**Django form validation error message**

The **error\_message** attribute permits you to add custom error messages to the fields. This feature allows you to override the predefined error message and define your own. And, the error message is sent in the form of a dictionary data type.

Forms.py

from django import forms

class EmpDetailsForm(forms.Form):

    Name=forms.CharField(max\_length=10,error\_messages = {'required':'Enter your Full Name'})

    Username=forms.CharField(min\_length=5)

    Password=forms.CharField(min\_length=8,widget=forms.PasswordInput, error\_messages = {'required':'Not use name as password'})

Views.py

from django.shortcuts import render

# Create your views here.

from .forms import EmpDetailsForm

def emp\_form(request):

    error = {}

    if request.method == "POST":

        form = EmpDetailsForm(request.POST)

        error['form'] = form

        if form.is\_valid():

            name = form.cleaned\_data.get("Name")

            print(name)

            username = form.cleaned\_data.get("Username")

            print(username)

            password = form.cleaned\_data.get("Password")

            print(password)

        return render(request,'form.html',error)

    else:

        form=EmpDetailsForm()

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

**form.html**

<form action="/eform" method="post">

    {%csrf\_token%}

{{form.as\_p}}

<input type="submit" value="send">

</form>

**Urls.py**

from django.contrib import admin

from django.urls import path

from app1 import views

urlpatterns = [

    path('admin/', admin.site.urls),

    path('eform',views.emp\_form),

]

**Settings.py**

INSTALLED\_APPS = [

    'django.contrib.admin',

    'django.contrib.auth',

    'django.contrib.contenttypes',

    'django.contrib.sessions',

    'django.contrib.messages',

    'django.contrib.staticfiles',

    'app1'

]

import os.path

template\_path=os.path.join(BASE\_DIR,"templates")

TEMPLATES = [

    {

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

        'DIRS': [template\_path],

**Advanced Templates**

* Template Inheritance
* Template Filters
* Template tags for relative URLs
* Block-end block
* extends
* Advantages of Template Inheritance
* Why Template Filters?
* What is Template Filter?
* How to Create Customized Template Filters?

**Template Inheritance**

One template can derived from another template, which is called template inheritance

Inheritance allows reusability and extensibility

It allows using the content of one template inside another template.

It allows to some new features the existing template without modifying it.

Template inheritance allows you to build a base “skeleton” template that contains all the common elements of your site and defines **blocks** that child templates can override..

**How one template can inherit another template?**

{% extends “templatename” %}

This template tag is used to inherit one template inside another template.

**How to define the block in template?**

{% block <block-name> %}

Content

{% endblock %}

Base.html

<!DOCTYPE html>

<html lang="en">

<head>

    <link rel="stylesheet" href="style.css">

    <title>{% block title %}My amazing site{% endblock %}</title>

</head>

<body>

    <div id="sidebar">

        {% block sidebar %}

        <ul>

            <li><a href="/">Home</a></li>

            <li><a href="/blog/">Blog</a></li>

        </ul>

        {% endblock %}

    </div>

    <div id="content">

        {% block content %} {% endblock %}

    </div>

</body>

</html>

Child1.html

{% extends "base.html" %}

{% block title %}My amazing blog{% endblock %}

{% block content %}

<h1> This is content </h1>

{% endblock %}

**Views.py**

from django.shortcuts import render

from django.http import HttpResponse

# Create your views here.

def display(request):

    return render(request,"child1.html")

urls.py

from django.contrib import admin

from django.urls import path

from app1 import views

urlpatterns = [

    path('admin/', admin.site.urls),

    path('display/',views.display)

]

Settings.py

INSTALLED\_APPS = [

    'django.contrib.admin',

    'django.contrib.auth',

    'django.contrib.contenttypes',

    'django.contrib.sessions',

    'django.contrib.messages',

    'django.contrib.staticfiles',

    'app1',

]

import os.path

template\_path=os.path.join(BASE\_DIR,"templates")

TEMPLATES = [

    {

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

        'DIRS': [template\_path],

        'APP\_DIRS': True,