1. Installation de Django - Django Wednesdays

⚠️ Installer en amont la dernière version de Python et Django sur leur site respectif.

Installation de l'environnement de travail.

1. J'installe Git Bash : https://git-scm.com/download/win et ensuite j'ouvre le logiciel.
2. Dans Git Bash, j'entre la commande "**pwd**" : Present Working Directory. Celle-ci permet de savoir où est situé le fichier actuel.
3. Ensuite je crée un "***directory***" : Un dossier de mon projet en utilisant la commande "**mkdir**" : Make a Directory et je nomme le projet.

```
mkdir /c/mongroupe
```

4. Puis j'utilise la commande "**cd**" : Change Directory afin d'exécuter du code dans le bon dossier.
5. Maintenant il faut créer un environnement virtuel. Pour cela j'entre la commande "**python -m venv virt**". Pour vérifier si l'environnement virtuel est bien installé, j'utilise la commande " **ls**" : Lists. Cette commande listera tous ce qui contient dans ce dossier.
6. Pour naviguer dans l'environnement virtuel, il faut l'activer. Pour se faire j'utilise cette commande "**source virt/Scripts/activate**". Pour le désactiver, j'utilise simplement "**desactivate**".
7. Pour finalement installer Django, j'utilise la commande "**pip install**" : Install a Python Package et je l'écris comme ceci "**pip install django**". Le "**pip install**" installe toujours la **dernière version** de la librairie.
8. "**pip freeze**" permet de lister les différentes librairies installées sur l'environnement virtuel.

Création du projet.

1. J'utilise la commande "**django-admin.py startprojet monclub**" ou "**django-admin startprojet monclub**" pour créer mon projet.

⚠️ Toujours vérifier avec la commande "**ls**" si tout a bien été créé.

2. J'utilise la commande "**cd monclub/**" afin de naviguer dans ce dossier.
3. Maintenant je souhaite savoir si mon projet est lisible sur internet. Pour se faire j'utilise la commande "**python manage.py runserver**". J'ai ensuite un petit message "*Watching for file changes with StatReloader*" et sur le navigateur web de mon choix (Google Chrome), j'entre l'url "localhost:8000".

🙌🏽 Il est tant d'insérer des lignes de codes pour donner vie au projet.

4. J’ouvre le dossier “**c/monclub**” dans mon IDE (Atom).
5. J’utilise la commande “**python manage.py startapp events**” pour créer une application (c’est plutôt une partie du site). Ça a créer un nouveau dossier dans mon projet.
6. Ensuite, je l’ajoute au fichier “settings.py” du dossier “events” dans la rubrique “**INSTALLED_APPS**” de cette façon “ **‘events’ ,** ” et je sauvegarde.
7. Je crée l’url de la page **events** donc je vais dans le fichier urls.py, j’ajoute **include** dans l’en-tête “**from django.urls import path, include**” puis je crée l’url avec le **path**.

```
from django.contrib import admin
from django.urls import path, include

urlpatterns = [
    path ('admin/', admin.site.urls),
    path ('', include('events.urls')),
]
```

💡 **path** : Renvoie un élément à inclure dans *urlpatterns*. Il est définit comme ceci :

path (route, view, kwargs=None, name=None)

- route : Le nom de domaine du site ou le chemin préalable de la page.
- views : La fonction lié à cet url dans le dossier views.py.
- kwargs : Le paramètre qui permet de créer une fonction qui accepte un nombre indéfini d'arguments nommés.
- name : Le nom donner à cet url.

8. Il faut créer le fichier events.urls dans le dossier events. Je le remplie de cette façon.

```
from django.urls import path
from . import views

urlpatterns = [
    path ('', views.home, name="home"),
]
```

9. Il faut créer la fonction “**home**” de views.home que j’ai indiqué dans le **path**.

```
from django.shortcuts import render

def home(request):
    return render (request, 'home.html', {})
```

💡 def : Définir.

Par exemple : je définis la requête de **home** que je souhaite passer dans **request** puis je retourne la requête de **home** avec **render** qui inclut request et template_name (la page html dans le dossier **template.py**).

- **request** : L'objet requête utilisé pour générer la réponse.
- **render**
  (request, template_name, context=None, content_type=None, status=None, using=None)

  Combine un gabarit donné avec un dictionnaire contexte donné et renvoie un objet **HttpResponse** avec le texte résultant.
- **template_name** (la fonction charge le gabarit ayant le nom donné et renvoie un objet template.

10. Je crée un dossier "template" dans le dossier "monclub".
11. Dans celui-ci, je crée le fichier "home.html".

```
<h1>Hello world !</h1>
```

12. Sur Git Bash, j'utilise la commande "**python manage.py runserver**" et lorsque je vais sur mon navigateur web en rafraichissant la page, je trouve "Hello world !".