Implement Bulit-in Permissions

Tuesday, February 27, 2024 10:55 AM

Purpose

Learn how to implement role-based access control permissions

Scope

Covers the following topics:

- Updating model attributes
-

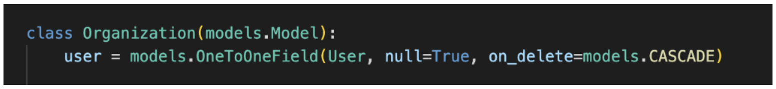
Procedure

Step 1: In the *models.py* file of you django app - import the user model class from Django authentication library

- `from django.contrib.auth.models import User`

Step 2: Write a "user" (or any other name you'd like to give it) attribute to the model you want to give permission to

- `user = models.OneToOneField(User, null=True, on_delete=models.CASCADE)`

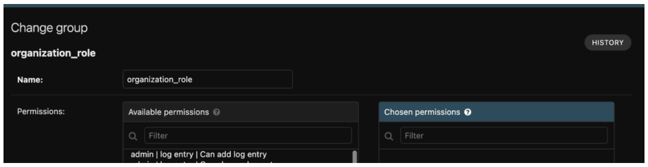


```
class Organization(models.Model):
    user = models.OneToOneField(User, null=True, on_delete=models.CASCADE)
```

Step 3: Migrate model changes to database

- `python manage.py makemigrations`
- `python manange.py migrate`

Step 4: In your app



Change group

organization_role

Name: organization_role

Permissions:

admin | log entry | Can add log entry

Resources

Step by step model updating and how to control from terminal	https://permify.co/post/rbac-in-django/
Concept explanations	https://theswed.se/implementing-permissions-and-user-roles-in-django-4/
Implementing control through it's own models	https://medium.com/@subhamx/role-based-access-control-in-django-the-right-features-to-the-right-users-9e93feb8a3b1
What is it	https://www.digitalguardian.com/blog/what-role-based-access-control-rbac-examples-benefits-and-more