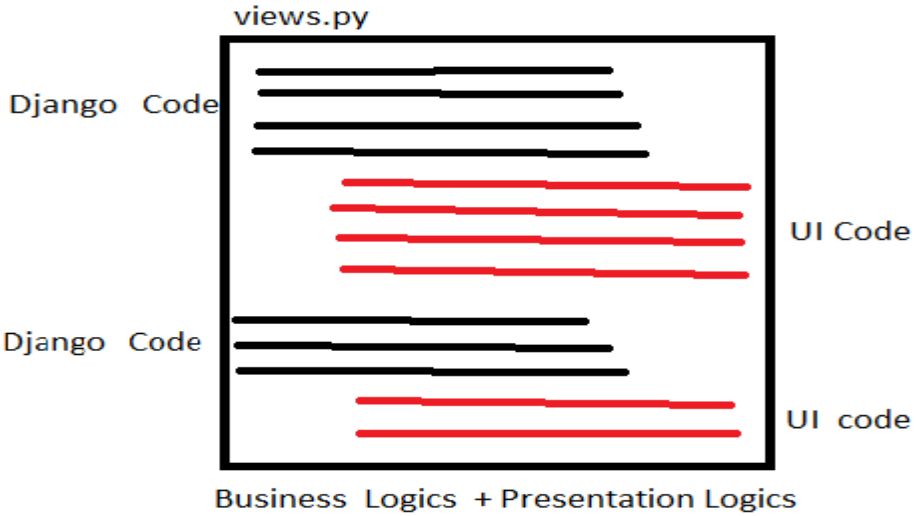
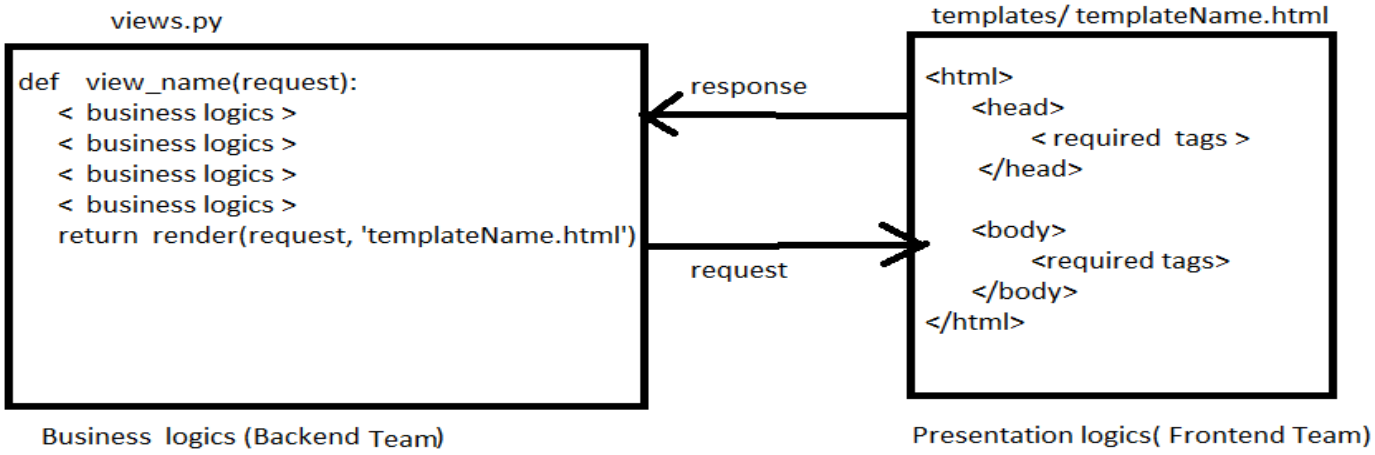
Django Templates Concepts:

- Django Templates are HTML files. Templates are used to write the Presentation coding.
- Eventhough Django follows MVT arch, django brings only views and models files when we create django project and django application.
- Templates are not default files or folders in django because views.py file allows both Bussiness and Presentation coding by default, thats why Templates is an optional concept in the django.
- Generally, All django developers no need to know UI coding and also all UI developers no need to know django coding, so it makes more complex when both django developer (backend developer) and UI developer(frontend developer) use same views.py file.



- Here both UI developers and django developers using the same views.py file, so there is a chance to get errors.
- It is also become overload to both the developers to solve this problem we can separate web development code from views.py file and take into a new template(html file).
- Now we will write the UI coding in the template files and finally we will give the Template file name in the views.py file by using the **render().**

  Syntax : return render(request , 'templateName.html')
- So every organization has two kinds of teams (frontend and backend teams) for a single web development application or project,
- Here, backend developers develop backend part of website in views.py files and frontend developers develop frontend part logics in html files.



What is a Template?

Being a web framework, Django needs a convenient way to generate HTML dynamically.
The most common approach relies on templates. A template contains the static parts of the desired HTML output as well as some special syntax describing how dynamic content will be inserted.

Template:

- A Template is a raw string which contains "N" number of "variables" and “N” number of "tags".
- Any text surrounded by pair of curl braces {{ }} is called as variable.
- For example : {{ name }} , {{ salary }}}
- A Variable representing "inserting something". It means dynamically binding variableName data into our template.
- Template getting variable names by using Context object.

Context:

- A Context is a Dictionary object which containes collection of key : value paires.
- The keys of dict must be same as variable name in the template , because the variables will take the values by using keys in the dict.
- We can create one or more contexts for single Templates.
  Syntaxt : { key1 : value1 , key2 : value2 , ….}
  For example : {"eno" : 10, "ename" : "Virat" ,"salary" : 10000}

Tags:

- Any text surrounded by single curl braces { } inside with % symbol then that is called as "tag" .
- A Tag says "do something or performing something".
- Django Template Language provides so many template tags like **if , elif, else, for, block , extends, include, load, commit, now** and so on.

For example :

```
{% for emp  in employees %}
       <h1> Employee Number :  {{ emp.eno }} </h1>
        <h1> Employee Name :    {{ emp.ename }} </h1>
 {% endfor %}

 {%  if 10 > 5   %}
     <p> if block tags executed </p>
 {% else %}
       <p> else block tags executed </p>
 {%  endif   %}
```

Generally we can creates template in different ways like ,

1. static template:
   - way of template we can create by using python shell and it is not useful in realtime
2. dynamic template :
   - dynamic way of template we can create by using template(html) files and which is useful in realtime

Creating Static Templates:

- We have to import the **'Template'** class and **'Context'** class to create template objects and context objects
- Templates is a raw string which contains “N” number of variables and tags.
- Context is a dictionary object which contains a collection of Key-Value paires.

render()

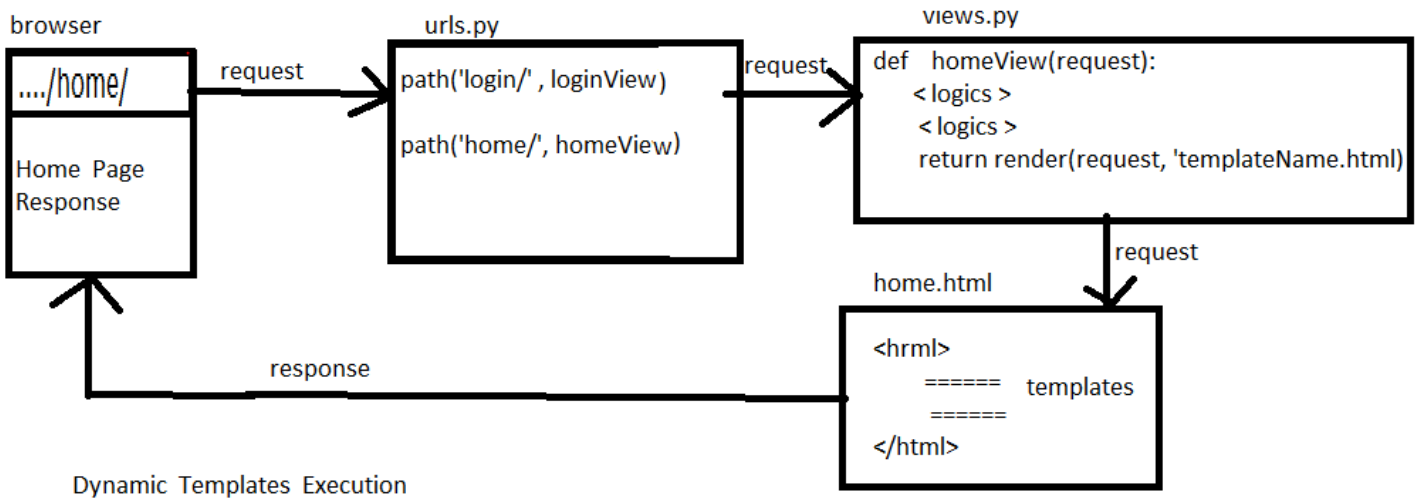
- It is used to map the context object keys with variables names in the template.
- Static way : template_object.render(context_object)
- Dynamic way : render(request, 'base.html' , { Key : Value})

Static way by using python shell

```
========= ========= ==========
E:\fourthfolder\datepro> python manage.py shell
>>> from django.template import Template, Context
>>> template_object = Template("He is {{ name }} and he got job in {{ company }}
and getting Rs{{ salary }} salary per month")
>>> context_object = Context({ 'name' : 'Srinivas', 'company' : 'TCS' , 'salary'
: 100000 })
>>> result  =  template_object.render(context_object)
>>> print(result)
He is Srinivas and he got job in TCS and getting Rs100000 salary per month.
>>> c = Context({'name':'nani','com':'INFOSYS','sal':20000})
>>> result = template_object.render(context_object)
>>> print(result) --->> He is nani and he got job in INFOSYS and getting 20000
per month
```

Creating Dynamic Templates:

- We can use Html files to create dynamic templates.



Dynamic Templates Execution

Create a project using dynamic template concept:

Step1: Create a Django ProjectName like **Template_Project**

Step2: Create a Application Name like **Template_App**

Step3: Goto **settings.py** file and configure the **Template PATH** details under **TEMPLATE** section.

```
TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',

        'DIRS': [BASE_DIR / 'templates'] ,

        '------': ----,
        '------' ----
    },
]
```

Step4: Open the views.py file and create required views logics

```
from django.shortcuts import render

def homeView(request):
    return render(request,'home.html')
```

Step5 : Create home.html isnside templates folder.

Right click on **templates** folder---->> click on **New** --->> click on **HTML File** --->> Give file name as **home.hml** ----->> click **enter** key

- **Note :** Keep the cursor on home.html file name and click **alt+enter.**
- It will create the home.html file in template folder automatically.

Step6: Open home.html file write the following code .

```
<html>
   <head>
     <title> Home Page </title>
   </head>
   <body>
       <h1> Welcome to Home Page </h1>
   </body>
</html>
```

Step7: Open the urls.py and create required *urls* for executing the required *views*

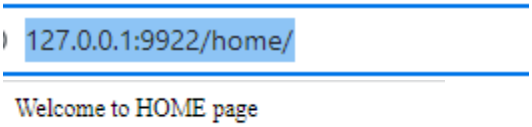
```
from django.contrib import admin
from django.urls import path
from appName import views

urlpatterns = [
    path('admin/', admin.site.urls),
    path('home/', views.homeView),
]
```

Step8: Now execute the server using runserver command.

```
python manage.py runserver 8000
```

Step9: Now send the **home/** URL request from the browser and see the Output.



Can django separate html?

- Django makes it possible to separate python and HTML, the python goes in views and HTML goes in templates.
- To link these two, Django relies on the **render()** function and the Django Template language.