

PROJET PYTHON

Site web Check My Lawyer



30 SEPTEMBRE 2021
UNIVERSITÉ PARIS 1 :
Yasmine GMAR

Table des matières

| | |
|---------------------------------------|--------------------------|
| Présentation | 2 |
| Domaine fonctionnel | 3 |
| Mode administrateur | 3 |
| Accès Public | 3 |
| Environnement | 3 |
| Environnement de développement | 3 |
| Langages et Frameworks | 3 |
| Django | 3 |
| VENV | 3 |
| Système de gestion de base de données | 4 |
| Connexion | 4 |
| Configuration | 4 |
| Migration | 5 |
| Serveur HTTP | 5 |
| Configuration minimale recommandée | 5 |
| Architecture de la solution | 6 |
| Modèle de données | 6 |
| Template | 7 |
| Test et validation | 7 |
| Evolutions prévues | 10 |

I. Présentation

Check My Lawyer est un projet qui consiste en l'élaboration d'un site web permettant de consulter une liste d'avocats enregistrés en France et importés à partir de Google Maps. Le site permet également de rédiger des commentaires pour chaque avocat présent dans la liste.

Ce site est une plateforme collaborative qui permet aux utilisateurs d'ajouter des avocats, de les noter et de commenter leurs prestations. L'objectif est de permettre aux clients d'avocats de donner leurs retours par rapport aux prestations de leurs avocats et de donner une visibilité sur les avocats les mieux réputés. Ainsi, les avocats les plus compétents et les plus corrects bénéficieront de plus de confiance et ceux qui ne le sont pas verront leurs notes diminués.

I. Domaine fonctionnel

Il est possible de dégager deux modes d'utilisation de cette plateforme.

1. Mode administrateur

Il est accessible via login et mot de passe et permet de gérer la liste des professions juridiques de la plateforme. Ceci impliquera des traitements tels que la modération des commentaires, la saisie/consultation/ mise à jour et suppression (CRUD) des cabinets présents dans la base de données.

2. Accès Public

Ceci correspond au mode ouvert aux personnes inscrites en tant qu'utilisateurs sur la plateforme.

Cet accès permet uniquement de rechercher, consulter et modifier les fiches des avocats français et notamment y laisser des avis.

II. Environnement

Nous dégageons dans cette partie deux types d'environnements

1. Environnement de développement

C'est l'environnement dans lequel ce projet a été réalisé. Ceci correspond également à l'environnement dans lequel la maintenance sera possible.

1.1. Langages et Frameworks

Le langage imposé pour le développement de cette plateforme est Python en version 3.9 sous l'IDE PyCharm et sous le système d'exploitation Windows 10 sous sa version 21H1.

1.1.1. GIT

C'est l'utilitaire de gestion de versions le plus connu dans le monde. Il est utilisé pour sauvegarder et retrouver l'avancement de la réalisation tout au long de ce projet. GIT permet également le partage du code source avec les personnes intéressées par ce dernier.

Pour retrouver l'ensemble du projet, il suffit de cliquer sur ce lien:

<https://github.com/Lunatique122/Lawyerfinder.git> et choisir la branche master.

1.1.2. Django

Nous avons utilisé Django framework (version 3.2.5) pour réaliser la partie web du projet.

1.1.3. VENV

Venv ou bien virtualenv est un outil de virtualisation d'environnement permettant l'isolation et la portabilité de ce dernier. Il permet ainsi de gérer indépendamment les dépendances de chaque projet en les installant dans des répertoires locaux dédiés à chacun.

L'activation d'un environnement se fait à travers la commande « activate » se trouvant dans son répertoire spécifique, et la désactivation se fait via « deactivate » du même répertoire.

Ainsi, sur la console, un seul environnement est actif à un moment donné.

1.2. Système de gestion de base de données

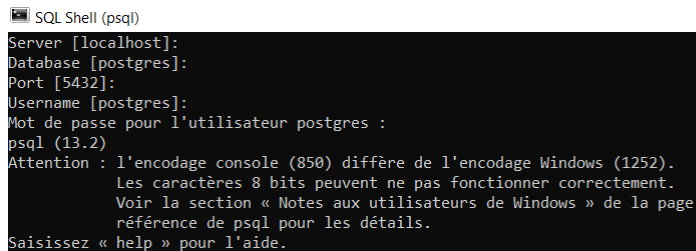
Le système de gestion de base de données choisi est PostgreSQL, choisi en tant que SGBD libre et gratuit. La version choisie est la dernière disponible pendant les développements et il s'agit de la 13.2-2.

PostgreSQL s'intègre bien à Python grâce au pilote psycopg2 permettant de traduire les requêtes SQL faites en python et de les diriger vers la base de données PostgreSQL.

Psycopg2 s'intègre automatiquement à Django.

1.2.1. Connexion

Il faut ouvrir postgres en mode terminal puis se connecter sur la base, il n'est pas nécessaire de remplir les informations demandées, il suffit de taper sur "entrer" pour chaque ligne. Quand le mot de passe est demandé, il faut taper "postgres". Voici ce que cela donne:



```
SQL Shell (psql)
Server [localhost]:
Database [postgres]:
Port [5432]:
Username [postgres]:
Mot de passe pour l'utilisateur postgres :
psql (13.2)
Attention : l'encodage console (850) diffère de l'encodage Windows (1252).
          Les caractères 8 bits peuvent ne pas fonctionner correctement.
          Voir la section « Notes aux utilisateurs de Windows » de la page
          référence de psql pour les détails.
Saisissez « help » pour l'aide.
```

Nom de la base de données : lawyerfinder

Utilisateur : postgres

Mot de passe : postgres

Par souci de simplicité on a opté pour l'utilisateur par défaut mais on souhaite renforcer encore plus la sécurité de la solution, on peut créer un autre utilisateur et lui attribuer des droits limités sur la table.

1.2.2. Configuration

Il faut ensuite importer la base de données à partir du fichier mydb.sql. en tapant la commande suivante : `psql -h hostname -p port_number -U username -f your_file.sql databasename`

C'est fait !

La base de données postgres est connectée à django grâce à cet ajout dans le fichier setting.py:

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.postgresql_psycopg2',
        'NAME': 'lawyerfinder',
        'USER': 'postgres',
        'PASSWORD': 'postgres',
        'HOST': 'localhost',
        'PORT': '5432',
    }
}
```

| yasmine, 25/08/2021 11:02 • first commit

1.2.3. Migration

La migration permet de synchroniser Django avec la base de données postgresSQL.

Pour effectuer la migration de la base de données nous avons utilisé la commande “python makemigrations” pour créer le fichier de migration et la commande “python migrate” pour appliquer les modifications sur ce fichier.

1.3. Serveur HTTP

Nous utilisons le serveur d’application Apache Tomcat v10 afin d’appeler le contenu web disponible localement sur un navigateur internet.

Le module python « mod_wsgi » permet la configuration et l’utilisation de ce serveur HTTP automatiquement via mod_wsgi-express .

2. Configuration minimale recommandée

Django prévoit un outil permettant de tester la compatibilité d’un système donné à sa framework.

La configuration système minimale communiquée est la suivante :

- RAM: 8 Go
- Carte graphique: AMD Radeon R7 A10-7850K
- CPU: Intel Core i5-4400E
- Espace libre: 3 Go
- OS: Windows 10

La configuration minimale pour un système utilisateur nécessite un simple navigateur web respectueux des standards peu importe le système d’exploitation (bureau ou mobile).

III. Architecture de la solution

III.1. Modèle de données

Ce site web contient deux modèles.

- Un modèle d'utilisateurs (users)

```
lawyerfinder=# \d+ users
```

| Colonne | Type | Collationnement | NULL-able | Table % public.users ¶ Par défaut | Stockage | Cible de statistiques | Description |
|--------------|------------------------|-----------------|-----------|---------------------------------------|----------|-----------------------|-------------|
| user_id | integer | | not null | nextval('users_user_id_seq':regclass) | plain | | |
| last_login | character varying(32) | | | | extended | | |
| username | character varying(30) | | not null | | extended | | |
| first_name | character varying(30) | | not null | | extended | | |
| last_name | character varying(30) | | not null | | extended | | |
| occupation | character varying(50) | | | | extended | | |
| biography | character varying(150) | | | | extended | | |
| website | character varying(30) | | | | extended | | |
| linkedin | character varying(30) | | | | extended | | |
| youtube | character varying(30) | | | | extended | | |
| researchgate | character varying(30) | | | | extended | | |
| twitter | character varying(30) | | | | extended | | |

```
Index :
"users_pkey" PRIMARY KEY, btree (user_id)
"users_email_0ea73cca_like" btree (email varchar_pattern_ops)
"users_email_key" UNIQUE CONSTRAINT, btree (email)
"users_username_e8658fc8_like" btree (username varchar_pattern_ops)
"users_username_key" UNIQUE CONSTRAINT, btree (username)
```

- Un modèle d'avocats (lawyers)

| Colonne | Type | Collationnement | NULL-able | Par défaut | Stockage | Cible de statistiques | Description |
|---------------|--------------------------|-----------------|-----------|------------|----------|-----------------------|-------------|
| lawyer | character varying(200) | | not null | | extended | | |
| field | character varying(200) | | not null | | extended | | |
| address | character varying(200) | | not null | | extended | | |
| city | character varying(200) | | not null | | extended | | |
| phone | integer | | not null | | plain | | |
| score | integer | | not null | | plain | | |
| creation_date | timestamp with time zone | | not null | | plain | | |
| update_date | timestamp with time zone | | not null | | plain | | |

```
Index :
"lawyers_pkey" PRIMARY KEY, btree (lawyer)
Référéncé par :
TABLE "reviews" CONSTRAINT "reviews_lawyer_id_fb951876_fk_lawyers_lawyer" FOREIGN KEY (lawyer_id) REFERENCES lawyers(lawyer) DEFERRABLE INITIALLY DEFERRED
Méthode d'accès : heap
```

- Un modèle de commentaires

```
lawyerfinder-# \d+ reviews
```

| Colonne | Type | Collationnement | NULL-able | Table % public.reviews ¶ Par défaut | Stockage | Cible de statistiques | Description |
|---------------|--------------------------|-----------------|-----------|--|----------|-----------------------|-------------|
| id | integer | | not null | nextval('reviews_id_seq'::regclass) | plain | | |
| title | character varying(200) | | not null | | extended | | |
| content | character varying(2000) | | not null | | extended | | |
| lawyer_id | character varying(200) | | not null | | extended | | |
| creation_date | timestamp with time zone | | not null | | plain | | |
| score | double precision | | not null | | plain | | |
| update_date | timestamp with time zone | | not null | | plain | | |

Index :

```
"reviews_pkey" PRIMARY KEY, btree (id)
"reviews_lawyer_id_fb951876" btree (lawyer_id)
"reviews_lawyer_id_fb951876_like" btree (lawyer_id varchar_pattern_ops)
```

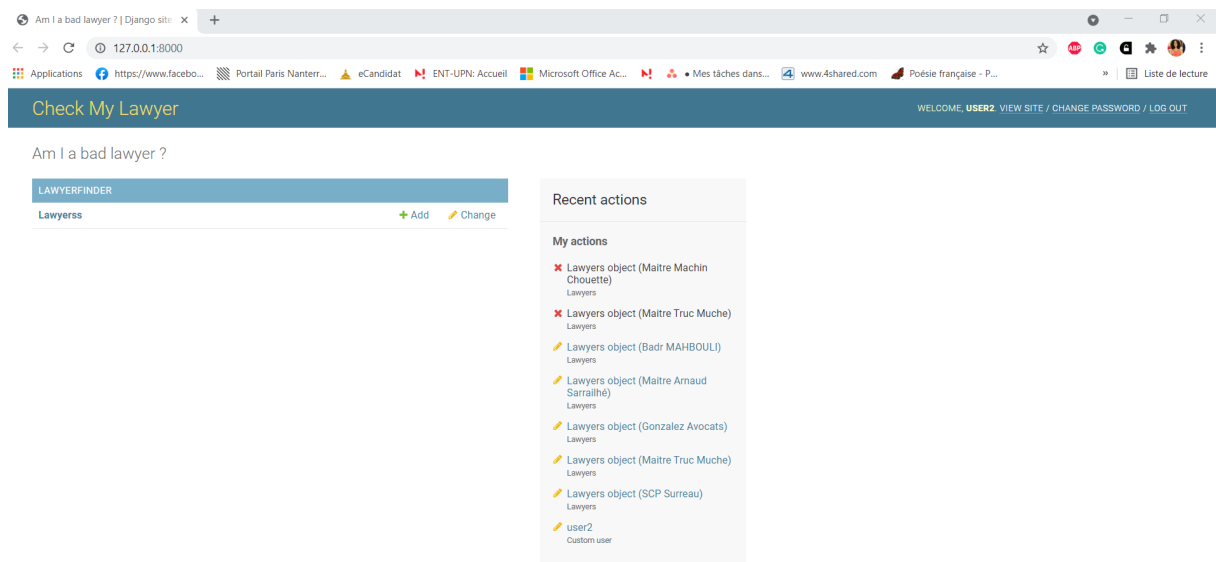
Contraintes de clés étrangères :

```
"reviews_lawyer_id_fb951876_fk_lawyers_lawyer" FOREIGN KEY (lawyer_id) REFERENCES lawyers(lawyer) DEFERRABLE INITIALLY DEFERRED
```

Méthode d'accès : heap

III.2. Template

Le template choisi est le template par défaut du framework Django. On a changé le titre et le sous-titre du template. Voici ce que ça donne:



IV. Test et validation

Partie qui explique comment lancer le projet avec des captures d'écran comme preuve de test

1- Activer l'environnement virtuel

Il faut commencer par activer l'environnement virtuel en ouvrant le projet dans pycharm et tapant dans le terminal de pycharm cette commande:

cd lawyerfinder/env/Scripts

```
Terminal: Local x +
(env) (base) C:\Users\Admin\Desktop\MIMO\Python\projet python\lawyerfinder>cd ..
(env) (base) C:\Users\Admin\Desktop\MIMO\Python\projet python>cd lawyerfinder/env/Scripts
```

puis tapez activate

```
(base) C:\Users\Admin\Desktop\MIMO\Python\projet python\lawyerfinder\env\Scripts>activate
```

l'environnement virtuel est maintenant activé.

2- lancer le site web

Pour lancer le site web il faut aller dans le répertoire apjango en tapant cd ../../ puis cd apjango

```
(env) (base) C:\Users\Admin\Desktop\MIMO\Python\projet python\lawyerfinder\env\Scripts>cd ../../
(env) (base) C:\Users\Admin\Desktop\MIMO\Python\projet python\lawyerfinder>cd apdjango
(env) (base) C:\Users\Admin\Desktop\MIMO\Python\projet python\lawyerfinder\apdjango>
```

Ensuite il suffit de taper python manage.py runserver

```
(env) (base) C:\Users\Admin\Desktop\MIMO\Python\projet python\lawyerfinder\apdjango>python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...
```

Le terminal affichera ensuite le lien HTTPS permettant d'ouvrir le site, il suffit de cliquer dessus.

```
System check identified 1 issue (0 silenced).
August 31, 2021 - 17:15:25
Django version 3.2.5, using settings 'apdjango.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

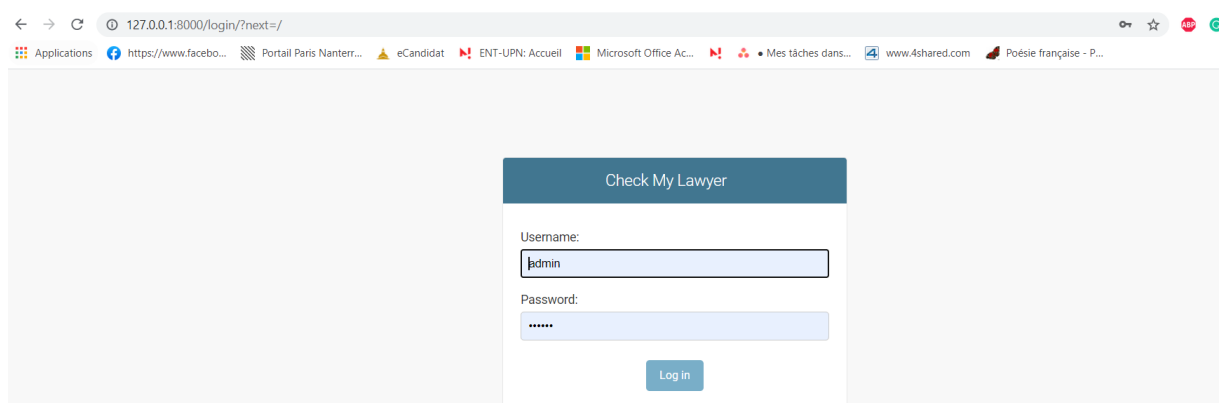
3- Se connecter au site

Une fois le site ouvert, on peut se connecter soit en mode admin soit en mode utilisateur.

- **en mode admin:** pour gérer les utilisateurs (ajouter, modifier, supprimer un utilisateur) et la base de données des avocats (ajouter, supprimer, modifier un avocat)

Les informations de connexion sont:

- le login : admin
- le mot de passe: azerty



- **en mode utilisateur:** pour ajouter/supprimer un avocat ou ajouter des commentaires sur un avocat

Les informations de connexion sont:

- le login : user2
- le mot de passe: 123456

V. Evolutions prévues

Nous avons souhaité intégrer l'API de Google Maps pour afficher l'adresse des différents avocats sur la carte de Google Maps ainsi que leurs notes. Nous n'avons pas pu le faire faute de temps. Cette évolution est envisagée pour l'avenir.

Nous avons également souhaité créer des vues spécifiques mais là aussi, au regard du temps et de la complexité nous avons préféré adapter le template par défaut généré par le framework Django. Des vues développées avec VueJs seront implémentées dans les évolutions à venir.