Framework Altair

> Calendrier de gestion d'évènements pour organisme de formation.
>
> - En cours de développement

Sommaire :

Prérequis {#prerequis}

> - Connexion internet
> - Connaissance Git
> - Wamp, Xamp ou Lamp
> - Un éditeur de code

Langages utilisés {#techno}

> - **HTML5, CSS3**
> - **PHP v7.3.12, MySQL v8.0.18** (tables relationnelles)
> - **Javascript et jquery** pour la partie Bootstrap
> - Material Design for Bootstrap
> - Architecture: **MVC**

Démarrage {#start}

- **composer** - installer les dépendances. `composer install`

- créer la base de donnée **framework_altair** avec le fichier sql. (avec phpMyAdmin)
- mettre à jour le chemin vers le dossier racine du fichier **config.php** pour un déploiement (local ou hébergement web ligne : 33).
- mettre à jour **PdoSql.php** (dbname, host, user, password).
- dans **php.ini** décommenter la ligne `user_agent` (pour les requètes HTTP a openstreetmap pour la localisation).

!! Infos importantes !! {#info}

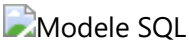
A la première utilisation la personne qui va s'enregistrer en premier sera **superAdmin** (*c'est lui qui aura accès à toute les fonctionnalités : ajout d'admin, modification des utilisateurs et des évènements, validation d'évènement, gestion des tokens pour les réseaux sociaux etc...*). Tous les futurs enregistrements seront considérés comme utilisateurs (avec la possibilité de demander à être "admin / coach"). Dans **layout.phtml** j'ai comfiguré la redirection sur `addAdmin` par défaut si aucune `$_SESSION['auth']` n'est trouvée.

- L'utilisateur est obligé de s'enregister et se logger pour avoir accès au site.

La base de donnée comprend : {#bdd}

> A la date du 28/05/2020

- La table **admin** = les utilisateurs (infos basique: nom, prénom, email, image, mot de passe, role, token, demande admin).
- La table **admininfo** = les infos communes aux users et aux admins (adresse, société, diplomes, site web, réseaux sociaux).
- La table **adminpro** = les coachs (infos sur leurs parcours, spécialités etc...).
- La table **coaching** = elle va fusionner avec la table adminpro.
- La table **custom_site** va enregistrer des paramatres de design globales pour le site (couleur du header, du footer, des titres, type de police, couleur des textes etc...).
- La table **events** = les évènements (titre, contenu, date, theme, image, est validé, adresse).
- La table **history** enregistre les connexions des utilisateurs.
- La table **interested** fait la jonction entre l'évènement et l'utilisateur si ce dernier est interréssé.
- La table **register** fait la jonction entre l'évènement est l'utilisateur qui s'est inscrit à cet évènement.
- La table **social_media** stockes les infos sensibles pour les réseaux sociaux et leur utilisation via l'API (token, clé secrète etc...).
- La table **vote** *permettra* d'enregistrer une note donnée à un coach par un utilisateur (*pas encore développé*).


Modele SQL

Utilisation des routes {#road}

> Dans le fichier **.htaccess**
>
> Cette modification est appelée réécriture d'URL. C'est un moyen de mettre en œuvre le mappage ou le routage d'URL dans une application Web. `<IfModule mod_rewrite.c> RewriteEngine On RewriteCond %{REQUEST_FILENAME} !-f RewriteCond %{REQUEST_FILENAME} !-d RewriteCond %{REQUEST_FILENAME} !-l RewriteRule ^(.+) index.php?r=$1 [QSA,L] </IfModule>`

> Dans le fichier **road.php**
>
> - On commence par le nom de la route `'route' => [`
> - Ensuite on ouvre un tableau avec le controller `'controller' => 'NomDuController',`
> - On le lie à la vue `'view' => 'nom_de_la_vue.phtml']`
> - Ca donne :

```
'route' => [
           'controller' => 'MonController',
          'view' => 'mavue.phtml'
           ]
```

> Dans le fichier **index.php**
>
> La route par défaut est `home`(*retour si l'url demandé n'existe pas*) il faut donc penser a créer cette page ou modifier les informations de redirection.

Utilisation du calendrier {#calendar}

> - Inspiré du calendrier de Grafikart

Les évènements sont gérés par le Model Events : {#events}

- Créer, Afficher, Mettre à jour, Effacer.
- S'inscrire ou s'interésser à un évènement.
- Retourne les évènements en cours de validation, le nombre d'inscrit(s) ou d'interéssé(s).

> Utilisé dans les controller :
>
> - AddEvent `App\Controller\AddEvent.php`
> - AwaitEvents `App\Controller\AwaitEvents.php`
> - MyEvents `App\Controller\MyEvents.php`
> - EditEvent `App\Controller\EditEvent.php`
> - Delete `App\Controller\Delete.php`
> - CurrentEvent `App\Controller\CurrentEvent.php`
> - Calendar `App\Controller\Calendar.php`

Le Model Event : {#event}

- Appelé par la fonction `find($id)` de la classe **Events**. `App\Model\Date\Events.php`
- getter et setter pour les élements de le table events.

> Utilisé dans les controller :
>
> - AddEvent `App\Controller\AddEvent.php`
> - EditEvent `App\Controller\EditEvent.php`

Le Model Month : {#month}

- Gére les dates en général (jour, mois, année) version fr.
- Permet de naviguer de mois en mois sur le calendrier.

> Utilisé dans le controller :
>
> - Calendar `App\Controller\Calendar.php`

Le Model EventValidator : {#validator}

- Utilise le Model **Validator** `App\Model\App\Validator.php`
- Valide les données des évènements (titre, format de la date et vérifie si la date de fin n'est pas avant la date de début).

> Utilisé dans les controller :
>
> - AddEvent `App\Controller\AddEvent.php`
> - EditEvent `App\Controller\EditEvent.php`

Le ModelMyEvents : {myEvents}

- Liste les évènements crées par l'admin ou les évènements ou l'utilisateur s'est inscrit.

> Utilisé dans les controller :
>
> - FullAccount `App\Controller\FullAccount.php`
> - MyEvents `App\Controller\MyEvents.php`
> - Search `App\Controller\Search.php`

Le ModelSearch : {#msearch}

- Outil de recherche de coach et d'évènement.
- Permet de filtrer les coachs en fonction de leur type de coaching ou de leur catégorie.

> Utilisé dans les controller :
>
> - Members `App\Controller\Members.php`
> - Search `App\Controller\Search.php`

La gestion des utilisateurs {#users}

Le ModelAdmin : {#modelAdmin}

- Créer, Afficher, Mettre à jour, Effacer.
- Gestion des infos Utilisateur, Coach.
- Liste les coachs en attentent de validation.
- Nombres de coach en attentent de validation. `config.php` `$GLOBALS['new'] = $admin->countRequestAdmin()`
- Récupère le mail pour la connexion.

> Utilisé dans les controller :

> - Account `App\Controller\Account.php`
> - AdminEdit `App\Controller\AdminEdit.php`
> - AdminPro `App\Controller\AdminPro.php`
> - Coaching `App\Controller\Coaching.php`
> - Connect `App\Controller\Connect.php`
> - FullAccount `App\Controller\FullAccount.php`
> - Members `App\Controller\Members.php`
> - Register `App\Controller\Register.php`
> - Search `App\Controller\Search.php`

Pour le MVC : {#mvc}

Création d'un controller

> Lors de la création d'un controller il faudra implémenter ces méthodes :
>
> ```
> * public function __construct(){}
> * public function httpGetRequest(){}
> * public function httpPostRequest(){}
> ```

Note: {#note}

> A savoir:
>
> - Pour le partage d'évènement il faut que le site soit hébergé afin de générer une vrai URL pour le partage (astuce on peut ouvrir le port 443 en local).
>
> ---
>
> - Projet en cours de développement.
>
> ---
>
> - Réalisé par Jean-christophe BARRET pendant mon stage chez Altaïr Communication avec l'aide de mes tuteurs de stage Benjamin Roumagne et Yoann Martinez.
>
> ---