Authentication

We have 2 types of Security

1. Authentication

2. Authorization

Authentication:

- To enter into particular software applications by using login is nothing but authentication.

Authorization:

- After entering into an application to which links we have to give permissions and to which links not to give permissions is about authorization.

Authentication

1. Add the auth urls

2. Create the login form

3. Secure the views

4. Create the users

5. Test it

Django provides enough urls to authentication (login and logout)

urls.py at project level:

urls.py

urlpatterns = [

```
  path('accounts/',include('django.contrib.auth.urls')),
]
```

- Go to templates folder and create new folder name registration and create new file name login.html

login.html(templates/registration)

```
<html>
<head>
</head>
<body>
  <form method="POST">
    <table>
    {{form.as_table}}
    </table>
    <button type="submit" name="button">Login</button>
    {% csrf_token %}
    </form>
</body>
</html>
```

Secure the views:

views.py

from django.contrib.auth.decorators import login_required

- without login if we sends the request directly we have to show the login form
- Django provides decorator (@login_required).This decorator stops direct access before that login page will open.
- According to our requirement we have to write this decorator for views.

views.py

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

@login_required

def homepage(request):

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

For class based views:

urls.py(application level)

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

urlpatterns = [

path('getteacherdetail/<int:pk>/',login_required(Teacherdetail.as_view())
),

]
```

After logout where we have to render the page:

settings.py (at project level)

LOGOUT_REDIRECT_URL='/userlogout'

- Go to templates folder and create new file logout.html

logout.html (templates)

```
<h1>User successfully logged out</h1>
```

views.py (application level)

```
def logoutuser(request):
        return render(request,'logout.html')
```

urls.py (at project level)

```
from admissions.views import logoutuser
urlpatterns = [
   path('userlogout/',logoutuser),
]
```

index.html (templates)

```
<a href="./accounts/logout">Logout</a>
```

Authorization:

```
from django.contrib.auth.decorators import
login_required,permission_required

@login_required

@permission_required('admissions.add_student')

def addadmission(request):

        form=StudentModelForm

        studentform= {'form':form}

        if request.method=="POST":

                form=StudentModelForm(request.POST)

                if form.is_valid():

                        form.save()

        return
render(request,'admissions/addadmissions.html',studentform)


@login_required

@permission_required('admissions.view_student')

def admissionreport(request):

        result=student.objects.all();

        students={ 'allstudents':result}

        return render(request, 'admissions/admissionreport.html',students)
```

```
@login_required
@permission_required('admissions.delete_student')
def delstudent(request,id):
    s=student.objects.get(id=id)
    s.delete()
    return admissionreport(request)


@login_required
@permission_required('admissions.change_student')
def updatestudent(request,id):
    s=student.objects.get(id=id)
    form=StudentModelForm(instance=s)
    dict={ 'form':form}
    if request.method== "POST":
        form=StudentModelForm(request.POST,instance=s)
        if form.is_valid():
            form.save()
    return render(request, 'admissions/updateadmissions.html',dict)
```

To give Permissions

- Before that we have to create super user for admin UI

python manage.py createsuperuser

It will asks like this

username:

email:

password:

password(again):

Here I have to create username as **admin** and password as **fiit@123**

- After creating the super user and run the server

  python manage.py runserver

- Go to browser and open localhost/admin/
- Enter the username and password of admin UI what you have to created by using super user.
- Once login It will shows groups and users and corresponding models what you are register in admin UI(admin.py)

admin.py (application level)

```
from django.contrib import admin

from admissions.models import student,Teacher

# Register your models here.

admin.site.register(student)

admin.site.register(Teacher)

class studentAdmin(admin.ModelAdmin):

    list_display=['id','name','fathername','classname','contact']
```

Django Administration:

localhost/admin

create users

first user:

username:user1

password:fiit@321

confirm password:fiit@321

click save and user1 is created

second user:

username:user2

password:fiit@000

confirm password:fiit@000

click save and user2 is created

- select the user1 and give the permission for accessing views

user1->select filter in user permissions and select the required action(for ex: can change student)like wise you can give many actions after choosing actions and save it.

user2->select filter in user permissions and select the required action (for ex: can view student) likewise you can give many actions after choosing actions and save it.

- Once you will give permissions to users and open browser and type the particular url pattern it will ask to login the user first and

then go to that corresponding url pattern and select the actions what you have to provide to user and check it.

For ex:127.0.0.1/ad/admreport

- It will ask the login for that corresponding user
- Here I am giving user2 login and password after that only I will see the admission report view details.