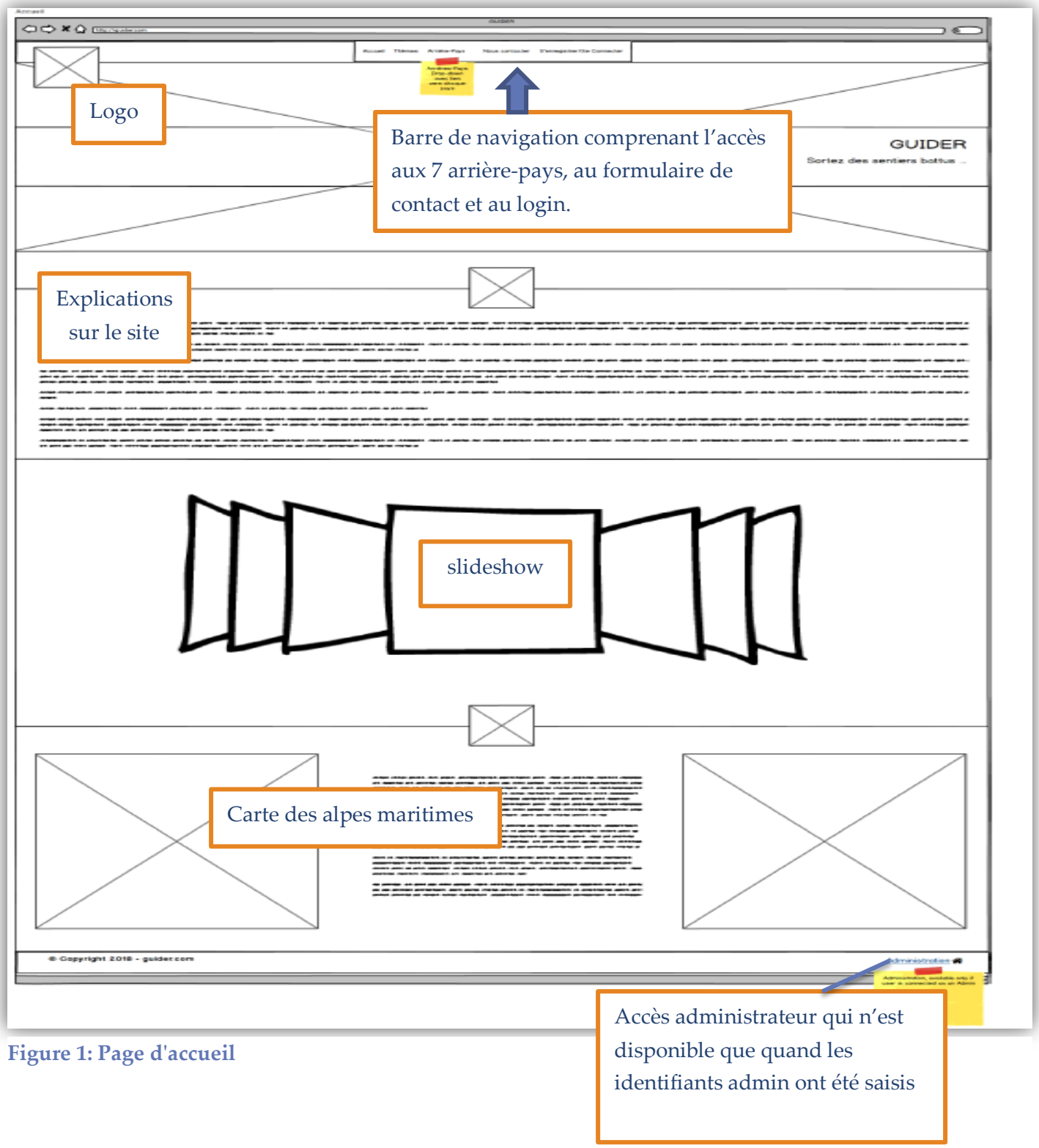


Figure 1: Page d'accueil

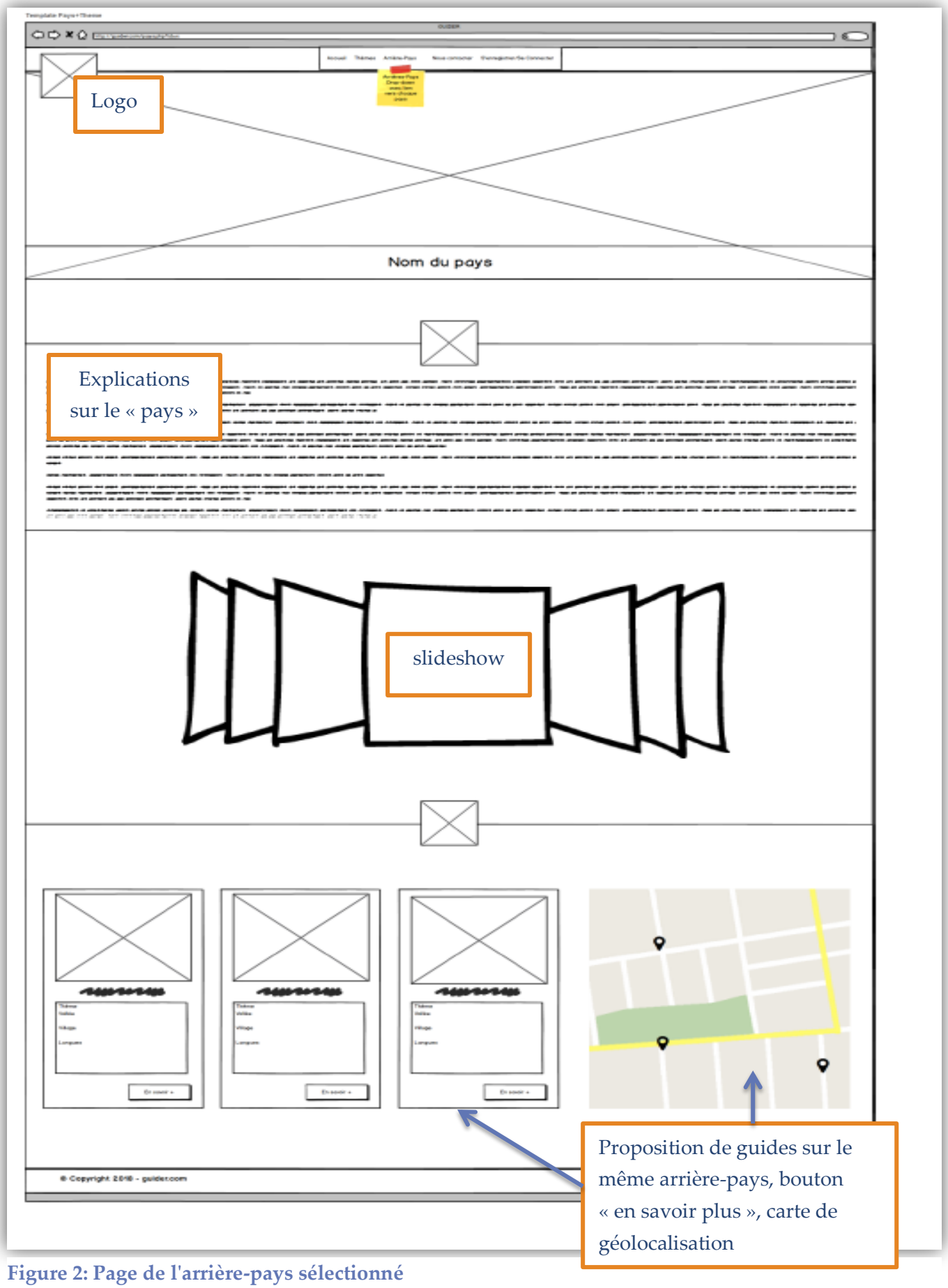


Figure 2: Page de l'arrière-pays sélectionné

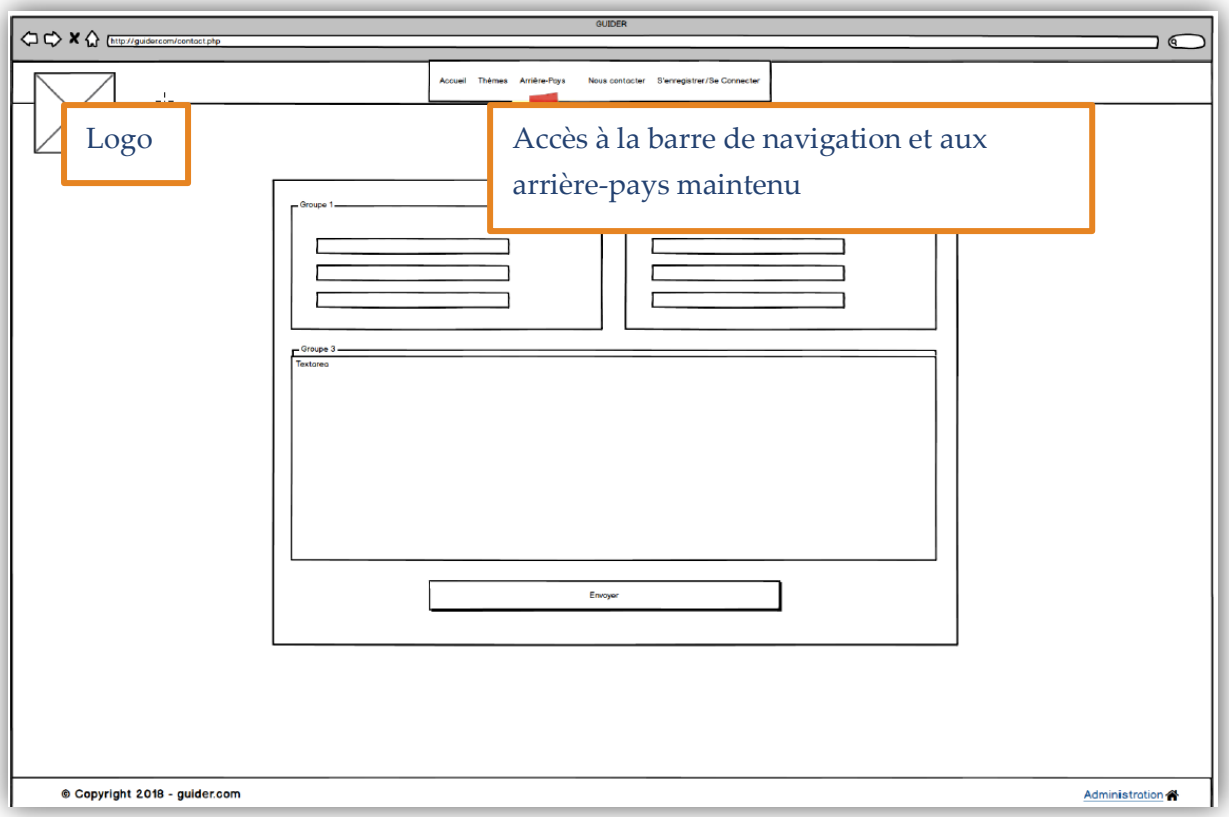


Figure 3: formulaire de contact.

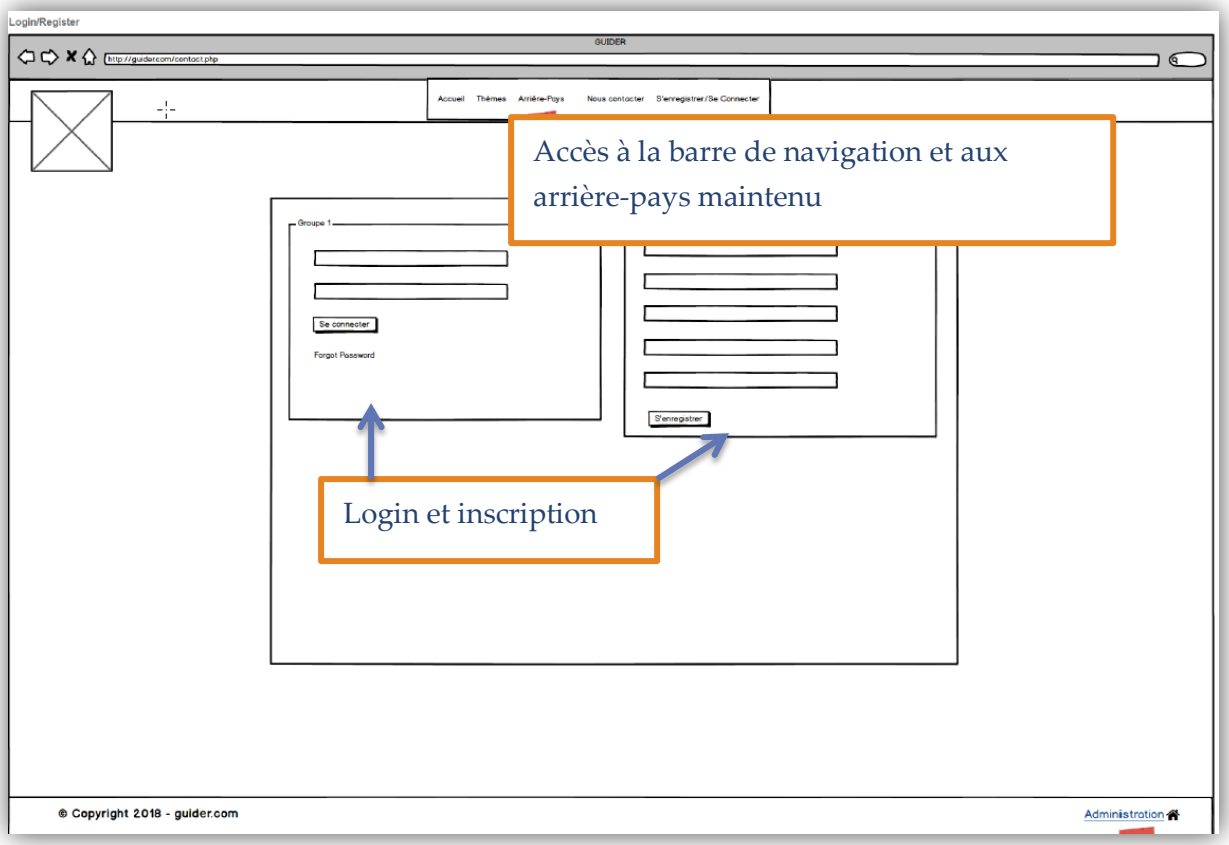


Figure 4: Espace inscription et Login

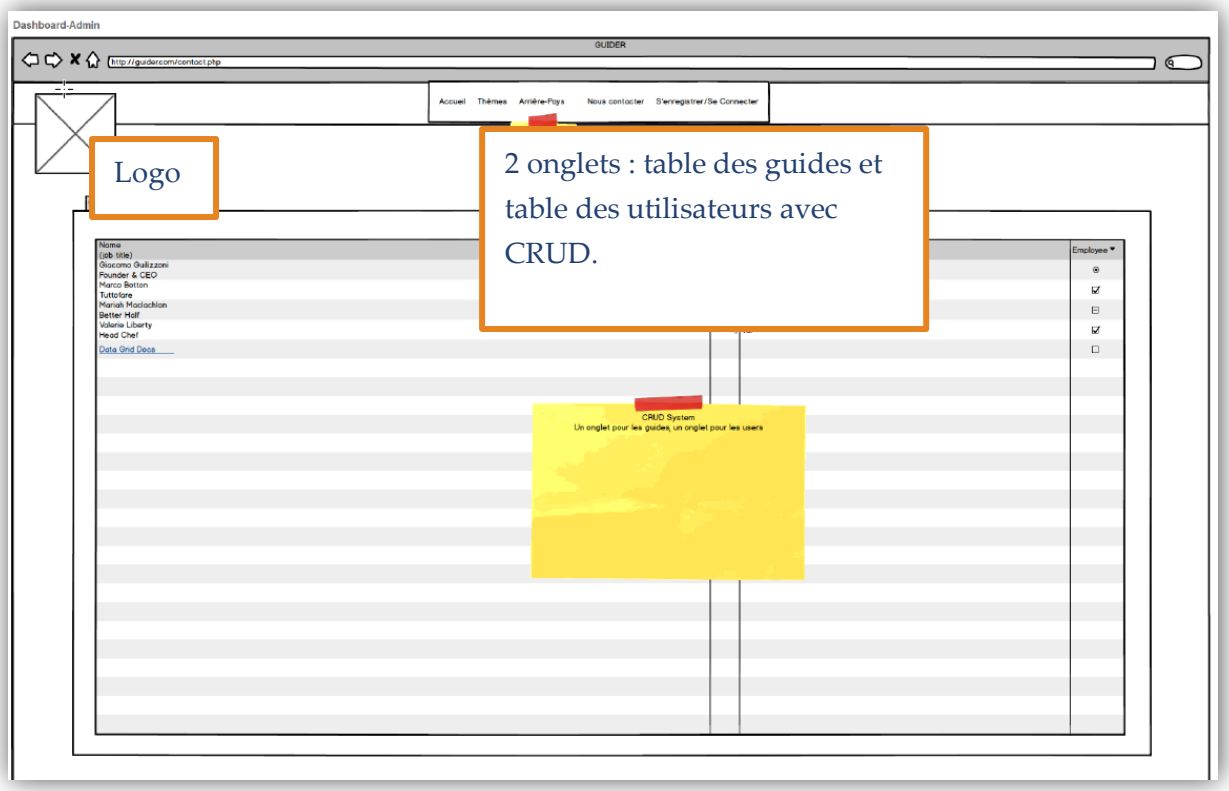
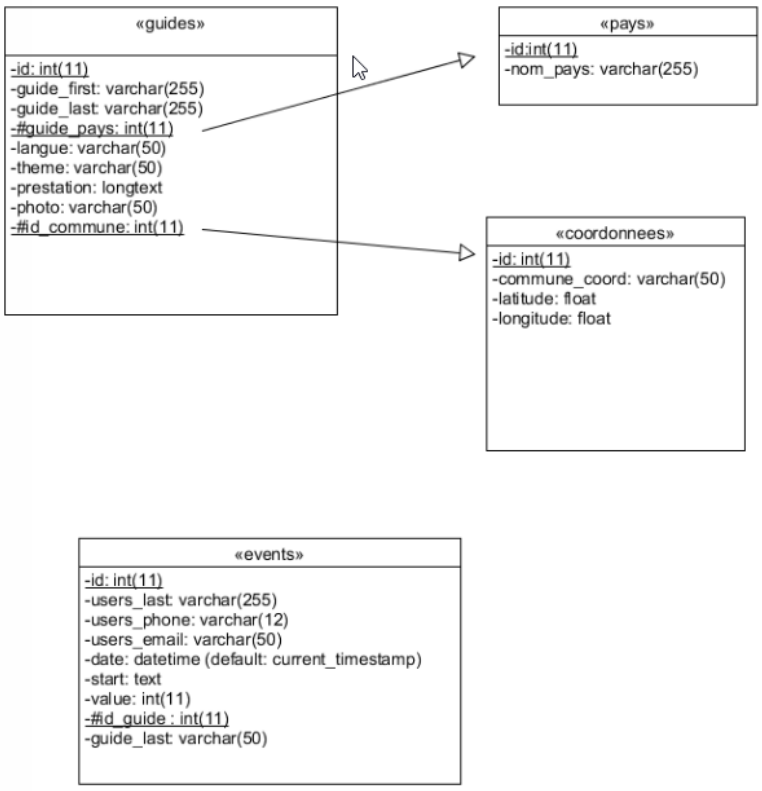


Figure 5: Dashboard administrateur

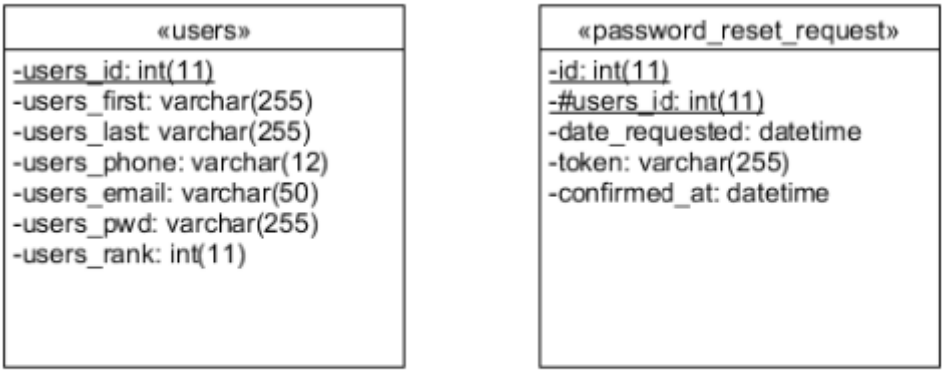
b) Conception de la base de données

Je suis partie des fonctionnalités attendues afin de déterminer les éléments dont nous avions besoin pour la base. J'ai construit le schéma suivant avant la création de la base comprenant 6 tables :



-La table des « **guides** » est reliée aux tables « **pays** » et « **coordonnées** » par une clé étrangère.

-La table « **events** » va contenir toutes les réservations par guide.



La table « users » contient tous les champs d'identification de l'utilisateur, la table password reset request va enregistrer toutes les demandes de nouveau mot de passe suite à un mot de passe oublié.

1.3 Techniques mises en œuvre et planification des activités

Je dispose d'un ordinateur portable, sous Linux Ubuntu avec une connexion internet.

Afin de réaliser ce projet, j'ai eu besoin de :

Papier et stylo pour les idées, liste d'idées et ébauche des schémas ;

- l'éditeur de texte Sublime Text 3
- un serveur Lamp
- MySQL,
- PhpMyAdmin
- Un hébergement web incluant un panel de gestion (o2switch) ;

Les maquettes ont été conçues avec :

- GlooMaps (pour l'architecture)
- Balsamiq 3 (pour les wireframes)
- UMLet (Schéma conception de la base de données)
- Photoshop (pour le Logo).

Le site **Guider** a été développé en **HTML5**, **CSS3** (et son framework **Bootstrap** 4.1), **Javascript** ES5 (et son framework **jQuery**) ainsi qu'en **PHP** 5 et 7.1. J'ai choisi ces langages par rapport à mes connaissances et celles de mon collègue au moment de

la conception. Je devais rendre un embryon de projet à la fin du mois de stage et faire une présentation au **Sictiam**, il était donc plus judicieux de rester sur des langages connus plutôt que de se lancer dans l'apprentissage de nouveaux langages ou Frameworks qui aurait retardé l'avancée du projet.

Mon collègue et moi-même avons travaillé ensemble pour la conception, le choix des photos, le maquettage.

Le front-end a été réalisé en partie par mon collègue Jean-Louis (partie accueil du site, pages des pays). J'ai réalisé la partie front-end côté administrateur et la partie Login.

Mon collègue étant tombé malade, j'ai dû créer intégralement le back-end en Php.

Lors de la 2eme partie de la création du site (continuité du stage en projet pédagogique), j'ai pu modifier la partie User et utiliser **la programmation orientée objet** que nous avions alors abordé en cours.

Organisation :

La **partie 1** (stage) a été planifiée avec un diagramme de Gantt (rétroplanning).

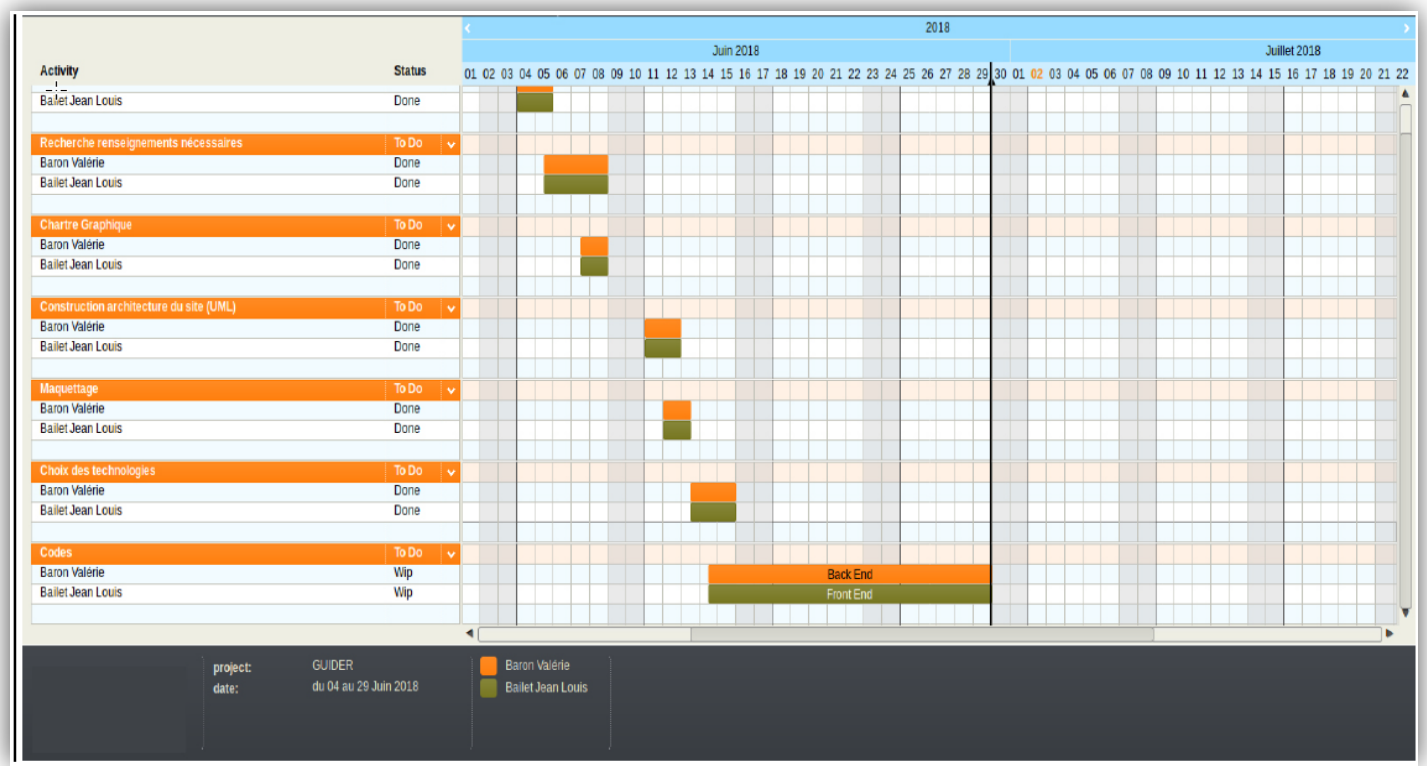


Figure 6: Activité sur un mois de stage

La **partie 2** (projet pédagogique et chef d'oeuvre) a été planifiée avec Trello (méthode Kanban pour le workflow). Sur cette partie, Mathieu Money m'a rejointe et nous avons pu continuer à développer le site et y apporter des améliorations.

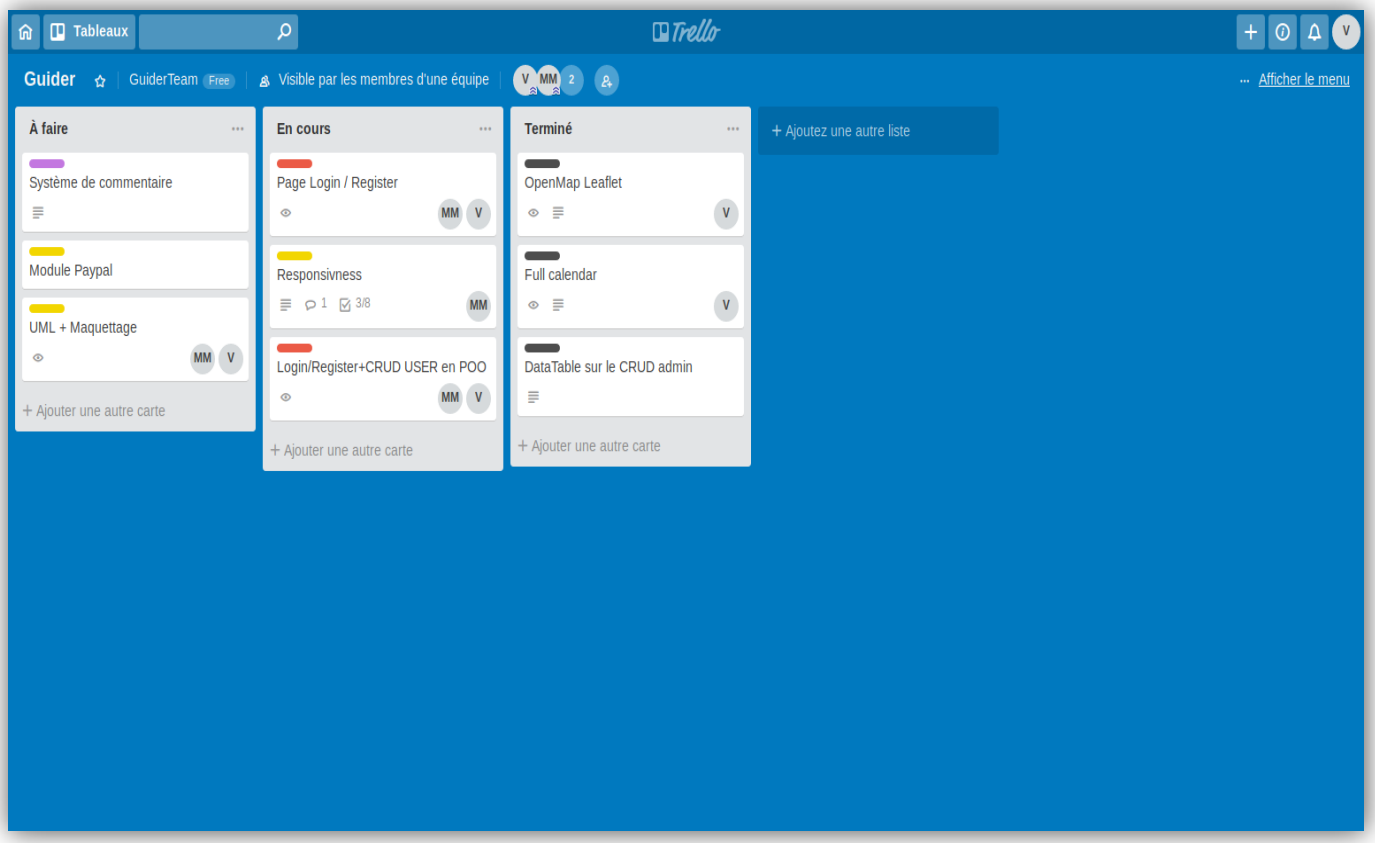


Figure 7: organisation du workflow de début Août jusqu'au 15 Septembre

1.4 Déroulement du stage et rapports avec le Sictiam

L'organisme appelé SICTIAM, qui a récompensé d'un troisième prix l'idée de création de ce site, m'a accueillie dans ses locaux pendant un mois afin de démarrer dans la conception et le développement du projet en toute tranquillité.

Le chef de projet du SICTIAM nous a aiguillé sur la façon d'établir une stratégie, de toucher les bons interlocuteurs, afin de diffuser l'application d'un point de vue commercial et m'a également donné des conseils sur les éléments qui lui paraissaient indispensable pour attirer le maximum de touristes vers les guides locaux dans l'arrière-pays des Alpes Maritimes.

Nous avons eu **2 réunions** au cours du mois de stage avec le chef de projet, Mr **Clément Seite**, qui nous a demandé de rédiger un cahier des charges et de faire les maquettes du site, d'utiliser le diagramme de Gantt afin de gérer l'évolution du projet et nous donner nous-mêmes des deadlines.

Nous avons eu une réunion de compte rendu de l'avancement du projet au mois de juillet durant lequel nous avons présenté la conception du projet en présence des représentants de l'école Simplon et de notre maitre de stage.

Le résultat de ce mois de stage nous a permis de faire une démonstration en direct et Mr Seite s'est improvisé utilisateur : il a testé l'ergonomie du site et nous a donné des conseils sur les boutons de retour ou les parties qu'il aurait aimé voir sur la page consultée. L'approche et le rendu de notre projet correspondent bien à ce qu'il attendait notre approche du projet.

1.5 Les résultats obtenus et avancement du projet

Je peux dire que le projet est presque terminé.

Reste à faire :

-Mise en place le module de paiement par **Paypal** non effectuée (mais j'ai été voir différents tutoriels notamment chez Grafikart sur le sujet), étant donné que je ne compte pas me lancer dans une création d'entreprise, mes objectifs professionnels étant différents. Néanmoins le Sictiam est prêt à soutenir le lancement parce que le projet leur

- Partie gestion des rendez-vous, consultation et éventuellement annulation, au niveau des guides, n'a pas été développée.

Quelques screenshots du site Guider ci-dessous

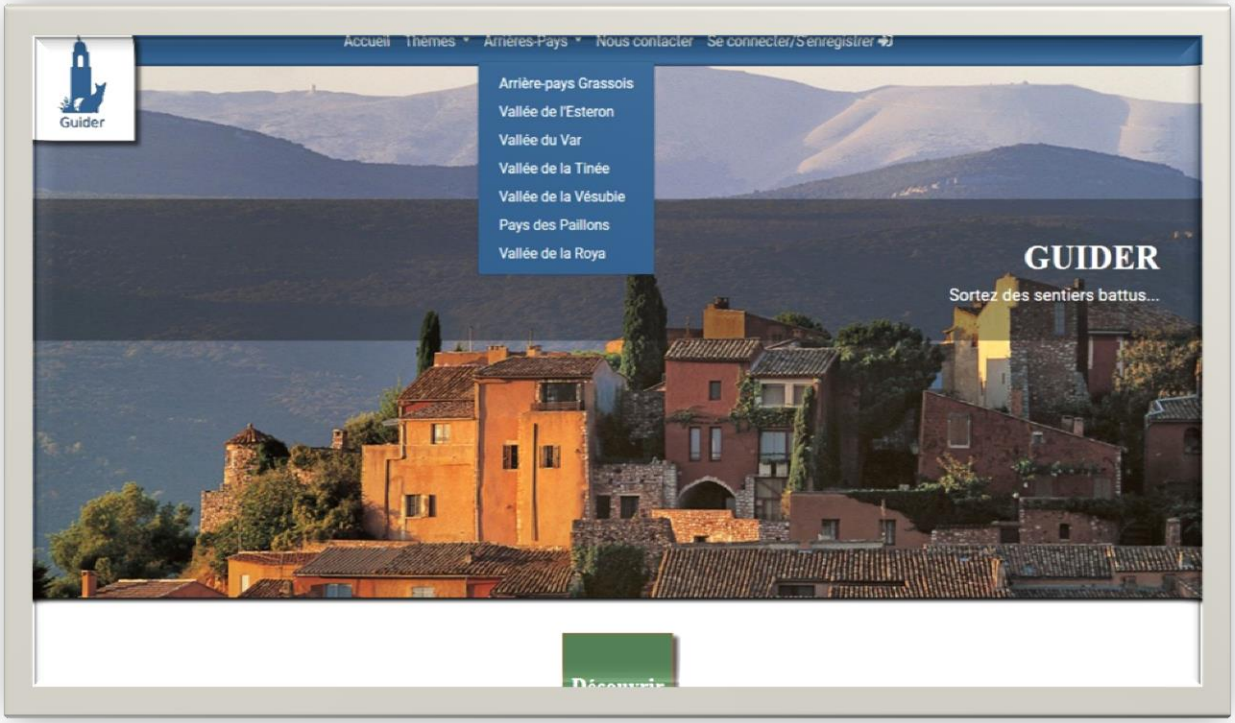


Figure 8: Page d'accueil

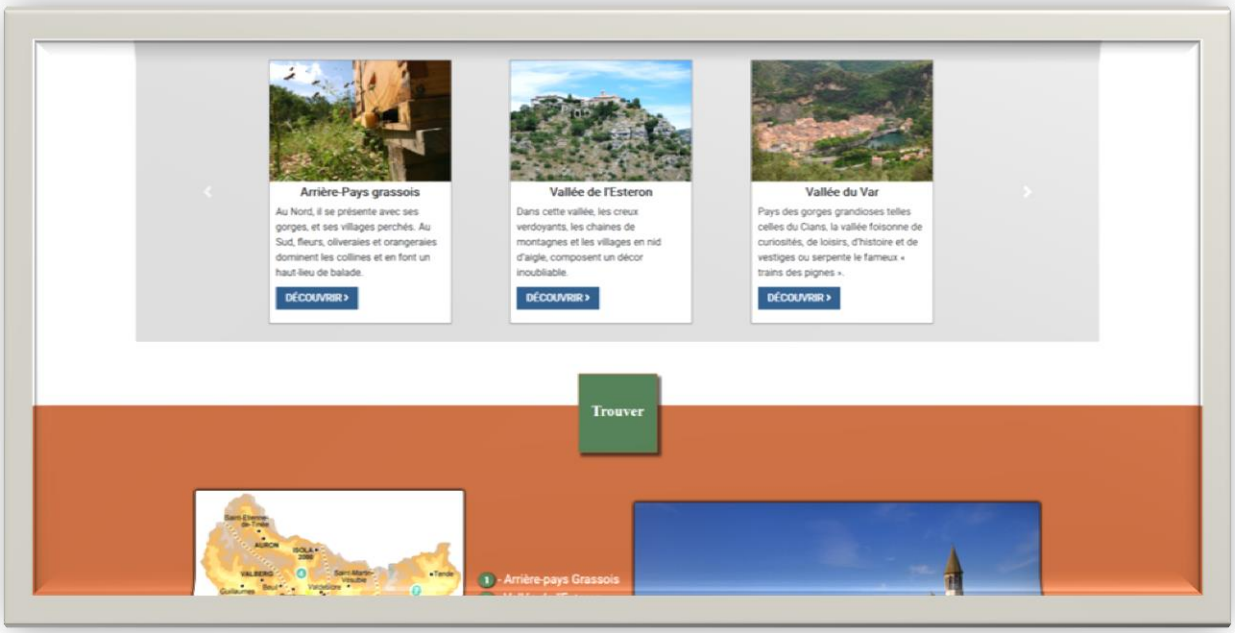


Figure 9: Page d'accueil, le slideshow manuel

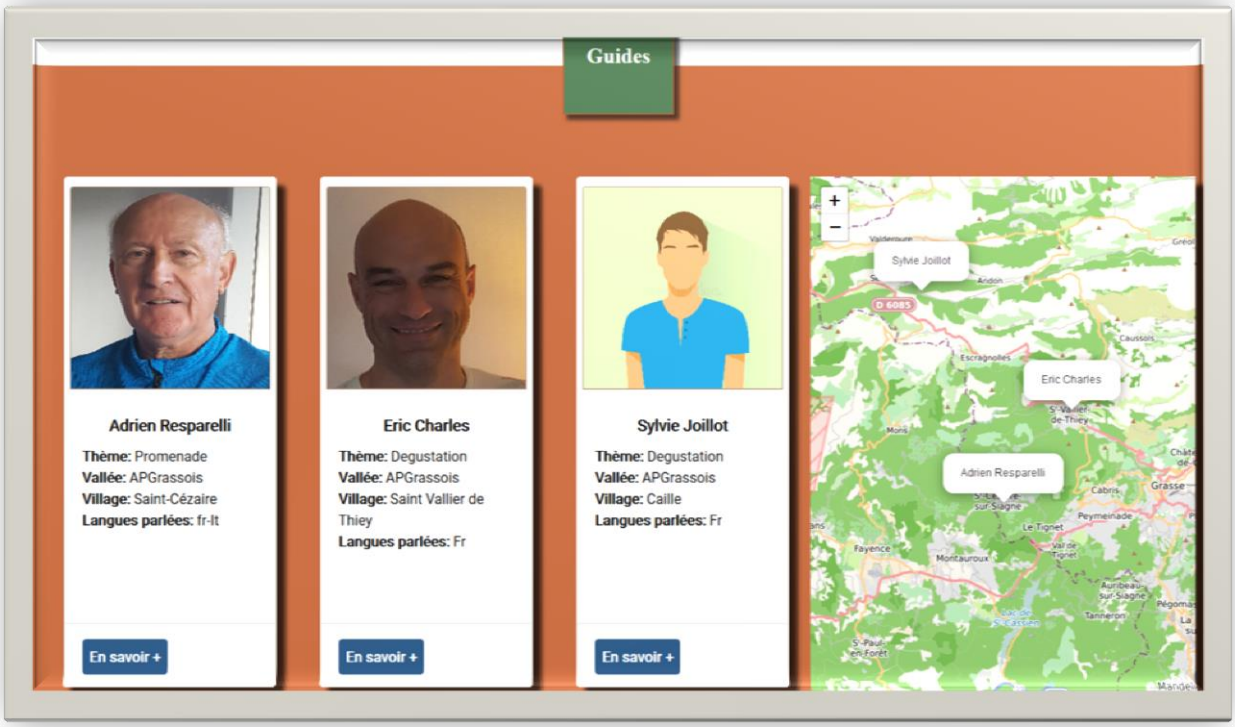


Figure 10: Choix de guides dans un arrière-pays et géolocalisation

Figure 11: Formulaire de réservation

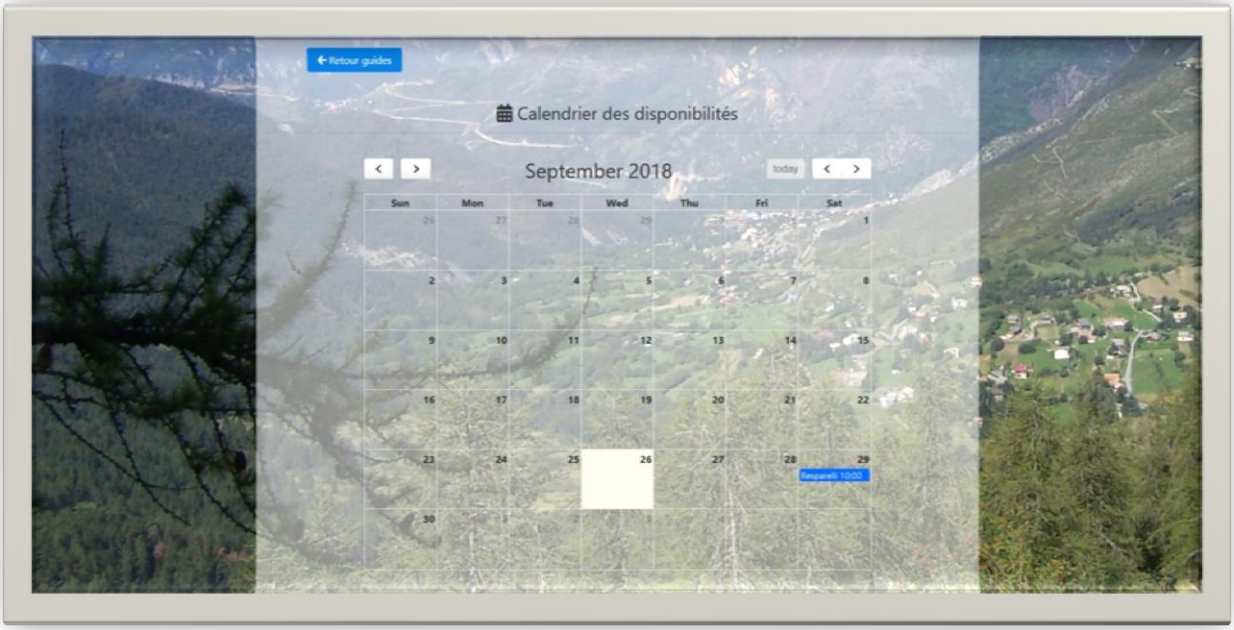


Figure 12: Calendrier des réservations

1.6 Problèmes rencontrés et évaluation personnelle

Le problème le plus difficile à gérer fut que mon premier collègue est tombé malade. Certaines parties dont il avait la charge n'ont pu être effectuées comme prévu et je n'ai pas pu avancer aussi vite que je le voulais.

J'ai dû terminer le front-end sans lui et sans le consulter puisqu'il n'était pas joignable, dans l'espoir que, à son retour, mon travail et ma vision du site correspondent bien à ce que nous avions prévu et corresponde aussi à ses propres attentes.

Du coup, j'ai élaboré le back-end en solo sans pouvoir lui expliquer et sans l'aider à monter en compétences, alors que c'était un des objectifs que je m'étais fixé.

Le projet du site s'est transformé en projet chef d'oeuvre . Les cours de la journée et des nouveaux petits projets ponctuels en cours ont ralenti la création. Mais finalement, le fait d'avoir été seule a été très formateur.

Un nouveau collègue, Mathieu Money, s'est greffé au projet et j'ai enfin pu rattraper le retard. Nous avons travaillé le côté « responsive » qui n'était pas parfait, réorganisé les fichiers pour plus de lisibilité, et de sécurité au niveau des données, le code a été refactorisé quand cela était nécessaire. Nous sommes passés du Php procédural à la POO pour tout ce qui touche à l'utilisateur.

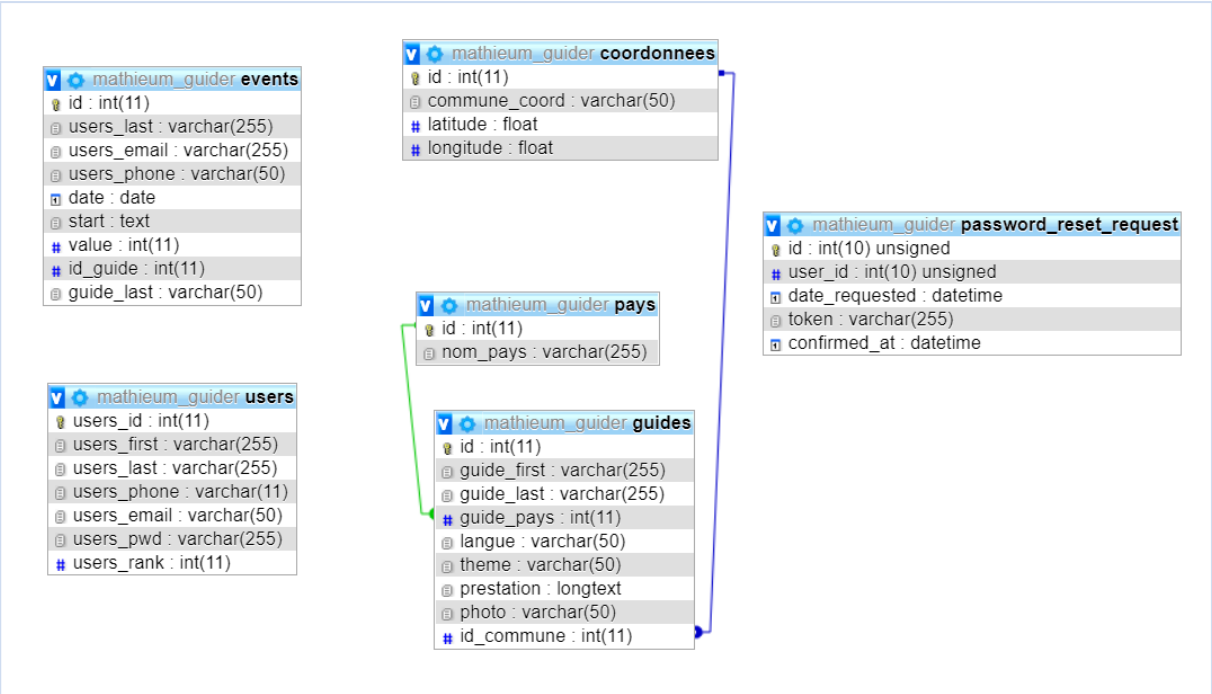
J'ai beaucoup appris et apprécié le développement en Php. Cela m'a permis de trouver des ressources fiables sur Internet (majoritairement en anglais), des tutoriels intéressants (merci Grafikart !) et des réponses aux bugs que j'ai rencontrés. Grâce à Mathieu, j'ai pu refaire un peu de front-end (notamment du JQuery) et sortir la tête du Php !

En conclusion, j'ai vu la création d'un projet de A à Z et c'est une belle et intensive expérience. On se rend compte qu'il y a énormément de choses à prévoir à l'avance et qu'il faut soigner la conception, sinon on perd du temps sur le développement et il arrive qu'on soit obligé de revenir sur ce que l'on a fait parce qu'il manque des éléments.
Mais je ne vais pas m'arrêter là ! Je pense qu'après la formation je développerai le module manquant et je coderai le même site en utilisant Laravel que je n'ai fait que survoler pour l'instant.

II. CONCEPTION/PROGRAMMATION DU PROJET

1.1 Mise en place de la base de données

Comme je l'ai indiqué dans le chapitre précédent, la base de donnée de Guider contient 6 tables créées sous **MySQL 5.6** (avec le moteur **innoDB** pour les besoins en base de données relationnelle) et l'encodage **utf8_unicode_ci** pour la plus grande compatibilité possible. Grâce à l'outil « Concepteur » de MySQL j'ai pu obtenir le diagramme relationnel suivant :



Toute modification ou suppression dans la structure des tables sera mise à jour automatiquement dans le concepteur.

-La table « **guides** » contient toutes les données sur les guides recrutés ainsi que 2 clés étrangères permettant une relation avec les tables « pays » et « coordonnées ».

-La table « **pays** » est une table intermédiaire, chaque pays pouvant regrouper plusieurs guides.

-La table « **coordonnées** » a été créée avec les champs « latitude » et « longitude » qui serviront à positionner les guides et leur commune sur une **carte de géolocalisation** grâce à **l'API LeafLet**. **OpenstreetMap**

-La table « **events** » est utilisée par le plugin **FullCalendar.js** et répertorie toutes les réservations effectuées.

-La table « **users** » possède un champ users_rank qui définit les droits d'accès aux fonctionnalités.

-La table « **password_reset_request** » permettra d'enregistrer toute modification de mot de passe demandée par un utilisateur et gérer l'utilisation du **token** fourni dans le mail de modification.

J'ai utilisé le serveur **mySQL** (sous Lamp pour moi et Xamp pour mon collègue) pour l'intégration de la base de données et l'interface graphique **PhpMyAdmin.**

J'ai tout d'abord créé le script de création de la base de données en langage SQL. J'ai, dans un premier temps, utilisé la base en « **localhost** » afin de développer toutes les fonctionnalités du site. Ensuite, mon collègue et moi avons transféré le site sur un hébergement payant (02switch) et la base de données distante a été créée par l'import du fichier SQL via l'onglet « import » de l'interface **PhpMyAdmin**.

☐	Éditer	Copier	Supprimer	16	Delcros	ale.delcros1142@gmail.com	0613118432	2018-09-22 14:00	1	17	Joillot	Sylvie
☐	Éditer	Copier	Supprimer	17	Delcros	ale.delcros1142@gmail.com	0613118432	2018-08-25 14:00	1	19	Baron	Albert
☐	Éditer	Copier	Supprimer	18	Delcros	ale.delcros1142@gmail.com	0493121144	2018-09-29 10:00	1	2	Resparelli	Adrien

Figure 13: Extrait table « Events »

☐	Éditer	Copier	Supprimer	39	8	2018-08-30 19:42:37	NULL	2018-08-30 19:43:20
☐	Éditer	Copier	Supprimer	40	8	2018-08-31 11:18:22	NULL	2018-08-31 11:28:58
☐	Éditer	Copier	Supprimer	41	8	2018-08-31 11:30:30	NULL	2018-08-31 11:31:02
☐	Éditer	Copier	Supprimer	42	8	2018-08-31 11:47:57	NULL	2018-08-31 11:48:55
☐	Éditer	Copier	Supprimer	43	8	2018-09-01 16:55:31	NULL	2018-09-01 16:55:47

Figure 14: Extrait table « password_reset_request

users_id	users_first	users_last	users_phone	users_email	users_pwd	users_rank
3	Alexis	Delcros	0493121144	ale.delcros1142@gmail.com	$2y$10$kfbCgU4awjV5iCp0dmcXZOUdyC15UplG63N2dzdlwCT...	1
8	Valérie	Baron	0609637613	valerie.baron67@gmail.com	$2y$10$pfnvQLaM9ydPpti9FcBqKehzFmIcGnrvKG3s1MsvhFV...	3
9	Mathieu	Money	0633251334	mmoney@devcannes.com	$2y$10$WwZ1NDHTip0OJhCqSW/Qkeo1TYkdaV9Xx2BgvPBz/xf...	3
19	qsdd	qsddfsfsdfsdsdfsdffsd	0000000000	qsd@qsd.qsd	$2y$10$8WJzNz.Hbgbh/t8FUlRiB.vp5RPegdgMD7H5n6gqiQC...	2

Figure 15: Extrait table « Users »

1.2 Connexion à la base de données

La Class **Database** a été créée en **Poo** et est nécessaire à la connexion à la base de données : Cette classe contient deux méthodes statiques de connexion et de déconnexion dont la valeur est stockée dans une variable $connection.

```php
<?php

class Database
{
    private static $dbHost = "localhost";
    private static $dbName = "mathieum_guider";
    private static $dbUser = "mathieum_guider";
    private static $dbUserpassword = "[redacted]";

    private static $connection = null;

    public static function connect()
    {
        try{
                //echo "try connexion\n";
                self::$connection = new PDO("mysql:host=" . self::$dbHost . ";charset=UTF8;dbname=" . self::$dbName, self::$dbUser,self::$
                    dbUserpassword);


            }
            catch (PDOException $e)
            {

                die($e->getMessage());

            }
            return self::$connection;
    }

    public static function disconnect(){
        self::$connection = null;

    }

}
```

Propriétés nécessaires à la connexion en privé pour ne pas pouvoir y accéder en dehors de la classe.

Exception : récupère le message d'erreur de la fonction de connexion et l'affiche.

```php
header.php
<?php

    require './class/Database.php';
    // va chercher le fichier database.php pour connexion
    $db = Database::connect();
    require('functions.php');
    session_start();
?>

<!DOCTYPE html>
<html lang="fr">

    <head>
        <title>GUIDER - v2.1.0</title>
        <meta charset="UTF-8">
```

Connexion à la base de données indiquée dans le « **header** » qui est inclus dans tous les fichiers du site.

1.3 Développement du site

Le site Guider a été développé en HTML5, CSS3 (et son framework Bootstrap 4.1) Javascript ES5 (et son framework jQuery) ainsi qu'en PHP 5.
La structure du dossier contenant le code se trouve en Annexe.

Les structures html header, footer et la barre de navigation ont été codés dans des fichiers séparés pour éviter les répétitions du code et sont appelés grâce à la fonction Php **include**('chemin et nom du fichier').

```
1 <?php include('includes/header.php');
2
```

```
<?php include('includes/footer.php'); ?>
```

```
<body>
        <?php include('nav.php'); ?>
```

1.3.1 La page d'accueil

La page d'accueil permet d'avoir accès à plusieurs zones du site, certaines zones ne seront accessibles qu'en fonction des droits utilisateurs. Les droits vont de 1 à 3 :

1= utilisateur
2 = guide
3= administrateur du site.

- les 7 arrière-pays : les **liens** pour les arrière-pays sont consultables à partir du « **dropdow menu** » et également du slideshow : bouton « en savoir + » ;
- le login et l'inscription : **lien** vers la page de login ;
- mon compte/ déconnexion : menu qui apparait une fois que la **session utilisateur** est ouverte et que l'utilisateur s'est connecté ;
- le compte administrateur : lien qui apparait dans le **footer** une fois que l'admin s'est connecté (droits admin attribués à l'utilisateur dans la base de données) et donnant accès au **CRUD** administrateur.

```
29            Arrières-Pays
30          </a>
31          <div class="dropdown-menu" aria-labelledby="paysDropDown">
32            <a class="dropdown-item text-custom-white" href="pays.php?url=1">
              Arrière-pays Grassois</a>
33            <a class="dropdown-item text-custom-white" href="pays.php?url=2">Vallée de
              l'Esteron</a>
34            <a class="dropdown-item text-custom-white" href="pays.php?url=3">Vallée du
              Var</a>
35            <a class="dropdown-item text-custom-white" href="pays.php?url=4">Vallée de
              la Tinée</a>
36            <a class="dropdown-item text-custom-white" href="pays.php?url=5">Vallée de
              la Vésubie</a>
37            <a class="dropdown-item text-custom-white" href="pays.php?url=6">Pays des
              Paillons</a>
38            <a class="dropdown-item text-custom-white" href="pays.php?url=7">Vallée de
              la Roya</a>
39          </div>
40        </li>
41
42        <li class="nav-item">
43          <a class="nav-link text-custom-white" href="conta
44        </li>
45        <?php
46          if(!isset($_SESSION['user'])){
47        ?>
48        <li class="nav-item">
49          <a class="nav-link text-custom-white" href="login-register.php">Se connecter/
            S'enregistrer <i class="fas fa-sign-in-alt"></i></a>
50        </li>
51      <?php } ?>
52      <?php
53          if(!empty($_SESSION['user'])){
54        ?>
55        <li class="nav-item dropdown">
56          <a class="nav-link dropdown-togg
            accountDropDown" role="button" data-toggle="dropdown" aria-haspopup="true"
            aria-expanded="false">
57            Mon Compte
58          </a>
59          <div class="dropdown-menu" aria-labelledby="accountDropDown">
60            <a class="dropdown-item text-custom-white" href="account.php">Mon Compte</a>
61            <a class="dropdown-item text-custom-white" href="disconnect.php">Déconnexion
              </a>
```

url récupérée en Php : **include** (fichier correspondant au pays)

Session user ouverte : accès au menu Mon Compte

```
// We check if a GET request was sent.
if (isset($_GET['url'])){

    // Here we use an if, else if, else condition instead of a switch.
    // Each value is the same as the id into the database.
    if ($_GET['url'] == 1){

        include('includes/pays/grasse.php');

    }elseif($_GET['url'] == 2){

        include('includes/pays/esteron.php');
```

On récupère les paramètres passés dans l'url pour afficher la page

```php
<!-- FOOTER -->
<section class="container-fluid">
    <footer class="col-xl-12">

        <p class="text-custom-white col-sm-12 col-xl-10">© Copyright 2018 - DevCannes.com
        </p>
        <?php

        if (isset($_SESSION['user']) && $_SESSION['user']['RANK'] == 3){


            ?>
            <a href="admin/index.php" class="nav-link text-custom-white col-sm-12
            col-xl-2">Administration <span class="fas fa-sign-in-alt"></span></a>
        <?php } ?>

    </footer>
</section>
```

Si l'utilisateur se connecte et ses **droits** sont de niveau 3, l'accès à la page d'administration apparait.

1.3.2 Un arrière-pays

Toutes les pages arrière-pays ont été conçues sur le même modèle. J'ai dû en créer 7 car chaque page donne des informations différentes et fait tourner un carousel automatique avec des images propres à chaque arrière-pays.

```php
<section class="container-fluid">
    <section class="bannerCountry">
        <img src="./img/pays/grasse/gourdon-600.jpg" alt="Plan Large de Gourdon">
        <div class="bannerCountryCaption">
            <p class="bannerCountryCaptionText">Vallée de l'Estéron</p>
        </div>
    </section>
    <!-- <div class="separation"></div> -->
    <section class="presentCountry">
        <div class="square title-font"><h1>Nature</h1></div>
        <p>Les grands espaces sauvages aux pentes boisées alternent avec plateaux
        arides. C'est le pays des clues et des villages belvédères vivant encore au
        rythme ancestral de l'élevage des moutons. Laissez-vous charmer par les villages
        de caractère.<br><br>Notre sélection de villages incontournables : <i>
        Roquesteron, Tourette du château, Gilette, Sigale</p>
    </section>
    <!-- Carousel Fade -->


    <div id="carouselPaysFade" class="carousel slide carousel-fade" data-ride="carousel">
        <div class="carousel-inner">
            <div class="carousel-item active">
                <img class="d-block w-100" src="./img/pays/esteron/slideshow/slide1.jpg"
                height="400px" alt="First slide">
            </div>
            <div class="carousel-item">
                <img class="d-block w-100" src="./img/pays/esteron/slideshow/slide2.jpg"
                height="400px" alt="Second slide">
            </div>
            <div class="carousel-item">
                <img class="d-block w-100" src="./img/pays/esteron/slideshow/slide3.jpg"
                height="400px" alt="Third slide">
```

```
41      <!-- FIn Carousel -->
42      <section class="presentGuide">
43          <div class="square title-font"><h1>Guides</h1></div>
44          <div class="row col-xl-12">
45              <div class="localGuide col-sm-12 col-md-7 col-lg-7 col-xl-8 mb-5">
46
47                  <?php
48                  countryListGuides($db, (int)$_GET['url']);
49                  // appel fonction avec comme parametres la connexion a la bdd et l'url
                    du pays(id)
50                  ?>
51
52              </div>
53              <!-- Map -->
54              <div id="mapid" class="
```

Appel à la **fonction** PHP **countryListGuides (fichier functions.php)** qui prend en paramètres la connexion à la base de données et **l'URL** du pays qui correspond à **l'id du pays dans la base de données.** Vont être affichés tous les guides qui correspondent par ex. à l'id 2 soit « la vallée de l'Estéron »

```
10  function countryListGuides($db,$id){
11      // j'ai besoin de 2 paramètres: la variable de connexion à la bd et l'id du pays
        queje vais récupérer avec le Get Url sur la page pays choisie.
12
13      $statement = $db->prepare('SELECT guides.id, guides.guide_first, guides.guide_last,
            coordonnees.commune_coord AS commune, guides.langue,guides.prestation,guides.
            theme, guides.photo,pays.nom_pays AS pays ,coordonnees.latitude, coordonnees.
            longitude FROM guides, coordonnees, pays WHERE  guides.id_commune = coordonnees.
            id AND guides.guide_pays = pays.id AND guides.guide_pays = :id');
14
15      $statement->bindValue(":id", $id, PDO::PARAM_INT);
16      $statement->execute();
17
18      while ($item = $statement->fetch())
19  // prend la premiere ligne du statement et le met dans le html, s'execute tant qu'il y a
    des lignes
20      {
21          echo '<div class="card js-marker" data-lat = "' . $item['latitude'] . '"
                data-lng = "' . $item['longitude'] . '" data-text = "' . $item['guide_first']
                 . ' ' . $item['guide_last'] . '">
22          <img class="card-img-top" src="./img/guides/' . $item['photo'] . '" alt="Card
                image cap">
23          <div class="card-body">
24          <h4 class="card-title"><strong>' . $item['guide_first'] . '</strong>'. ' ' . '<
                strong>'. $item['guide_last'] . '</strong>' . '</h4>
25          <p class="card-text"><strong>Thème: </strong>' . $item['theme'] . ' <br><strong>
                Vallée:  </strong>' . $item['pays'] . ' <br><strong>Village:  </strong>' . $
                item['commune'] . ' <br><strong>Langues parlées: </strong>' . $item['langue']
                 . '</p>
26          </div>
27          <div class="card-footer" id="ensavoirplus">
28          <a class="btn p-2 rounded" href="includes/pays/view_guide.php?id=' .$item['id'] .
                '">En savoir +</a>
29          </div>
30          </div>';
31      }
32
33  }
```

Requête demandant les guides par l'id du pays

Explication de la requête SQL :

- je dois récupérer tous les guides de la table « guides » dont le champ « guide_pays » a la même valeur (le même id). Le champ guide_pays

(clé étrangère) pointe vers la clé primaire de la table « pays » qui donne les noms des arrière-pays. Au lieu d'intégrer un id dans mon document HTML je vais intégrer le nom du pays.

-je dois récupérer les coordonnées géographiques des villages auxquels appartiennent les guides. Le champ **id_commune** de la table **« guides » (clé étrangère)** pointe vers la clé primaire de la table « coordonnées » qui contient le nom des villages et leurs coordonnées géographiques.

-Je vais **intégrer** dans mon document HTML **le nom du village** au lieu de son id et les coordonnées que je vais utiliser en **data attribute** (data-lng et data-lat) pour ensuite pouvoir implémenter la carte de géolocalisation des guides.

Résultat d'une carte **GUIDE** et la géolocalisation :

J'ai utilisé ***Leaflet.js*** (https://leafletjs.com/reference-1.3.4.html) qui est une librairie Javascript de cartographie interactive open-source. Elle est notamment utilisée par le fournisseur de fonds de carte :

OpenStreetMap

```
1  let $map = document.querySelector('#mapid')
2  /*creation leafletMap class + async function that will display the map on the screen when map is loaded */
3  class leafletMap {
4
5      constructor () {
6
7          this.map = null
8          this.bounds = []
9      }
10
11     async load (element){
12
13         return new Promise((resolve,reject) => {
14             $script('https://unpkg.com/leaflet@1.3.3/dist/leaflet.js', () =>{
15             this.map = L.map(element).setView([43.799999,6.95], 10)
16             /*setView is necessary to center the map on an area: here middle of Alpes maritimes*/
17             // tileLayer:damier dans lequel on va charger diff cartes, on utilise ici openstreetmap
18             L.tileLayer('//{s}.tile.openstreetmap.fr/osmfr/{z}/{x}/{y}.png', {attribution: '&copy; <a href="https://
19             maxZoom: 18 }).addTo(this.map)
20             /*max zoom level going from 1 to 20*/
21             resolve()
22             })
23         })
24
25     }
26 /*markers being displayed on the map, using data-lat et data-lng attributes*/
27     addMarker (lat,lng,text){
28
29         let point = [lat, lng]
30         this.bounds.push(point)
```

```
31 /*params of popup that can be changed, here we want a static popup*/
32         L.popup({
33
34             autoClose: false,
35             closeOnEscapeKey: false,
36             closeOnClick: false,
37             closeButton: false,
38             className: 'marker',
39             maxWidth: 200
40         })
41
42         .setLatLng(point)
43         .setContent(text)
44         .openOn(this.map)
45
46     }
47
48     center(){
49         this.map.fitBounds(this.bounds)
50     }
51
52 }
53
54 const initMap = async function(){
55     // // variable servant à afficher la map dans ma div id 'mapid', set view determine le point central de la map, 10 correspond au niveau
56     // // de zoom choisi
57
58     let map = new leafletMap()
59     await map.load($map)
60     Array.from(document.querySelectorAll('.js-marker')).forEach((card) => {
61
62         map.addMarker(card.dataset.lat, card.dataset.lng, card.dataset.text)
63     })
64
65     map.center()
66 }
67
68 if ($map !== null){
69
70     initMap()
71 }
72
```

Script Leaflet permettant d'afficher la carte et de la centrer sur un point précis. Affichage des markers qui correspondent aux coordonnées des villages

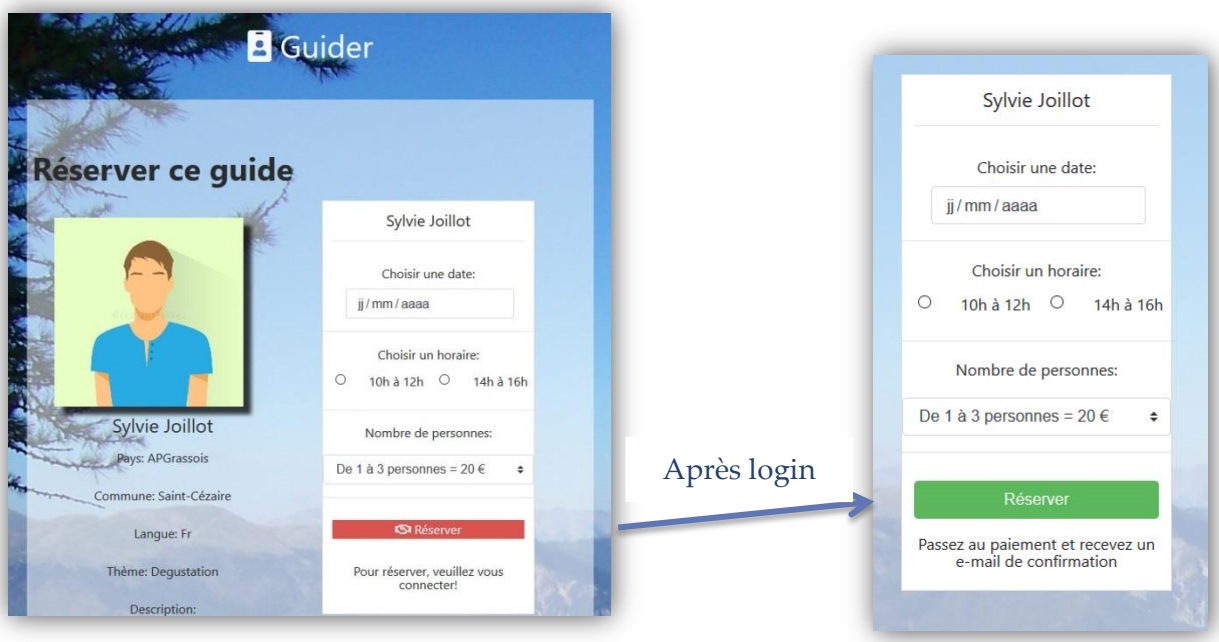
1.3.3 Fiche « guide » et partie réservation

```php
<?php

session_start();

require '../../class/Database.php';
// get guide's id in order to select each data concerning this guide
if(!empty($_GET['id'])){

    $id = checkInput($_GET['id']);
}

$db = Database::connect();
$statement = $db->prepare('SELECT guides.id, guides.guide_first, guides.guide_last, coordonnees.commune_coord AS commune, guides.lang
    photo,pays.nom_pays AS pays ,coordonnees.latitude, coordonnees.longitude FROM guides, coordonnees, pays WHERE guides.id = ?');

// prepare page guide with all elements concerning guide in order to fill the card, selection by id
$statement->execute(array($id));
$item = $statement->fetch();

$_SESSION['guide'] = $item['id'];
$_SESSION['name'] = $item['guide_last'];
$_SESSION['first_name'] = $item['guide_first'];

Database::disconnect();
```

La fiche du guide est créée grâce à la **superglobale** $_GET qui va récupérer l'id du guide sélectionné dans l'url et par une requête SQL préparée puis exécutée qui va récupérer les informations. L'utilisateur a accès à un formulaire de réservation qu'il ne pourra valider que s'il est connecté sinon le bouton de réservation reste indisponible :

```php
<?php
    if(!empty($_SESSION['user']['ID'])){

        echo '<input id="reservation" class="btn btn-order bg-success w-100 p-2 text-light" type="submit" name="reserve" value="Réserver"> <
            h6>Passez au paiement et recevez un e-mail de confirmation</h6>';


    } else {

        echo '<p class="bg-danger text-center text-light"><i class="far fa-handshake"></i> Réserver</a> <h6>Pour réserver, veuillez vous
            connecter!</h6></p>';

    }

?>
```

Je vérifie si la session « user » existe



Le formulaire va envoyer les données dans la base de données et remplir les champs de la table « events ».

Partie non développée : paiement de la réservation et confirmation de commande par envoi de mail au guide et à l'utilisateur.

Le **calendrier des disponibilités** est créé par le **plugin Jquery FullCalendar.js**. :

Les données sont récupérées grâce à une requête SQL. Le résultat sera stocké dans une variable contenant un tableau associatif nommé $data. Les données seront converties au format JSON par la fonction **json_encode()** et seront récupérées et intégrées par le plugin directement dans le layout du calendrier.

```php
<?php

session_start();

require '../../class/Database.php';



$id = $_SESSION['guide'];

$data = array();

// prepare and execute statement query on guide events
$db = Database::connect();
$statement = $db->prepare('SELECT * FROM events WHERE id_guide = ?');
$statement->execute(array($id));
$result = $statement->fetchAll();


foreach($result as $row)
{
// datas that will be implemented into calendar
    $data[] = array(

        'id' => $row['id'],
        'title' => $row['guide_last'] . ' ' . $row['start'],
        'start' => $row['date']


    );
}
   //   echo "<pre>";
        // $statement->debugDumpParams();
        // echo "</pre>";


// // conversion data into string to be displayed into event calendar
// }
echo json_encode($data);
// function cleaning datas
```

Je récupère toutes les réservations effectuée pour un guide et les stocke dans un tableau. Les propriétés **title** et **start** sont obligatoires pour faire fonctionner le calendrier

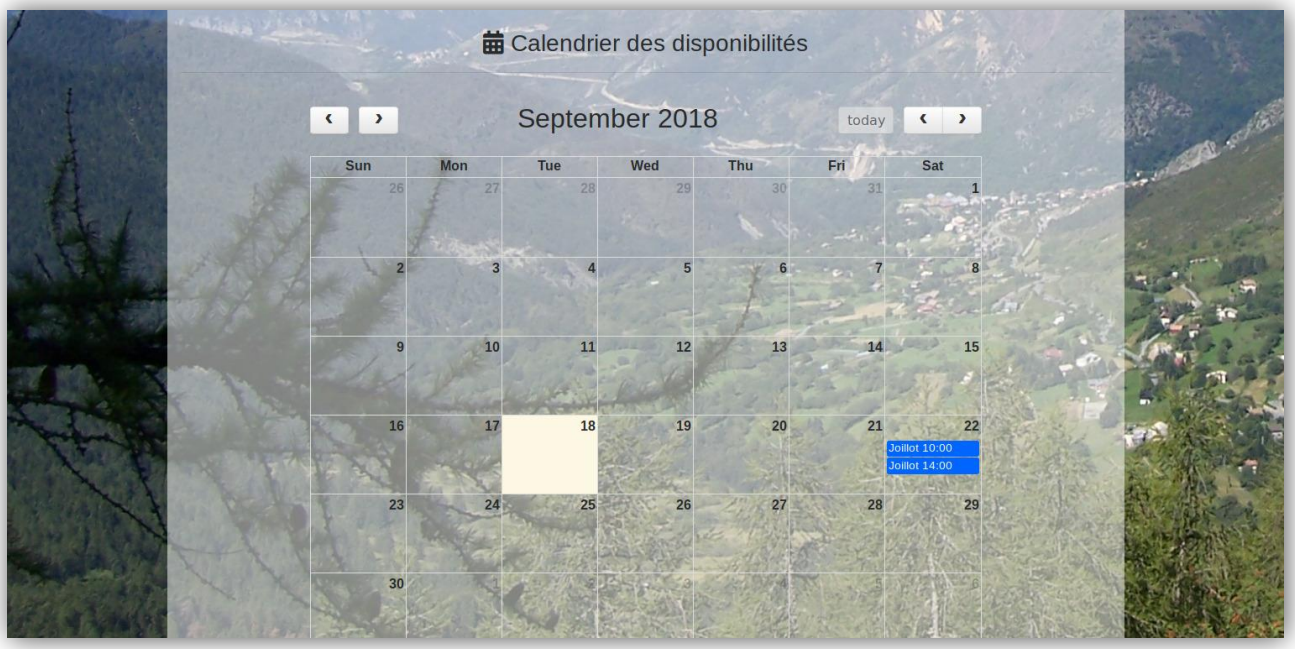


Figure 16: vue du calendrier

1.3.4 Partie Administrateur

L'administrateur se connecte sur la page d'accueil et a accès à la gestion. Une petite sécurité a été mise en place pour éviter qu'une personne mal intentionnée se positionne directement dans le fichier admin via l'URL :

```php
<?php

include ('header_admin.php');

// SECURITE: On teste si la variable de session existe et si elle existe si la valeur du rang est 3
if(empty($_SESSION['user']) && $_SESSION['user']['RANK'] != 3)
{
  // Si c'est le cas, on redirige vers le formulaire de login
  header('Location: ../index.php');
  exit();
}
?>
```

L'administrateur a la gestion complète des **guides** locaux et une partie de la gestion du **User** (modification et suppression) dans un **CRUD**.

J'ai utilisé le **plugin jQuery Datatable** pour l'affichage des tableaux du CRUD pour obtenir 2 onglets. Ce plugin ajoute un bouton de recherche ainsi que des flèches permettant le classement par ordre croissant ou décroissant. Il affiche 10 entrées par page pour une lecture plus agréable.

Guides | Utilisateurs

Liste des guides + Ajouter

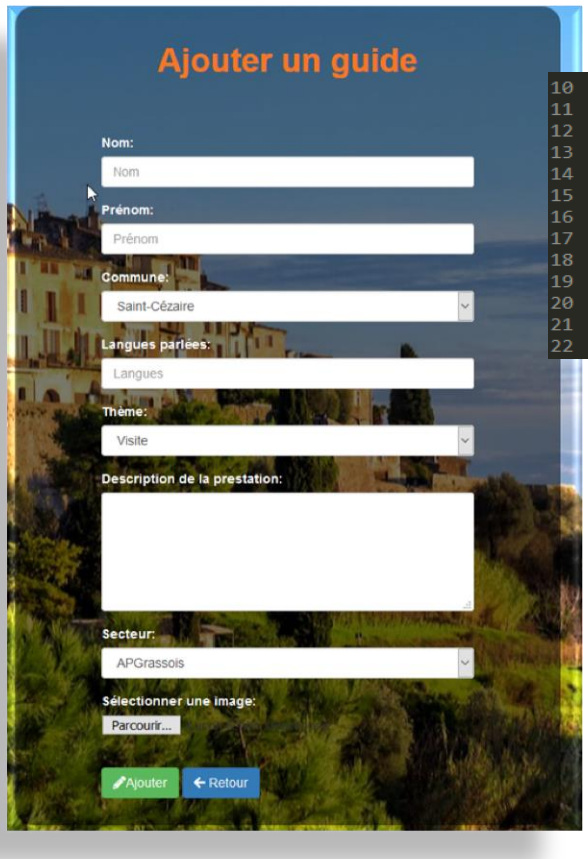
Show 10 entries Search:

Prénom	Nom	Village	Thème	Pays	Action
Clémence	Antaro	Touët-sur-Var	Visite	Var	Voir Modifier Supprimer
Albert	Baron	Tourette-du-Château	Visite	Estéron	Voir Modifier Supprimer
Laura	Barège	Saint-Martin-Vésubie	Visite	Vésubie	Voir Modifier Supprimer
Nicolas	Benoist	Saint-Sauveur-de-Tinée	Visite	Tinée	Voir Modifier Supprimer
Jeannette	Boutoux	Saint Dalmas de Tende	Visite	Roya	Voir Modifier Supprimer
Roberto	Cantelli	Gilette	Visite	Estéron	Voir Modifier Supprimer
Jean	Capitan	Lucéram	Visite	Paillon	Voir Modifier Supprimer
Eric	Charles	Saint Vallier de Thiey	Degustation	APGrassois	Voir Modifier Supprimer
Michel	Delfino	La Bollène-Vésubie	Produits locaux	Vésubie	Voir Modifier Supprimer
Joe	Demailly	Touët de l'Escarène	Dégustation	Paillon	Voir Modifier Supprimer

Showing 1 to 10 of 21 entries Previous 1 2 3 Next

Données implémentées par requête SQL

Exemple de création d'un guide :




```php
if(!empty($_POST))
{
    $guide_first = checkInput($_POST['prenom']);
    $guide_last = checkInput($_POST['nom']);
    $commune = checkInput($_POST['id_commune']);
    $langue = checkInput($_POST['langue']);
    $theme = checkInput($_POST['theme']);
    $prestation = checkInput($_POST['prestation']);
    $pays = checkInput($_POST['guide_pays']);
    $image = checkInput($_FILES['image']['name']);
    $imagePath = '../img/guides/' . basename($image);
    $imageExtension = pathinfo($imagePath, PATHINFO_EXTENSION);
    $isSuccess = true;
```

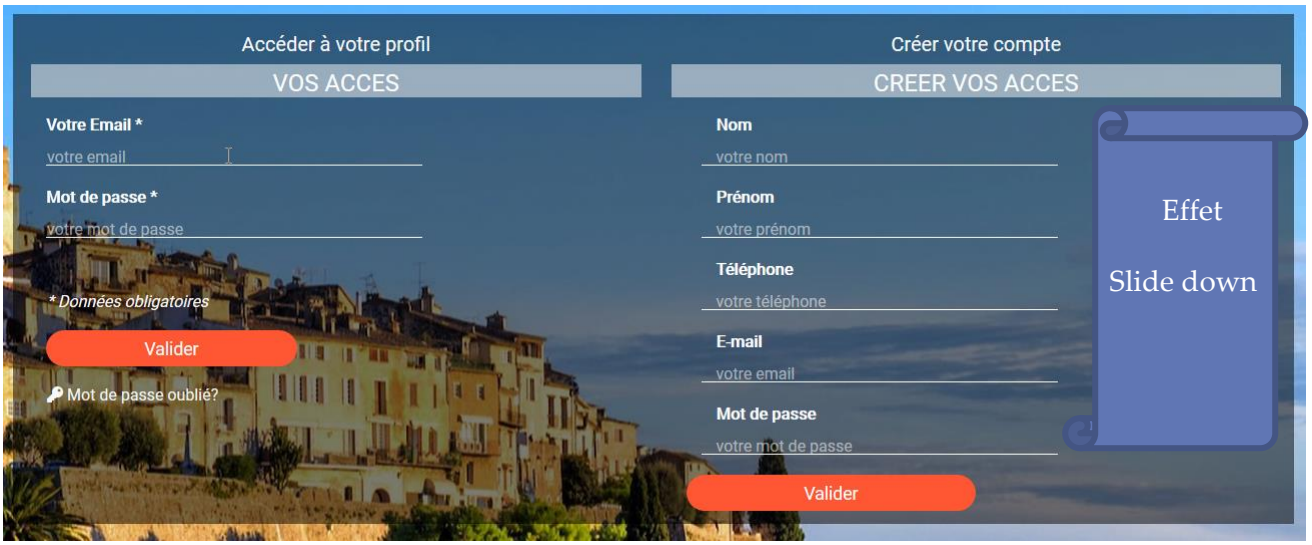
Les champs sont vérifiés un par un par une fonction de « nettoyage », ensuite je vérifie s'ils sont vides. Si oui un message d'erreur s'affiche en dessous du ou des champs vides. Tous les champs sont **obligatoires**. Le format de l'image doit être jpeg, jpg ou png, la taille de l'image inférieure à 5mo. Si l'image n'existe pas dans nos fichiers, elle est chargée.

1.3.5 Partie User

La partie User a été conçue en langage procédural puis a été refondée en POO. La classe User est utilisée dans les formulaires d'**inscription** et de **login**, dans la partie **Mon compte** et dans le CRUD User pour l'administrateur.

L'utilisateur créée son compte grâce au formulaire suivant :

La partie inscription est animée en **Jquery**, quand l'utilisateur appuie sur le bouton s'inscrire, la partie texte disparait en glissant et donne place au formulaire d'inscription :



```
jQuery(document).ready(function($){

    var screen = $(window);

    if(screen.width() > 740){
        // je cache le formulaire d'inscription à l'ouverture de la page
        document.getElementById('registerFormL');

        $('#registerFormL').hide();

        $('#registerBtnDisplayL').on('click', function(){

            //$('#registerMessageL').fadeOut();
            $('#registerMessageL,#registerBtnDisplayL').hide('slide',{direction:"up"}, 700)

            $('#registerFormL').delay(750).show('slide',{direction:'up'},700);


        });
    }
```

Le formulaire est caché dès le départ

Quand le bouton s'inscrire est cliqué, le message et le bouton s'effacent et laissent place au « register form » qui apparait doucement à l'écran(en 0.7 secondes)

-Validation côté client: L'utilisateur rentre ses données et celles-ci sont vérifiées par le plugin jQuery Validator .

```
1  jQuery(document).ready(function($){
2
3      // Method to add regex rule for validation
4      jQuery.validator.addMethod(
5          "regex",
6          function(value, element, regexp) {
7              if (regexp.constructor != RegExp)
8                  regexp = new RegExp(regexp);
9              else if (regexp.global)
10                 regexp.lastIndex = 0;
11                 return this.optional(element) || regexp.test(value);
12         },"Le format ne correspond pas !"
13     );
14
15     // Call the validate plugin method
16     $('#registerCheck').validate({
17         rules:{
18             'users_last':{
19                 'required':true,
20                 'regex':/^([a-z ééêêñçîì]{2,30})$/i
21             },
22             'users_first':{
23                 'required':true,
24                 'regex':/^([a-z ééêêñçîì]{2,30})$/i
25             },
26             'users_phone':{
27                 'required':true,
28                 'regex':/^([+(\d]{1})(([\d+() -.]){5,16})([+(\d]{1})$/
29             },
30             'users_email':{
31                 'required':true,
32                 'email':true
33             },
34             'users_pwd':{
35                 'required':true,
36                 'minlength':8,
37                 'maxlength':16
38             }
39         },
40         messages:{
41             users_last:{
42                 required:"Le champ est requis !",
43                 regex:"Doit être compris entre 2 et 30 caractères maximum. Doit uniquement comporter des lettres."
44
45             },
46             users_first:{
47                 required:"Le champ est requis !",
48                 regex:"Doit être compris entre 2 et 30 caractères maximum. Doit uniquement comporter des lettres."
49             },
50             users_phone:{
```

Script à intégrer dans le code

Le plugin appelle la méthode quand mon formulaire qui a l'id « registerCheck »est validé.

Règles de validation que nous avons définies :
- sont acceptées toutes les lettres de a à z, les lettres avec accent, le champ doit être compris entre 2 et 30 caractères. Le « i » indique que le **champ n'est pas sensible à la casse.**
- le regex du format téléphonique s'applique à un format international
- le script vérifie que le champ email soit bien au format e-mail(xxx@xxx.xxx)
-le mot de passe doit être compris entre 8 et 16 caractères.

-Validation côté serveur :

```php
if(isset($_POST['submitRegister']) && !empty($_POST['submitRegister'])){

    $first = cleanInput($_POST['users_first']);
    $last = cleanInput($_POST['users_last']);
    $email = cleanInput($_POST['users_email']);
    $phone= cleanInput($_POST['users_phone']);
    $pwd = $_POST['users_pwd'];

    // hashage du password pour sécurité

    $passHash = password_hash($pwd, PASSWORD_DEFAULT);


    $user = new User($email,$passHash,$id='',$first,$last,$phone);
    if($user->exist($db)){

        $_SESSION['errors']['user_exist'] = "Cet utilisateur existe déjà";


        header('Location:login-register.php');
    } else {
        $regInput = '/^([a-z ééêñçîì]{2,30})$/i';
        $regPhone = '/^([+(\d]{1})(([\d+() -.]){5,16})([+(\d]{1})$/';

        if ((isset($first) && !empty($first)) && (isset($last) && !empty($last))){

            if(preg_match($regInput, $first) && preg_match($regInput, $last)){

                if(isset($phone) && !empty($phone)){

                    if(preg_match($regPhone, $phone)){

                        if(filter_var($email, FILTER_VALIDATE_EMAIL)){

                            if(isset($pwd) && !empty($pwd) && strlen($pwd) >= 8 && strlen($pwd) <= 16){

                                // All verification has been passed, we created a new DB entry
                                try{

                                    $user->add($db);
                                    header('Location:login-register.php');
                                    //echo "OK en DB <a href='login-register.php'> Retout a la page login</a>";
                                }
                                catch(CustomException $e){

                                    die("<br /><strong>Error ! Details: </strong><br />" . $e->getMessage());
```

Quand le champ **input** de type « **submit** » appelé « submitRegister » est cliqué, les champs du formulaire passent dans une fonction de nettoyage qui supprime les espaces, les anti slashes et convertit les caractères spéciaux en entité HTML. Le mot de passe est crypté à l'aide la fonction php **password_hash().**

Un nouvel objet de classe « **User** » est créé, utilisant les paramètres du constructeur. Nous vérifions dans un premier temps que cet utilisateur n'existe pas dans la base de données et affichons un message d'erreur si tel est le cas. Si

les champs sont remplis, une vérification champ par champ, par rapport au REGEX que nous avons déterminé, est effectuée par la fonction **preg_match() .** L'e-mail passe également par une fonction de vérification, **filter_var(),** qui passe l'email par un filtre **FILTER_VALIDATE _EMAIL** qui valide l'adresse courrier. La fonction **strlen()** vérifie la longueur du password (nous avons demandé un password allant de 8 à 16 caractères). Si tout est ok, **la méthode add() de la classe User** est utilisée pour rentrer les données en base de données.

```
public function add($bdd){

    $addUser = $bdd->prepare('INSERT INTO `users`(`users_first`, `users_last`, `users_phone
        `, `users_email`, `users_pwd`, `users_rank`) VALUES ( :first, :last, :phone,
        :mail, :pass, :rank)');
    $addUser->bindValue(':first', $this->getFirst(), PDO::PARAM_STR);
    $addUser->bindValue(':last', $this->getLast(), PDO::PARAM_STR);
    $addUser->bindValue(':phone', $this->getPhone(), PDO::PARAM_STR);
    $addUser->bindValue(':mail', $this->getMail(), PDO::PARAM_STR);
    $addUser->bindValue(':pass', $this->getPass(), PDO::PARAM_STR);
    $addUser->bindValue(':rank', $this->getRank(), PDO::PARAM_INT);

    $exec = $addUser->execute();

    if(!$exec){

        throw new CustomException("An error occured, please check your query in User.php
            -> add()");
    }
}
```

Construction du diagramme d'activité d'inscription pour l'utilisateur :

Figure 17: Diagramme d'activité de la classe User

Classe User :

```php
class User
{
    private $_id, $_first, $_last, $_phone, $_mail, $_password, $_rank;

    function __construct($mail,$password,$id='',$first='',$last='',$phone='',$rank=1)
    {
        $this->setMail($mail);
        $this->setPass($password);
        $this->setId($id);
        $this->setFirst($first);
        $this->setLast($last);
        $this->setPhone($phone);
        $this->setRank($rank);
    }

    // Accesseurs/Mutateurs
    public function getId(){
        return $this->_id;
    }

    public function getFirst(){
        return $this->_first;
    }

    public function getLast(){
        return $this->_last;
    }

    public function getPhone(){
        return $this->_phone;
    }

    public function getMail(){
        return $this->_mail;
    }
```

Méthode d'authentification d'un utilisateur :

```php
// Class method to authenticate a user when sign-in.
public function authenticate($bdd){
    // We check if we have any errors generated during a previous attempt and unset those error if necessary.
    if (isset($_SESSION['errors']['user_LogError']) && !empty($_SESSION['errors']['user_LogError'])){
        unset($_SESSION['errors']['user_LogError']);
    }

    // We select into the database all entries which match with the email that user has entered.
    $auth = $bdd->prepare("SELECT * FROM `users` WHERE `users_email` = :mail");
    $auth->bindValue(':mail', $this->getMail());
    $exec = $auth->execute();

    // If our query sends an error (code 400), we throw a custom exception.
    if (!$exec)
        throw new CustomException("An error occured, please check your query in User.php -> authenticate()");

    // We count our results.
    $result = $auth->rowCount();

    // If we have only one result, we try to create a session else we redirect to the login page.
    if ($result == 1) {

        $user = $auth->fetch();

        // We check if the password that user has entered match with which is into the database.
        $verifPassword = password_verify($this->getPass(), $user['users_pwd']);
        // If it's true, we create the session. Otherwise, we redirect the user to the login page.
        if($verifPassword){
            $_SESSION['user']['ID'] = $user['users_id'];
            $_SESSION['user']['MAIL'] = $user['users_email'];
            $_SESSION['user']['FIRST'] = $user['users_first'];
            $_SESSION['user']['LAST'] = $user['users_last'];
            $_SESSION['user']['PHONE'] = $user['users_phone'];
            $_SESSION['user']['RANK'] = $user['users_rank'];
            $_SESSION['user']['PASS'] = $user['users_pwd'];
            $_SESSION['user']['CONNECT'] = Date("Y-m-d-H-i-s");
            $file = '../logs/connections.txt';
            $newLine = "\nConnexion : ".$_SESSION['user']['CONNECT']." - ".$_SESSION['user']['ID']." ".$_SESSION['user']['MAIL']." Statut : OK ";
            file_put_contents($file, $newLine, FILE_APPEND | LOCK_EX);
            // file put content: utilisé à la place de file open, file write et file close
            header('Location: index.php');
        } else {
            $_SESSION['errors']['user_LogError'] = 'Nom d\'utilisateur ou mot de passe incorrect';
            header('Location: login-register.php');

        }


    } else {
        $_SESSION['errors']['user_LogError'] = 'Nom d\'utilisateur ou mot de passe incorrect';
        header('Location: login-register.php');
```

```
92  if(isset($_POST['submitLogin']) && !empty($_POST['submitLogin'])){
93
94      $email = cleanInput($_POST['users_email']);
95      $pwd = $_POST['users_pwd'];
96
97      $user = new User($email,$pwd);
98
99
100
101     try{
102
103         $user->authenticate($db);
104     }
105     catch(CustomException $e){
106         die("<br /><strong>Error ! Details: </strong><br />" . $e->getMessage());
107
108     }
109 }
```

La méthode **authenticate()** utilise en paramètre la base de données. Elle vérifie s'il y a une session « errors » en cours et dans l'affirmative, la détruit.

Une requête SQL préparée puis exécutée (afin d'éviter les **injections SQL**) va chercher toutes les informations liées au user en utilisant la méthode **getMail()** qui va prendre le mot de passe entré dans le formulaire et les retourne sous la forme d'un **array**. Si la requête rencontre une erreur, elle va afficher l'exception relative à cette erreur.

Si le résultat de la requête est égal à 1, la méthode Php **fetch()** nous retourne cette ligne . On vérifie ensuite si le mot de passe entré par l'utilisateur correspond bien au mot de passe entré dans le base de données avec la méthode **password_verify().** Elle retourne un **booléen** et vérifie que le hachage fourni corresponde bien au mot de passe. Si le résultat renvoyé est TRUE, on créée une session User avec la variable superglobale **$_SESSION** qui comprendra toutes les données rangées dans un tableau associatif.

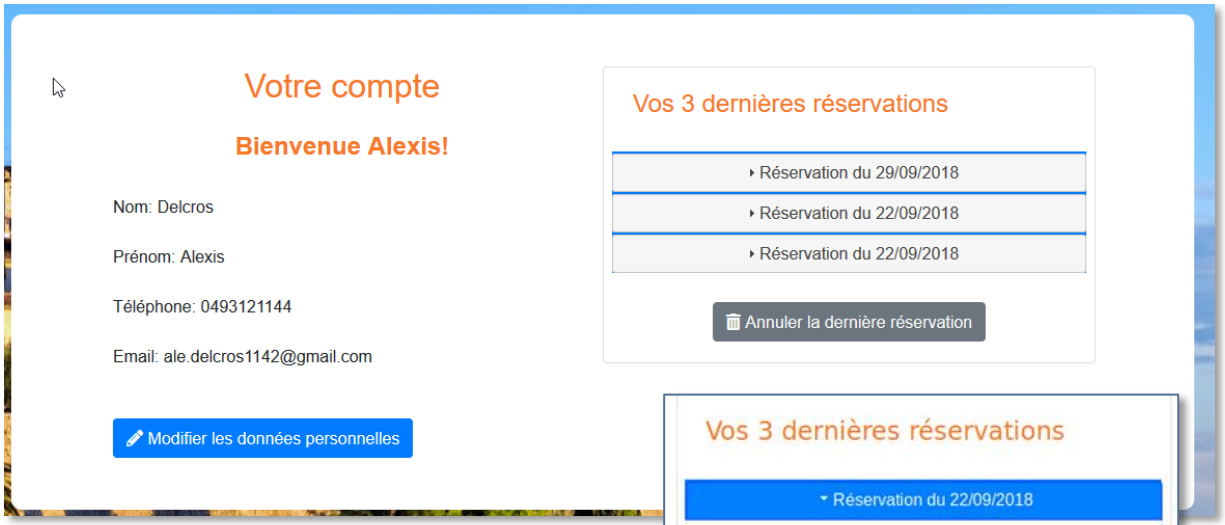
Les données de connexion sont enregistrées dans un fichier connections.txt.

Accès au compte User

L'utilisateur a accès à ses données personnelles et à ses 3 dernières réservations :

-il peut modifier ses données ,
-il peut annuler la dernière réservation.

La partie réservation est traitée de façon dynamique, j'ai utilisé la librairie **JQuery UI** pour rajouter la fonctionnalité appelée « accordion » qui permet d'ouvrir et fermer les onglets en douceur :



Les éléments indiqués dans la « card » **Bootstrap** sont générés dans le document HTML par une requête SQL.
Si l'utilisateur modifie ses données, celles-ci sont modifiées dans la **table** Users par la méthode **update()** qui contient la requête **SQL** suivante :

```
public function update($bdd){

    $updateUser = $bdd->prepare("UPDATE `users`SET `users_first`= :first,
        users_id` = :id");
    $updateUser->bindValue(':first', $this->getFirst(), PDO::PARAM_STR);
    $updateUser->bindValue(':last', $this->getLast(), PDO::PARAM_STR);
    $updateUser->bindValue(':phone', $this->getPhone(), PDO::PARAM_STR);
    $updateUser->bindValue(':email', $this->getMail(), PDO::PARAM_STR);
    $updateUser->bindValue(':rank', $this->getRank(), PDO::PARAM_INT);
    $updateUser->bindValue(':id', $this->getId(), PDO::PARAM_INT);

    $exec = $updateUser->execute();
```

Elles se réactualisent ensuite dans la page « Votre compte » grâce à la méthode **resync()** de la classe User, qui va chercher les nouvelles données dans la table Users.

1.3.6 Les fonctionnalités à venir

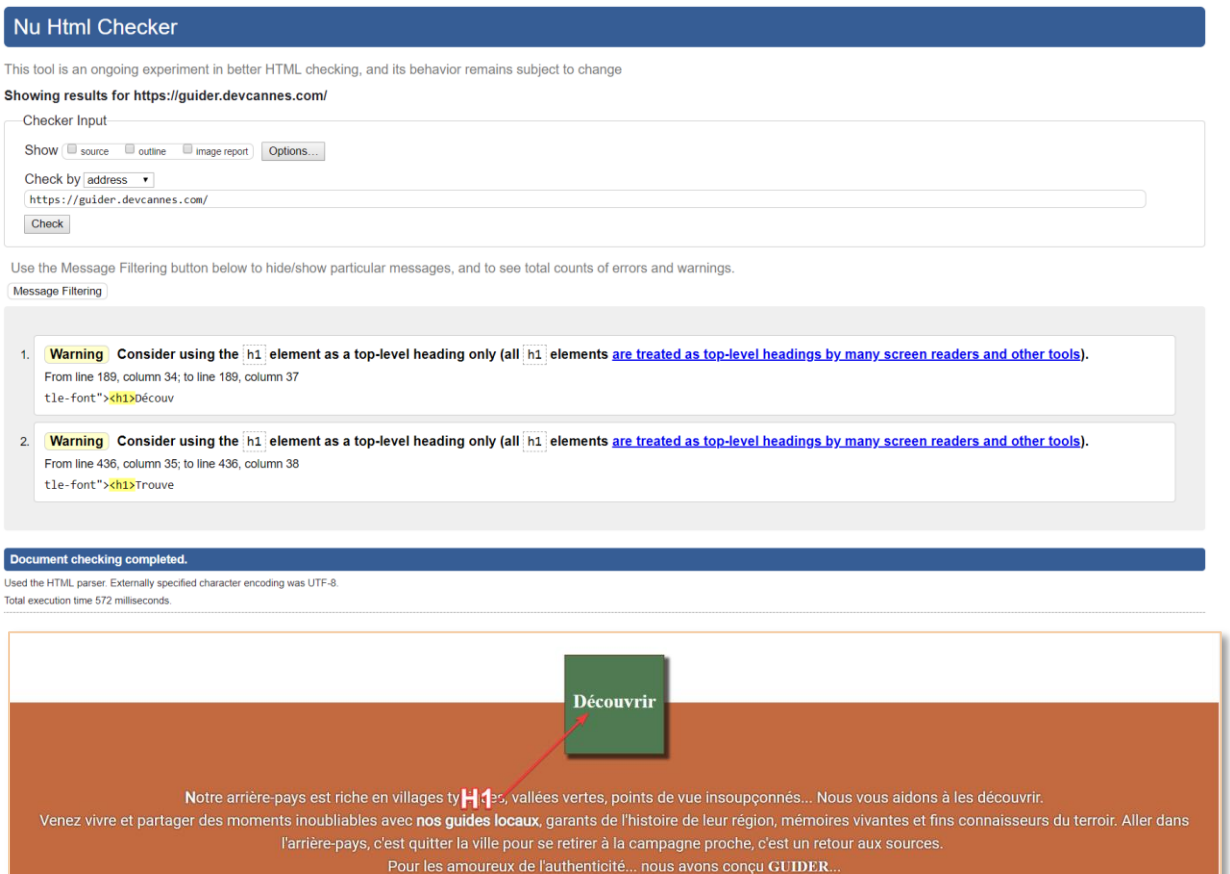
Le module d'annulation de la dernière réservation n'a pas encore été développé. Ce que nous prévoyons :

- quand l'utilisateur validera son annulation, il y aura une contrainte de traitement. Par exemple, il ne pourra annuler au maximum que 24h avant la prestation.
- si la réservation est annulable, l'utilisateur pourra valider, recevra un email de confirmation et le remboursement.
- le guide recevra également un email de confirmation d'annulation et l'annulation apparaitra dans son compte « guide » , module qui n'a également pas été développé.

Le module de consultation et gestion des réservations du guide sont aussi à développer.

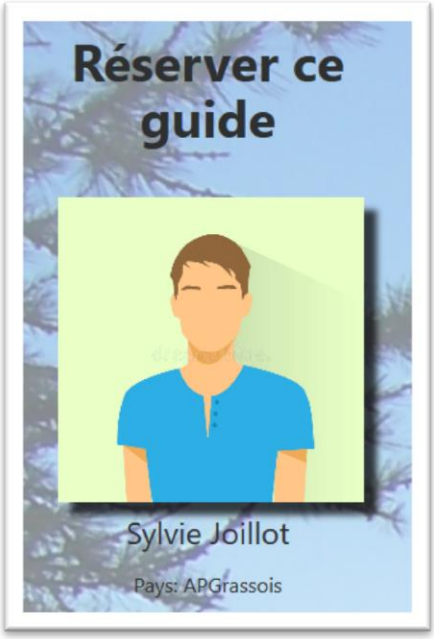
1.4 Responsivité

Le site Guider a été développé en HTML5, CSS3 (et son framework Bootstrap 4.1), Javascript ES5 (et son framework jQuery) ainsi qu'en PHP 7.1. Les conventions de nommage (camelCase) et les normes W3C ainsi que l'accessibilité ont été respectés dans la mesure du possible.
Le résultat du test de validation W3C de la page d'accueil, test réalisé sur **https://validator.w3c.org** est le suivant :



De la même façon nous avons dû faire un compromis entre le contraste des normes d'accessibilité et le design avec la couleur de fond des sections et la couleur de leur texte. Je dois encore mettre en place un système TTS (Text To Speech) afin de permettre aux malvoyants de profiter pleinement de nos services et améliorer l'accessibilité du site.

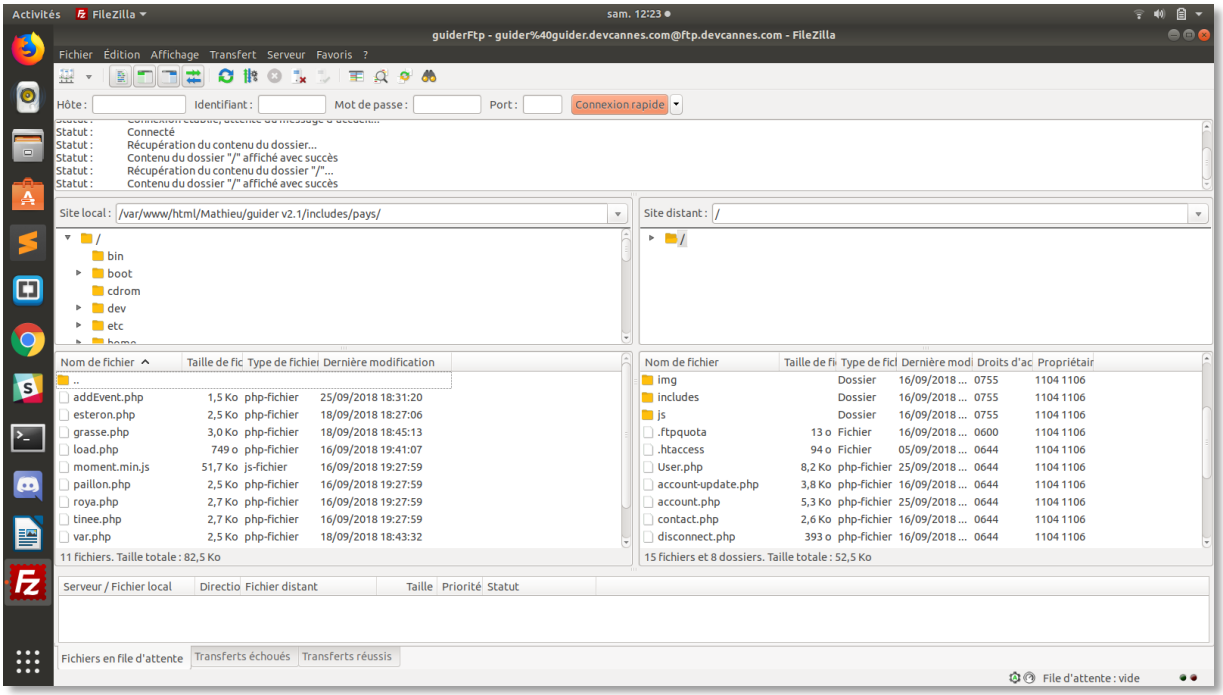
Le site est construit en Bootstrap pour la responsivité, des @Media queries ont été rajoutées quand cela a été nécessaire. Ci-dessous le rendu d'une page en format tablette et en format mobile :



1.5 Déploiement du site

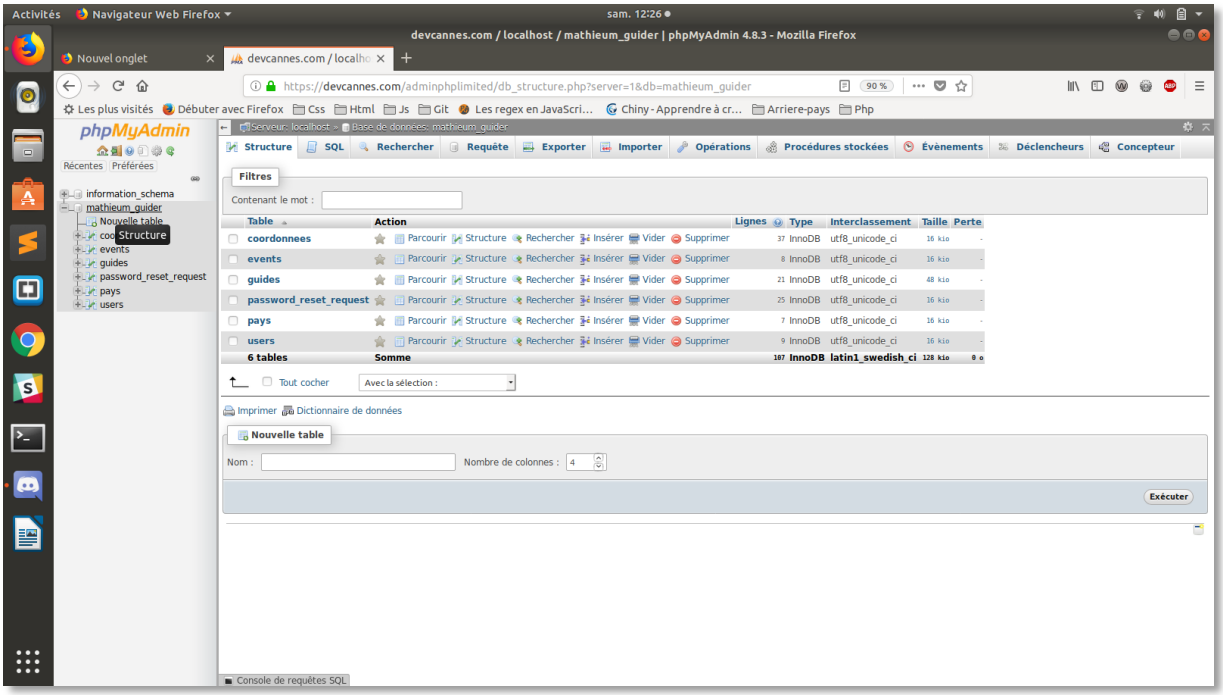
Mon collègue Mathieu a mis en ligne le site sur son hébergement chez **02switch.**

Nous avons utilisé **Filezilla** afin de migrer les fichiers chez l'hébergeur. Il m'a donné accès à la base de données qui a été créée via le Cpanel control de 02switch.

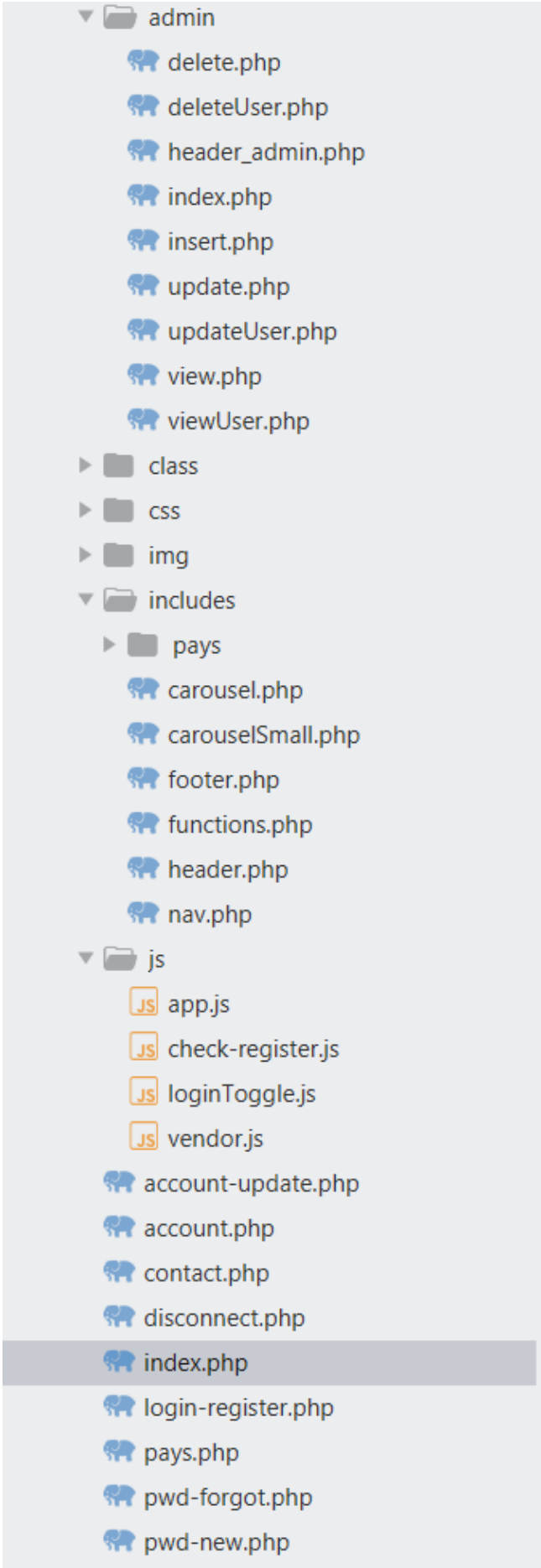


Nous avons utilisé l'outil de versioning **GIT** en local sur nos propres pc, avons pushé le code au fur et à mesure de l'avancement sur **GITHUB** afin de pouvoir récupérer le total des fichiers. Comme nous travaillions sur des fichiers séparés, nous n'avons pas créé de branche. Nous sommes restés sur la branche MASTER tout au long du développement.

Dès que nous avons mis le site en ligne, nous avons travaillé sur nos fichiers en localhost puis ensuite les avons transférés via **Filezilla**, serveur **FTP** open-source, vers le site.



Le site est visible sur https://www.guider.devcannes.com/index.php



Arborescence du site Guider