

# Application de Base de Données Relationnelles

## Guide de Maintenance

### Application el ligne :

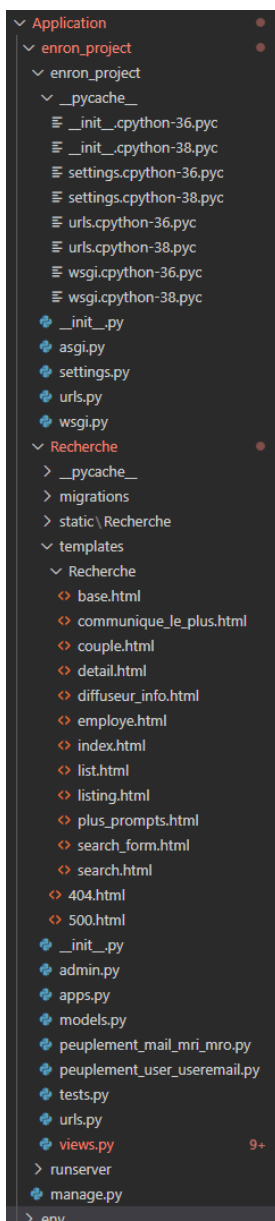
L'application peut-être utilisée en ligne ou en local. L'utilisation en ligne se fait via l'URL 'txzm1820.odns.fr/BDDR/Recherche', l'utilisation en local est décrite ci-après.

### Présentation et arborescence :

Cette application web utilise Django 3.1 lié à une base de donnée (BDD) PostgreSQL à l'Université d'Angers (savennieres.math.univ-angers.fr). Elle se nomme Recherche, l'arborescence de fichiers enron\_project est decrite ci-après. Les principaux paramètres pré-réglés se trouvent dans le fichier enron\_project/settings.py .

N.B. : Un env pour Visual Studio Code a été installé permettant de faciliter la gestion/le lancement de l'application.

### Arborescence des fichiers de code :



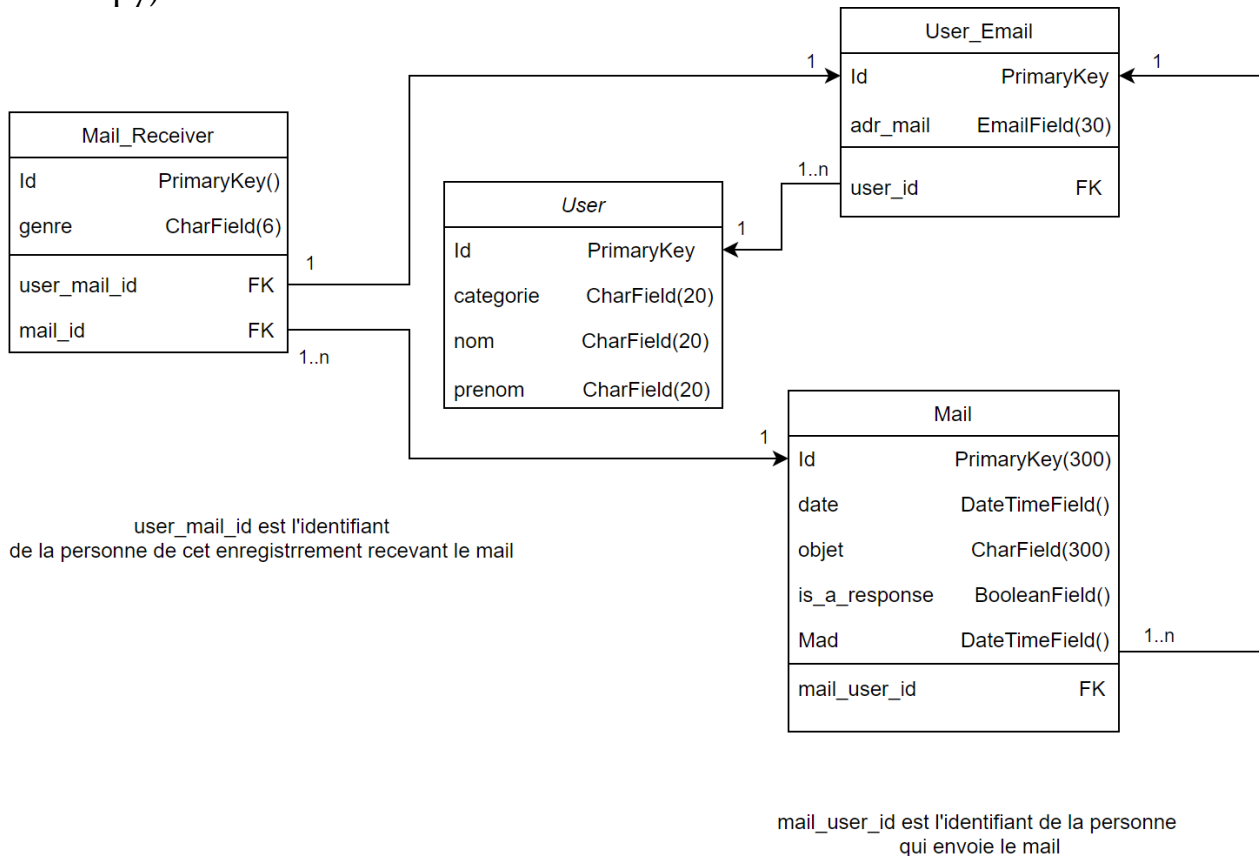
Fichiers de paramétrisation générale, lien avec la base de données, URLs générales de l'application, etc...

Fichiers HTML/CSS d'affichage des informations (utilisation de templates Django)

Scripts Python de configuration et peuplement de la base de donnée, d'utilisation des scripts HTML précédents (ces scripts seront décrits dans les pages suivantes), gestion des URLs, etc...

## La base de donnée :

La BDD est sous PostgreSQL hébergée sous le compte de Quentin Geindreau à l'Université d'Angers. Le schéma de la base de donnée (initialisée dans le script models.py) :



Chaque utilisateur est renseigné à la base dans un fichier XML, et sera enregistré ici dans les tables User, qui enregistre son nom/sa catégorie, et User\_Email qui lui relie son/ses adresse(s) mail.

Les informations de chaque mail sont enregistrées dans la table Mail.

N.B. : Mad est la date du mail en cas de réponse.

Enfin pour chaque mail, chaque personne ayant reçu ce mail lui est liée via ForeignKey dans Mail\_Receiver.

N.B. : genre renseigne sur le type de réception (CC, CCI, envoi direct).

Les scripts de peuplement se trouvent dans Application/enron\_project/Recherche. Il y en a 2 :

un premier, 'peuplement\_user\_useremail.py' qui peuple les tables User et User\_Email grâce au fichier XML.

Un deuxième, 'peuplement\_mail\_mri\_mro.py', peuplant le reste.

## Lancement de l'application :

Avant tout il faut reconfigurer la BDD à attaquer afin de faire fonctionner l'application par rapport aux paramètres du fichier GIT :

dans enron\_project/settings.py remplacer tout le DATABASES={} par :

```

DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.postgresql', # on utilise l'adaptateur postgresql
        'NAME': 'qgeindreau', # le nom de notre base de donnees creee precedemment
        'USER': 'qgeindreau', # attention : remplacez par votre nom d'utilisateur
        'PASSWORD': '455 521',
        'HOST': 'savennieres.math.univ-angers.fr',
        'PORT': '',
    }
}

```

Le paramétrage devrait alors être bon.

Le plus simple est ensuite d'utiliser l'env sous Visual Studio Code :  
se placer dans le dossier .../ProjetBDDR-main/Application/  
lancer « ./env/bin/activate »

L'env est alors activé, il suffit maintenant de se placer dans le dossier  
'enron\_project' puis lancer la fonction runserver de manage.py en renseignant  
l'adresse IP et le numéro de port sur lesquels lancer l'application (par exemple  
./manage.py runserver 0.0.0.0:8000 )

L'application est alors lancée et il suffit de taper sur un navigateur web l'URL  
localhost:8000/Recherche pour arriver sur la page d'accueil (renseigner l'adresse IP  
paramétrée).

### Gestion des URLs :

Chaque URL possible est nommée dans Recherche/urls.py et sa  
fonctionnalité définie dans views.py. Ces URLs mènent à des pages HTML décrites  
ci-dessous. En dehors de la pge d'accueil il existe les URLs :

```

0.0.0.0:8000/Recherche/search
                        /couple
                        /employe
                        /jour
                        /<user_id>    avec l'identifiant d'un employé renseigné

```

L'utilité de chacune de ces URLs est décrite dans le manuel utilisateur. Des  
paramètres sont passés à ces fichiers HTML, ces paramètres sont calculés dans  
views.py et envoyés aux page HTML via la fonction render.

### Affichage :

Chaque page associée aux URLs ci-dessus sont définies en HTML/CSS sur la  
base d'un template 'Recherche/base.html' définissant principalement le fond, le  
header, le footer et le style. Les fiches CSS associées sont dans  
Recherche/static/Recherche/css.

## Modules principaux :

Les principaux modules à installer pour Python se trouvent en premières lignes des fichiers `urls.py`, `views.py` et `Auxiliaire.py` du dossier Recherche, en voici une liste :

`django`, `pandas`, `numpy`, `json`, `matplotlib`, `email.parser`, `dateutil.parser`, `io`, `igraph`, `plotly`.

D'autres modules sont à installer si vous souhaitez relancer les scripts de peuplement de la BDD (`xml.etree.ElementTree`, `os`, `may`, `datetime`,...)