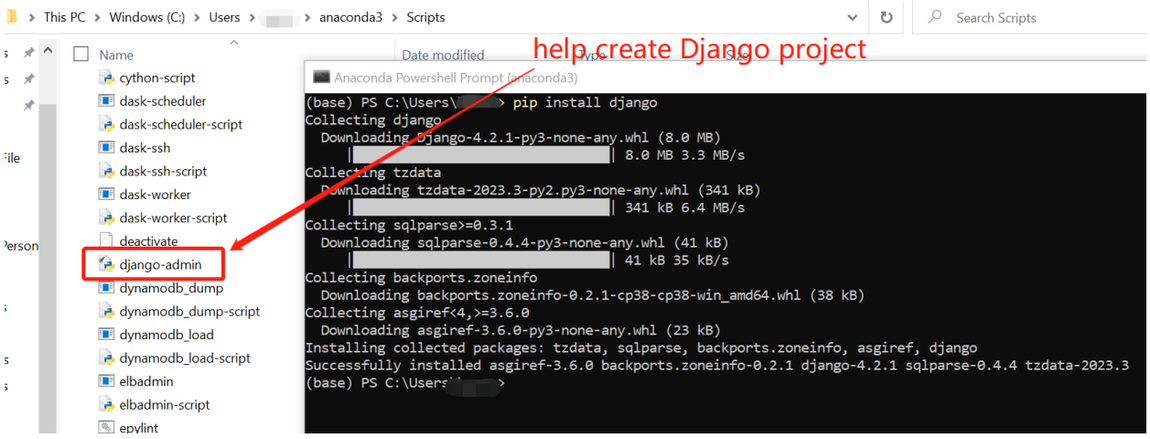
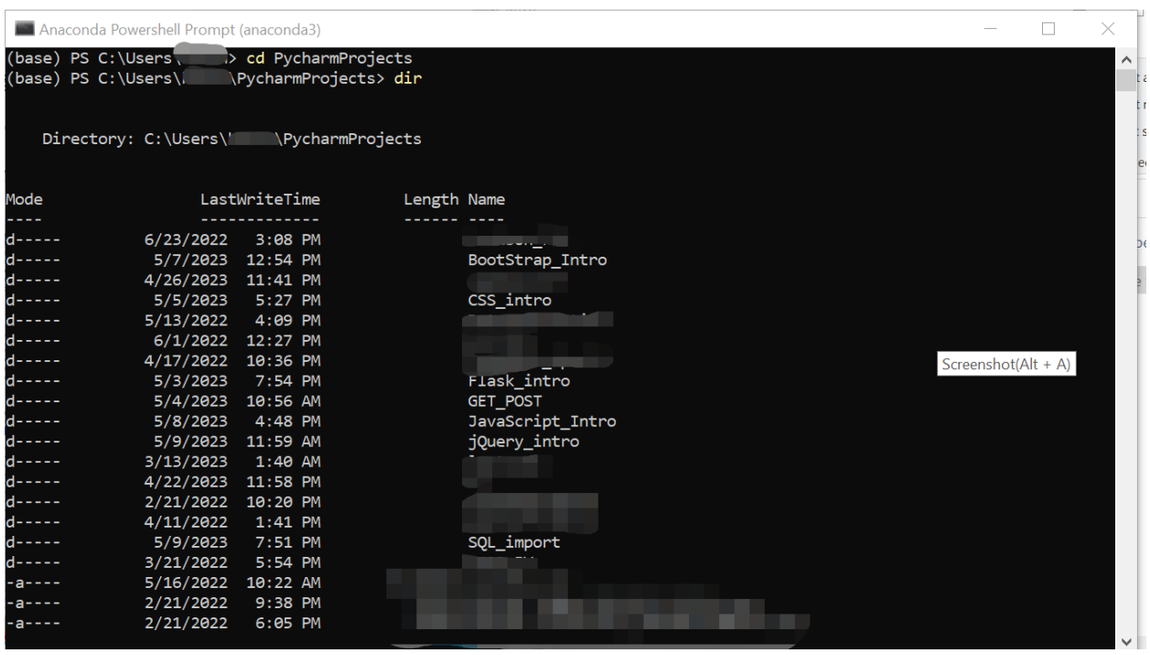
Django

1. install Django



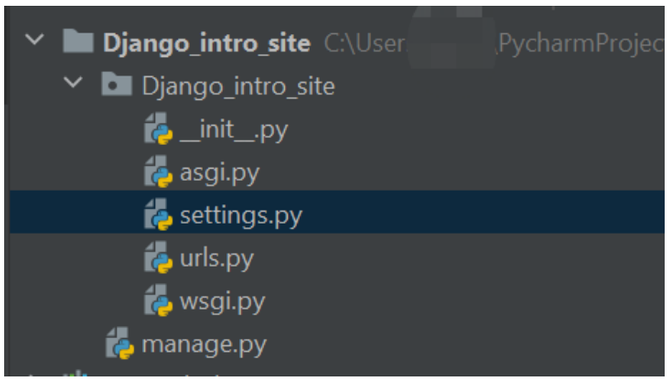
2. create Django project




```
(base) PS C:\Users\[redacted]\PycharmProjects> cd Django_intro
(base) PS C:\Users\[redacted]\PycharmProjects\Django_intro> pwd

Path
----
C:\Users\[redacted]\PycharmProjects\Django_intro


(base) PS C:\Users\[redacted]\PycharmProjects\Django_intro> C:/Users/[redacted]/anaconda3/Scripts/django-admin startproject Django
_intro_site
(base) PS C:\Users\[redacted]\PycharmProjects\Django_intro>
```

1. get into the direction you want to put the project
2. create project using django-admin
3. please make sure: 1) slash “/” 2) please do not add “.exe”

2. file introduction



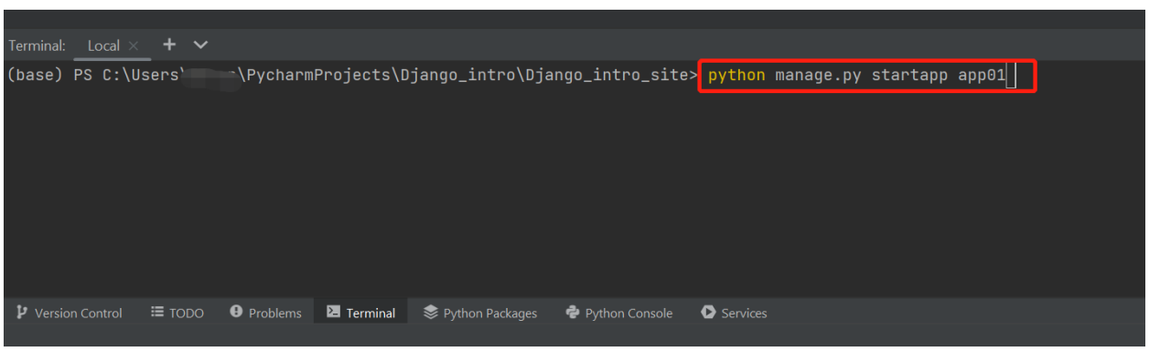
1. manage.py: project management, app dev,
2. asgi.py, wsgi.py receive web request, do not make any changes
3. urls.py URL→function (@app.route())…
4. settings.py setting

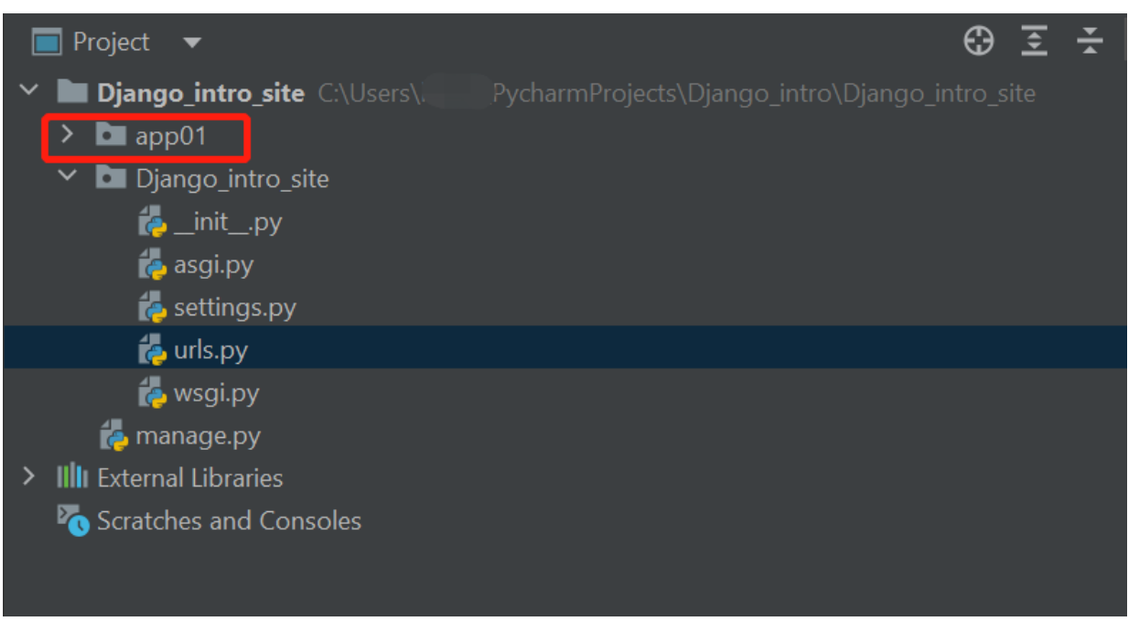
3. app

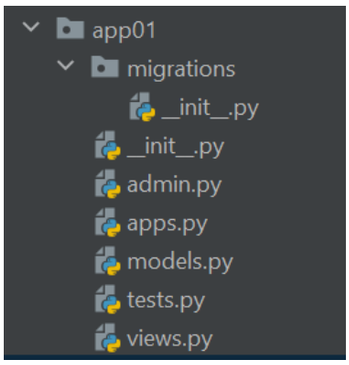
create app

open terminal

python manage.py startapp appname

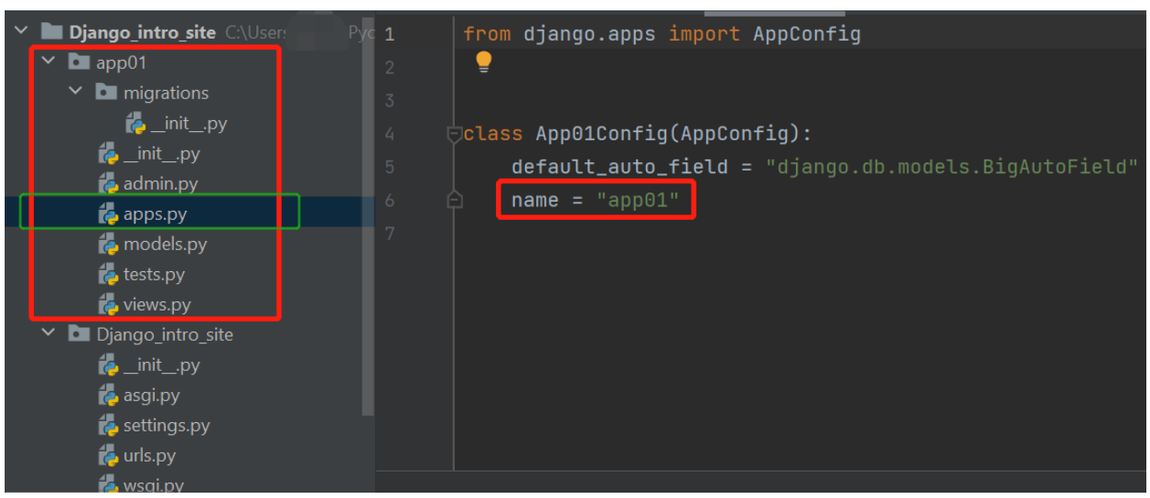


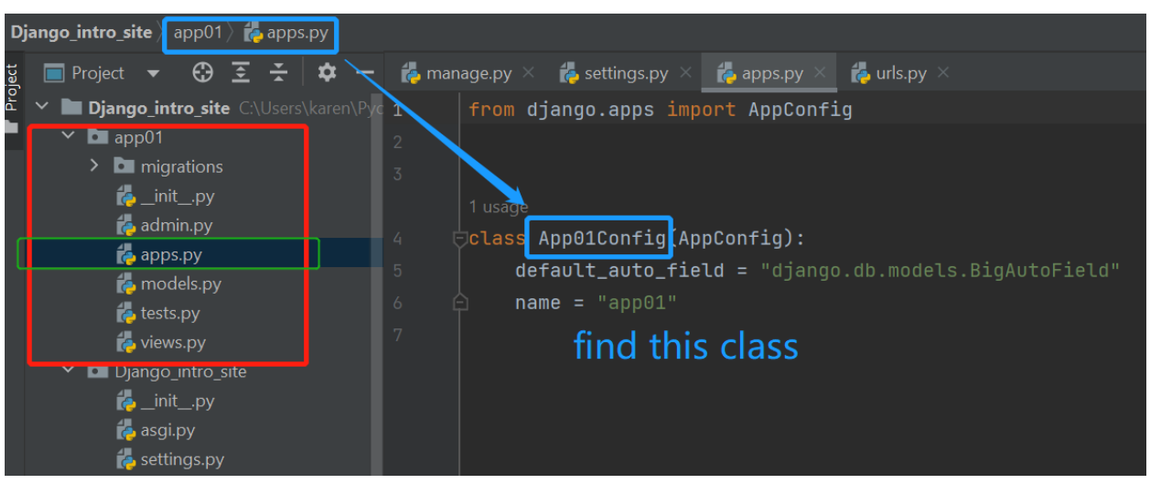




4. quick start

1) register app

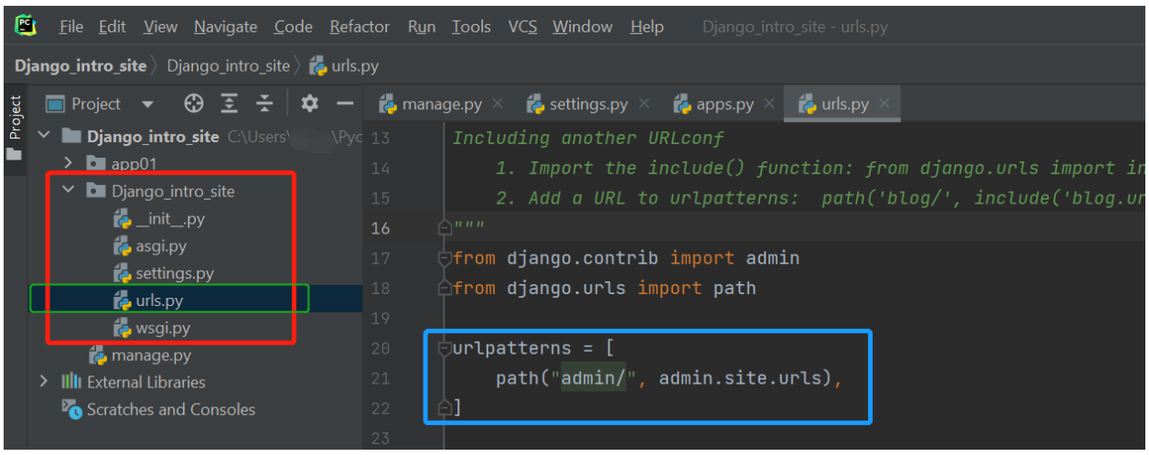


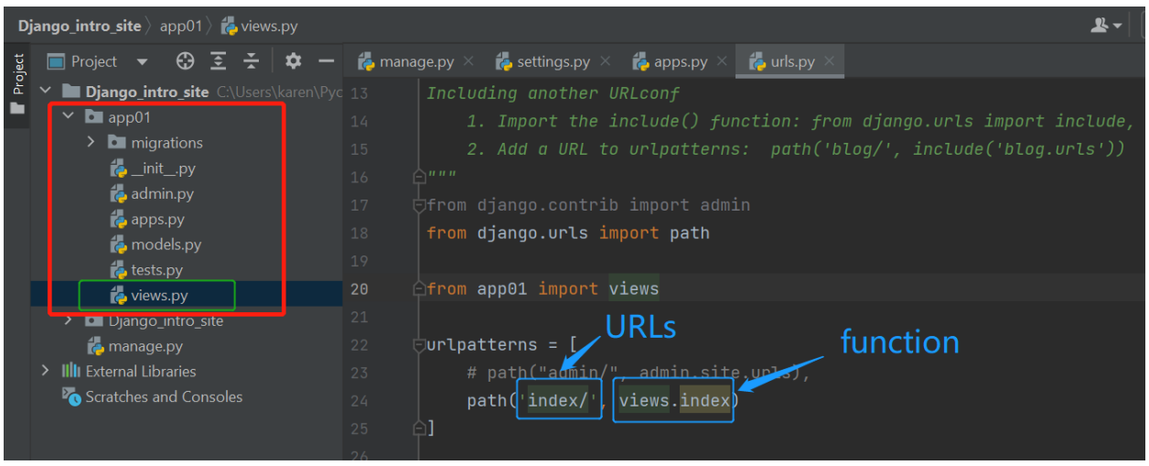


ADD

```
"app01.apps.App01Config"
```

2) build relationships between urls and functions

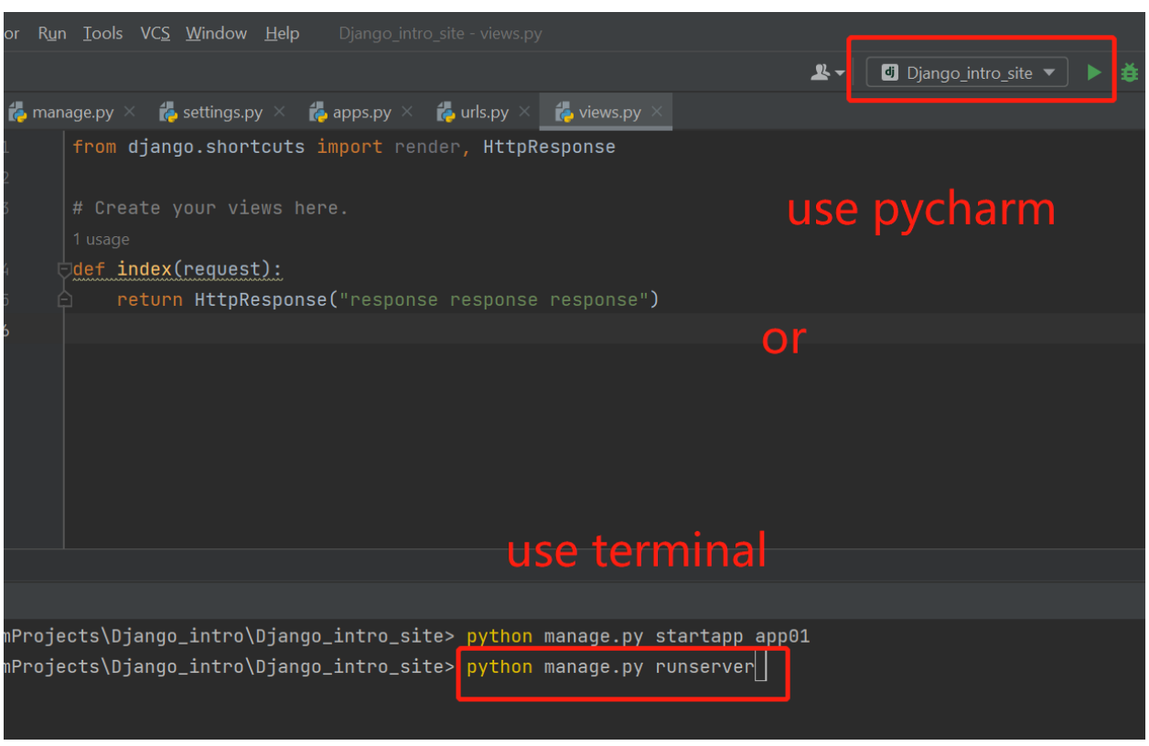


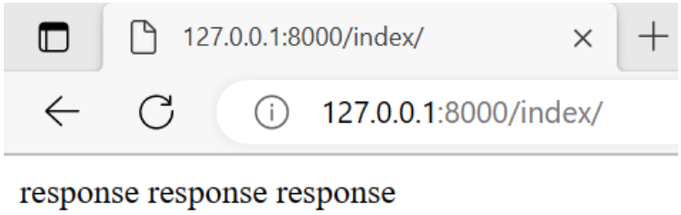


3) create function in views.py

```
from django.shortcuts import render, HttpResponse

# Create your views here.
def index(request):
    return HttpResponse("response response response")
```

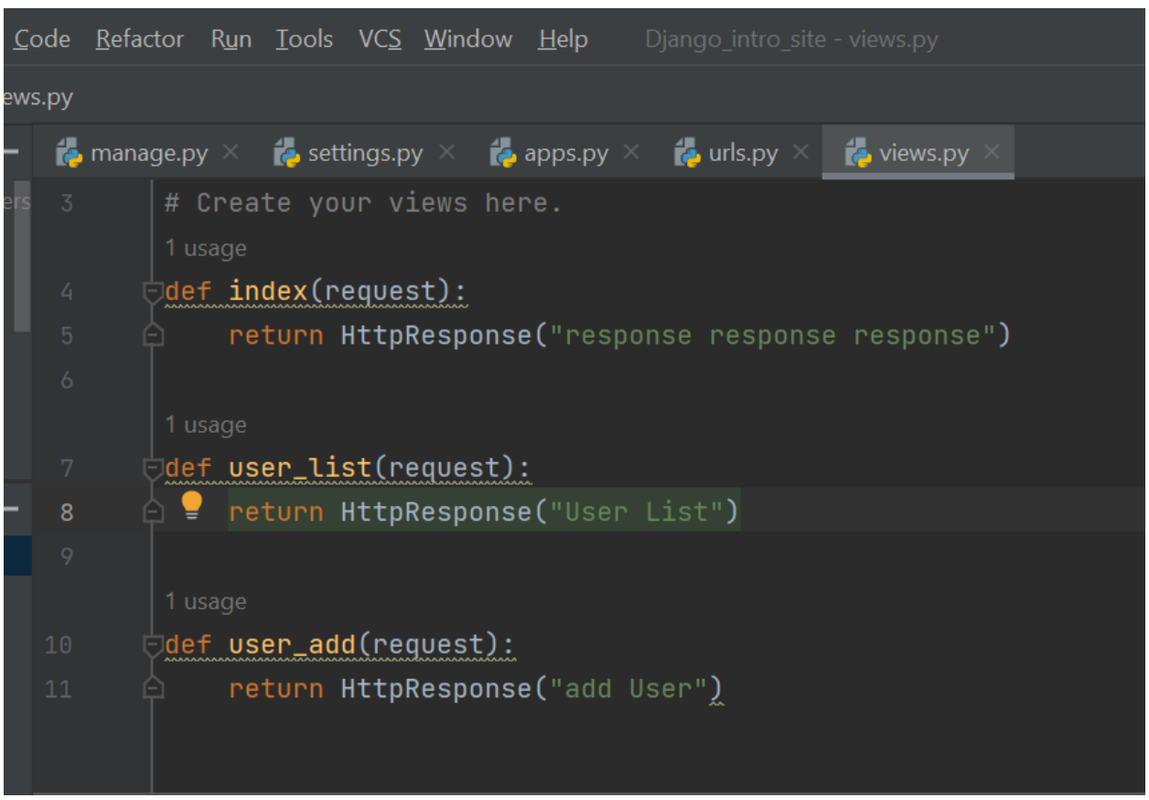
4）run





5. add previous SQL example

1) add function & bulid relationships



```python
    1. Import the include() function: from django.urls import in
    2. Add a URL to urlpatterns:  path('blog/', include('blog.ur
"""
from django.contrib import admin
from django.urls import path

from app01 import views

urlpatterns = [
    # path("admin/", admin.site.urls),
    path('index/', views.index),
    path('user/list/', views.user_list),
    path('user/add/', views.user_add),
]
```

2) add html files

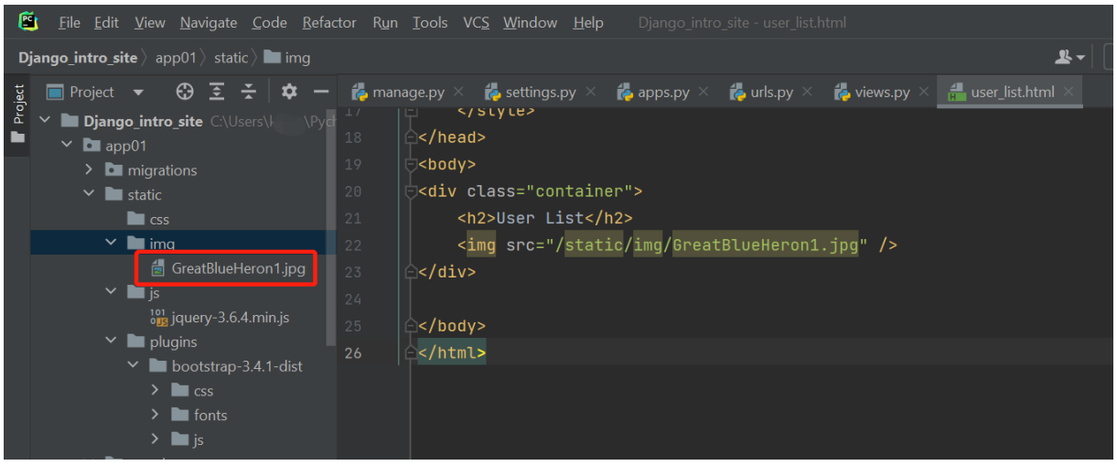
```python
# Create your views here.
1 usage
def index(request):
    return HttpResponse("response response response")

1 usage
def user_list(request):
    return render(request, "user_list.html")

1 usage
def user_add(request):
    return HttpResponse(request, "add_user.html")
```

add html files in app/templates

3) import css & js

similarly, use folder static.

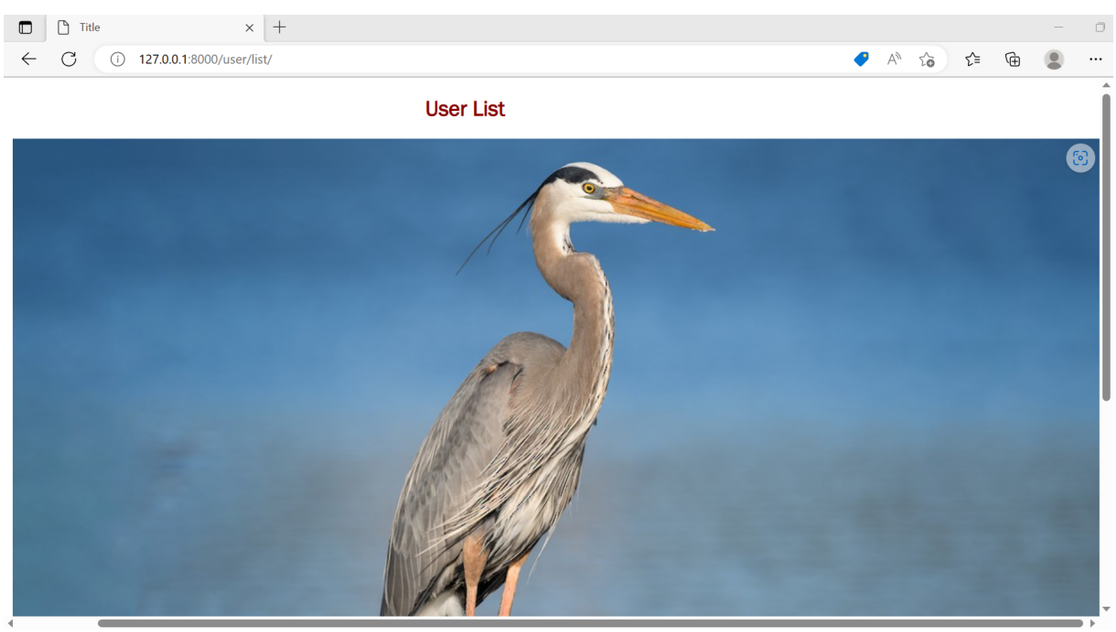
> if cannot loading image



use the following code, if your STATIC_URL in setting.py is set to "static/"

```
{% load static %}
<img src="{% static "img/GreatBlueHeron1.jpg" %}" />
```

```
<div class="container">
    <h2>User List</h2>
    {% load static %}
    <img src="{% static "img/GreatBlueHeron1.jpg" %}" />
</div>
```



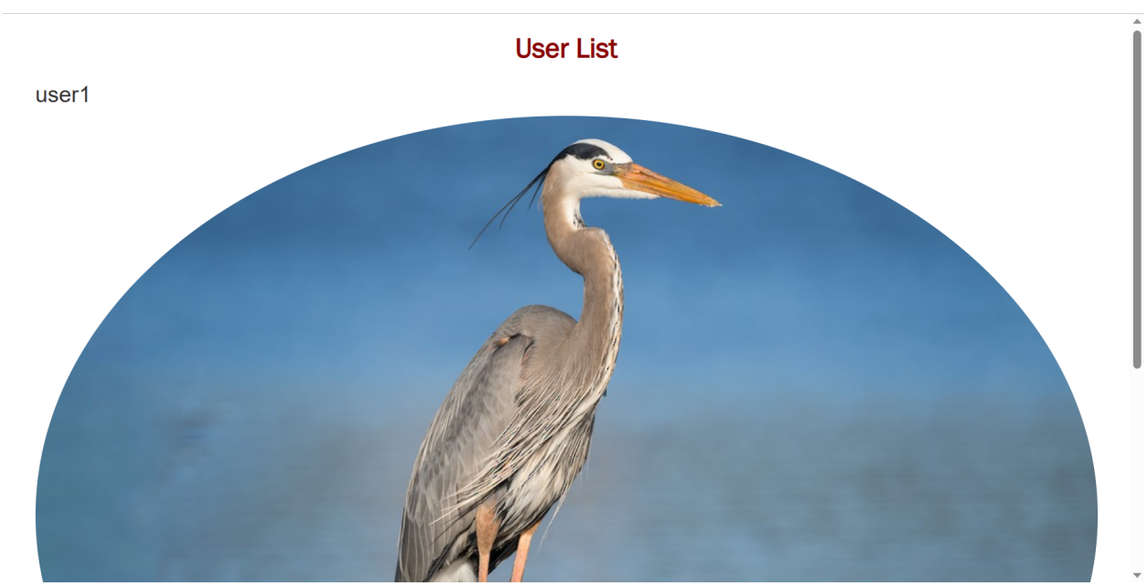
This only work when run the project.

> **django not displaying image**
>
> I started with a django app and tried displaying an image in the home page. But the image is not getting displayed.
>
> https://stackoverflow.com/questions/20848640/django-not-displaying-image

4) placeholder {{ }}

```
<!-- user_list.html -->
{% load static %}

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
    <link rel="stylesheet" href="{% static 'plugins/bootstrap-3.4.1-dist/css/bootstrap.css' %}">
    <style>
        .container h2 {
            color: darkred;
            font-weight: bold;
            font-family: "Franklin Gothic Book";
            text-align: center;
        }
    </style>
</head>
<body>
<div class="container">
    <h2>User List</h2>
    <h3>{{ u1 }}</h3>
    <img src="{% static "img/GreatBlueHeron1.jpg" %}" class="img-circle img-responsive"/>


</div>

<script src="{% static 'js/jquery-3.6.4.min.js' %}"></script>
<script src="{% static 'plugins/bootstrap-3.4.1-dist/js/bootstrap.js' %}"></script>
</body>
</html>
```

```
# views.py
from django.shortcuts import render, HttpResponse

# Create your views here.
import pymysql
def index(request):
    return HttpResponse("response response response")

def user_list(request):
    user1 = 'user1'
    return render(request, "user_list.html", {'u1': user1})

def user_add(request):
    return HttpResponse(request, "add_user.html")
```



multiple placeholder

.index

```
{% load static %}

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
    <link rel="stylesheet" href="{% static 'plugins/bootstrap-3.4.1-dist/css/bootstrap.css' %}">
    <style>
        .container h2 {
            color: darkred;
            font-weight: bold;
            font-family: "Franklin Gothic Book";
            text-align: center;
        }
    </style>
</head>
<body>
<div class="container">
    <h2>User List</h2>
    <h3>{{ u1 }}</h3>
    <div>{{ r1.0 }}</div>
    <div>{{ r1.1 }}</div>
    <div>{{ r1.2 }}</div>
    <img src="{% static "img/GreatBlueHeron1.jpg" %}" class="img-circle img-responsive"/>


</div>

<script src="{% static 'js/jquery-3.6.4.min.js' %}"></script>
<script src="{% static 'plugins/bootstrap-3.4.1-dist/js/bootstrap.js' %}"></script>
</body>
</html>
```

```
from django.shortcuts import render, HttpResponse

# Create your views here.
import pymysql
def index(request):
    return HttpResponse("response response response")

def user_list(request):
    user1 = 'user1'
    roles = ['role1', 'role2', 'role3']
    return render(request, "user_list.html", {'u1': user1, 'r1': roles})

def user_add(request):
    return HttpResponse(request, "add_user.html")
```

for loop

```
def user_list(request):
    user1 = 'user1'
    roles = ['role1', 'role2', 'role3', 'role4', 'role5']
    return render(request, "user_list.html", {'u1': user1, 'r1': roles})
```

```
{% load static %}

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
    <link rel="stylesheet" href="{% static 'plugins/bootstrap-3.4.1-dist/css/bootstrap.css' %}">
    <style>
        .container h2 {
            color: darkred;
            font-weight: bold;
            font-family: "Franklin Gothic Book";
            text-align: center;
        }
    </style>
</head>
<body>
<div class="container">
    <h2>User List</h2>
    <h3>{{ u1 }}</h3>
    <div>
```

```
        {% for item in r1 %}
            <span>{{ item }}</span>
        {% endfor %}
    </div>
    <img src="{% static "img/GreatBlueHeron1.jpg" %}" class="img-circle img-responsive"/>


</div>

<script src="{% static 'js/jquery-3.6.4.min.js' %}"></script>
<script src="{% static 'plugins/bootstrap-3.4.1-dist/js/bootstrap.js' %}"></script>
</body>
</html>
```

User List

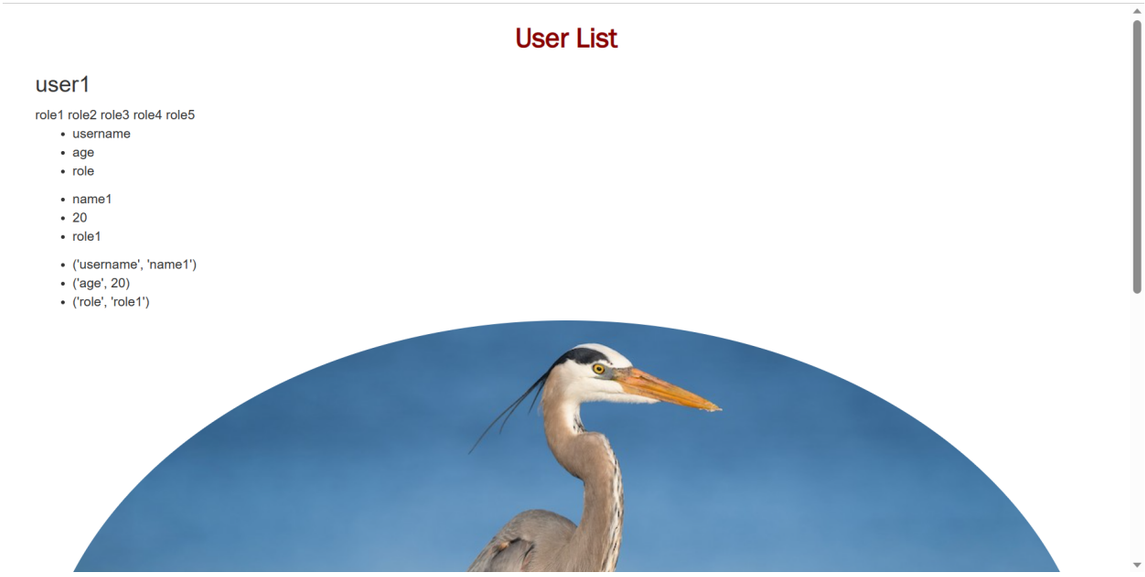
user1

role1 role2 role3 role4 role5

data in dictionary

```
<div>{{ u2.name }}</div>
    <ul>
        {% for item in u2.keys %}
            <li>{{ item }}</li>
        {% endfor %}
    </ul>
    <ul>
        {% for item in u2.values %}
            <li>{{ item }}</li>
        {% endfor %}
    </ul>
    <ul>
        {% for item in u2.items %}
            <li>{{ item }}</li>
        {% endfor %}
    </ul>
```

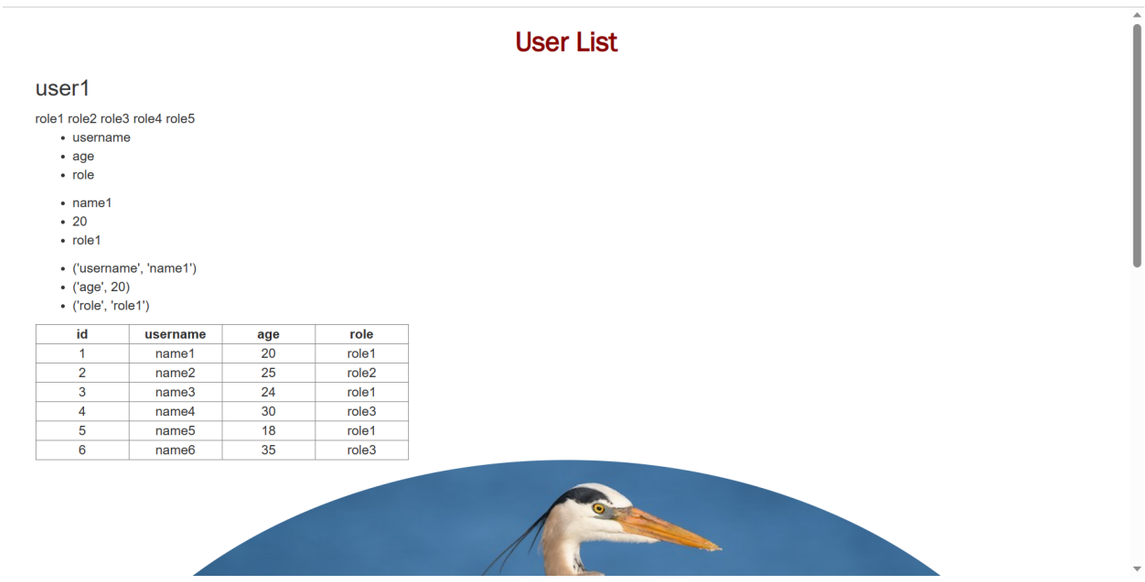
```
def user_list(request):
    user1 = 'user1'
    roles = ['role1', 'role2', 'role3', 'role4', 'role5']
    user_info = {'username': 'name1', 'age': 20, 'role': 'role1'}
    return render(request, "user_list.html", {'u1': user1, 'r1': roles, 'u2': user_info})
```



index

```
{% load static %}

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
    <link rel="stylesheet" href="{% static 'plugins/bootstrap-3.4.1-dist/css/bootstrap.css' %}">
    <style>
        .container h2 {
            color: darkred;
            font-weight: bold;
            font-family: "Franklin Gothic Book";
            text-align: center;
        }
        .container th {
            width: 100px;
            text-align: center;
        }
        .container td {
            text-align: center;
        }
    </style>
</head>
<body>
<div class="container">
    <h2>User List</h2>
    <h3>{{ u1 }}</h3>
    <div>
        {% for item in r1 %}
            <span>{{ item }}</span>
        {% endfor %}
    </div>
    <div>{{ u2.name }}</div>
    <ul>
        {% for item in u2.keys %}
            <li>{{ item }}</li>
        {% endfor %}
    </ul>
    <ul>
        {% for item in u2.values %}
            <li>{{ item }}</li>
        {% endfor %}
    </ul>
    <ul>
        {% for item in u2.items %}
            <li>{{ item }}</li>
        {% endfor %}
    </ul>

    <div>
        <table border="1">
            <thead>
                <tr>
                    <th>id</th>
                    <th>username</th>
                    <th>age</th>
                    <th>role</th>
                </tr>
            </thead>
            <tbody>
                {% for row in u3 %}
```

```
                    <tr>
                        <td>{{ forloop.counter }}</td>
                        <td>{{ row.username }}</td>
                        <td>{{ row.age }}</td>
                        <td>{{ row.role }}</td>
                    </tr>
                {% endfor %}
            </tbody>

        </table>
    </div>

    <img src="{% static "img/GreatBlueHeron1.jpg" %}" class="img-circle img-responsive"/>


</div>

<script src="{% static 'js/jquery-3.6.4.min.js' %}"></script>
<script src="{% static 'plugins/bootstrap-3.4.1-dist/js/bootstrap.js' %}"></script>
</body>
</html>
```



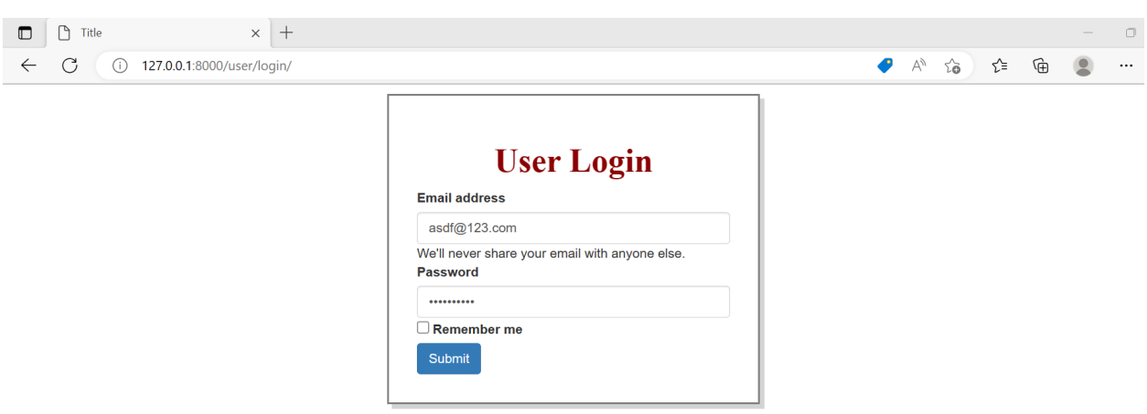
request

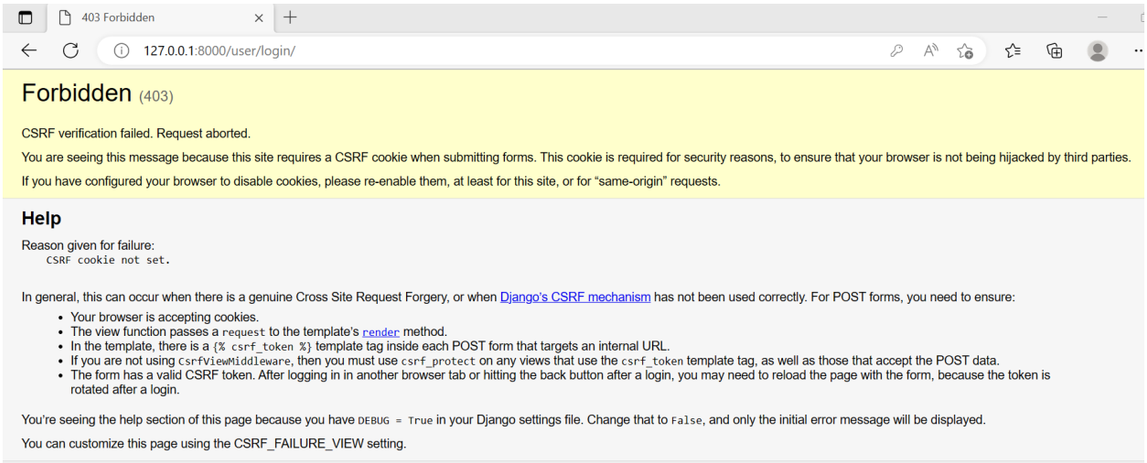
```
request.method
# what kind of request
request.GET
# GET data from URLs
request.POST
# submit data
return redirect("url")
# redirect to "url"
```

ex: User Login

use previous code

BootStrap Example

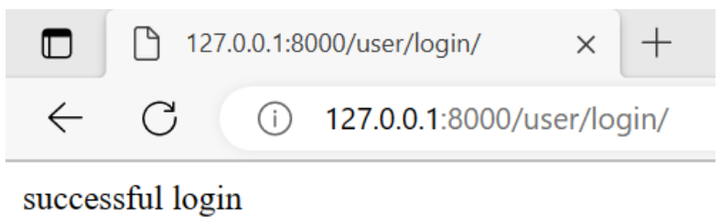




should add CSRF token in <form></form>

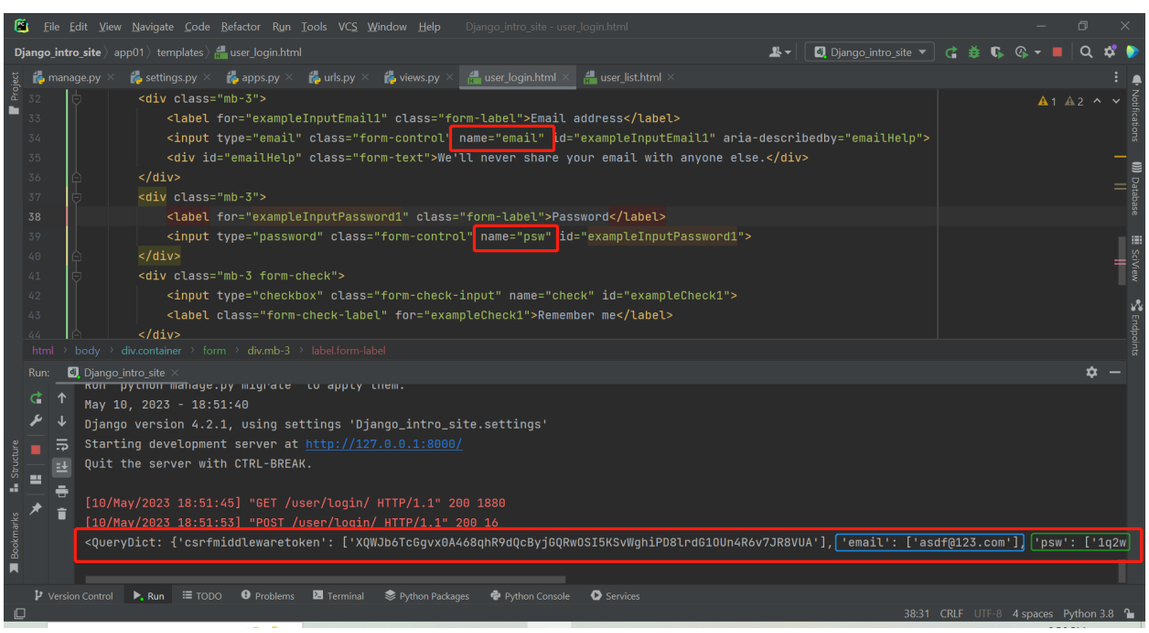
```
{% csrf_token %}
```

also, previous code cannot return any other variable except token:




```
[10/May/2023 18:14:56] "POST /user/login/ HTTP/1.1" 200 16
<QueryDict: {'csrfmiddlewaretoken': ['lP2CI8JqGL0DiErflsakpssOB25mkZ15gHbDpxMuhNkJqpOmT3HnDjtIvBvxC3pR']}>
```

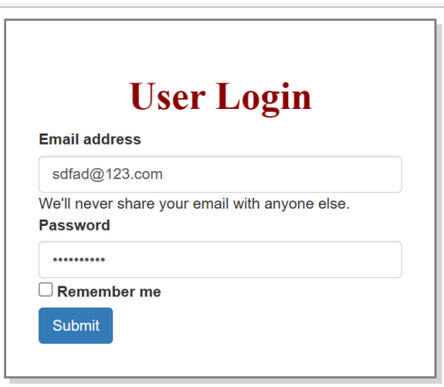
to get other variables, should add name to each input label:

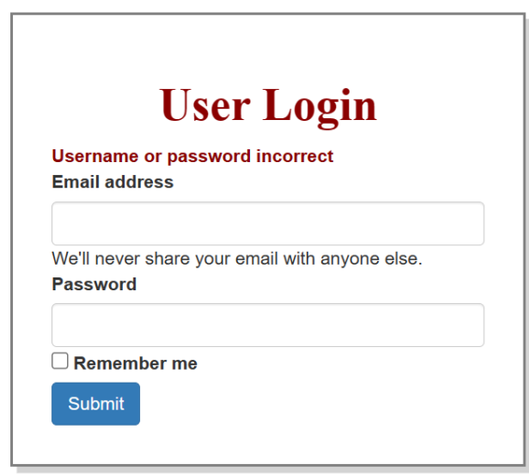



```
def user_login(request):
    if request.method == 'GET':
        return render(request, 'user_login.html')
    else:
        print(request.POST)
        user_name = request.POST.get('email')
        psw = request.POST.get('psw')
```

```
        if user_name == 'asdf@123.com' and psw == '1q2w3elkjh':
            return redirect('https://www.allaboutbirds.org/news/')
        else:
            return render(request, 'user_login.html', {'error_message': 'Username or password incorrect'})
```

```
{% load static %}

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
    <link rel="stylesheet" href="{% static 'plugins/bootstrap-3.4.1-dist/css/bootstrap.css' %}">
    <style>
        .w_title {
            color: darkred;
            font-family: "Eras Light ITC";
            font-weight: bold;
        }

        .container {
            width: 400px;
            padding: 30px;
            margin: 0 auto;
            margin-top: 10px;
            border: 2px solid gray;
        }
    </style>
</head>
<body>

<div class="container" style="box-shadow: 5px 5px lightgray">
    <h1 class="w_title" style="text-align: center;">User Login</h1>
    <form method="post" action="/user/login/">
        {% csrf_token %}
        <div style="color: darkred; font-weight: bold">{{ error_message }}</div>
        <div class="mb-3">
            <label for="exampleInputEmail1" class="form-label">Email address</label>
            <input type="email" class="form-control" name="email" id="exampleInputEmail1" aria-describedby="emailHelp">
            <div id="emailHelp" class="form-text">We'll never share your email with anyone else.</div>
        </div>
        <div class="mb-3">
            <label for="exampleInputPassword1" class="form-label">Password</label>
            <input type="password" class="form-control" name="psw" id="exampleInputPassword1">
        </div>
        <div class="mb-3 form-check">
            <input type="checkbox" class="form-check-input" name="check" id="exampleCheck1">
            <label class="form-check-label" for="exampleCheck1">Remember me</label>
        </div>
        <button type="submit" class="btn btn-primary">Submit</button>
    </form>
</div>

<script src="{% static 'js/jquery-3.6.4.min.js' %}"></script>
<script src="{% static 'plugins/bootstrap-3.4.1-dist/js/bootstrap.js' %}"></script>
</body>
</html>
```

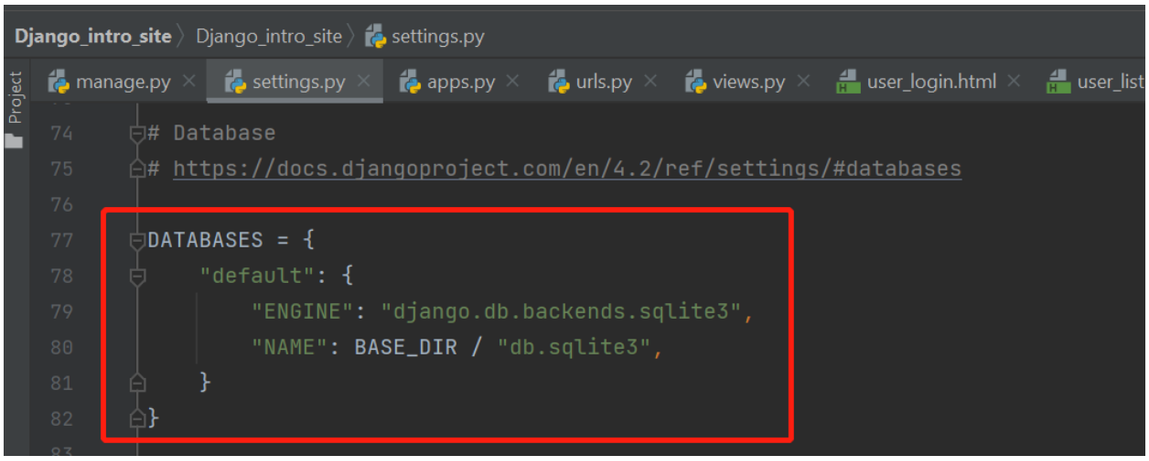




5) connect to database

💡 pip install mysqlclient

1. change database settings



Django

The web framework for perfectionists with deadlines.

https://docs.djangoproject.com/en/4.2/ref/databases/

```
# settings.py
DATABASES = {
    "default": {
        "ENGINE": "django.db.backends.mysql",
        "OPTIONS": {
            "read_default_file": "/path/to/my.cnf",
        },
    }
}
```

```
# my.cnf
[client]
database = NAME
user = USER
password = PASSWORD
default-character-set = utf8
```

ex:

```
DATABASES = {
    "default": {
        "ENGINE": "django.db.backends.mysql",
        "NAME": 'web_intro',
        "USER": 'root',
        "PASSWORD": '****',
        "HOST": '127.0.0.1',
        "PORT": 3306,
    }
}
```

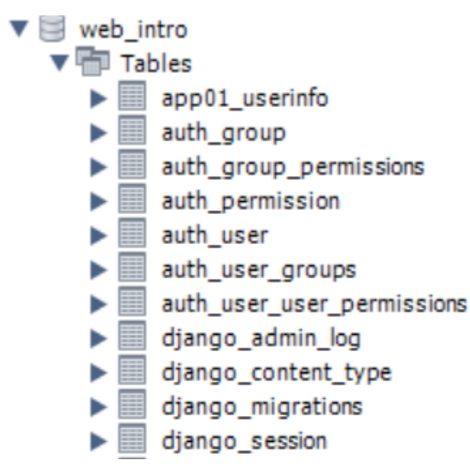
create table

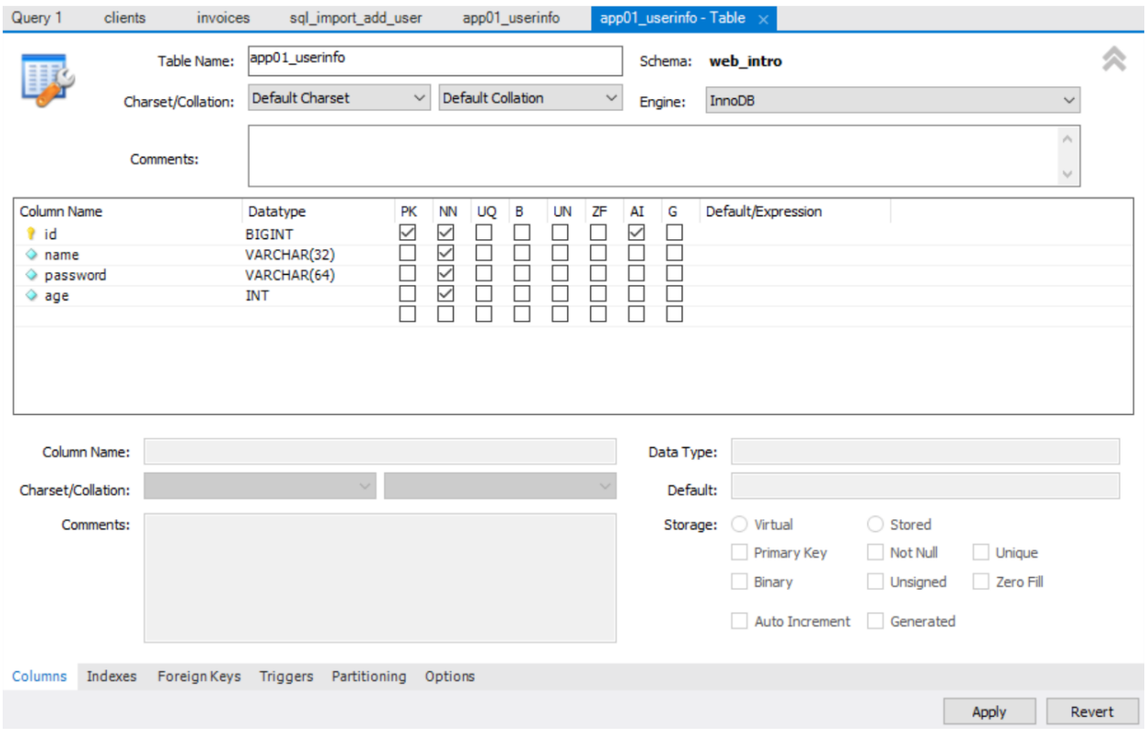
```
"""
create a table named app01_userinfo
SQL:
CREATE TABLE app01_userinfo(
    id bigint autoincrement primary key,
    name VARCHAR(32),
    password VARCHAR(32),
    age int
```

```
)
"""
class UserInfo(models.Model):
    name = models.CharField(max_length=32)
    password = models.CharField(max_length=64)
    age = models.IntegerField()
```

```
# terminal
python manage.py makemigrations

python manage.py migrate
```

```
Terminal:  Local
(base) PS C:\Users\[redacted]\PycharmProjects\Django_intro\Django_intro_site> python manage.py makemigrations
Migrations for 'app01':
  app01\migrations\0001_initial.py
    - Create model UserInfo
(base) PS C:\Users\[redacted]\PycharmProjects\Django_intro\Django_intro_site> python manage.py migrate
  Applying auth.0006_require_contenttypes_0002... OK
  Applying auth.0007_alter_validators_add_error_messages... OK
  Applying auth.0008_alter_user_username_max_length... OK
  Applying auth.0009_alter_user_last_name_max_length... OK
  Applying auth.0010_alter_group_name_max_length... OK
  Applying auth.0011_update_proxy_permissions... OK
  Applying auth.0012_alter_user_first_name_max_length... OK
  Applying sessions.0001_initial... OK
(base) PS C:\Users\[redacted]\PycharmProjects\Django_intro\Django_intro_site>
```





everytime you want to add tables, change tables (remove col or add col, always use previous two lines in terminal).

If you want to add cols to an exsist table, it might cause problems because other columns have already have values while this new col do not. You can simply make a default value for this very column. (default =) Or, we can allow this column to be NA by using (null=True, blank=True)

insert value into table

```
UserInfo.objects.create(name='name1', password='8uhuyg7y', age=25)
```

delete

```
# # delete record where id = 3
UserInfo.objects.filter(id=3).delete()
# # delete all records in the table
UserInfo.objects.all().delete()
```

find

```
# get data
UserInfo.objects.filter(id=1)
UserInfo.objects.all()
"""
SELECT *
FROM app01_userInfo
WHERE age=16
LIMIT 1
"""
UserInfo.objects.filter(age=16).first()
```

update

```
# update
UserInfo.objects.filter(id=1).update(password='kjhgf789')
```

6) get data from database

```
from django.shortcuts import render, HttpResponse, redirect
from app01.models import UserInfo

def user_info(request):
```

```
    data_list = UserInfo.objects.all()
    print(data_list)
    return render(request, 'user_info.html', {"data_list": data_list})
```

```
[11/May/2023 20:34:26] "GET /favicon.ico HTTP/1.1" 404 2561
<QuerySet [<UserInfo: UserInfo object (1)>, <UserInfo: UserInfo object (2)>, <UserInfo: UserInfo object (3)>, <UserInfo: UserInfo object (4)>]>
[11/May/2023 20:34:55] "GET /user/info/ HTTP/1.1" 200 701
```

what we get directly are objects

```
<tbody>
{% for obj in data_list %}
    <tr>
    <th>obj.id</th>
    <th>obj.user</th>
    <th>obj.email</th>
    <th>obj.password</th>
    </tr>
{% endfor %}
</tbody>
```

if put insert clause in model.py may cause problem like this:

User Info

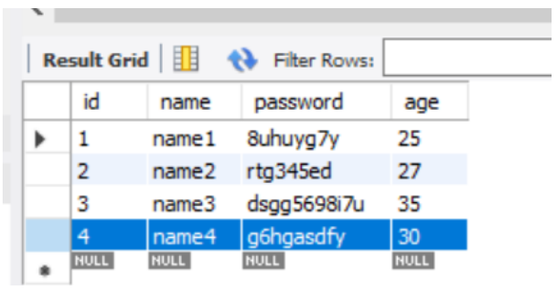
ID	name	age	password
1	name1	25	8uhuyg7y
2	name2	27	rtg345ed
3	name3	35	dsgg5698i7u
4	name4	30	g6hgasdfy
5	name1	25	8uhuyg7y
6	name2	27	rtg345ed
7	name3	35	dsgg5698i7u
8	name4	30	g6hgasdfy
9	name1	25	8uhuyg7y
10	name2	27	rtg345ed
11	name3	35	dsgg5698i7u
12	name4	30	g6hgasdfy
13	name1	25	8uhuyg7y
14	name2	27	rtg345ed
15	name3	35	dsgg5698i7u
16	name4	30	g6hgasdfy
17	name1	25	8uhuyg7y
18	name2	27	rtg345ed
19	name3	35	dsgg5698i7u
20	name4	30	g6hgasdfy

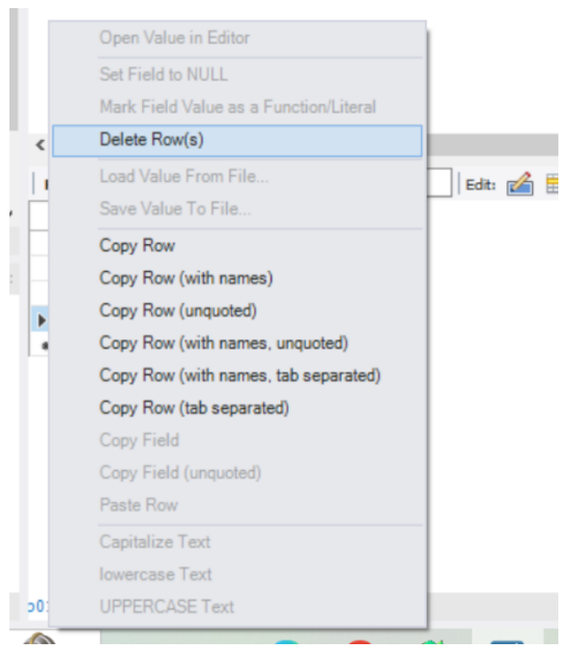
but what we realy want is :

User Info

ID	name	age	password
1	name1	25	8uhuyg7y
2	name2	27	rtg345ed
3	name3	35	dsgg5698i7u
4	name4	30	g6hgasdfy

This might because we use insert clause in model.py, you can delete that in SQL(what I done), also, you can truncate the table and insert later.

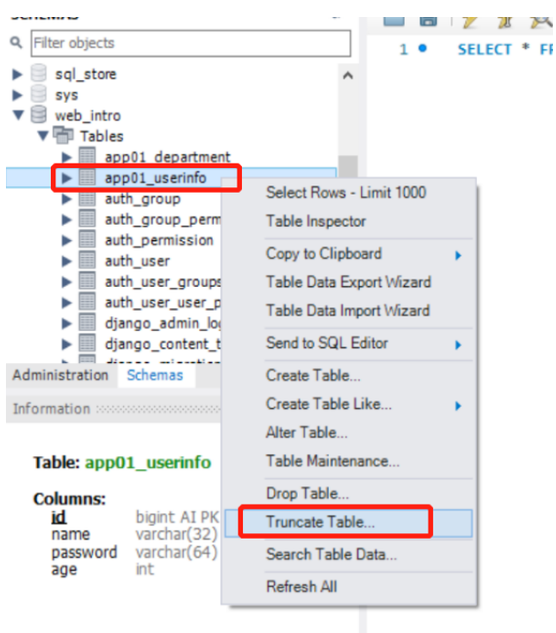




If delete rows in MySQL, when you close the table MySQL will ask you if you want to keep the changes you made, do remenber to delete "`" in the SQL code provided by MySQL.

`db_name`.`table_name` `col_name`

Please remember to delete these three pairs of "`"



also, if you have problem like this:

```
  File "C:\Users\[redacted]\anaconda3\lib\site-packages\MySQLdb\cursors.py", line 206, in execute
    res = self._query(query)
  File "C:\Users\[redacted]anaconda3\lib\site-packages\MySQLdb\cursors.py", line 319, in _query
    db.query(q)
  File "C:\Users\[redacted]anaconda3\lib\site-packages\MySQLdb\connections.py", line 254, in query
    _mysql.connection.query(self, query)
MySQLdb.ProgrammingError: (1146, "Table 'web_intro.app01_emp_info' doesn't exist")

The above exception was the direct cause of the following exception:
```

Please try this method:

How to fix 'Django: table doesn't exist' when migrating from someone else's project?

I get the following error while trying to run the command python manage.py makemigrations invigilators:

https://stackoverflow.com/questions/54143552/how-to-fix-django-table-doesnt-exist-when-migrating-from-someone-elses-proj

cut and paste code in model.py to somewhere else, commen related code in views.py and url.py

```
Terminal:  Local
  File "<frozen importlib._bootstrap>", line 219, in _call_with_frames_removed
  File "C:\Users\[redacted]\PycharmProjects\Django_intro\Django_intro_site\Django_intro_site\urls.py", line 20, in <module>
    from app01 import views
  File "C:\Users\[redacted]\PycharmProjects\Django_intro\Django_intro_site\app01\views.py", line 2, in <module>
    from app01.models import UserInfo
ImportError: cannot import name 'UserInfo' from 'app01.models' (C:\Users\[redacted]PycharmProjects\Django_intro\Django_intro_site\app01\models.py)
(base) PS C:\Users\[redacted]\PycharmProjects\Django_intro\Django_intro_site> python manage.py makemigrations
No changes detected
(base) PS C:\Users\[redacted]\PycharmProjects\Django_intro\Django_intro_site> python manage.py migrate
Operations to perform:
  Apply all migrations: admin, app01, auth, contenttypes, sessions
Running migrations:
  No migrations to apply.
Version Control  Run  Python Packages  TODO  Python Console  Problems  Terminal  Services
```

add data by hands

```
In [3]: from app01.models import UserInfo
In [4]: UserInfo.objects.create(name='name1', password='8uhuyg7y', age=25)
Out[4]: <UserInfo: UserInfo object (1)>
In [5]: UserInfo.objects.create(name='name2', password='rtg345ed', age=27)
Out[5]: <UserInfo: UserInfo object (2)>
In [6]: UserInfo.objects.create(name='name3', password='dsgg5698i7u', age=35)
Out[6]: <UserInfo: UserInfo object (3)>
In [7]: UserInfo.objects.create(name='name4', password='g6hgasdfy', age=30)
Out[7]: <UserInfo: UserInfo object (4)>
In [8]: from app01.models import Department
In [9]: Department.objects.create(department_name='department1')
Out[9]: <Department: Department object (1)>
In [10]: Department.objects.create(department_name='department2')
Out[10]: <Department: Department object (2)>

In [11]:
```

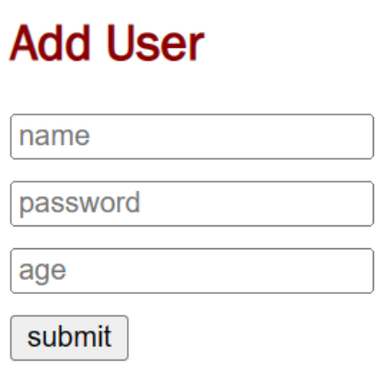
add user

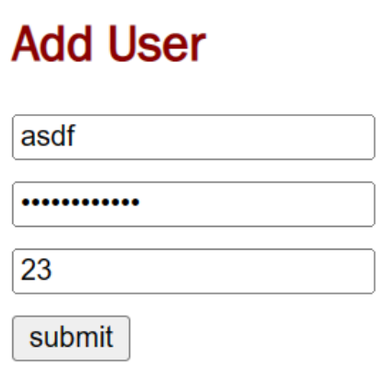
```
# urls.py
path('user/add/', views.user_add),
```

```
# views.py
def user_add(request):
    if request.method == 'GET':
```

```
        return render(request, 'add_user.html')
    user = request.POST.get("user")
    psw = request.POST.get("psw")
    age = request.POST.get("age")
    UserInfo.objects.create(name=user, password=psw, age=age)
    return redirect('/user/info/')
```

```
<!-- add_user.html -->
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
    <style>
        .container {
            width: 1000px;
            margin: 0 auto;
        }

        .container h2 {
            color: darkred;
            font-weight: bold;
            font-family: "Franklin Gothic Book";
        }
        .container form {
            text-align: center;
        }

        .container input {
            margin-top: 10px;
            display: block;
        }
    </style>
</head>
<body>
<div class="container">
    <h2>Add User</h2>
    <form method="post" action="/user/add/">
        {% csrf_token %}
        <input type="text" name="user" placeholder="name">
        <input type="password" name="psw" placeholder="password">
        <input type="text" name="age" placeholder="age">
        <input type="submit" value="submit">
    </form>
</div>
</body>
</html>
```





User Info

ID	name	age	password
1	name1	25	8uhuyg7y
2	name2	27	rtg345ed
3	name3	35	dsgg5698i7u
4	name4	30	g6hgasdfy
5	asdf	23	1a2s3d4flkjh

delete user

```
# urls.py
path('user/delete/', views.user_delete),
```

```
# views.py
def user_delete(request):
    nid = request.GET.get('nid')
    UserInfo.objects.filter(id=nid).delete()
    return redirect('/user/info/')
```

```
<!-- user_info.html -->
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
    <style>
        .container {
            width: 1000px;
            margin: 0 auto;
        }
        .container th {
            text-align: center;
            width: 100px;
        }
        .container h1 {
            color: darkred;
        }
    </style>
</head>
<body>
<div class="container">
    <h1>User Info</h1>
    <a href="/user/add/">
        <button style="margin-bottom: 10px">Add User</button>
    </a>
    <table border="1">
        <thead>
            <tr>
                <th>ID</th>
                <th>name</th>
                <th>age</th>
                <th>password</th>
                <th>oprations</th>
            </tr>
        </thead>
        <tbody>
        {% for obj in data_list %}
            <tr>
            <th>{{ obj.id }}</th>
            <th>{{ obj.name }}</th>
            <th>{{ obj.age }}</th>
            <th>{{ obj.password }}</th>
            <th>
                <a href="/user/delete/?nid={{ obj.id }}">delete</a>
            </th>
            </tr>
        {% endfor %}
        </tbody>
    </table>
</div>
```

```
</body>
</html>
```

User Info

Add User

ID	name	age	password	oprations
1	name1	25	8uhuyg7y	delete
2	name2	27	rtg345ed	delete
3	name3	35	dsgg5698i7u	delete
4	name4	30	g6hgasdfy	delete
5	asdf	23	1a2s3d4flkjh	delete
6	lkjh	31	0o9k8u7yasdr	delete

User Info

Add User

ID	name	age	password	oprations
1	name1	25	8uhuyg7y	delete
2	name2	27	rtg345ed	delete
3	name3	35	dsgg5698i7u	delete
4	name4	30	g6hgasdfy	delete
6	lkjh	31	0o9k8u7yasdr	delete

User Management Ex

User Management-continue

User Management - continue - Administrator

User Management - continue - login

User Management - continue - logout

User Management - continue - Info

ajax intro