Create Template Folder and Files

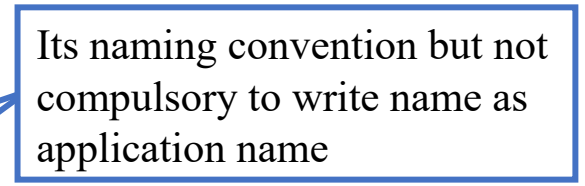
We can create separate folder inside templates folder for applications then each application will contain only those html file which are related to them. This will enhance readability and separate html files according to applications.

geekyshows

- **templates**
  - **courseone.html**
  - **coursetwo.html**
  - **feesone.html**
  - **feestwo.html**
- geekyshows
  - __init__.py
  - settings.py
  - urls.py
  - wsgi.py
- manage.py
- course
- fees

geekyshows

- **templates**
  - **course**
    - **html files related to course app**
  - **fees**
    - **html files related to fees app**
- geekyshows
  - __init__.py
  - settings.py
  - urls.py
  - wsgi.py
- manage.py
- course
- fees



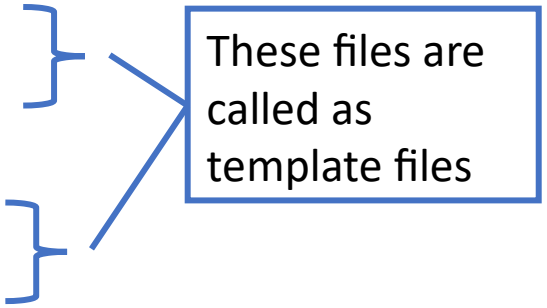
Create Template Folder and Files

geekyshows

- **templates**
  - **courseone.html**
  - **coursetwo.html**
  - **feesone.html**
  - **feestwo.html**
- geekyshows
  - __init__.py
  - settings.py
  - urls.py
  - wsgi.py
- manage.py
- course
- fees

geekyshows

- **templates**
  - **course**
    - **courseone.html**
    - **coursetwo.html**
  - **fees**
    - **feesone.html**
    - **feestwo.html**
- geekyshows
  - __init__.py
  - settings.py
  - urls.py
  - wsgi.py
- manage.py
- course
- fees



Add Templates in settings.py

geekyshows

- **templates**
  - **course**
    - **courseone.html**
    - **coursetwo.html**
  - **fees**
    - **feesone.html**
    - **feestwo.html**
  - geekyshows
    - __init__.py
    - settings.py
    - urls.py
    - wsgi.py
  - manage.py
  - course
  - fees

settings.py

Old Version Django 3.0:
TEMPLATES_DIR = os.path.join(BASE_DIR, 'templates')

```
TEMPLATES_DIR = BASE_DIR / 'templates'

INSTALLED_APPS = [
      'course',
      'fees'
]

TEMPLATES = [
    {
      'DIRS': [TEMPLATES_DIR],
    }
]
```

Django Version 3.1

Directories where the engine should look for template source files, in search order.

Geeky Steps

- Create Django Project: *django-admin startproject geekyshows*
- Create Django Application1: *python manage.py startapp course*
- Create Django Application2: *python manage.py startapp fees*
- Add/Install Applications to Django Project (course and fees to geekyshows) using settings.py INSTALLED_APPS
- Create **templates** folder inside Root Project Directory
- Add **templates** directory in settings.py

Old Django Version 3.0

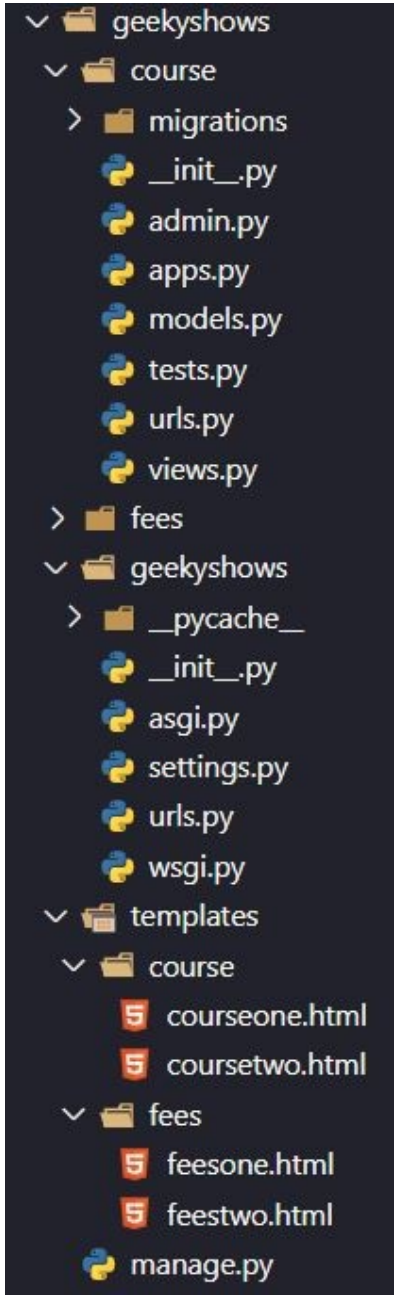
```
TEMPLATES_DIR = os.path.join(BASE_DIR, 'templates')
```

```
TEMPLATES_DIR = BASE_DIR / 'templates'
```

Django Version 3.1

```
TEMPLATES = [ {
        'DIRS': [TEMPLATES_DIR],      } ]
```

- Create Separate Directory for each application, inside templates directory
- Create template files inside templates/temp_application folder/directory
- Write View Function inside views.py file
- Define url for view function of application using urls.py file



Write Templates Files

When we create Template file for application we separate business logic and presentation from the application *views.py* file.

Now we will write business logic in *views.py* file and presentation code in template file.

templates/course

courseone.html

```
<html>
    <head>
        <style> ……. </style>
    </head>
    <body>
        <h1>Hello Django</h1>
    </body>
    <script>…….</script>
</html>
```

templates/course
coursetwo.html

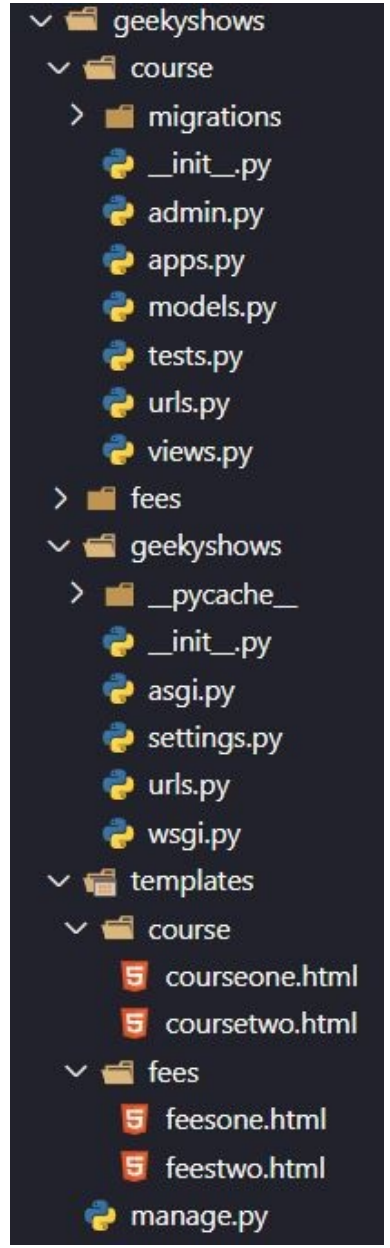
```
<html>
    <body>
        <h1>Hello Python</h1>
    </body>
</html>
```

templates/fees
feesone.html

```
<html>
    <body>
        <h1>300</h1>
    </body>
</html>
```

templates/fees
feestwo.html

```
<html>
    <body>
        <h1>200</h1>
    </body>
</html>
```



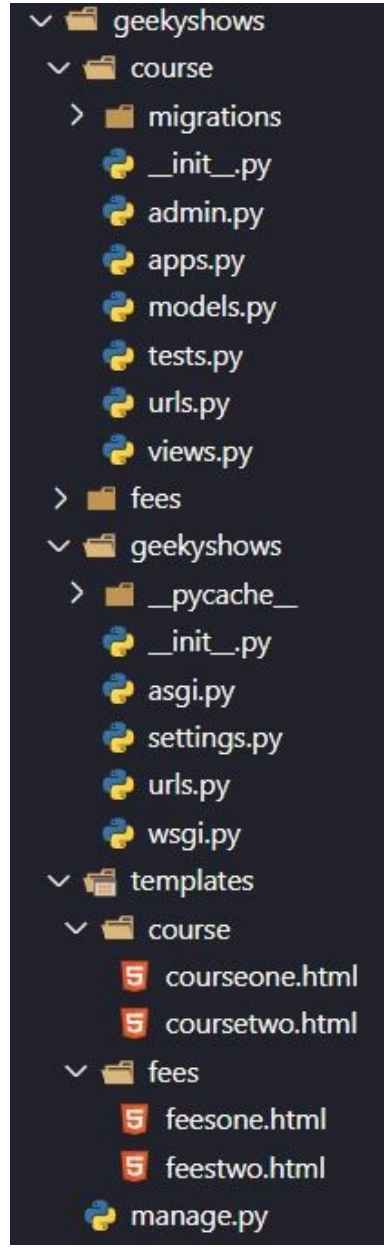
Rendering Templates Files

By Creating Template file for application we separate business logic and presentation from the application *views.py* file. Now we will write business logic in views.py file and presentation code in html file.

Still *views.py* will be responsible to process the template files for this we will use *render()* function in *views.py* file.

views.py

```
from django.shortcuts import render

def function_name(request):

    Dynamic Data, if else, any python code logic

    return render(request, template_name, context=dict_name,
content_type=MIME_type, status=None, using=None)


def learn_django(request):

    return render(request, 'course/courseone.html')
```



render ( )

render ( ) Function - It combines a given template with a given context dictionary and returns an HttpResponse object with that rendered text.

Syntax:-

render(request, template_name, context=dict_name, content_type=MIME_type, status=None, using=None)

Where,

request – The request object used to generate this response.*

template_name – The full name of a template to use or sequence of template names. If a sequence is given, the first template that exists will be used. *

context – A dictionary of values to add to the template context. By default, this is an empty dictionary. If a value in the dictionary is callable, the view will call it just before rendering the template.

content_type – The MIME type to use for the resulting document. Defaults to 'text/html'.

status – The status code for the response. Defaults to 200.

using – The NAME of a template engine to use for loading the template.

render ( )

Syntax:-

render(request, template_name, context=dict_name, content_type=MIME_type, status=None, using=None)

Example:-

render(request, ‘courseone.html’, context=cname, content_type=‘application/xhtml+xml’)

render(request, ‘course/courseone.html’, context=cname, content_type=‘application/xhtml+xml’)

render(request, ‘course/coursetwo.html’)

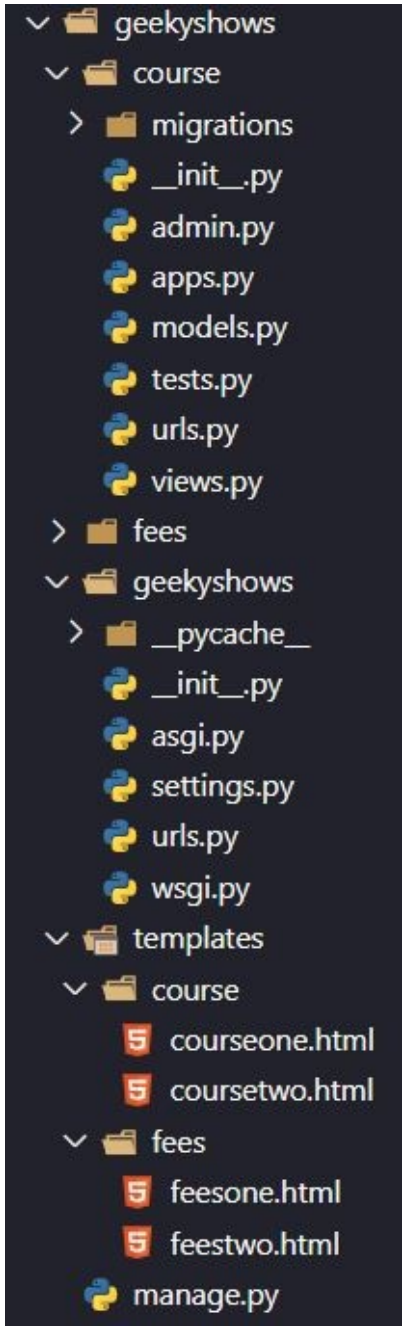
templates/course

courseone.html

```
<html>
    <head>
        <style> ……. </style>
    </head>
    <body>
        <h1>Hello Django</h1>
    </body>
    <script>…….</script>
</html>
```

views.py

```
from django.shortcuts import render
def learn_django(request):
    return render(request, 'course/courseone.html')
```



Geeky Steps

- Create Django Project: *django-admin startproject geekyshows*
- Create Django Application1: *python manage.py startapp course*
- Create Django Application2: *python manage.py startapp fees*
- Add/Install Applications to Django Project (course and fees to geekyshows) using settings.py INSTALLED_APPS
- Create **templates** folder inside Root Project Directory
- Add **templates** directory in settings.py

Old Django Version 3.0

```
TEMPLATES_DIR = os.path.join(BASE_DIR, 'templates')
```

Django Version 3.1

```
TEMPLATES_DIR = BASE_DIR / 'templates'
TEMPLATES = [ {
        'DIRS': [TEMPLATES_DIR],      } ]
```

- Create Separate Directory for each application, inside **templates** directory
- Create template files inside templates/temp_application folder/directory
- Write View Function inside views.py file
- Define url for view function of application using urls.py file
- Write Template files code