

Tp 06 Sécurité et site administration

AMONA Birewa Audrey

27 novembre 2023

Table des matières

1 Authentification

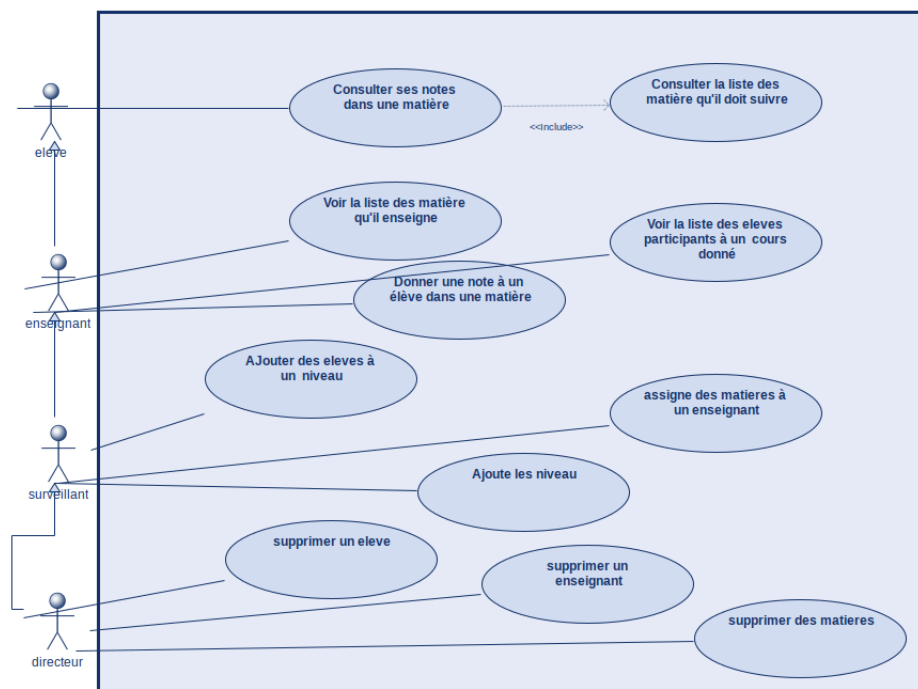
1

Résumé

Ce TP contient 3 parties indépendantes. D'abord nous nous pencherons sur l'authentification. Ensuite nous explorerons les problèmes de sécurité : CSRF, HTTPS, clé de chiffrement et détournement de clic. Enfin nous verrons comment customiser le site d'administration.

1 Authentification

1. Dans le Tp précédent, nous avons spécifié les différentes fonctionnalités en fonction des utilisateurs du système :



Il était convenu que seuls les enseignants, les surveillants ou encore le directeur peuvent avoir accès au formulaire d'ajout de note.

2. Dans le site d'administration, nous remarquons qu'il y a deux modèles de plus **Users** et **Groups**, que nous n'avons pas ajoutés nous-mêmes. Le modèle **Users** possède les attributs suivants : username, last-name, first-name, email address, staff, status. Il possède un enregistrement qui a pour username le nom de notre superutilisateur créé.
3. Nous allons créer des groupes avec les différents rôles du diagramme de cas d'utilisation précédent.

eleve

HISTORY

Name:

eleve

Permissions:

Available permissions ?

Q

Filter

auth | user | Can add user
auth | user | Can change user
auth | user | Can delete user
auth | user | Can view user
contenttypes | content type | Can add content type
contenttypes | content type | Can change content type
contenttypes | content type | Can delete content type
contenttypes | content type | Can view content type
notes | Eleve | Can add Eleve
notes | Eleve | Can change Eleve
notes | Eleve | Can delete Eleve
notes | Enseignant | Can add Enseignant

Chosen permissions ?

Q

Filter

notes | Eleve | Can view Eleve
notes | Matiere | Can view Matiere
notes | niveau | Can view niveau
notes | Note | Can view Note

enseignant

HISTORY

Name:

enseignant

Permissions:

Available permissions ?

Q

Filter

auth | permission | Can delete permission
auth | permission | Can view permission
auth | user | Can add user
auth | user | Can change user
auth | user | Can delete user
auth | user | Can view user
contenttypes | content type | Can add content type
contenttypes | content type | Can change content type
contenttypes | content type | Can delete content type
contenttypes | content type | Can view content type
notes | Eleve | Can add Eleve
notes | Eleve | Can delete Eleve

Chosen permissions ?

Q

Filter

notes | Eleve | Can change Eleve
notes | Eleve | Can view Eleve
notes | Enseignant | Can view Enseignant
notes | Matiere | Can view Matiere
notes | niveau | Can view niveau
notes | Note | Can add Note
notes | Note | Can change Note
notes | Note | Can delete Note
notes | Note | Can view Note

surveillant

HISTORY

Name:

surveillant

Permissions:

Available permissions ?

Q

Filter

auth | permission | Can change permission
auth | permission | Can delete permission
auth | permission | Can view permission
auth | user | Can add user
auth | user | Can change user
auth | user | Can delete user
auth | user | Can view user
contenttypes | content type | Can add content type
contenttypes | content type | Can change content type
contenttypes | content type | Can delete content type
contenttypes | content type | Can view content type
sessions | session | Can add session

Chosen permissions ?

Q

Filter

notes | Eleve | Can add Eleve
notes | Eleve | Can change Eleve
notes | Eleve | Can delete Eleve
notes | Eleve | Can view Eleve
notes | Enseignant | Can add Enseignant
notes | Enseignant | Can change Enseignant
notes | Enseignant | Can delete Enseignant
notes | Enseignant | Can view Enseignant
notes | Matiere | Can add Matiere
notes | Matiere | Can change Matiere
notes | Matiere | Can delete Matiere
notes | Matiere | Can view Matiere

Choose all ?

Remove all

- Django fournit une classe User qui contient l'ensemble des informations nécessaires à l'authentification et aux autorisations. Nous pouvons étendre ce modèle sans modifier nos modèles utilisateurs par défaut soit :

- En créant une relation d'héritage entre notre modèle et le modèle par défaut **Users**
 - En créant une relation OneToOne entre notre modèle et le modèle par défaut **Users**
5. Nous allons procéder à la création OneToOne entre les deux modèles, lorsqu'on lance **py manage.py makemigrations**, on a l'erreur suivante :

```

○ (app) audrey@LEFREREDUCOUZIN:~/Documents/django/TP/tp6/my_first_django_project$ py manage.py makemigrations
It is impossible to add a non-nullable field 'user' to eleve without specifying a default. This is because the database needs something to populate existing rows.
Please select a fix:
  1) Provide a one-off default now (will be set on all existing rows with a null value for this column)
  2) Quit and manually define a default value in models.py.
Select an option: █

```

Ensuite on corrige le modèle en rendant l'attribut **user** nullable :

```

1 class Personne(models.Model):
2     nom = models.CharField(max_length=50)
3     prenom = models.CharField(max_length=50)
4     sexe = models.CharField(choices=(("M", "Masculin"), ("F", "Feminin")), max_length=20)
5     date_naissance = models.DateField()
6     user = models.OneToOneField(User, on_delete=models.CASCADE, null=True)
7
8     class Meta:
9         abstract = True

```

6. Nous allons maintenant créer une instance de **User** pour chaque personne que nous avons créé :

Select user to change

Search

Action: ----- Go 0 of 9 selected

<input type="checkbox"/>	USERNAME	EMAIL ADDRESS	FIRST NAME	LAST NAME	STAFF STATUS
<input type="checkbox"/>	aa				✗
<input checked="" type="checkbox"/>	audrey				✓
<input type="checkbox"/>	az				✗
<input type="checkbox"/>	bt				✗
<input type="checkbox"/>	ep				✗
<input type="checkbox"/>	ka				✗
<input type="checkbox"/>	sa				✗
<input type="checkbox"/>	sc				✗
<input type="checkbox"/>	td				✗

7. Nous allons maintenant enlever la propriété nullable sur l'attribut précédemment ajouter. Lorsqu'on essaie de relancer les migration, on a le message suivant au terminal :

```

Applying notes1000s_eleve_user_enseignant_user111... ok
(app) audrey@LEFREREDUCOUZIN:~/Documents/django/TP/tp6/my_first_django_project$ py manage.py makemigrations
SystemCheckError: System check identified some issues:

ERRORS:
notes.Eleve.user: (fields.E320) Field specifies on_delete=SET_NULL, but cannot be null.
      HINT: Set null=True argument on the field, or change the on_delete rule.
notes.Enseignant.user: (fields.E320) Field specifies on_delete=SET_NULL, but cannot be null.
      HINT: Set null=True argument on the field, or change the on_delete rule.
(app) audrey@LEFREREDUCOUZIN:~/Documents/django/TP/tp6/my_first_django_project$

```