TP1 Python Django
Shell et site d'administration

TOYI Francois

October 2024

Mise en route

Je me suis mis dans mon projet pour commencé le nvo projet .

L'interface de programmation(API)

1. Une **API** est un ensemble de règles et de protocoles qui permet à deux applications ou services de communiquer entre eux.

Un **shell** est un programme qui sert d'interface entre l'utilisateur et le système d'exploitation(en ligne de commande).

2. Cette commande ouvre un shell de notre projet django .

La différence entre la commande alias uniquement et alias manage.py shell est que l'alias ouvre l'interpreteur python lui même tandis que l'alias manage.py shell ouvre l'interpreteur python de notre projet django(c'est la particularité entre les deux).

```
(test_env) toyi@toyi-desktop:~/Documents/L3/sem_5/django/ifnti_l3$ py manage.py shell
Python 3.8.10 (default, Sep 11 2024, 16:02:53)
[GCC 9.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
(InteractiveConsole)
>>>
KeyboardInterrupt
>>> exit()
(test_env) toyi@toyi-desktop:~/Documents/L3/sem_5/django/ifnti_l3$ py
Python 3.8.10 (default, Sep 11 2024, 16:02:53)
[GCC 9.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

3-a) l'attribut nom ne contiens rien. Oui c'est normal car on n'a pas définis de valeur par défaut pour l'attribut nom. Il n'y a pas eu d'erreur car l'attribut nom n'est pas déclaré comme obligatoire c'est le pourquoi.

b) Ici on à une erreur car on est en train d'enrégistré un niveau avec le même nom or on à indiqué que notre attribut nom doit être unique dans notre table niveau.

c) Supprimons le niveau sauvegardé en base de données, on a :

```
>>> niveau = Niveau.objects.last()
>>> niveau.delete()
(1, {'notes.Niveau': 1})
```

d) Créons 3 Niveau et sauvegardons les, on a:

```
>>> from notes.models import Niveau
>>> niv1 = Niveau()
>>> niv1.nom = "L1"
>>> niv1.save()
>>> niv2 = Niveau()
>>> niv2.nom = "L2"
>>> niv2.save()
>>> niv3 = Niveau()
>>> niv3.nom = "L3"
>>> niv3.save()
>>>
```

Les valeurs des clé primaire sont les suivantes :

```
>>> nivaux = Niveau.objects.all()
>>> nivaux.values_list()
<QuerySet [(1, 'L1'), (2, 'L2'), (3, 'L3')]>
>>>
```

4.Voici se que j'ai tapé et se que sa donne car moi j'ai définit en créant ma classe la méthode **__str__()** doux le résultat là:

```
>>> eleJ = Eleve.objects.get(nom="Jean")
>>> print(eleJ)
Jean Aimar
>>>
```

Pour que le nom et le prenom s'affiche alors on doit définir la méthodes **__str__()** dans tout les modèles, les voici :

Modele Personne :

```
    def __str__(self):
        return f"{self.nom} {self.prenom} {self.date_naissance} {self.sexe}"
```

Modele Eleve

```
def __str__(self):
    parent_str = super().__str__()
    matieres = ", ".join([str(matiere) for matiere in self.matieres.all()])
    return f"{parent_str} - Niveau: {self.niveau.nom} - Matières: {matieres
    or 'Aucune matière'}"
```

Modele Enseignant

```
def __str__(self):
    parent_str = super().__str__()
    matieres = ", ".join([str(matiere) for matiere in self.matieres.all()])
    return f"{parent_str} - Matières: {matieres or 'Aucune matière'}"
```

Modele Matiere

```
    def __str__(self):
        return self.nom
```

Modele Niveau

```
    matieres = ", ".join([str(matiere) for matiere in self.matiere.all()])
    return f"Niveau: {self.nom} - Matières: {matieres or 'Aucune matière'}"
```

Modele Note

```
    return f"Note: {self.valeur} - Matière: {self.matiere.nom} - Élève: {self.eleve.prenom}
```

5. Non les niveau n'était pas sans nom mais plutot une chaine de charactère vide.

Pour corrigé se problème on peut utilisé la méthode suivante **get_or_create()** soit on peut plutôt ajouter des constructeur dans nos modèle ou classe .

Le site d'administration

1. Voici la commande à taper :

```
    py manage.py createsuperuser
```

2. La page est en **anglais**.

Pour ramener sa en francais on vas dans le setting de notre projet et on change la variable suivante comme telle :

```
LANGUAGE_CODE = 'fr'
```

3. Le fichier **admin.py** est utilisé pour configurer et personnaliser l'interface d'administration de notre application.

Voici les lignes que j'ai ajouter :

```
from .models import Niveau,Eleve,Enseignant,Matiere

# Register your models here.

admin.site.register(Niveau)
admin.site.register(Enseignant)
admin.site.register(Eleve)
admin.site.register(Matiere)
```

4.Oui il y a soucie d'orthographe sur les nouveaux modèles, C'est juste les **accents** où les **s** . Pour y remédier on vas utiliser la variable **verbose_name_plural** .

Reportons notre code .

Modele Niveau

```
class Meta:
    verbose_name_plural = "Niveaux"
```

Modele Matiere

```
class Meta:
    verbose_name_plural = "Matières"
```

Modele Eleve

```
class Meta:
    verbose_name_plural = "Elèves"
```