rendu (Creating Forms The Right Way): https://simpleisbetterthancomplex.com/series/2017/10/02/a-complete-beginners-guide-to-django-part-5.html#reply-post-view

Serie youtube django security : https://www.youtube.com/watch?v=JsyySe17gjE

Tutoriels OC NoSQL : https://openclassrooms.com/fr/courses/4462426-maitrisez-les-bases-de-donnees-nosql

Tutoriel NoSQL (alternative) : https://beginnersbook.com/2017/09/introduction-to-nosql/

Tutoriel listes python : https://openclassrooms.com/fr/courses/1206331-utilisation-avancee-des-listes-en-python

Tutoriel améiliorer son python : https://openclassrooms.com/fr/courses/4425111-perfectionnez-vous-en-python

Tutoriel POO python : https://openclassrooms.com/fr/courses/4302126-decouvrez-la-programmation-orientee-objet-avec-python

Lire livre sur python

open source wysiwyg jodit : https://xdsoft.net/jodit/play.html

open source wysiwyg summernote : https://summernote.org/

open source wysiwyg ckeditor : https://ckeditor.com/

autres : https://techwiser.com/best-wysiwyg-html-editor-open-source/

# Powershell

##### Powershell admin n’ouvre pas

go directly to the PowerShell exe and try to run that as an administrator:

C:\Windows\System32\WindowsPowerShell\v1.0\Powershell.exe

##### Autoriser l’exécution de scripts (ps1)

Set-ExecutionPolicy RemoteSigned

voir: https://www.windowscentral.com/how-create-and-run-your-first-powershell-script-file-windows-10

# Postgre sql

##### Installation

installation : https://www.enterprisedb.com/downloads/postgres-postgresql-downloads

Télécharger et lancer l’installateur. Choisir le port 2345 pour éviter les conflits. Tapper système dans le menu démarrer et aller à advanced system settings/variables d’environnements et sélectionner la variable path dans les variables du système. la modifier et lui ajouter les directory :

C:\Program Files\PostgreSQL\12\bin

C:\Program Files\PostgreSQL\12\bin

##### Lancer serveur/Arrêter serveur/Redémarrer serveur

|  |
| --- |
| pg\_ctl -D "C:\Program Files\PostgreSQL\12\data" start |
| pg\_ctl -D "C:\Program Files\PostgreSQL\12\data" stop |
| pg\_ctl -D "C:\Program Files\PostgreSQL\12\data" restart |

Si une erreur se fait et qu’on a un tip qui demande si le port 5432 ou autre est déjà utilisé, changer le numéro du port

##### Changer numéro du port

Voir: https://www.jamescoyle.net/how-to/3019-how-to-change-the-listening-port-for-postgresql-database

# Python

Vient avec le serveur SQLite

##### Ajouter environnement virtuel dans pycharm

https://www.jetbrains.com/help/pycharm/creating-virtual-environment.html

##### Shell

Quand on lance du code python dans powershell ou bash ou etc., on peut appeler le document voulu en ajoutant shell

python manage.py shell

Ce qui permet d’ajouter le sys.path et d’importer toutes les dépendances nécessaires.

##### \_\_str\_\_/\_\_repr\_\_

Méthode embed de python qui décrit comment python affiche les objets dans la console quand on demande leur affichage (si on retourne <MonObjet: NomObjet object> ou autre chose comme son nom, etc)

|  |
| --- |
| #Par exemple, pour retourner le nom de l’objet ajouter la méthode avec les autres de l’objet  class MyClass  …  def \_\_str\_\_(self):  return self.name |

Le \_\_repr\_\_ fait la même chose.

##### @

Prend la fonction et la modifier en la faisant passer dans la fonction/méthode/classe du décorateur en paramètre avant d’effectuer la fonction proprement dite.

|  |
| --- |
| class Pizza(object):  def \_\_init\_\_(self):  self.toppings = []  def \_\_call\_\_(self, topping):  # When using '@instance\_of\_pizza' before a function definition  # the function gets passed onto 'topping'.  self.toppings.append(topping())  def \_\_repr\_\_(self):  return str(self.toppings)  pizza = Pizza()  @pizza  def cheese():  return 'cheese'  @pizza  def sauce():  return 'sauce'  print pizza  # ['cheese', 'sauce'] |

# Django

##### Importer fichier

Chaque app de Django est en fait une librairie et donc on peut demander d’importer un fichier d’une app comme tel:

from boards import views #où board est le dossier de la librairie et views.py le fichier

et pas from ..boards import views.py

Si pycharm ne reconnait pas l’appel, c’est que le dossier mère du projet django doit être marqué comme source. Pour se faire, aller dans

File -> Settings -> Project: MyProjectName -> Project structure

et ajouter en source le dossier mère (e.g. celui à la même hauteur que le .idea)

##### lancer serveur

python manage.py runserver

##### Create app

django-admin startapp MyApp

puis ajouter le nom de l’app dans les installed\_app du fichier settings.py

##### Primary key

Django génère une PK automatiquement si on ne la spécifie pas.

##### Foreign Key

models.foreignKey(FKName, related\_name=’NewName’, on\_delete = models.CASCADE)

si on met pas le related\_name, le nom donné à la colonne sera le FKName + ‘\_set’.

##### Creation des tables

Instancier le code sql correspondant à nos requêtes Django

python manage.py makemigrations

Pour voir le code SQL que Django va executer pour créer les tables, dans powershell, faire:

python manage.py sqlmigrate MyApp 0001

Où 0001 est le numéro de version de notre app qui se trouve dans son dossier migrations.

Lancer le code SQL créé

python manage.py migrate

##### Ajouter une ligne dans table/MAJ

Le save est utilisé autant pour la creation que pour la mise-à-jours

maLigne = ClasseTableauDB(attribut1=’’, attribut2=’’, …) et maLigne.save()

ou juste

maLigne = ClasseTableauDB.objects.create(attribut1=’’, attribut2=’’, …)

##### Afficher lignes

ClasseTableauDB.object.all()

##### Sélection ligne

ClasseTableauDB.objects.get(colonnePK = 'valeurePK')

##### Accéder colonne d’une ligne/MAJ

maligne.macolonne

maligne.macolonne = nouvelle\_valeure

##### Regex

A regular expression for matching URL patterns in strings. Note that these regular expressions do not search GET or POST parameters. In a request to http://127.0.0.1:8000/boards/?page=2 only /boards/ will be processed

##### Mails

Pour simuler l’envoie de mails pour faire la page de reset password, on peut utiliser django en ajoutant dans les settings

EMAIL\_BACKEND = 'django.core.mail.backends.console.EmailBackend'

## Admin

##### Créer un admin

python manage.py createsuperuser

##### Ajouter une app

Dans le fichier admin de l’app ajouter :

from .models import Board

admin.site.register(Board)

## Forms

GET is used to retrieve data from the server.

POST is used when we want to change data on the server.

Django protects all POST requests using a CSRF Token (Cross-Site Request Forgery Token). C’est un tag qu’on met au début de la form

<form method="post">

{% csrf\_token %}

…

et qui sera remplacé par

<input type="hidden" name="csrfmiddlewaretoken" value="jG2o6aWj65YGaqzCpl0TYTg5jn6SctjzRZ9KmluifVx0IVaxlwh97YarZKs54Y32">

##### Créer un formulaire

|  |  |  |
| --- | --- | --- |
| Forms | Views | Template |
| #On crée la classe Form  class NewTopicForm(forms.ModelForm):  message = forms.CharField(widget=  forms.Textarea(), max\_length=4000)  class Meta:  model = Topic  fields = ['subject', 'message'] | #On utilise la classe form pour sauver les données entrées dans la BD  if request.method == 'POST':  form = NewTopicForm(request.POST)  if form.is\_valid():  topic = form.save(commit=False)  topic.board = board  topic.starter = user  topic.save()  post = Post.objects.create(  message=form.cleaned\_data.get('message'),  topic=topic,  created\_by=user  )  return redirect('board\_topics', pk=board.pk)  else:  form = NewTopicForm() | #On crée le formulaire avec django  <form method="post">  {% csrf\_token %}  <div class="form-group">  <label for="id\_subject">Subject</label>  <input type="text" class="form-control" id="id\_subject" name="subject">  </div>  <div class="form-group">  <label for="id\_message">Message</label>  <textarea class="form-control" id="id\_message" name="message" rows="5"></textarea>  </div>  <button type="submit" class="btn btn-success">Post</button>  </form> |
|  |  | ou juste  {% block content %}  <form method="post">  {% csrf\_token %}  {{ form.as\_p }}  <button type="submit" class="btn btn-success">Post</button>  </form>  {% endblock %} |

## Template

##### Ajouter custom template

Dans le fichier settings.py de MyProject, ajouter

|  |
| --- |
| os.path.join(BASE\_DIR, 'myTemplatesDirName') |
| À l’emplacement de :  TEMPLATES = [  {  'BACKEND': 'django.template.backends.django.DjangoTemplates',  'DIRS': [  os.path.join(BASE\_DIR, 'templates')  ], |

##### template filter

Le template réagit en appelant une classe python qui va lui renvoyé un string ou autre à afficher selon si la condition est respectée ou non.

|  |  |
| --- | --- |
| Template\_filter.py | template.html |
| from django import template  register = template.Library()  @register.filter  def field\_type(bound\_field):  return bound\_field.field.widget.\_\_class\_\_.\_\_name\_\_ | {% load template\_filter %}  {{ form.username|field\_type }}  #va retourner : 'TextInput' |

Exemple plus complexe qui varie selon ce qui est entré dans le formulaire : https://simpleisbetterthancomplex.com/series/2017/09/25/a-complete-beginners-guide-to-django-part-4.html#creating-custom-template-tags

##### Render/Url/Vue

On prend une vue et on la render avec un template sélectionné. On écrit

return render(request, , 'templatePath.html', {, 'arg1' : value})

|  |  |  |
| --- | --- | --- |
| urls.py | Views.py | homeTemplate.html |
| urlpatterns = [  path('', views.home, name='home'),  …  ] | def home(request):  boards = Board.objects.all()  return render(request, 'home.html', {'boards': boards})  #Ou les objet/ligne de la table boards ont un attribut name | <body>  …  {% for board in boards %}  {{ board.name }}<br>  { % endfor %}  … |

##### Ajouter url

<a href="{% url 'board\_topics' board.pk %}">{{ board.name }}</a>

Toujours utiliser la balise {% url %}

##### Ajouter Bootstrap

Ajouter le fichier boostrap.min.css dans le dossier static et mettre

|  |
| --- |
| STATICFILES\_DIRS = [  os.path.join(BASE\_DIR, 'static'),  ] |
| Puis ajouter {% load static %} au début du fichier html avant le doctype |

Voir : https://simpleisbetterthancomplex.com/series/2017/09/11/a-complete-beginners-guide-to-django-part-2.html

### Html

##### For loop

{% for board in boards %}

…

{% endfor %}

##### Variable

{{ variable }}

##### Embed template/Variable template mère

|  |  |
| --- | --- |
| Main template | sub template |
| <!doctype html>  …  {% block contentA %}  Mon titre  {% endblock %}  …  {% block contentB %}  {% endblock %}  … | #Pas besoin des balises html  {% block contentA %}  {{block.super}}  #Va retourner ce qui était inclus dans le contentA mère, soit : Mon titre  ….html code here  {% endblock %}  …  {% block contentB %}  ….html code here  {% endblock %} |

Permet d’ajouter le code d’un autre template dans un template. On peut accéder à la valeure spécifiée pour le main template dans le subtemplate avec {{block.super}} comme dans contentA.

###### On peut mettre {% block contentA %} {% endblock contentA %} pour bien savoir quel block finis et on peut aussi utiliser les blocks dans le head pour ajouter des feuilles css ou autre spécifiques à certaines pages.

## Test

##### Base de données test

Pour les tests, django n’utilise pas la BD actuelle, il faut en créer une bidon qui sera détruite à la fin des test

|  |
| --- |
| def setUp(self):  Board.objects.create(name='Django', description='Django board.') |

##### Lancer test

Dans powershell, écrire :

python manage.py test

python manage.py test --verbosity=2 # si on veut plus de détails

Verbosity 0 : no output , 1 normal, 2 ++

On peut tester seulement une app avec

python manage.py test myAppName

Ou un fichier test avec

python manage.py test myAppName.directory.myPythonTest #Ne pas écrire le .py

On pourrait même appeler un test spécifique en ajoutant la classe et la méthode après le nom du fichier.

#### Vues

##### Retourne la bonne vue

Tester si django retourne les vues pour le url demandé (le fait automatiquement pour toutes les vues associées) :

|  |
| --- |
| from django.urls import reverse  def test\_home\_view\_status\_code(self):  #On test pour l’url home  url = reverse('home')  response = self.client.get(url)  self.assertEquals(response.status\_code, 200) |

##### Retourne la vue associé à l’alias

Ex : s’assurer que si on écrit '/', on retourne la fonction home qui nous donne la view de home.

|  |
| --- |
| from django.urls import resolve  from .views import home  def test\_home\_url\_resolves\_home\_view(self):  view = resolve('/')  self.assertEquals(view.func, home) |

## Url

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

view argument : It also accepts the return of the django.conf.urls.include function, which is used to reference an external urls.py file.

Name argument : Permet d’identifier l’url uniquement, ce qui fait qu’on aura qu’à changer le regex pour modifier l’appel de l’url pour tout le site.

Le path peut contenir l’information d’un username, une variable, une pk, une date, etc.

toujour terminer le regex avec un /.

##### Logout/Login

Pour qu’après s’être logout, on revienne à la page home, dans le settings.py, à la fin du document, ajouter :

LOGOUT\_REDIRECT\_URL = 'home'

et pour être redirigé au home après le login réussi :

LOGIN\_REDIRECT\_URL = 'home'

## Vue

Créer une vue :

1. Ajouter son path/route dans le url.py du dossier du projet
2. Créer la classe de la vue dans le views.py de l’app
3. Ajouter le template à la vue créée

##### Fonction vue

view function used to process the user request for a matched URL.

##### Protéger vue

Pour restreindre une vue aux utilisateurs login seulement, on ajoute le décorateur

from django.contrib.auth.decorators import login\_required

@login\_required

avant la méthode de la vue.

##### View login

La vue du login demande un paramètre next pour savoir où se diriger après.

# PyCharm

##### change font style

voir : https://www.jetbrains.com/help/pycharm/configuring-colors-and-fonts.html