Requirements

- Install Python
- Install Django
- How to Create Django Project, Apps and do settings
- How to Create Templates and do settings

Django Template Language

Django's template language is designed to strike a balance between power and ease. It's designed to feel comfortable to those used to working with HTML.

***courseone*.html**

```
<html>
  <body>
    <h2> Course Name:{{nm}} Duration:{{du}} and Total Seats: {{st}}</h2>
  </body>
</html>
```

Jinja2 - Jinja is a modern and designer-friendly templating language for Python, modelled after Django's templates. It is fast, widely used and secure with the optional sandboxed template execution environment.

python pip install jinja2

'BACKEND': 'django.template.backends.jinja2.Jinja2',

Variables

Variables look like this: **{{ variable }}** When the template engine encounters a variable, it evaluates that variable and replaces it with the result.

Rules:-

Variable names consist of any combination of alphanumeric characters and the underscore.

Variable name should not start with underscore.

Variable name can not have spaces or punctuation characters.

Syntax:- {{variable}}

Example:- {{nm}}, {{name1}}, {{first_name}}

Dynamic Template Files

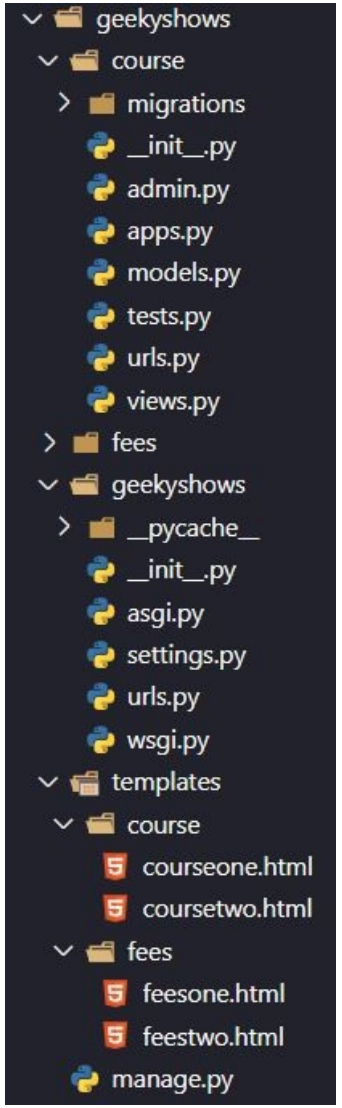
views.py

```
from django.shortcuts import render
def learn_django(request):
        cname = 'Django'
        duration = '4 Months'
        seats = 10
        django_details = {'nm':cname, 'du':duration, 'st':seats}
        return render(request, 'course/courseone.html', django_details)
```

templates/course

***courseone*.html**

```
<html>
  <body>
        <h2> Course Name:{{nm}} Duration:{{du}} and Total Seats: {{st}}</h2>
    </body>
</html>
```



Filters

When we need to modify variable before displaying we can use filters. Pipe '|' is used to apply filter.

Syntax:- {{variable | filter}}

Example:- {{name|upper}}

Some of filters take arguments.

Syntax:- {{variable | filter **:** argument}}

Example:- {{article|truncateword**:**40}}

Filter can be chained.

Syntax:- {{variable | filter | filter}}

Example:- {{article|upper}}

Example:- {{article|upper|truncateword**:**40}}

Filters

capfirst – It capitalizes the first character of the value. If the first character is not a letter, this filter has no effect.

Example:- {{value|capfirst}}

default - If value evaluates to False, uses the given default. Otherwise, uses the value.

Example:- {{ value|default:"nothing" }}

If value is "" (the empty string), the output will be nothing.

length - It returns the length of the value. This works for both strings and lists. The filter returns 0 for an undefined variable.

Example:- {{ value|length }}

lower - It converts a string into all lowercase.

Example:- {{ value|lower }}

Filters

upper - It converts a string into all uppercase.

Example:- {{ value|upper }}

slice - It returns a slice of the list. Uses the same syntax as Python's list slicing.

Example:- {{ some_list|slice:":2" }}

truncatechars - It truncates a string if it is longer than the specified number of characters. Truncated strings will end with a translatable ellipsis character ("…").

Argument: Number of characters to truncate to

Example:- {{ value|truncatechars:7 }}

truncatewords - It truncates a string after a certain number of words. Newlines within the string will be removed.

Argument: Number of words to truncate after

Example:- {{ value|truncatewords:2 }}

Filters

date – It formats a date according to the given format.

Example:- {{value|date:“D d M Y”}}

time – It formats a time according to the given format.

Example:- {{value|time:“H:i”}}

Day

Format Character	Description	Example

Week

Format Character	Description	Example

Month

Format Character	Description	Example

Year

Format Character	Description	Example

Time

Format Character	Description	Example

Timezone

Format Character	Description	Example

Date/Time

Format Character	Description	Example

Predefined Formats

Format	Description	Example

Example:- {{ value|date:"SHORT_DATE_FORMAT" }}

Example:- {{ value|time:“TIME_FORMAT" }}

Filters

floatformat

When used without an argument, rounds a floating-point number to one decimal place but only if there's a decimal part to be displayed.

Value	Template	Output

If used with a numeric integer argument, floatformat rounds a number to that many decimal places.

Value	Template	Output

Filters

floatformat

Particularly useful is passing 0 (zero) as the argument which will round the float to the nearest integer.

Value	Template	Output

If the argument passed to floatformat is negative, it will round a number to that many decimal places but only if there's a decimal part to be displayed.

Value	Template	Output

if Tag

{% if %} tag - The {% if %} tag evaluates a variable, and if that variable is “true” (i.e. exists, is not empty, and is not a false boolean value).

Syntax:-

```
{% if variable %}
      ……..
{% endif %}
```

Example:-

```
{% if nm %}
      </h1>Hello {{nm}}</h1>
{% endif %}
```

```
{% if nm and st %}
      </h1> For Course{{nm}} {{st}} Seat Available</h1>
{% endif %}
```

```
{% if nm or st %}
      </h1>Seat Available</h1>
{% endif %}
```

```
{% if not st%}
      </h1>Seat Not Available</h1>
{% endif %}
```

if Tag with condition

Syntax:-

```
{% if condition %}
      ……..
{% endif %}
```

Example:-

```
{% if nm == 'Django'  %}
      </h1>Hello {{nm}}</h1>
{% endif %}
```

```
{% if nm == 'Django' and st==5 %}
      </h1>{{nm}} Seat Available</h1>
{% endif %}
```

```
{% if nm == 'Django' or st == 5 %}
      </h1>{{nm}} Seat Available</h1>
{% endif %}
```

```
{% if not st == 5 %}
      </h1>Seat Not Available</h1>
{% endif %}
```

if tags may also use the operators **==, !=, <, >, <=, >=, in, not in, is,** and **is not**

if Tag with filter

Syntax:-

```
{% if variable|filter %}
        ……..
{% endif %}
```

Example:-

```
{% if nm|length >= 6 %}
        </h1>Hello {{nm}}</h1>
{% endif %}
```

if else Tag

Syntax:-

```
{% if variable %}
        ……..
{% else %}
        ……..
{% endif %}
```

Example:-

```
{% if nm %}
        </h1>Hello {{nm}}</h1>
{% else}
        <h1>No Course Available</h1>
{% endif %}
```

if else Tag with Condition

Syntax:-

```
{% if condition %}
        ……..
{% else %}
        ……..
{% endif %}
```

Example:-

```
{% if nm ==‘Django’ %}
        </h1>Hello {{nm}}</h1>
{% else}
        <h1>No Course Available</h1>
{% endif %}
```

if elif Tag

Syntax:-

```
{% if variable %}
      ……..
{% elif variable %}
      ……..
{% else %}
      ……..
{% endif %}
```

Example:-

```
{% if nm %}
      </h1>Hello {{nm}}</h1>
{% elif st %}
      </h1>Seats {{st}}</h1>
{% else}
      <h1>No Course Available</h1>
{% endif %}
```

if elif Tag with condition

Syntax:-

```
{% if condition %}
      ……..
{% elif condition %}
      ……..
{% else %}
      ……..
{% endif %}
```

Example:-

```
{% if nm==‘Django’ %}
      </h1>Hello {{nm}}</h1>
{% elif st==5 %}
      </h1>Seats {{st}}</h1>
{% else}
      <h1>No Course Available</h1>
{% endif %}
```

Dot Lookup

Technically, when the template system encounters a dot, it tries the following lookups, in this order:

- Dictionary lookup
- Attribute or method lookup
- Numeric index lookup

for loop Tag

Syntax:-

```
{% for variable in variables %}
    {{ variable }}
{% endfor %}
```

Example:-

```
<ul>
{% for stu in student %}
    <li>{{ stu }}</li>
{% endfor %}
</ul>
```

Syntax:-

```
{% for variable in variables %}
    {{ variable }}
{% empty %}
    Empty
{% endfor %}
```

Syntax:-

```
{% for key, value in data.items %}
    {{ key }}: {{ value }}
{% endfor %}
```

Predefined forloop Variable

Variable	Description

Example:-

```
{% for stu in student %}
    {{forloop.counter}}{{ stu }}
{% endfor %}
```