Django Part -2

Understanding GET and POST Methods

Introduction

- In this presentation we are going to create a simple app that allows an HTML form to send data to a page using GET and POST Methods.
- We are not going to make things complicated for the learner and if you follow the simple steps that are in continuation with our first presentation (Django Part -1), you will easily be able to build and test this app.
- In this presentation we are going to first use the GET method to send data to another page and display it.
- In the second part we are going to achieve the same using POST method

GET Method explained

- GET is used to request data from a specified resource
- http://localhost:8000/display?rollno=100&name=ashok
- In the above example the data is sent through the HTTP header and is visible in the address bar
  - GET request can be cached
  - GET request remain in browser history
  - GET request can be bookmarked
  - GET request should never be used for sending sensitive information
  - GET request have length restrictions
  - GET request is generally used to request data and no to modify it.

POST Method explained

- POST method is used to send data to a server to create or modify a resource
  - POST requests are never cached
  - POST request do not remain in browser history
  - POST requests cannot be bookmarked
  - POST request have no restrictions on length
  - POST request are not visible in the HTTP header and are send as an attachment to the server.

Creating the formapp

- Step 1: Create a Django project inside your selected virtual environment and name it as webproj `(env1) D:\projects\django-admin startproject webproj`
- Step 2: Create a Django app inside the webproj and name it as formapp

```
(env1) D:\projects>django-admin startapp formapp
```

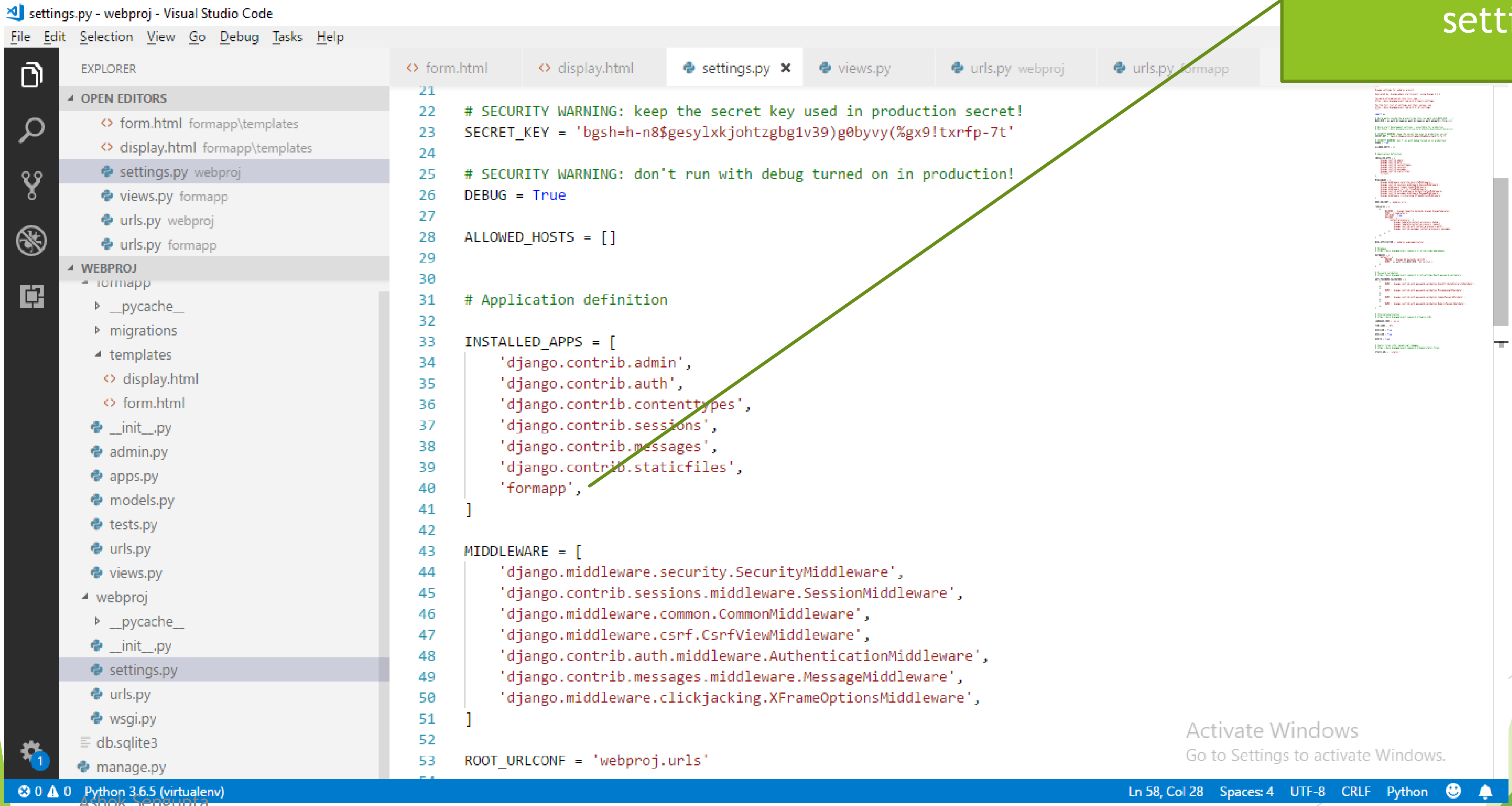
or

```
(env1) D:\projects>python manage.py startapp formapp
```

- Step 3: Now open your project in Visual Studio code (IDE used for python)

```
(env1) D:\projects>code .
```

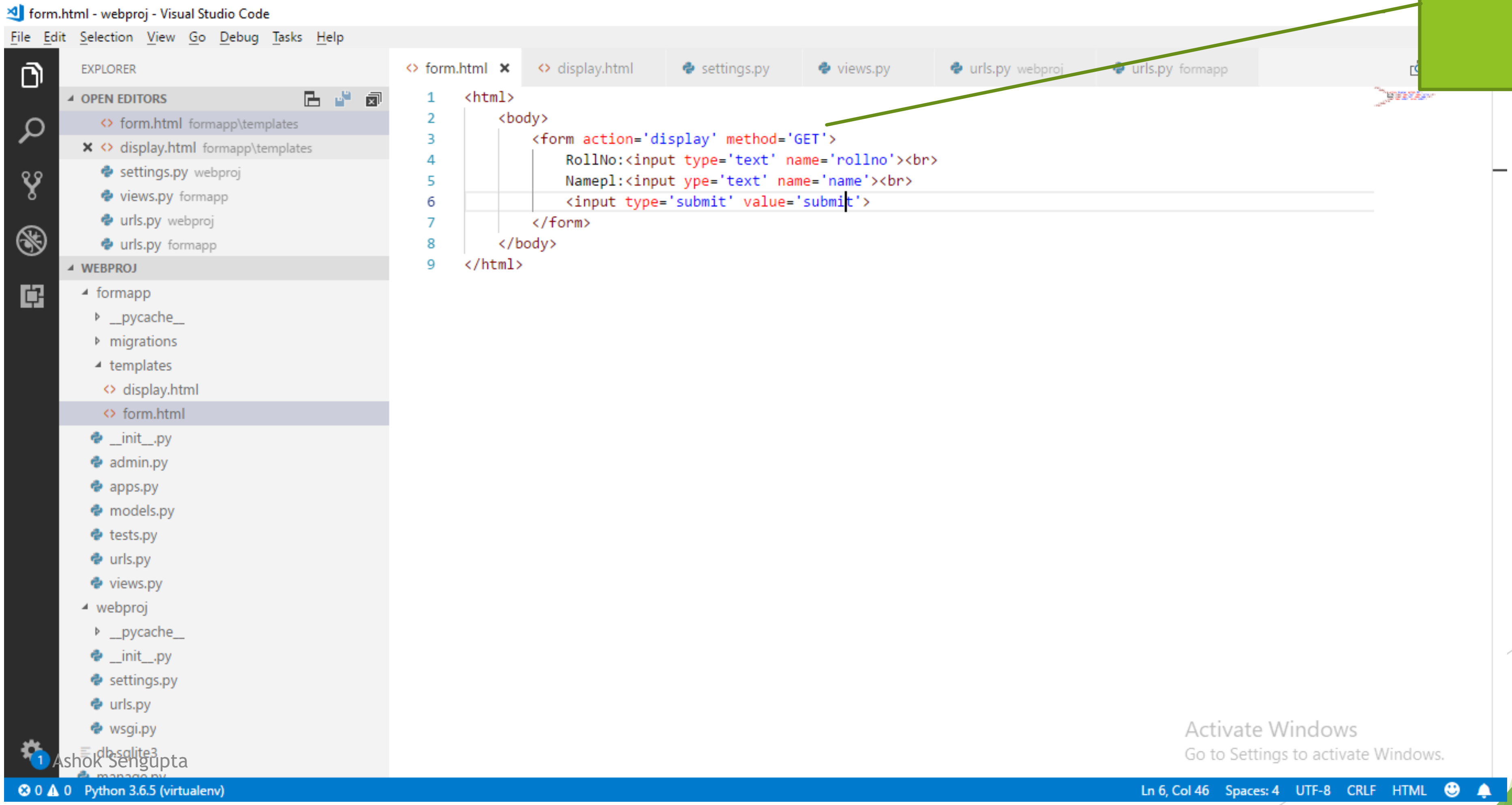
Register your app in settings.py



```
# SECURITY WARNING: keep the secret key used in production secret!
SECRET_KEY = 'bgsh=h-n8$gesylxkjohtzgbg1v39)g0byvy(%gx9!txrfp-7t'

# SECURITY WARNING: don't run with debug turned on in production!
DEBUG = True

ALLOWED_HOSTS = []


# Application definition

INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'formapp',
]

MIDDLEWARE = [
    'django.middleware.security.SecurityMiddleware',
    'django.contrib.sessions.middleware.SessionMiddleware',
    'django.middleware.common.CommonMiddleware',
    'django.middleware.csrf.CsrfViewMiddleware',
    'django.contrib.auth.middleware.AuthenticationMiddleware',
    'django.contrib.messages.middleware.MessageMiddleware',
    'django.middleware.clickjacking.XFrameOptionsMiddleware',
]

ROOT_URLCONF = 'webproj.urls'
```

Registering formapp in the settings.py

Create the templates folder in formapp folder and create the form.html

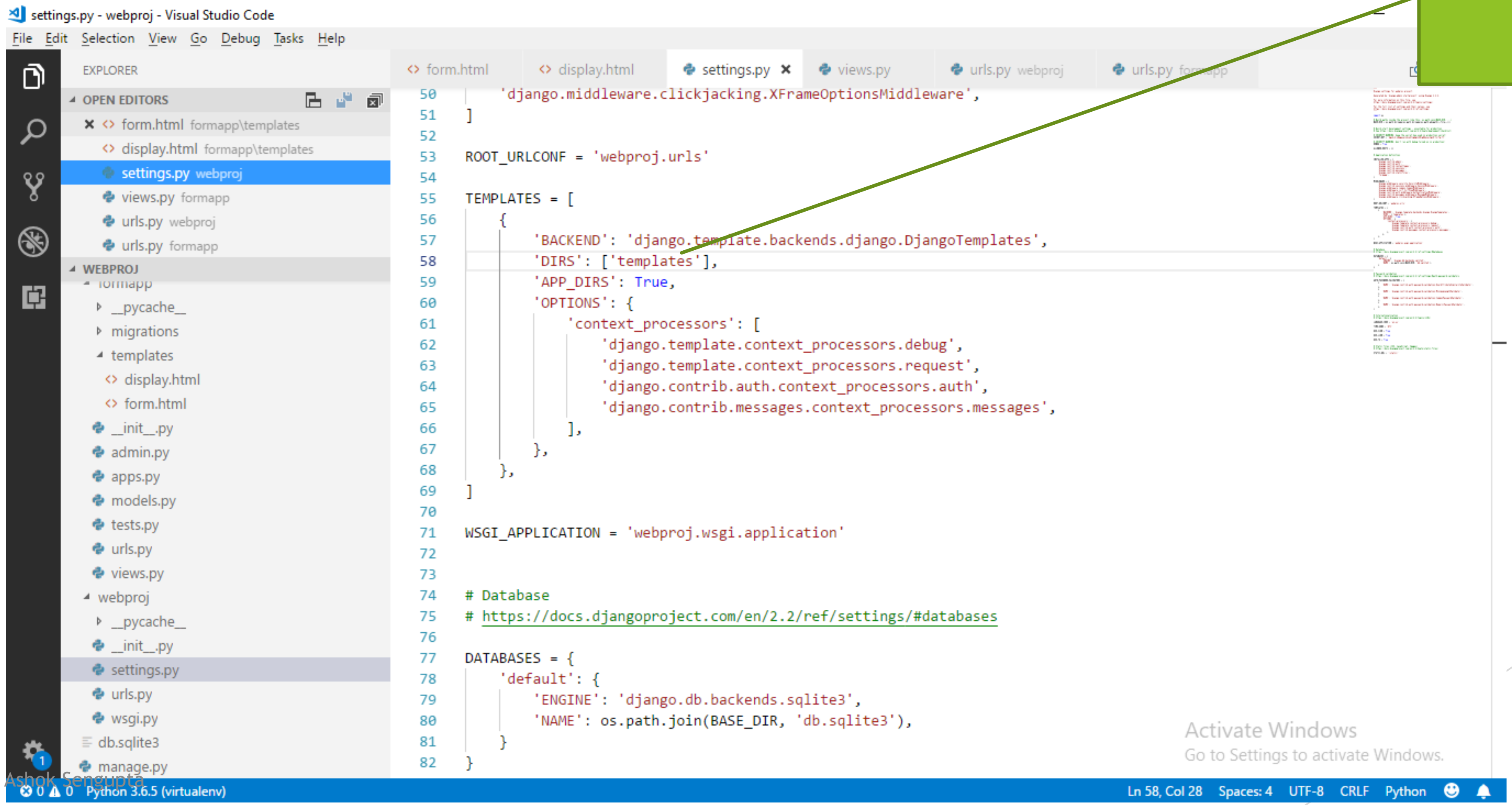


```
<html>
    <body>
        <form action='display' method='GET'>
            RollNo:<input type='text' name='rollno'><br>
            Namepl:<input ype='text' name='name'><br>
            <input type='submit' value='submit'>
        </form>
    </body>
</html>
```

Observe that we are using GET method

Ashok Sengupta

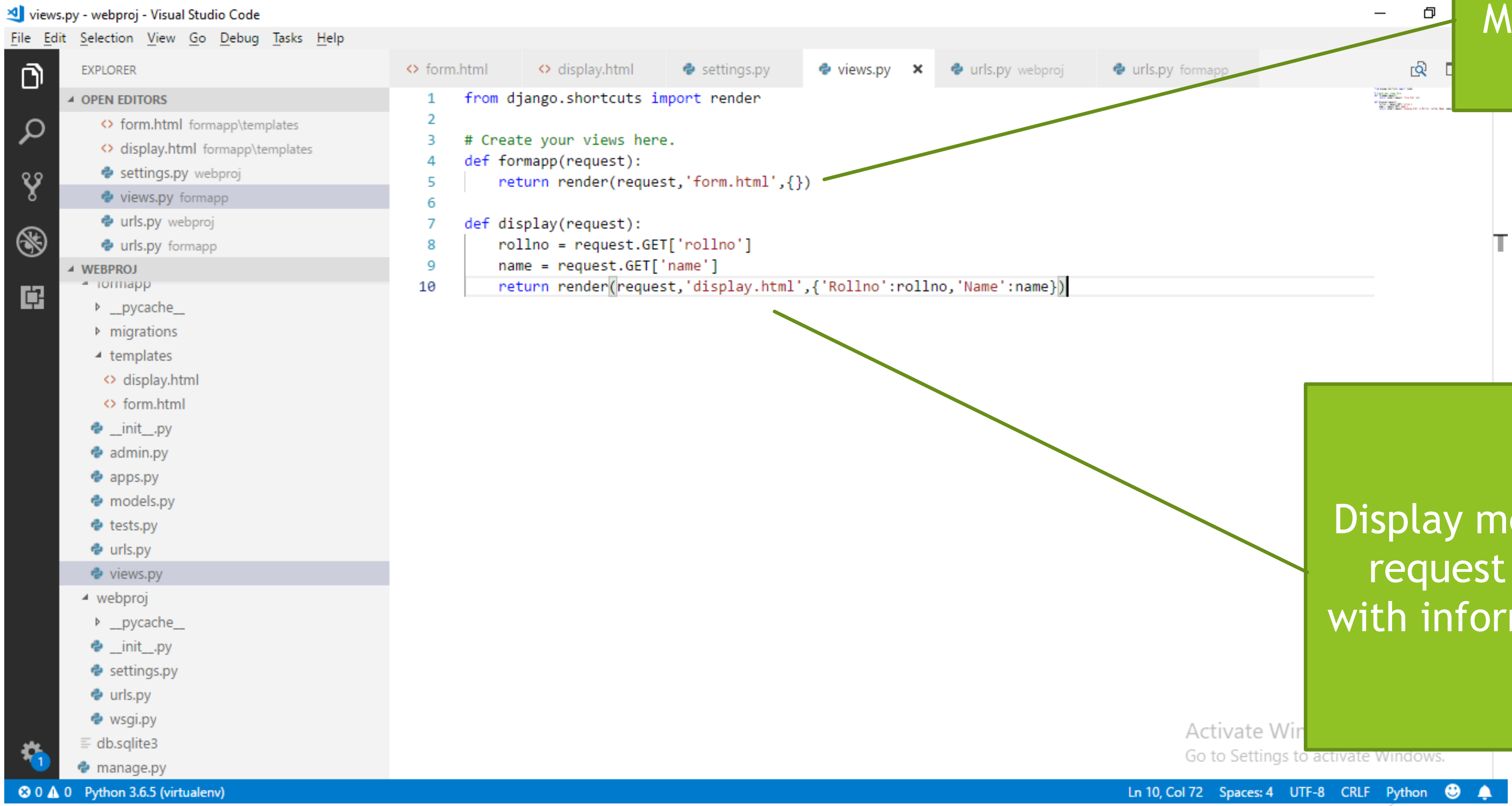
Register the templates folder in settings.py



```
    'django.middleware.clickjacking.XFrameOptionsMiddleware',
]

ROOT_URLCONF = 'webproj.urls'

TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': ['templates'],
        'APP_DIRS': True,
        'OPTIONS': {
            'context_processors': [
                'django.template.context_processors.debug',
                'django.template.context_processors.request',
                'django.contrib.auth.context_processors.auth',
                'django.contrib.messages.context_processors.messages',
            ],
        },
    },
]

WSGI_APPLICATION = 'webproj.wsgi.application'


# Database
# https://docs.djangoproject.com/en/2.2/ref/settings/#databases

DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.sqlite3',
        'NAME': os.path.join(BASE_DIR, 'db.sqlite3'),
    }
}
```

templates folder is registered in 'DIRS'

Edit the views.py to create two views. One for the formapp and other for the display

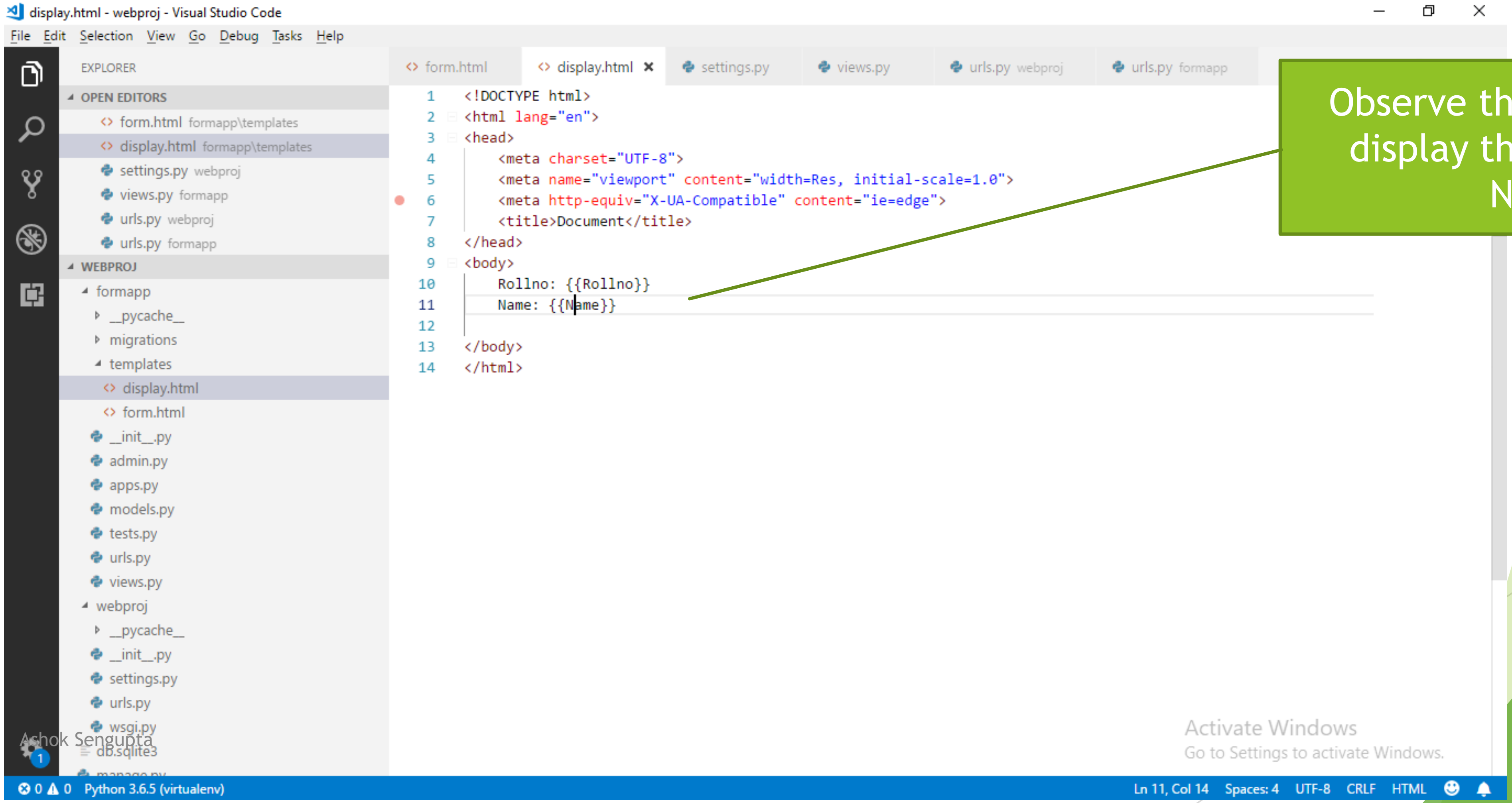


```python
from django.shortcuts import render

# Create your views here.
def formapp(request):
    return render(request,'form.html',{})

def display(request):
    rollno = request.GET['rollno']
    name = request.GET['name']
    return render(request,'display.html',{'Rollno':rollno,'Name':name})
```

Method to handle the form

Display method is to handle the GET request to the display.html page with information as Rollno and Name

Create the display.html page in the templates folder

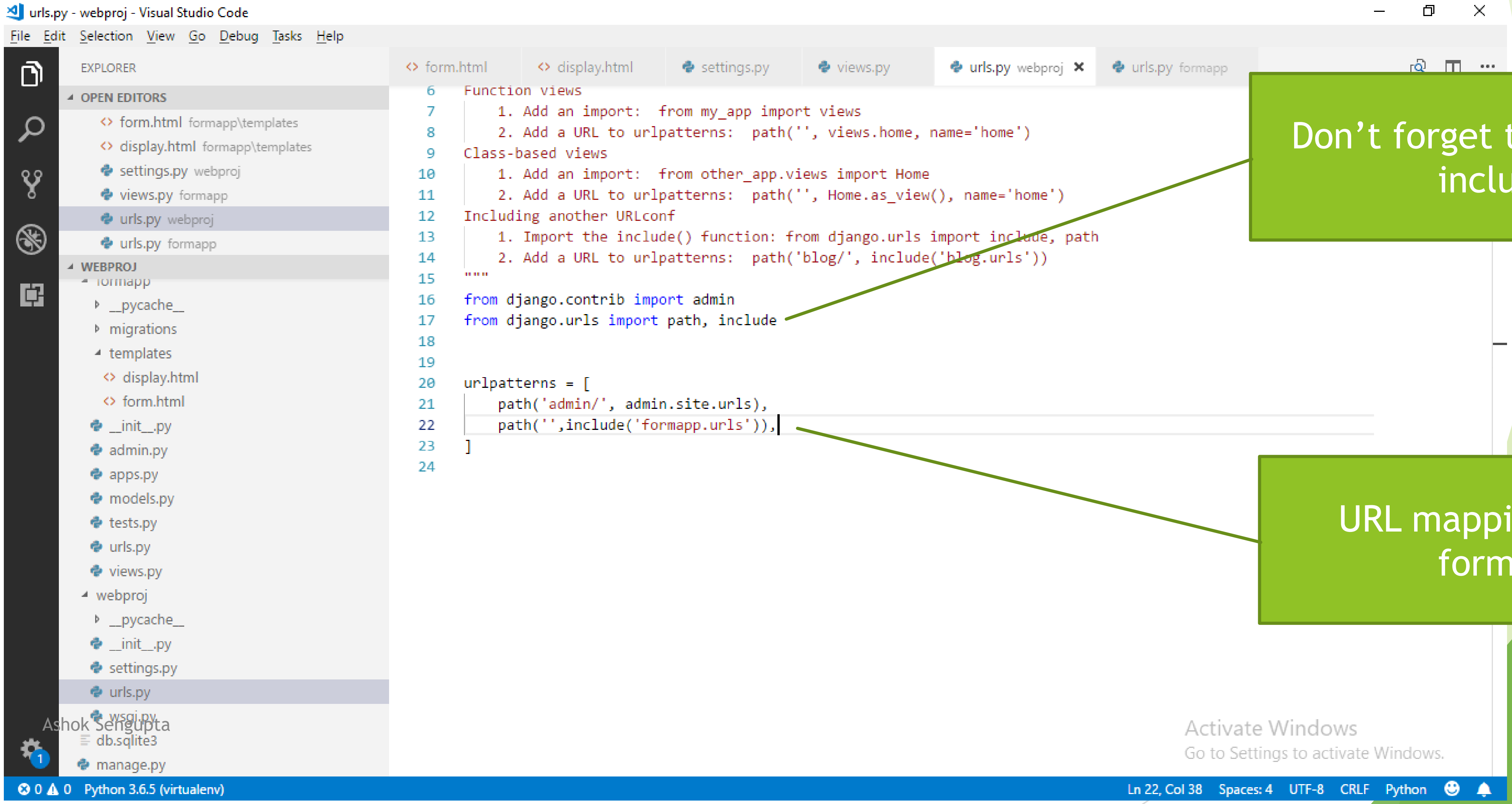


```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=Res, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Document</title>
</head>
<body>
    Rollno: {{Rollno}}
    Name: {{Name}}

</body>
</html>
```

Observe the DTL used to display the Rollno and Name

Ashok Sengupta

Edit the urls.py files in the webproj folder



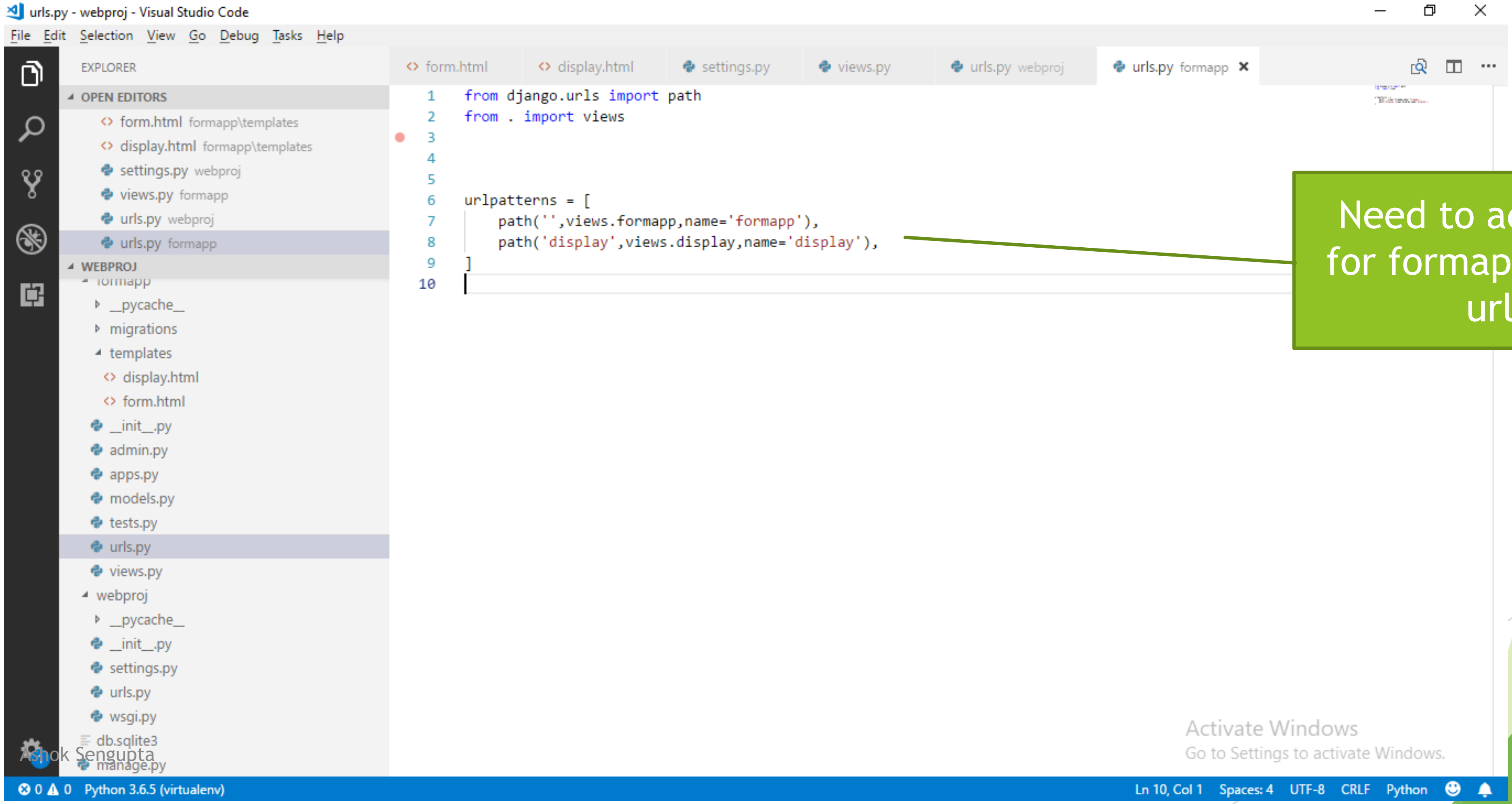
```python
Function views
    1. Add an import:  from my_app import views
    2. Add a URL to urlpatterns:  path('', views.home, name='home')
Class-based views
    1. Add an import:  from other_app.views import Home
    2. Add a URL to urlpatterns:  path('', Home.as_view(), name='home')
Including another URLconf
    1. Import the include() function: from django.urls import include, path
    2. Add a URL to urlpatterns:  path('blog/', include('blog.urls'))
"""
from django.contrib import admin
from django.urls import path, include


urlpatterns = [
    path('admin/', admin.site.urls),
    path('',include('formapp.urls')),
]
```

Don’t forget to imort the include

URL mapping for the formapp

Ashok Sengupta

Create urls.py in fromapp forlder and edit it.

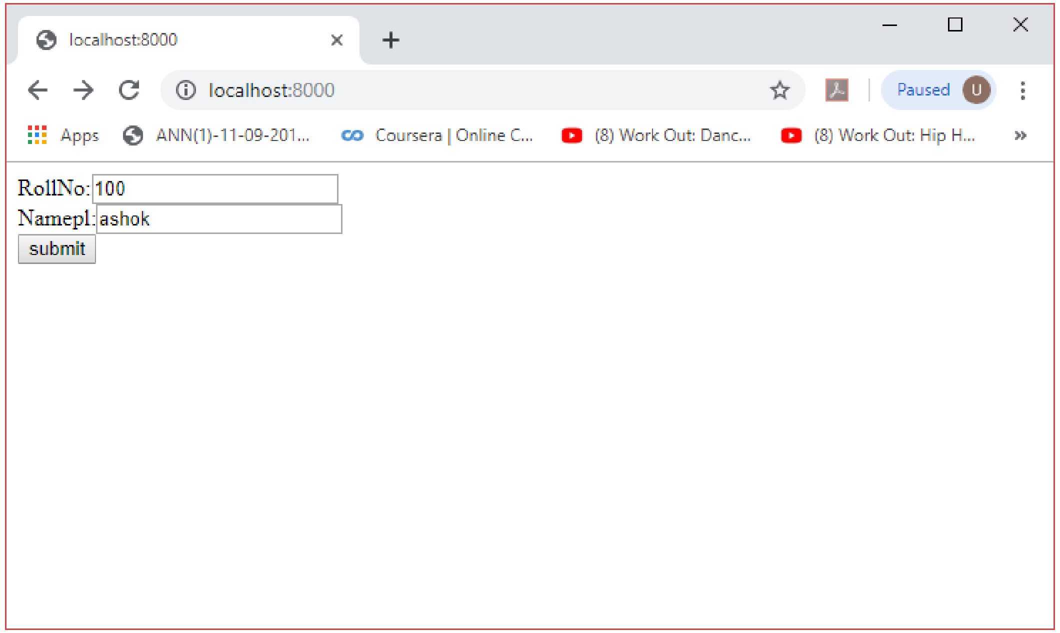


```
from django.urls import path
from . import views



urlpatterns = [
    path('',views.formapp,name='formapp'),
    path('display',views.display,name='display'),
]

```

Need to add path entries for formapp and display in urlpaterns

Ashok Sengupta

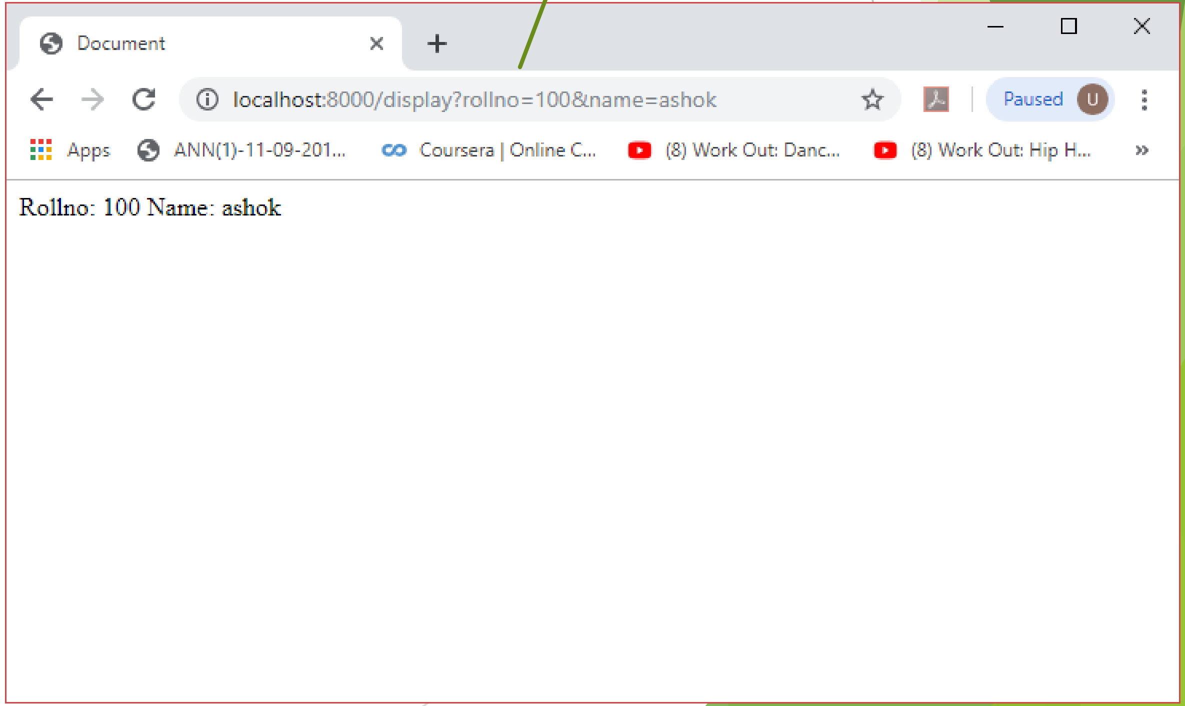
That’s it friends. You can test the app by running the server



RollNo: 100

Namepl: ashok

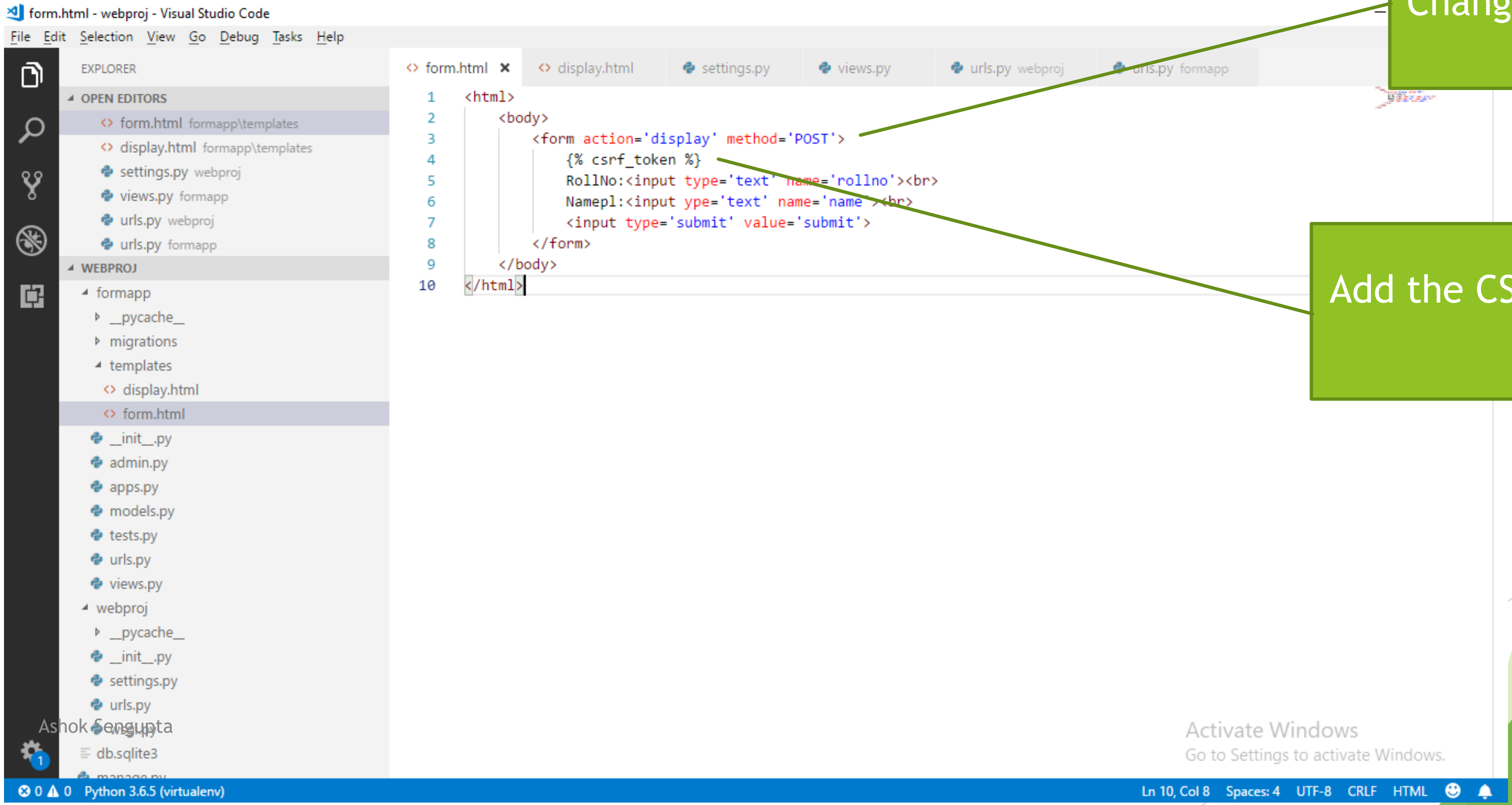
submit



Rollno: 100 Name: ashok

See in GET request the HTTP header contains the data

Only a few changes to achieve the same using POST method- first edit the form.html

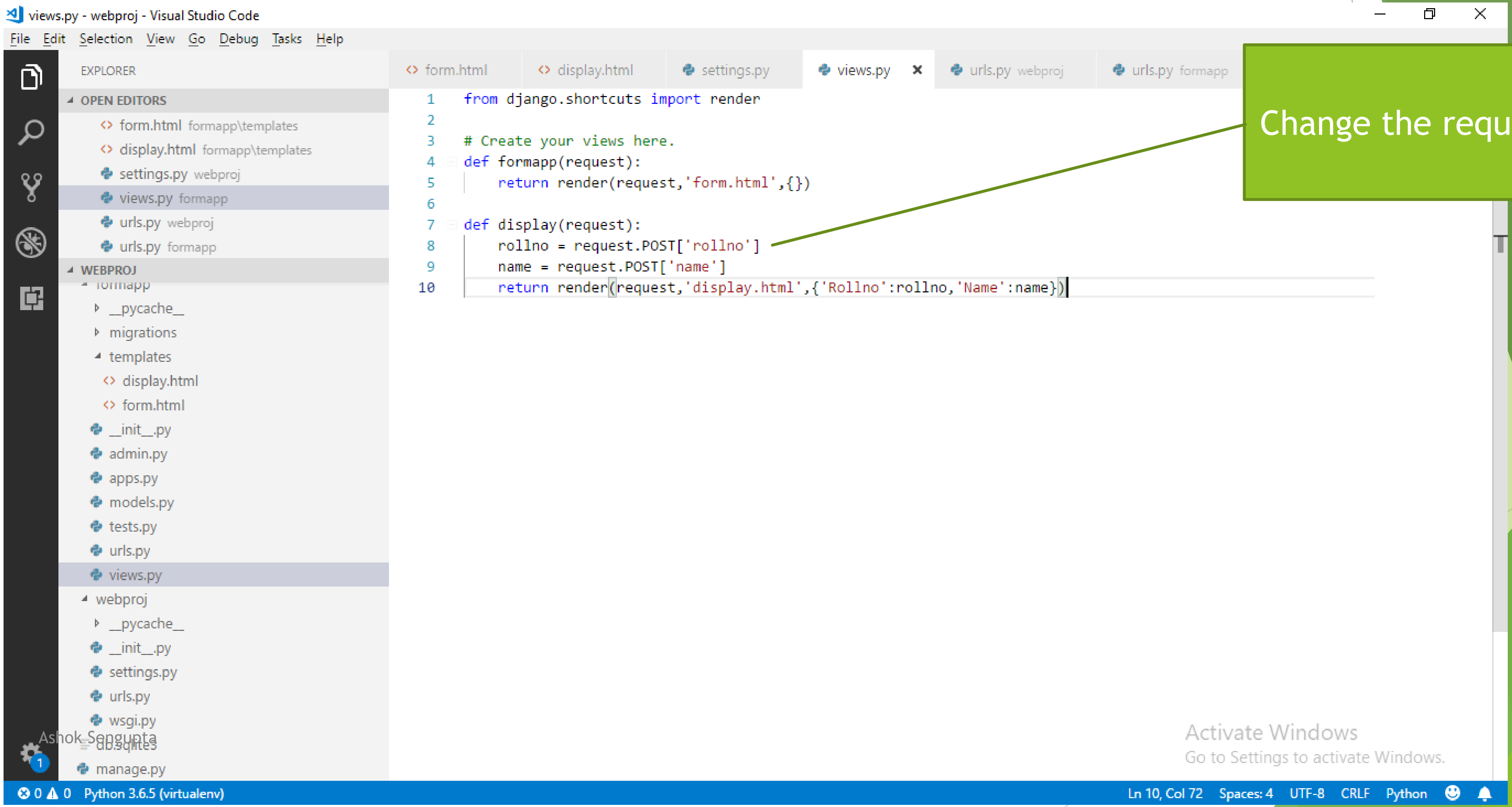


```
<html>
    <body>
        <form action='display' method='POST'>
            {% csrf_token %}
            RollNo:<input type='text' name='rollno'><br>
            Namepl:<input ype='text' name='name'><br>
            <input type='submit' value='submit'>
        </form>
    </body>
</html>
```

Change the method to POST

Add the CSRF token in zinza code

Now edit the views.html



```python
from django.shortcuts import render

# Create your views here.
def formapp(request):
    return render(request,'form.html',{})

def display(request):
    rollno = request.POST['rollno']
    name = request.POST['name']
    return render(request,'display.html',{'Rollno':rollno,'Name':name})
```

Change the request to POST

Ashok Sengupta

That’s all friends the app will work the same way.

Thank You