Django (Usuarios y Modelos Relacionados)

The web framework for perfectionists with deadlines.

Agenda

- Administración de Usuarios
- Autenticación con Django
- Relaciones en Modelos con Django
- Consultas a Tablas Relacionadas

Administración de Usuarios

El objeto "User" es el core del sistema de autenticación. Los atributos primarios de un usuario por defecto son:

- username
- password
- email
- first_name
- last_name

```
# python manage.py shell
>>> from django.contrib.auth.models import User
>>> user = User.objects.create_user(
    'john', 'lennon@thebeatles.com', 'johnpassword')
>>> user.last_name = 'Lennon'
>>> user.save()

$ python manage.py createsuperuser
```

Autenticación con Django

Para realizar la autenticación de los usuarios debemos importar el módulo django.contrib.auth que contiene todas las clases y funciones para implementar el sistema de autenticación.

```python
from django.contrib.auth import authenticate, login

def my_view(request):
    username = request.POST['username']
    password = request.POST['password']
    user = authenticate(request, username=username, password=password)
    if user is not None:
        login(request, user)
        # Redirect to a success page.
        ...
    else:
        # Return an 'invalid login' error message.
        ...
```

Autenticación con Django

Django utiliza sesiones y middleware para conectar el sistema de autenticación a objetos de solicitud. Estos proporcionan un atributo `request.user` en cada solicitud que representa al usuario actual. Si el usuario actual no ha iniciado sesión, este atributo se establecerá en una instancia de `AnonymousUser`, de lo contrario será una instancia de Usuario.

```
if request.user.is_authenticated:
    # Do something for authenticated users.
    ...
else:
    # Do something for anonymous users.
    ...
```

```
from django.contrib.auth import logout

def logout_view(request):
    logout(request)
    # Redirect to a success page.
```

Autenticación con Django

La manera simple de limitar acceso a las paginas es checando `request.user.is_authenticated` y despues redireccionar hacia la pagina del login.

```python
from django.conf import settings
from django.shortcuts import redirect

def my_view(request):
    if not request.user.is_authenticated:
        return redirect('%s?next=%s' % (settings.LOGIN_URL, request.path))
    # ...
```

Para validar que solo los usuarios logueados accedan a las vistas usamos lo siguiente:

```python
from django.contrib.auth.decorators import login_required

@login_required
def my_view(request):
    ...
```

Autenticación con Django (Avanzado)

```
# urls.py
from django.contrib.auth import views as auth_views
urlpatterns = [
    path('accounts/login/', auth_views.LoginView.as_view(), name='login'),
    path('accounts/logout', auth_views.LogoutView.as_view(), name='logout')
]

# settings.py
LOGIN_REDIRECT_URL = '/home’
LOGOUT_REDIRECT_URL = '/accounts/login/'

# templates/registration/login.html
<form>
    {% csrf_token %}
    {{form.username}}
    {{form.password}}
    <button type="submit">Login</button>
</form>
```