Mis a jour le 2023-01-09, 16:29

- django.db.backends.postgresql_psycopg2
- django.db.backends.mysql
- django.db.backends.oracle

et dans la variable NAME, on met le nom de la base (ou le fichier si sqlite). Il faut aussi USER, PASSWORD et HOST si ce n'est pas sqlite.

- INSTALLED_APPS : contient la liste des applications installées, y compris les applications systèmes comme django.contrib.admin qui permet de faire l'administration. Certaines appli nécessitent des tables dans la base, et pour les installer, il faut faire : `manage.py migrate` (il regarde les applications installées, INSTALLED_APPS, et crée les tables en conséquence).
- TEMPLATE_DIRS : une liste de directories contenant des templates. On peut par exemple créer un directory `templates` sous le directory projet, et le rajouter dans TEMPLATE_DIRS. Django recherche aussi les templates par défaut dans un sous-directory templates du directory du package.
- mettre la variable TIME_ZONE à la bonne valeur, par exemple : `TIME_ZONE = 'Europe/Paris'` pour la france.

* `manage.py runserver` :

- ça démarre un petit serveur écrit en python uniquement pour les tests. Attention, ce n'est pas un serveur pour la production (pas testé pour la sécurité !)
- le serveur écoute par défaut sur le port 8000 (sinon, faire `manage.py runserver 4000` pour écouter sur le port 4000.
- par défaut, le serveur est lancé pour l'adresse IP 127.0.0.1, donc accessible seulement en local. Si on veut y accéder depuis une autre machine : `manage.py runserver 0.0.0.0` ou `manage.py runserver 192.168.2.10` (avec l'adresse IP de la machine).
- pour lancer sur un autre port et y accéder depuis une autre machine : `manage.py runserver 0.0.0.0:4000`.
- avec ce serveur, le code est reloadé à chaque requête, mais si on rajoute des nouveaux fichiers, il faut relancer le serveur.

* Création d'applications :

- `manage.py startapp myApp` (à l'endroit où se trouve manage.py) : pour créer l'application myApp.
- un directory myApp est alors créé au même niveau que manage.py.
- myApp contient :
  - __init__.py, comme tout package.
  - admin.py
  - apps.py
  - migrations qui contient les fichiers de migration lors des mises à jour du modèle.
  - models.py qui contient les modèles, c'est à dire les classes correspondant aux tables qui seront créées dans la base de données (c'est Django qui s'occupe de créer les tables).
  - tests.py
  - views.py

* Création de la première vue :

- rajouter dans le fichier urls.py principal du projet : `from django.conf.urls import include, path` et dans la liste urlpatterns, rajouter l'élément : `path('myPath/', include('myApp.urls'))` : myPath/ est le chemin qui sera reconnu et retiré de l'URL avant de l'adresser à l'application myApp
- créer un fichier urls.py dans myApp et y mettre par exemple :

```
from django.conf.urls import path
from . import views
urlpatterns = [path('myTest', views.hello)]
```

  - le 1er argument dans path est la suite de l'url
  - le 2ème argument est le nom de la fonction à appeler dans l'application (à mettre dans le fichier views.py)
- dans le fichier views.py de l'application, rajouter :

```
from django.http import HttpResponse
def hello(httpRequest):
    return HttpResponse('bonjour')
```


- l'application sera alors disponible à l'adresse : `http://myServer/myPath/myTest`