Database Form

Created	@October 11, 2023 10:41 AM
Made By	Mariam Tamer

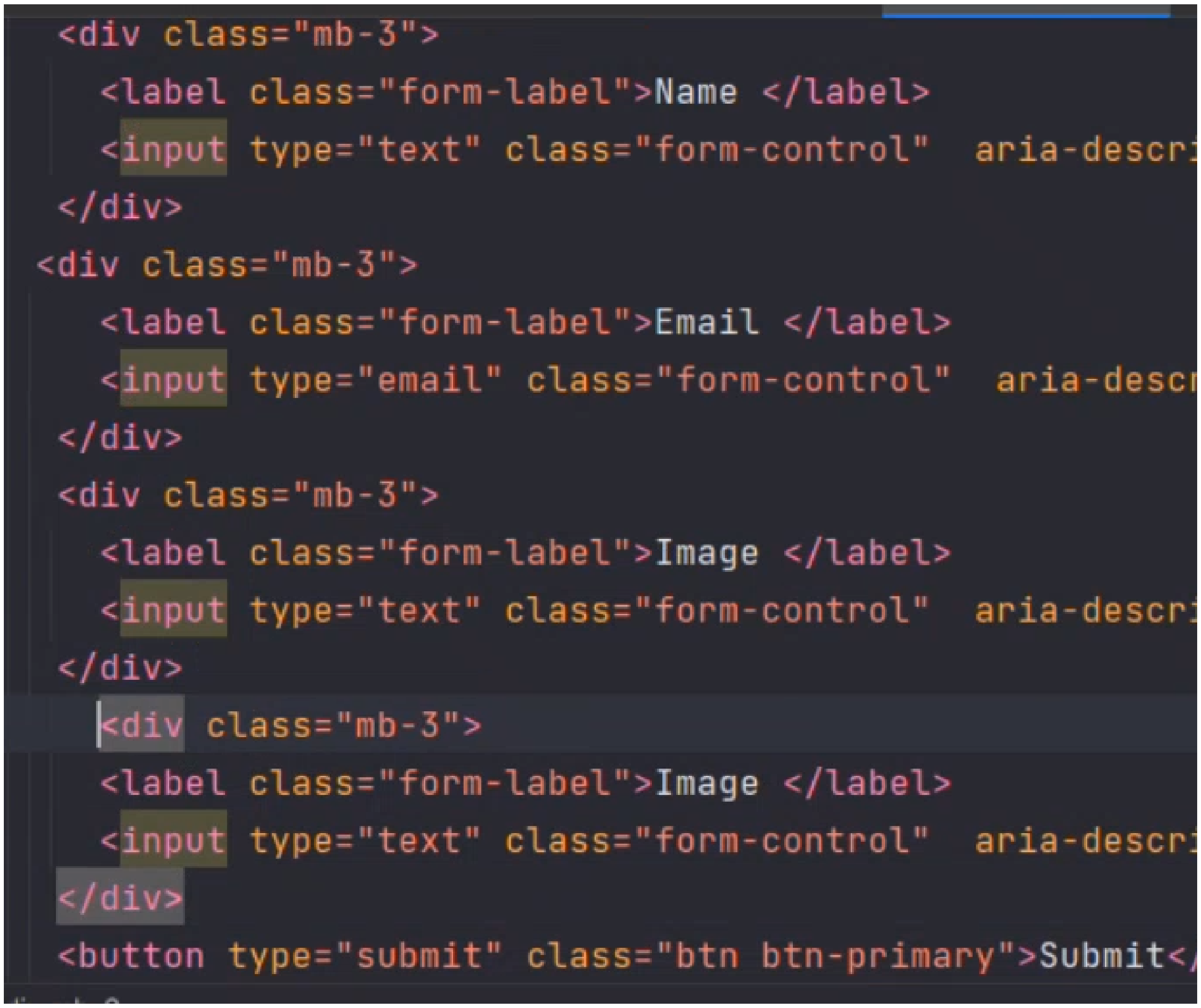
Steps to add new objects into your DB table using form:

1. **Make a button and make its href to send you to the form page**
2. **In your html file, make a bootstrap form**
    - **make it's method="post"**
    - **the type="" depending on your cols**
    - **add the csrf token**
    - **add name attributes in your html inputs so the data can be posted on the network Payload**
3. **Add the view function for the html page**
    - **check if the request is post or get**
    - **get data from the request body**
    - **add data to your DB**
4. **Add the view function in urls.py of your app**
5. **Add its url in your html form href**

Let's discuss them step by step :

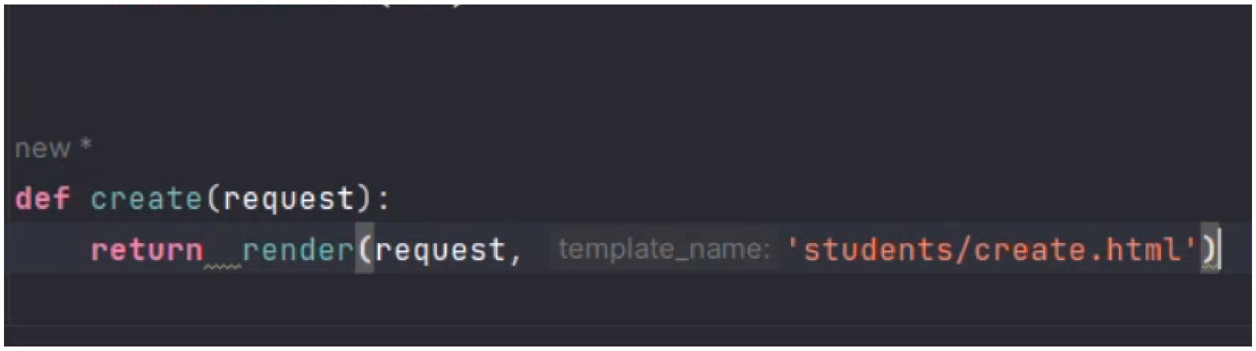
To add students objects into DB using form

1. **Make a button and make its href to send you to the form page**
2. **Make a bootstrap form and edit the type="" depending on your cols**



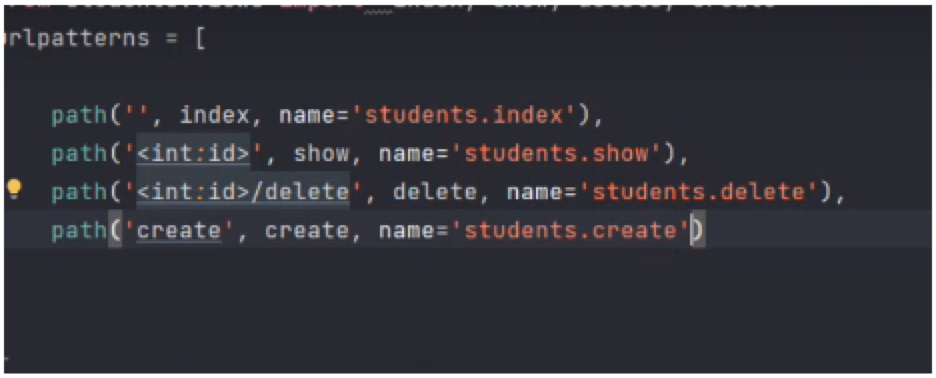
```
  <div class="mb-3">
    <label class="form-label">Name </label>
    <input type="text" class="form-control"  aria-descri
  </div>
 <div class="mb-3">
    <label class="form-label">Email </label>
    <input type="email" class="form-control"  aria-descr
  </div>
  <div class="mb-3">
    <label class="form-label">Image </label>
    <input type="text" class="form-control"  aria-descri
  </div>
    <div class="mb-3">
    <label class="form-label">Image </label>
    <input type="text" class="form-control"  aria-descri
  </div>
  <button type="submit" class="btn btn-primary">Submit</
```

3. **Add the view function for the html page**



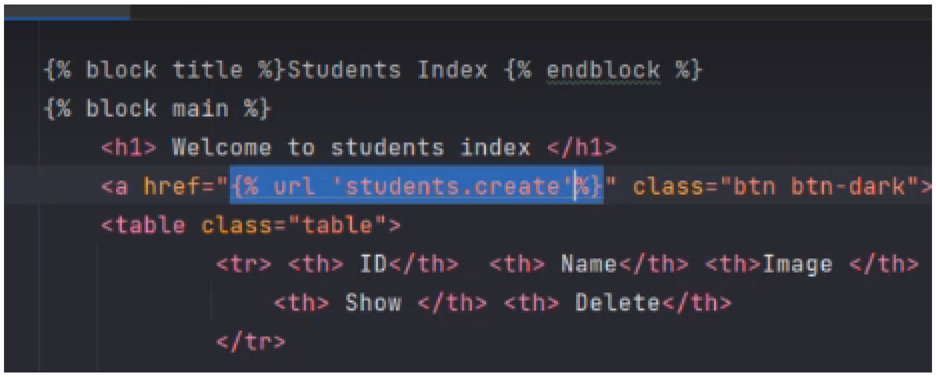
```
new *
def create(request):
    return render(request,  template_name: 'students/create.html')
```

4. **add it's urls.py in pages**



```
rlpatterns = [

    path('', index, name='students.index'),
    path('<int:id>', show, name='students.show'),
    path('<int:id>/delete', delete, name='students.delete'),
    path('create', create, name='students.create')
```

5. **give the href this url in your html page**



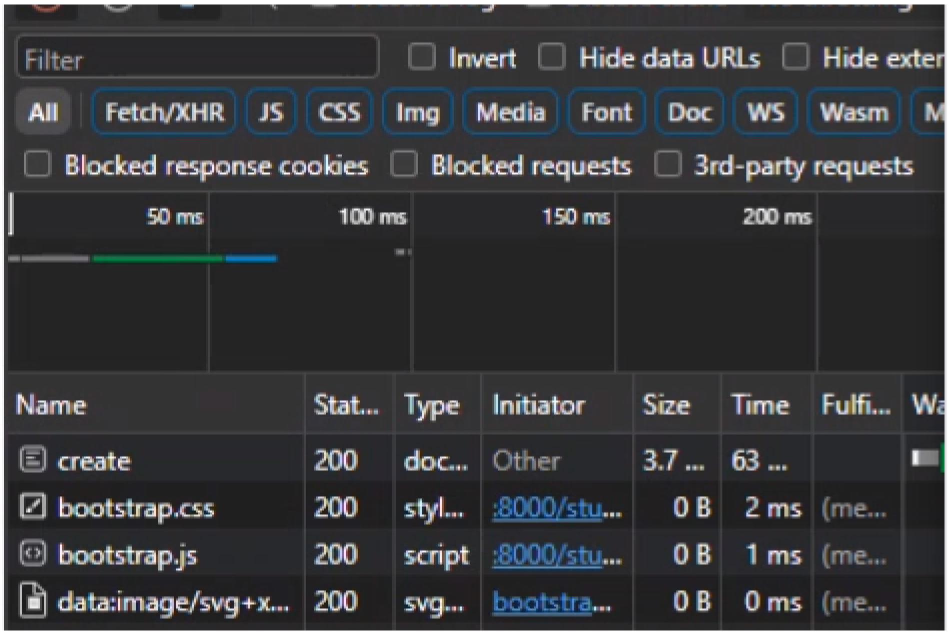
```
{% block title %}Students Index {% endblock %}
{% block main %}
    <h1> Welcome to students index </h1>
    <a href="{% url 'students.create'%}" class="btn btn-dark">
    <table class="table">
            <tr> <th> ID</th>  <th> Name</th> <th>Image </th>
                <th> Show </th> <th> Delete</th>
            </tr>
```

-Now if you go to your form and press Submit then open your network to check your function status, you'll find it's a get method and your data is sent in the **same page** .. so you want your method to be post so your data is posted on **another page** ..

method='get' —> data posted in the **same page**

method = 'post' —> data posted to **another page**

-You can find all this in your **inspect—network ..** (the create function you created in your views/whatever its name is)



Name	Stat...	Type	Initiator	Size	Time	Fulfi...	Wa
create	200	doc...	Other	3.7 ...	63 ...		
bootstrap.css	200	styl...	:8000/stu...	0 B	2 ms	(me...	
bootstrap.js	200	script	:8000/stu...	0 B	1 ms	(me...	
data:image/svg+x...	200	svg...	bootstra...	0 B	0 ms	(me...	

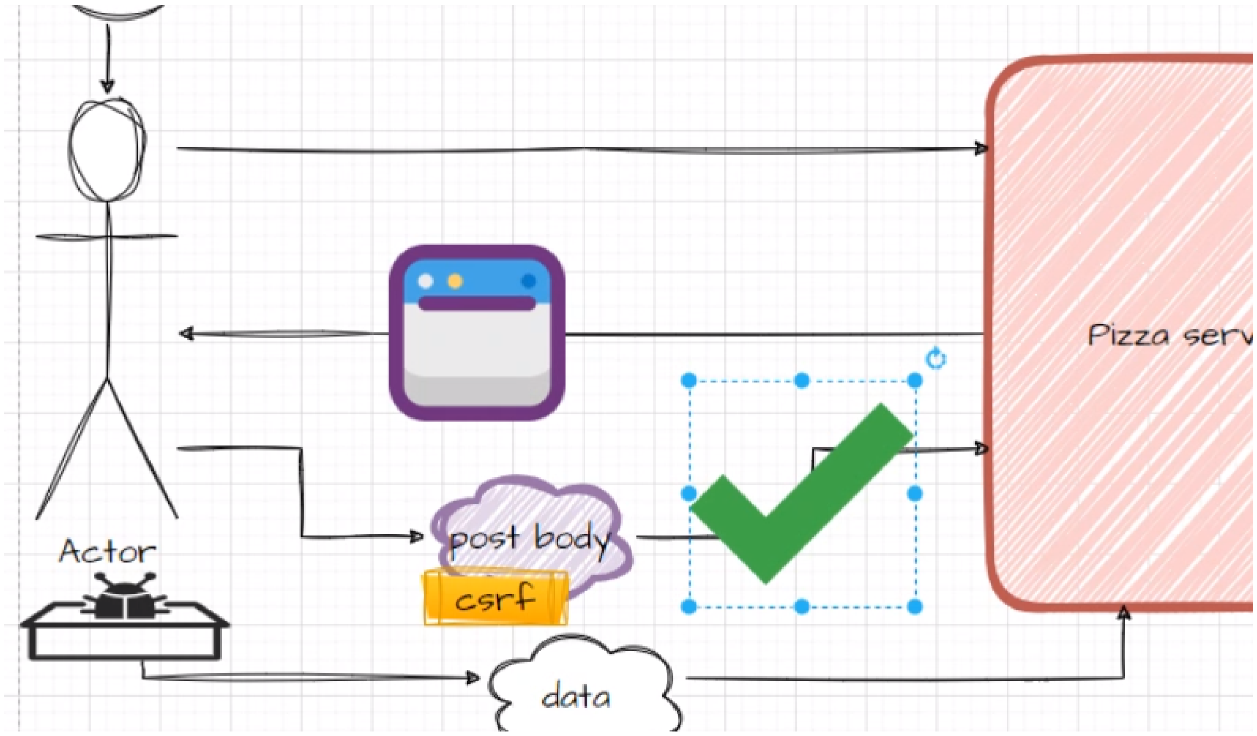
-After changing your method to post and pressing submit, you'll get an error called : csrf

csrf error —> *cross site forgery request*

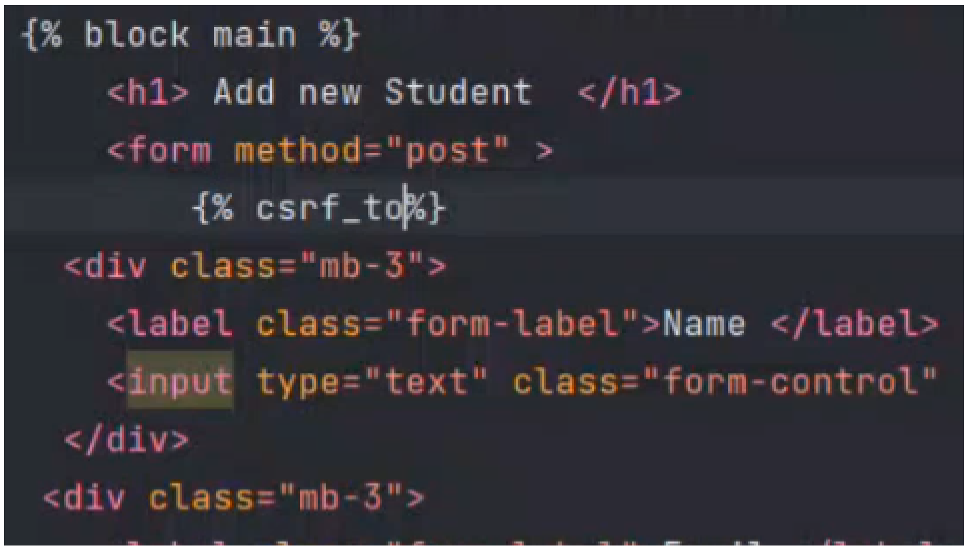
(related to certificate and authentication)

-When you post a form to server, you connect to a network. This network downloads mulware on your system. (A mulware is like a bad software or a software with virus for EX). This mulware takes the data in your post body and sends data to the server from their own data without checking if any changes happened to the data in the process which is not secure, so you get that csrf certification error.

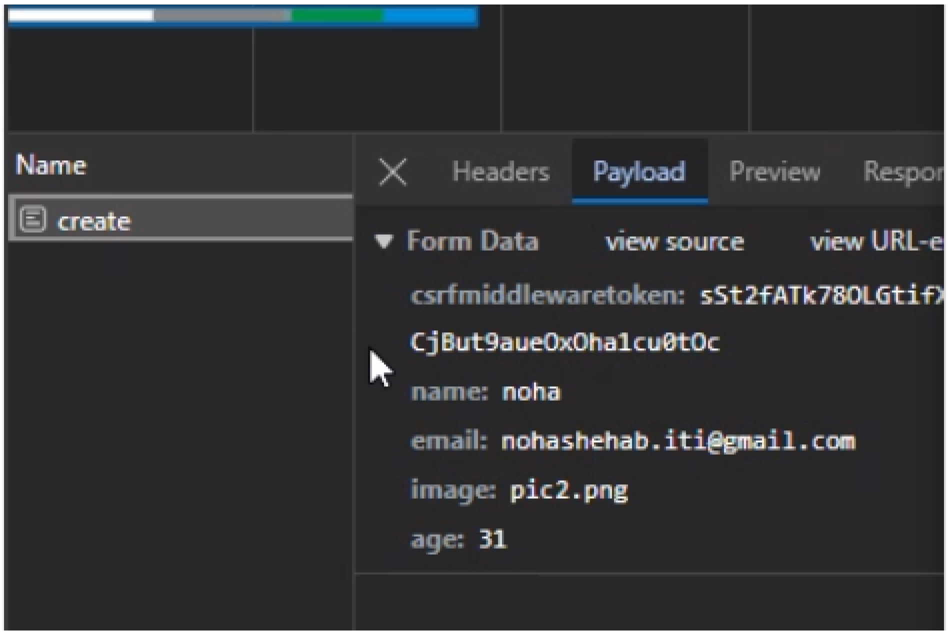
-Only those who have the **crsf token** will pass their post body data and go to the server.



6. **So add the csrf token to your form like this , and your post method will be sent to server.**

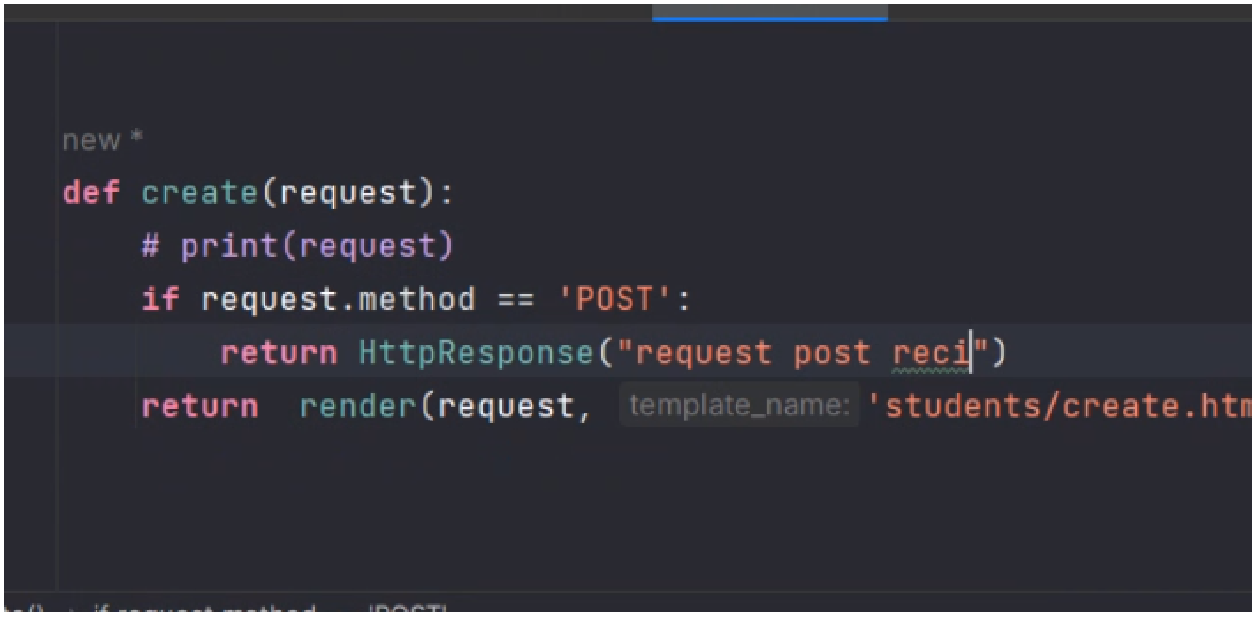


```
{% block main %}
    <h1> Add new Student  </h1>
    <form method="post" >
        {% csrf_to%}
  <div class="mb-3">
    <label class="form-label">Name </label>
    <input type="text" class="form-control"
  </div>
 <div class="mb-3">
```



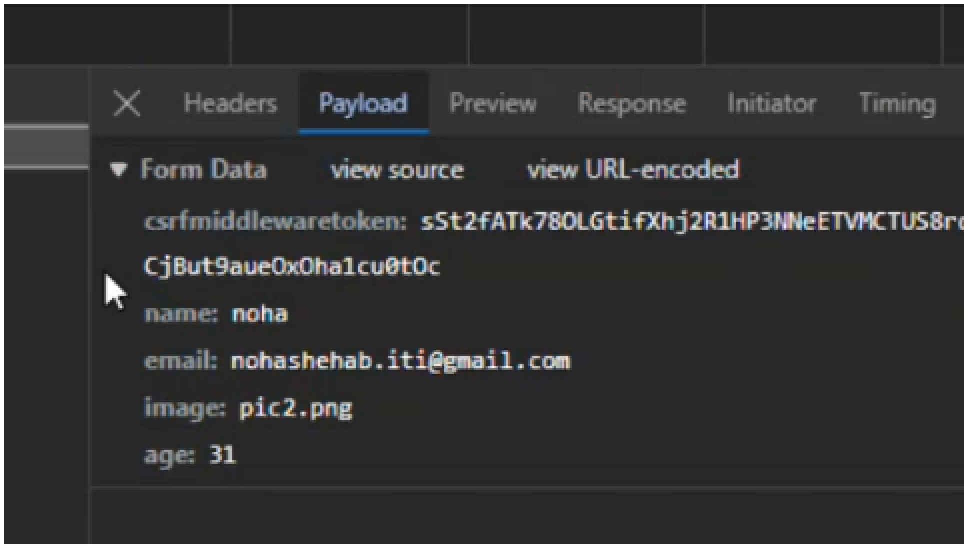
```
csrfmiddlewaretoken: sSt2fATk78OLGtif>
CjBut9aueOxOha1cu0tOc
name: noha
email: nohashehab.iti@gmail.com
image: pic2.png
age: 31
```

7. **To know if your method=get or post**



```
new *
def create(request):
    # print(request)
    if request.method == 'POST':
        return HttpResponse("request post reci")
    return  render(request,  template_name: 'students/create.htm
```

8. **add name attributes in your html inputs so the data can be posted on the network Payload**

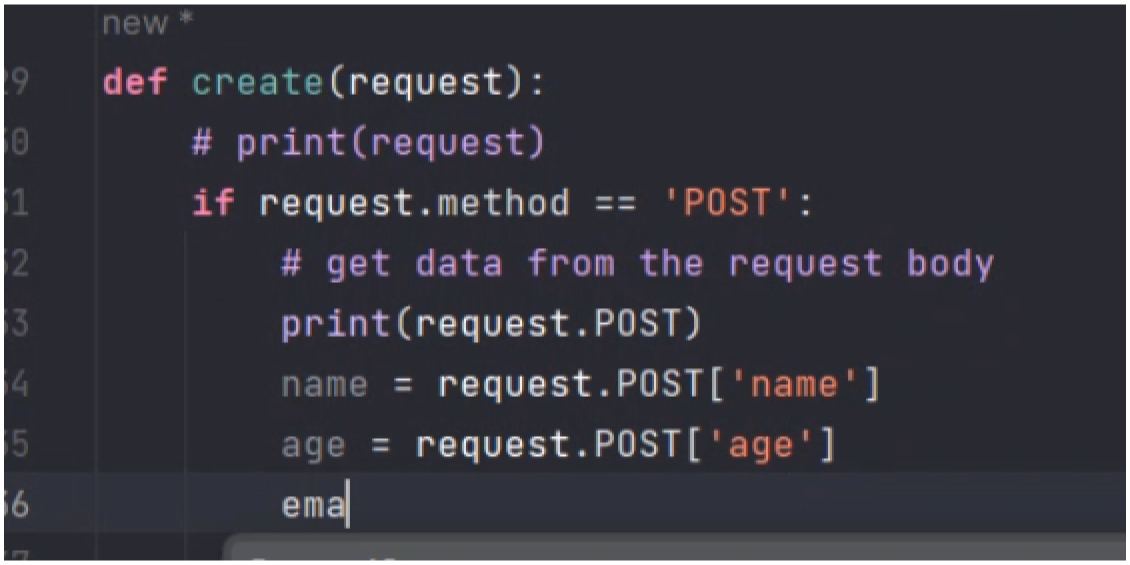


```
csrfmiddlewaretoken: sSt2fATk78OLGtifXhj2R1HP3NNeETVMCTUS8r
CjBut9aueOxOha1cu0tOc
name: noha
email: nohashehab.iti@gmail.com
image: pic2.png
age: 31
```

9. **Now, to post them to DB, get data from request body, then add it to DB**

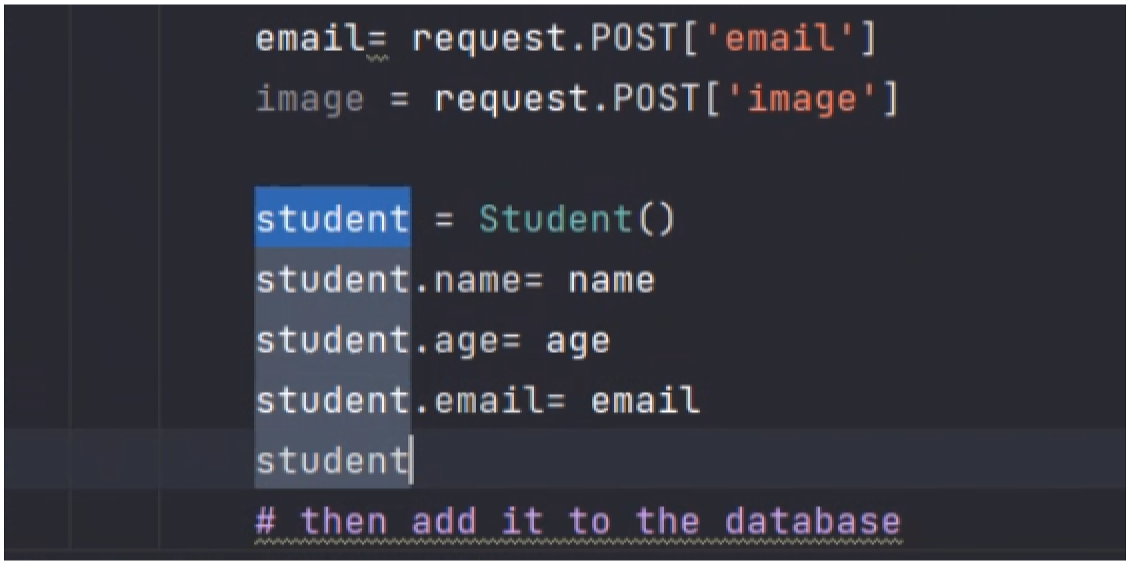
- **get it from request body**

(**request.POST**) .. POST is capitalized



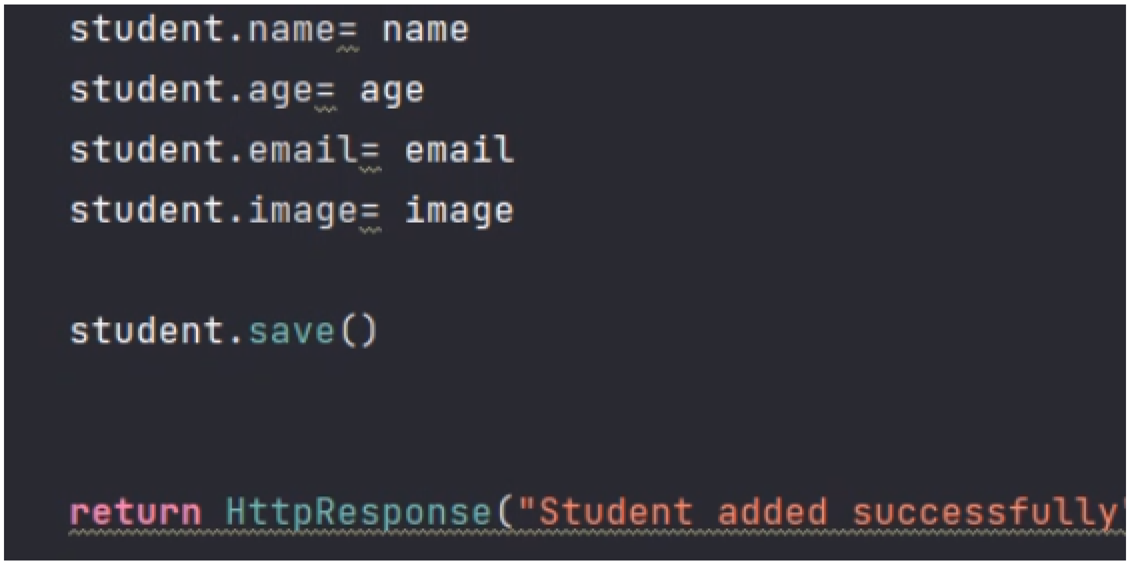
```
new *
def create(request):
    # print(request)
    if request.method == 'POST':
        # get data from the request body
        print(request.POST)
        name = request.POST['name']
        age = request.POST['age']
        ema
```

- **add it to DB**

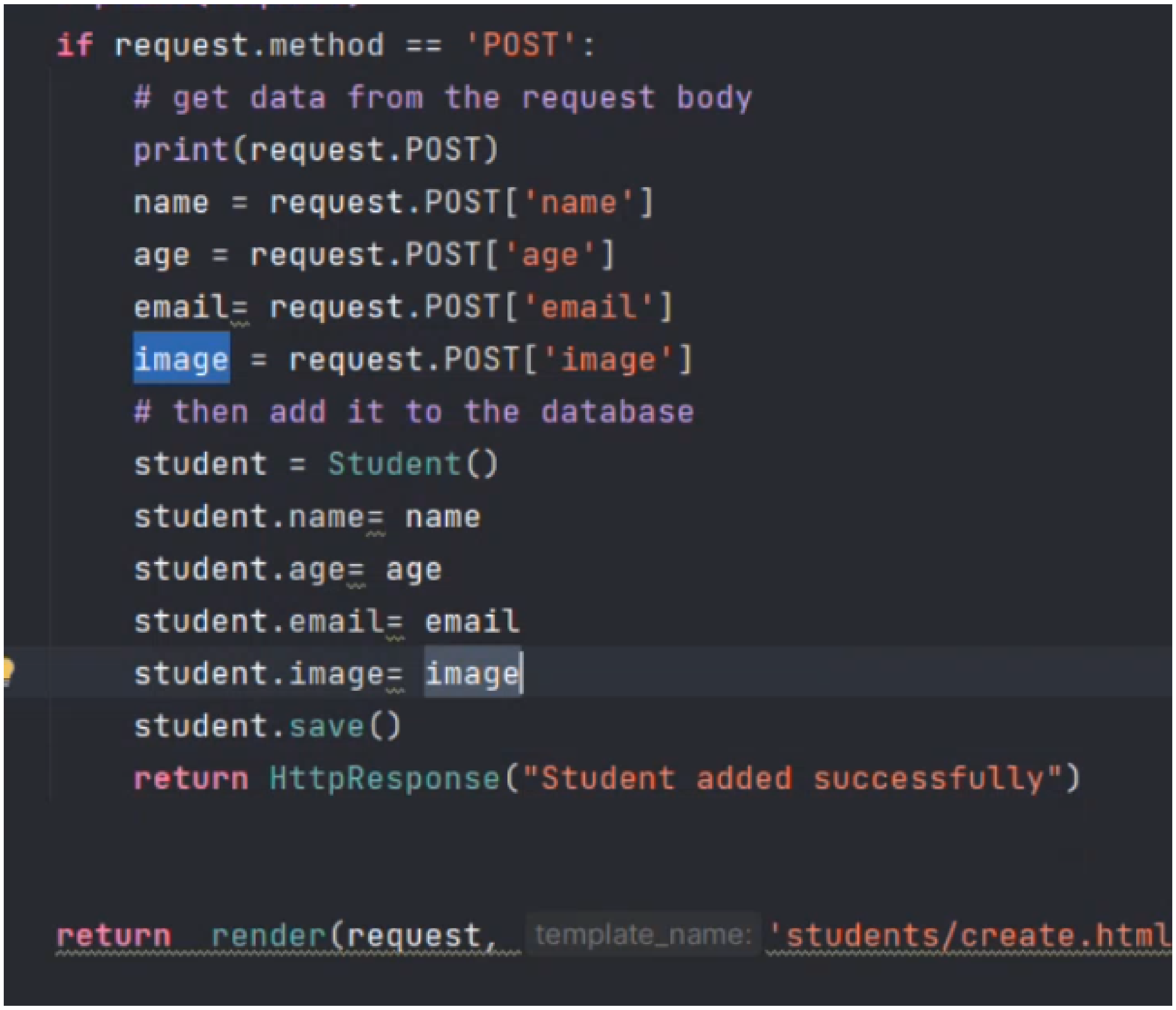


```
email= request.POST['email']
image = request.POST['image']

student = Student()
student.name= name
student.age= age
student.email= email
student
# then add it to the database
```

then student.save()



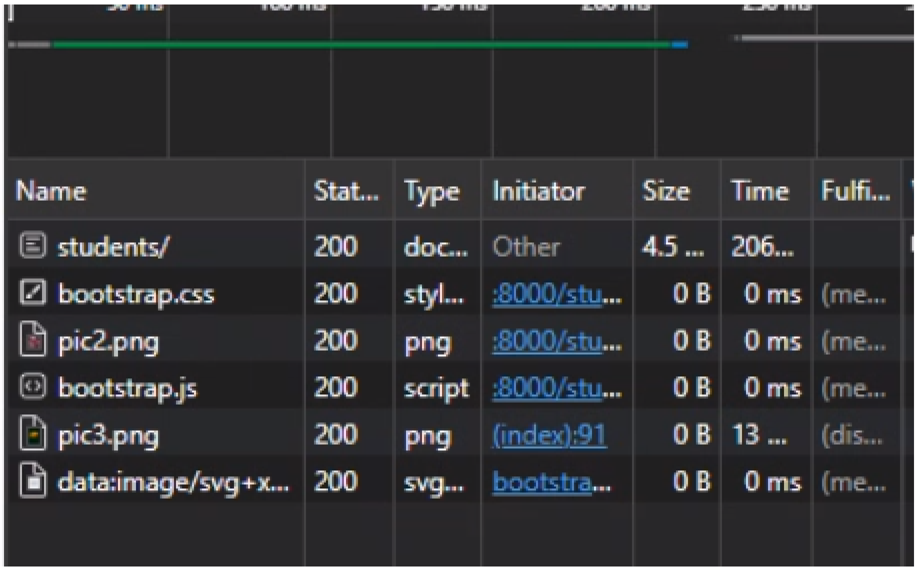
```
student.name= name
student.age= age
student.email= email
student.image= image

student.save()



return HttpResponse("Student added successfully
```



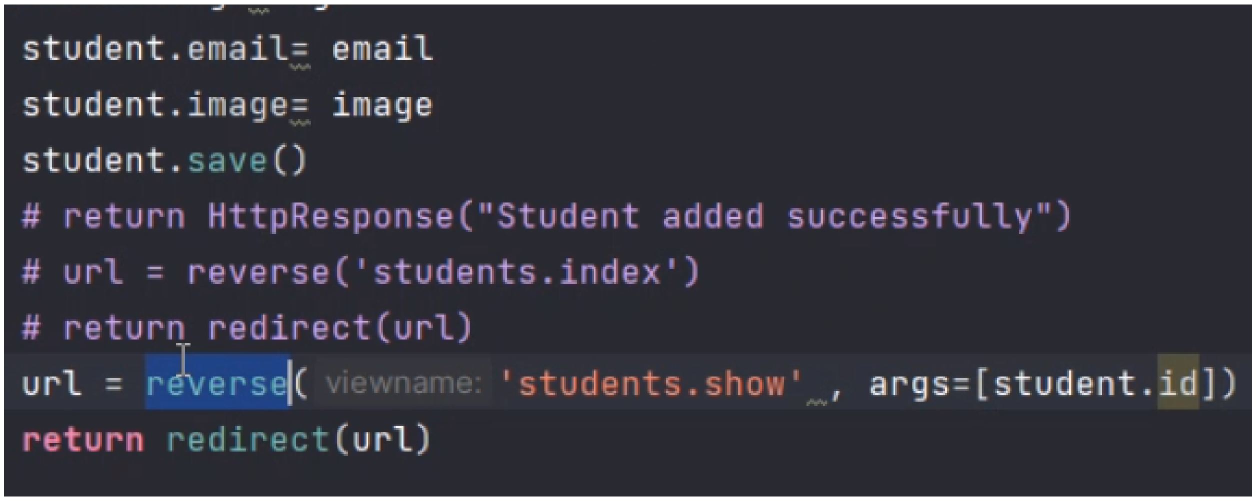
```python
    if request.method == 'POST':
        # get data from the request body
        print(request.POST)
        name = request.POST['name']
        age = request.POST['age']
        email= request.POST['email']
        image = request.POST['image']
        # then add it to the database
        student = Student()
        student.name= name
        student.age= age
        student.email= email
        student.image= image
        student.save()
        return HttpResponse("Student added successfully")


    return  render(request,  template_name: 'students/create.html
```

<u>Now if u submit data :</u>

a new student is added successfully to your DB and appears on your site page



Name	Stat...	Type	Initiator	Size	Time	Fulfi...
students/	200	doc...	Other	4.5 ...	206...	
bootstrap.css	200	styl...	:8000/stu...	0 B	0 ms	(me...
pic2.png	200	png	:8000/stu...	0 B	0 ms	(me...
bootstrap.js	200	script	:8000/stu...	0 B	0 ms	(me...
pic3.png	200	png	(index):91	0 B	13 ...	(dis...
data:image/svg+x...	200	svg...	bootstra...	0 B	0 ms	(me...



```
student.email= email
student.image= image
student.save()
# return HttpResponse("Student added successfully")
# url = reverse('students.index')
# return redirect(url)
url = reverse( viewname: 'students.show' , args=[student.id])
return redirect(url)
```

return redirect 3la el track

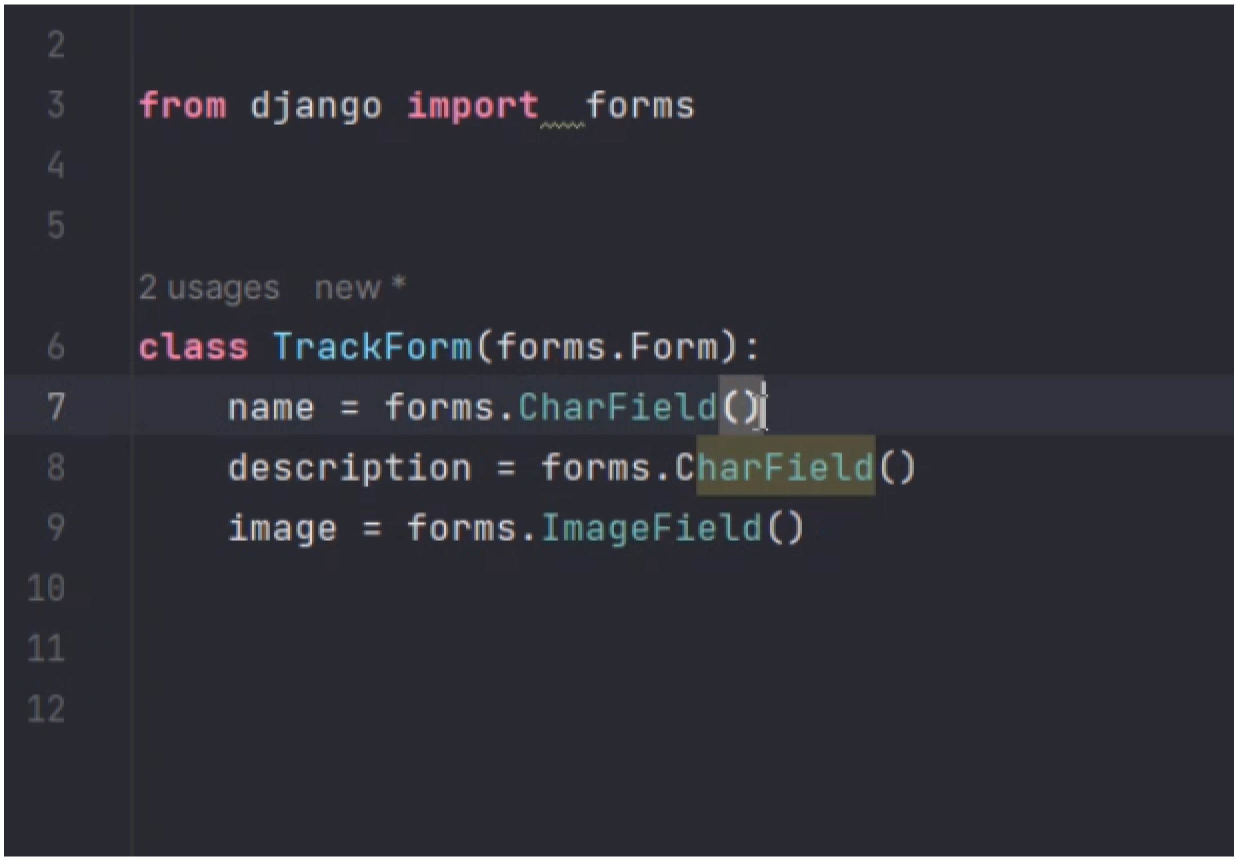
after break :

form validation

-Django made smth called **“Django Forms”**

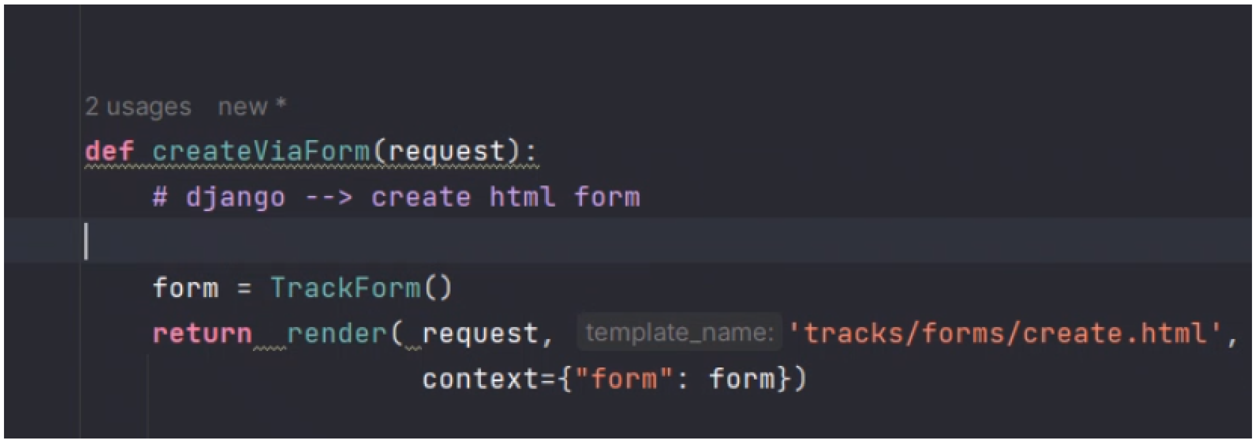
-inside your tapp, make : **forms.py**

```
from django import forms

class TrackForm(forms.Form):
  name=forms.charField()
  rest like model ....
```

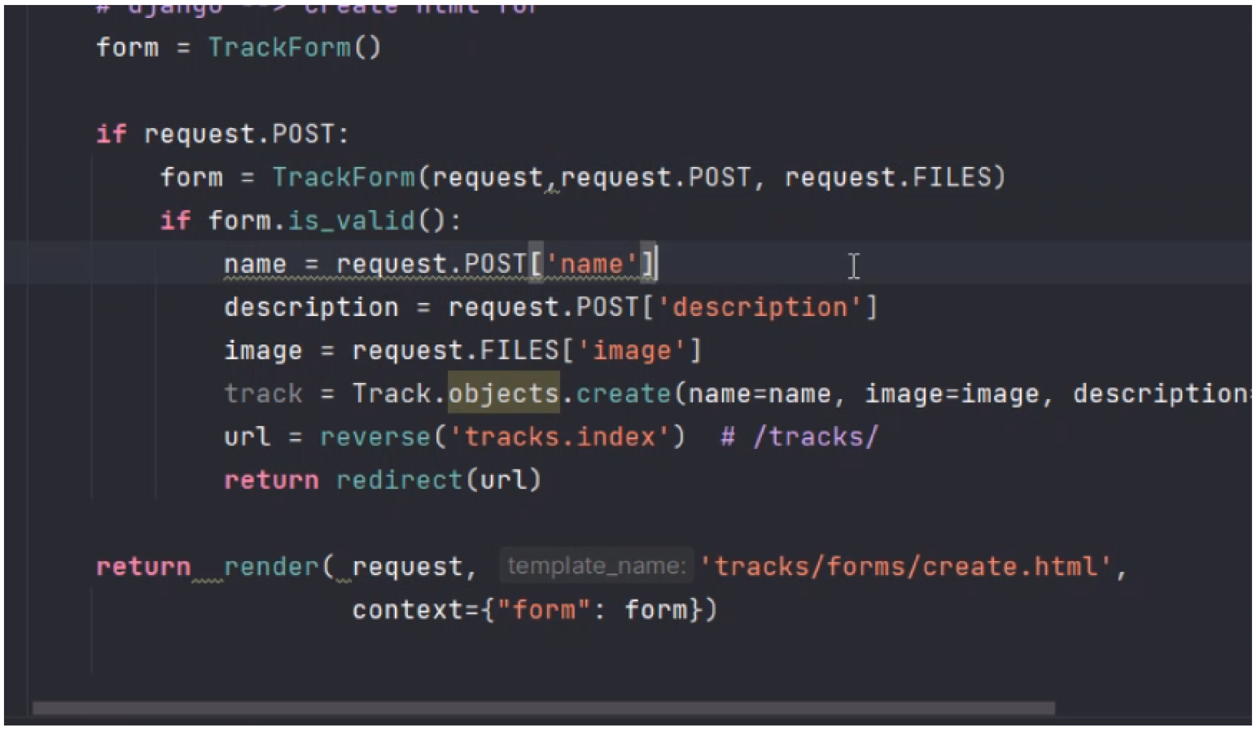


```python
from django import forms


2 usages   new *
class TrackForm(forms.Form):
    name = forms.CharField()
    description = forms.CharField()
    image = forms.ImageField()
```

in views :



```python
2 usages   new *
def createViaForm(request):
    # django --> create html form

    form = TrackForm()
    return render(request, template_name: 'tracks/forms/create.html',
                  context={"form": form})
```



```
    form = TrackForm()

    if request.POST:
        form = TrackForm(request,request.POST, request.FILES)
        if form.is_valid():
            name = request.POST['name']
            description = request.POST['description']
            image = request.FILES['image']
            track = Track.objects.create(name=name, image=image, description
            url = reverse('tracks.index')  # /tracks/
            return redirect(url)

    return  render( request,  template_name: 'tracks/forms/create.html',
                    context={"form": form})
```

Lab 3 :

-Using forms to : create product - edit product - modify image in product model to be image field (dont put url , instead : upload image)

-dont work in model forms

-modify products model to include category

-category has a model - index - show (1 to many)

-when u click show category, its page will display products in this category

-category has the following features : name - description - image