**53. How to create a template in Django, demonstrate using suitable example.**

**Steps to Create a Template in Django:**

1. **Set up Template Directory**: First, define the DIRS option in your settings.py to point to the location where your templates are stored.

In settings.py:

python

Copy code

TEMPLATES = [

{

'BACKEND': 'django.template.backends.django.DjangoTemplates',

'DIRS': [os.path.join(BASE\_DIR, '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',

],

},

},

]

1. **Create a Template**: Create a templates folder in your project directory, and then create HTML files with Django template tags.

For example, create a home.html in the templates folder:

html

Copy code

<!-- templates/home.html -->

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Home Page</title>

</head>

<body>

<h1>Welcome, {{ name }}</h1>

<p>Current year: {{ year }}</p>

</body>

</html>

1. **Render the Template in a View**: In your views.py, you can use Django's render() function to render the template and pass context data.

Example of a view rendering the template:

python

Copy code

from django.shortcuts import render

def home(request):

context = {

'name': 'John Doe',

'year': 2024

}

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

1. **Configure URL to Display the View**: In urls.py, link the view to a URL.

python

Copy code

from django.urls import path

from . import views

urlpatterns = [

path('', views.home, name='home'),

]

**54. How to create views in Django. Explain with example**.

 **Create a View Function**: Views are defined in the views.py file of your Django application.

Example of a simple view function:

python

Copy code

from django.http import HttpResponse

from django.shortcuts import render

def greeting(request):

return HttpResponse("Hello, World!")

def home(request):

return render(request, 'home.html', {'message': 'Welcome to the Home Page'})

 **Link the View to a URL**: In the urls.py file, map the view to a specific URL pattern.

python

Copy code

from django.urls import path

from . import views

urlpatterns = [

path('', views.home, name='home'),

path('greet/', views.greeting, name='greeting'),

]

 **Run the Server**: Start the Django development server and visit the respective URLs.

Copy code

python manage.py runserver

**55. How forms are created in Django. Implement it with suitable example**.

**Steps to Create and Use Forms in Django:**

1. **Define a Form**: Create a Python class that inherits from django.forms.Form or django.forms.ModelForm.

Example of a basic form:

python

Copy code

# forms.py

from django import forms

class UserForm(forms.Form):

name = forms.CharField(max\_length=100)

email = forms.EmailField()

message = forms.CharField(widget=forms.Textarea)

1. **Create a View to Handle the Form**: In your views.py, create a view function to handle form submission and rendering.

python

Copy code

# views.py

from django.shortcuts import render

from .forms import UserForm

def contact(request):

if request.method == 'POST':

form = UserForm(request.POST)

if form.is\_valid():

# Process the data (e.g., save it to the database)

name = form.cleaned\_data['name']

email = form.cleaned\_data['email']

message = form.cleaned\_data['message']

return render(request, 'thanks.html', {'name': name})

else:

form = UserForm()

return render(request, 'contact.html', {'form': form})

* + The view checks if the form was submitted (request.method == 'POST').
  + If the form is valid, it processes the data (e.g., saves to the database or sends an email).
  + Otherwise, it renders the form again for the user to fill out.

1. **Create the Form Template**: Create the HTML template for the form (contact.html).

html

Copy code

<!-- contact.html -->

<h1>Contact Us</h1>

<form method="post">

{% csrf\_token %}

{{ form.as\_p }}

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

</form>

The {% csrf\_token %} is necessary for security, and {{ form.as\_p }} renders the form fields as paragraphs.

1. **Create the Response Template**: After form submission, you can display a thank you message (thanks.html).

html

Copy code

<!-- thanks.html -->

<h1>Thank you, {{ name }}!</h1>

<p>Your message has been received.</p>

1. **Configure URL Patterns**: In urls.py, link the view to a URL.

python

Copy code

# urls.py

from django.urls import path

from . import views

urlpatterns = [

path('contact/', views.contact, name='contact'),

]