

Session 9 - 1/10/2025

## Authentication and authorization

تعريف مستخدم جديد

تسجيل دخول / تسجيل خروج

حماية البيانات من الوصول ..

: "الصادقة" - من أنت ؟ Authentication

عملية التحقق من هوية الشخص (المستخدم) ، حيث تتحقق أن المستخدم صدر من من يدعى أنه هو من خلال التحقق من صحة بياناته مثل اسم المستخدم، كلمة

النور ...

"التصويف" تعني الصلاحيات ما هي صلاحياتك ؟ Authorization

هل يحق الشخص الذي تم التعرف عليه القيام ببعض الإجراءات ؟

كثير ما ي Heard actions و الموارد التي يتعين على user بالوصول إلى authenticated user

ترى صلاحيات على مستوى policies أو roles

مثل الصلاحيات للتحكم عن بيانات ولكن من دون تغييرها .

Django # توفر نظام معنوي و سهل لدارة المستخدم

- تسجيل دخول / تسجيل خروج

صلاحيات

passwords

sessions

. Django يحتوي user model على ميزة بكل اختصار في

Session : المعلومات المحفوظة بين الطلبات يمكن استرجاع

User model : فهو الاسم و يحفظ المعلومات في رسالة رقم .

صفحت : settings.py

MIDDLEWARE = [ ]

يُوجَّه من البرامج الذي يعدل كمحسِّن بين التصييرات المختلفة لرسائل الاتصال بينها، يرجع التصييرات  
بيانات البيانات بالفعل معًا بلائحة يجتذب المترقب عن اختلاف الأيقونة، إذ التصييرات المختلفة

صيغة هذه الـ logic ( متوجهة في الترميز والأكواد )  
Middleware in Django framework of hooks

تقديم الـ request

كل # component middleware # كل #

\* 'django.middleware.security.SecurityMiddleware',

تعزز خصائص سلامة على درجة كل سلم عليه تعليم / تعليم  
يمكن تنفيذ من خلال إصدارات معينة، هنا الـ middleware # تأسية لتحسين العرض  
الأولي لتصييرات الـ django.

contrib

\* 'django.contrib.sessions.middleware.SessionMiddleware',

الوصول إلى session attribute خلال الـ view.

\* 'django.middleware.common.CommonMiddleware',

التعامل مع www / . / URL normalization

\* 'django.middleware.csrf.CsrfViewMiddleware',

Anti Cross-site Request Forgery (CSRF) تجنب من هجمات ا.

\* 'django.contrib.auth.middleware.AuthenticationMiddleware',

يوصل المترقب الذي قام بـ login() حول بارطبع request

\* 'django.contrib.messages.middleware.MessageMiddleware',

تحتوي الرسائل مؤقتة في طلب واحد واسترجادها للعنصراني على المدى (عادة العنصر التالي)  
تم دفع علامة على كل رسالة بمعنى عدد بحد أقصى (تسلسل المعلومات ذات الصلة) one-time notifications "flash messages" ←

]

Templates = [

!

'context\_processors': [

'django.template.context\_processors.request',

مجموعة من مطلبات العناصر التي تعيّن dictionaries لمحة request dictionary يأخذ مثلاً request object و هو صفات request.function إلى

context أو

request.path : في request.data في المدخلات request.user في templates

'django.contrib.auth.context\_processors.auth',

context messages > perm < user تعيين

في authentication إلى البيانات المتعلقة بالuser user.is\_authenticated.

'django.contrib.messages.context\_processors.messages',

context اطلاعات إرسال إلى templates

flash messages في المدخلات

]

لتكون ...

SESSION\_COOKIE\_SECURE = TRUE

لضمان أن HTTPS يتم إرساله معه من خلال اتصال session cookies،  
وتصديق معيار HTTPS يعني أنه تم تبديل سرقة الاتصال أو التسلل.

CSRF\_COOKIE\_SECURE = TRUE

HTTPS لضمان أن CSRF protection cookie يتم إرساله من خلال اتصال  
الاتصال من مصدر غير موثوق به.

الخطوة الخامسة

عند تشغيل جيد 59

dashboard app. → student app  
يتحقق المراحل التي نسبعد في كل مرة

(student app) views.py \* ضمن

```
from django.contrib.auth.views import LoginView, LogoutView
```

```
class UserLoginView(LoginView):
    template_name = 'student/login.html'
```

```
class UserLogoutView(LogoutView):
    template_name = 'student/logout.html'
```

(student app) urls.py \* ضمن

```
from django.urls import path
from .views import UserLoginView, UserLogoutView
```

```
urlpatterns = [  
    path('login/', UserLoginView.as_view(), name='login'),  
    path('logout/', UserLogoutView.as_view(), name='logout'),  
]
```

نقدم باصنيفة : settings.py ٣٤.  
LOGIN\_REDIRECT\_URL = 'dashboard/'

يعرف أي عن من تلقى الدخول

( dashboard app ) views.py ٣٥ \*

```
from core.django.contrib.auth.decorators import login_required  
from django.contrib.auth.mixins import LoginRequiredMixin  
from django.shortcuts import render  
from django.views.generic import TemplateView
```

```
class DashboardView(LoginRequiredMixin, TemplateView):  
    template_name = 'dashboard.html'
```

@login\_required

```
def dashboard(request):
```

```
    return render(request, 'dashboard.html')
```

function si class

جعل دالة

التي سمعت  
جاء من

: dashboard.html

```
{% if user.is_authenticated %}
```

```
<P>
```

Hello {{ user.name }}

```
</P>
```

```
{% else %}
```

```
<P>
```

Do you want to login?

```
</P>
```

```
{% endif %}
```

( dashboard :: student )

apps

accounts app

( accounts app ) forms.py

```
from django import forms
```

```
from django.contrib.auth.forms import AuthenticationForm
```

```
class LoginForm(AuthenticationForm):
```

```
    username = forms.CharField(
```

```
        label = 'Username',
```

```
        widget = forms.TextInput(attrs = { 'placeholder': 'Your
```

```
username' })
```

```
)
```

```
password = forms.CharField(  
    label='Password',  
    widget=forms.PasswordInput(attrs={'placeholder': 'Your  
password'})  
)
```

```
remember_me = forms.BooleanField(  
    required=False,  
    initial=False,  
    label='Remember Me?',  
)
```

(accounts app) views.py

```
from django.contrib.auth.views import LoginView  
from .forms import LoginForm
```

```
class UserLoginView(LoginView):  
    template_name = 'accounts/Login.html'  
    authentication_form = LoginForm  
    redirect_authenticated_user = True
```

ادخل بذوق API

```
def form_valid(self, form):  
    response = super().form_valid(form)  
    remember = form.cleaned_data.get('remember_me')  
    if not remember:  
        self.request.session.set_expiry(0)
```

بعد انتهاء اجل

اعطى التذكرة / طبع الكود

```
else:  
    self.request.session.set_expiry(60 * 60 * 24 * 14)
```

بعد انتهاء اجل

```
return response
```

(accounts app) urls.py

```
    path('Login/', LoginView.as_view(), name='login'),
```

(accounts app) login.html

```
<h2> Login </h2>
<form method="post" novalidate>
    {{% csrf_token %}}
    {{ form.non_field_errors }}
<p>
    {{ form.username.label_tag }} <br>
    {{ form.username }} {{ form.username.errors }}
</p>
<p>
    {{ form.password.label_tag }} <br>
    {{ form.password }} {{ form.password.errors }}
</p>
<p>
    {{ form.remember_me }} {{ form.remember_me.label_tag }}
</p>
{{% if next %}}
<input type="hidden" name="next" value="{{next}}"/>
{{% endif %}}
<button type="submit"> Log in </button>
</form>
```