

1. Préparation de l'environnement de développement

1.1. Choix d'IDE

La préparation de l'environnement de développement intégré est primordiale pour le développeur, cela afin d'augmenter la productivité. En effet, avant de se lancer dans le développement, j'ai préparé mon IDE, en l'occurrence PhpStorm, en installant la version 2017.1. mais avant d'installer le IDE, il est nécessaire d'installer php5.5.9 ou plus (nécessaire pour Symfony 3.4) ainsi que plusieurs modules et dépendances pour le bon fonctionnement de l'application.

Généralement on gère les dépendances moyennant un gestionnaire de version « **composer** », il permet de déclarer et d'installer les bibliothèques dont le projet principal a besoin.

Composer nous permet ainsi d'installer une application symfony tout simplement à travers la commande suivante:

```
$ composer create-project symfony/framework-standard-edition mon_projet "3.4.*"
```

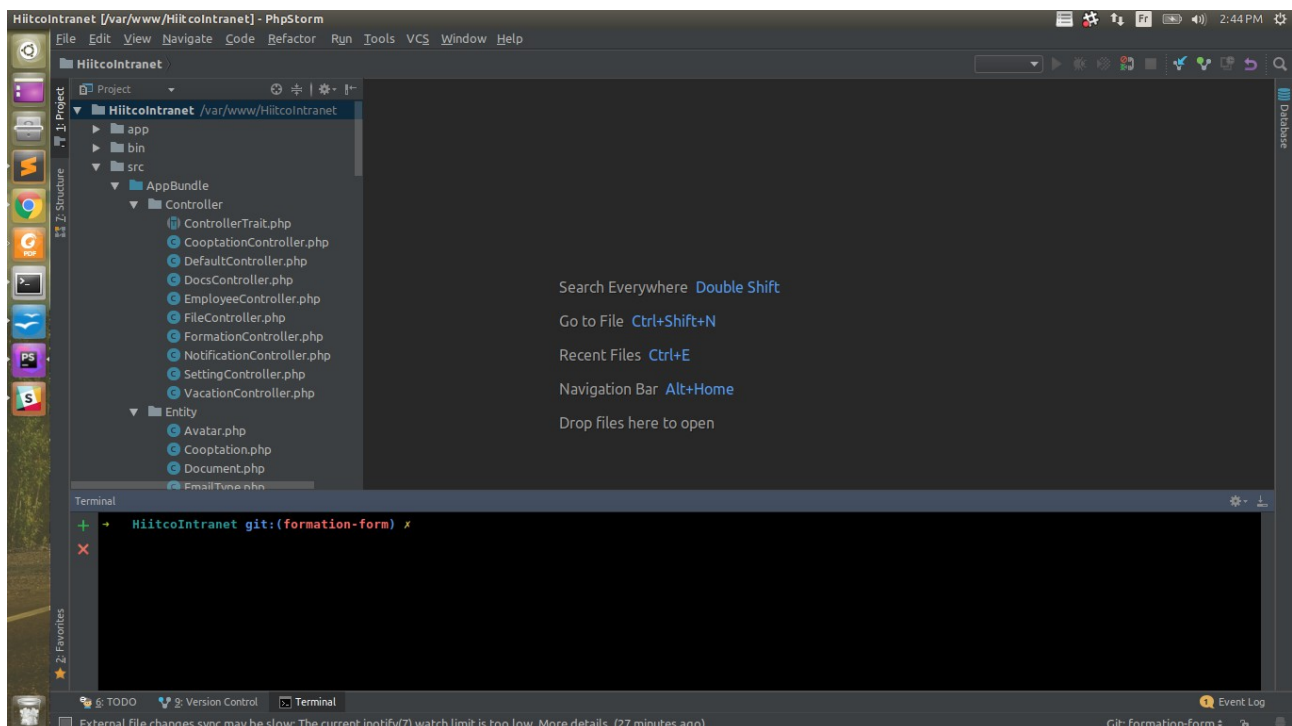


Figure 1

Pour gérer le code source, j'ai créé un repository sur mon

1.2. utilisation basic de symfony

on peut installer une application symfony avec plusieurs méthodes, la plus simple c'est d'utiliser composer, ou bien on peut utiliser Symfony installer téléchargeable à partir du site officiel :

```
$ sudo curl -Ls https://symfony.com/installer -o /usr/local/bin/symfony
$ sudo chmod a+x /usr/local/bin/symfony
```

Puis pour l'installation :

```
$ cd /var/www/
$ symfony new HiitCoIntranet "3.4.*"
```

Avec `HiitCoIntranet` est le nom du projet.

Et `3.4.*` est la version de symfony installer

Une fois installé: le projet contient les répertoires suivants

Le répertoire /bin: Ce répertoire contient tous les exécutable utilisés pendant le développement. Les exécutables sont des commandes PHP.

Le répertoire /src : C'est le répertoire dans lequel on met le code source.

Le répertoire /tests : Ce paragraphe contient tous les tests de l'application.

Le répertoire /var : Contient tout ce que Symfony va écrire durant son process : les logs, le cache, et d'autres fichiers nécessaires à son bon fonctionnement.

Le répertoire /vendor : Contient toutes les bibliothèques externes à notre application. Dans ces bibliothèques externes, et même Symfony! C'est là où résident tout les dépendances du projets, Doctrine, Twig, SwiftMailer, etc.

1.2. utilisation basic de git :

Une fois le projet est installé et avant même de commencer à écrire le code, il est bien conseillé et recommandé d'utiliser git ou un autre vcs.

On commence par initialiser un dépôt git dans le répertoire du projet :

```
$ cd /var/www/HiitCoIntranet
$ git init
```

Ensuite on indexe les fichiers et/ou répertoires qu'on veut avec la commande **`$ git add fichier`**.

En suite on enregistre les fichiers les répertoires et les modifications indexés(ées) avec la commande

dans le dépôt avec la commande :

```
$ git commit
```

github

Sur github on crée un dépôt «**HiitCoIntranet**». Puis on copie son url : «**git@github.com:amine-hiit/HiitcoIntranet.git**» ensuite on exécute la commande suivante

Puis on push le projet / les modifications sur ce répo distant

```
$ git push origin master
```

2. Description de l'interface IHM

2.1. L'interface d'authentification

L'utilisateur de l'application est directement dirigé vers la première page, celle de l'authentification. Elle offre à nos utilisateurs la possibilité de s'authentifier selon leurs profils en faisant entrer leurs noms d'utilisateur et mots de passe. C'est un passage obligatoire et il permet d'assurer la sécurité de l'application en accordant à chaque utilisateur les ressources et les droits qui lui correspondent.

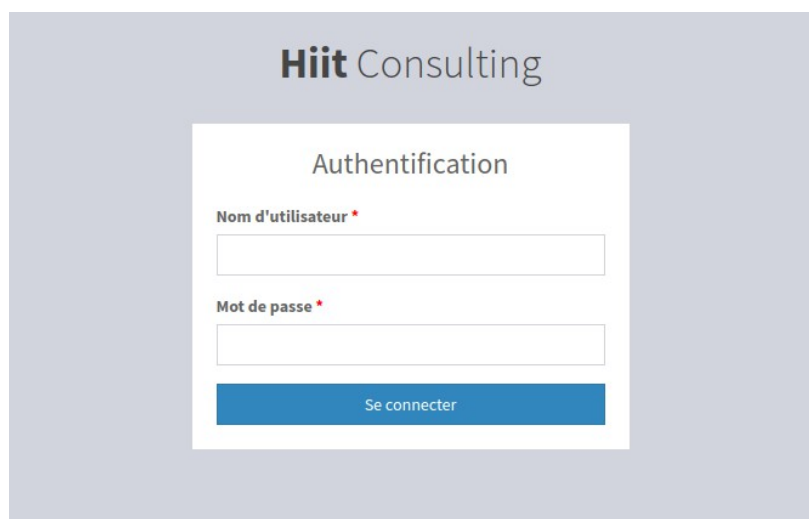
The image shows a web interface for 'Hiit Consulting'. At the top, the company name 'Hiit Consulting' is displayed in a bold, dark font. Below it, the word 'Authentification' is centered. There are two input fields: the first is labeled 'Nom d'utilisateur *' and the second is labeled 'Mot de passe *'. Both fields are empty. Below the password field is a blue button with the text 'Se connecter' in white.

Figure 2

2.2. Description de l'interface d'accueil

Une fois authentifié, l'utilisateur est redirigé vers la page d'accueil, et ensuite selon son état(valide

ou non) il aura le droit d'accéder aux autres parties de l'application ou non.



Figure 3

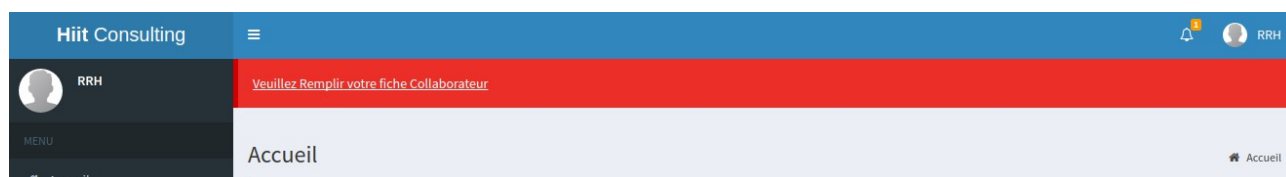


Figure 4

Si l'utilisateur est non valide, il sera redirigé après chaque tentative à accéder à une autre page à la page d'accueil contenant un lien vers la fiche collaborateur. Ce lien présent dans la page d'accueil tant que l'utilisateur n'est pas valide. (Figure ci-dessous).

La page d'accueil ainsi que toute autre page contient quatres composants:

- Navbar (statique)

Ce composent contient le nom de la société, l'accès aux notifications l'accès au profile et un bouton pour la deconnexion.



Figure 5

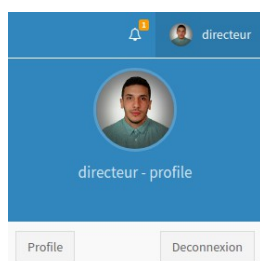


Figure 6

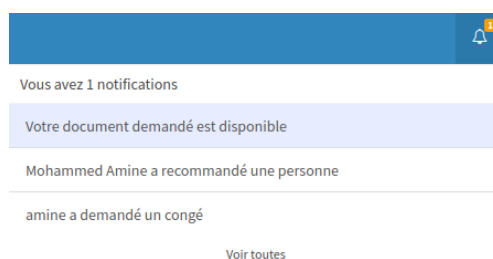


Figure 7

Concernant les notifications, on peut récupérer le nombre des notifications instantané en envoyant des requêtes ajax chaque 5 second au serveur, ce dernier cherche sur la base de données toutes les notifications non lues, et renvoie le nombre. Ensuite une fois on clique sur le bouton de notif, une deuxième requête ajax est envoyée pour récupérer les dix dernières notifications.

- Sidebar (statique)

Ce composent donne la possibilité à accéder à toute les fonctionnalités de l'application.

Selon le profile, le sidebar contient plusieurs boutons : chaque bouton redirèges vers une page ou un module dans l'application

La figure 8 montre le sidebar présenté à un utilisateur ayant le rôle d'un admin.

La figure 9 montre le sidebar présenté à un utilisateur ayant le rôle d'un responsable RH

La figure 10 montre le sidebar présenté à un utilisateur ayant le rôle d'un employée.

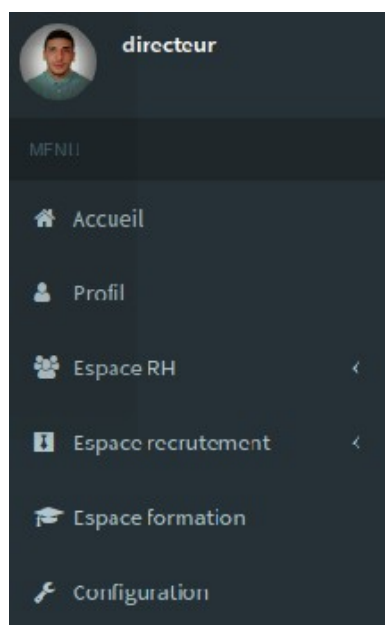


Figure 8

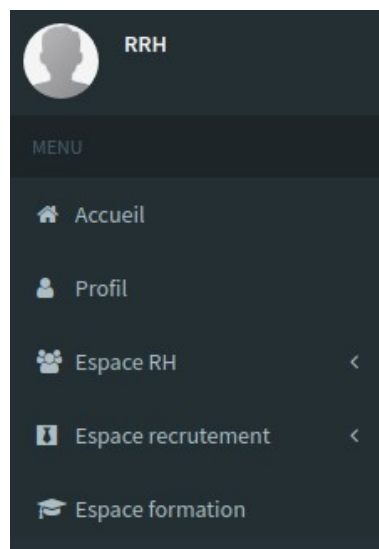


Figure 9

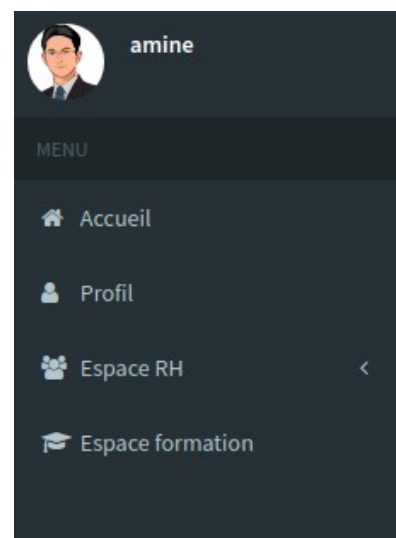


Figure 10

- Footer

Contient tout simplement les coordonnées de l'application.

Figure 11

- Contenu de la page (dynamique)

C'est dans ce composent que se présentent toutes les fonctionnalités de l'application

2.3. Description des différentes pages de l'application :