Styling Django Forms with django-crispy-forms

- Django by default doesn't provide any Django form styling method due to which it takes a lot of effort and precious time to beautifully style a form.
- **django-crispy-forms** solves this problem for us. It will let you control the rendering behavior of your Django forms in a very elegant and DRY way.

Modules required:

- pip install django
- pip install django-crispy-forms
- pip install crispy-bootstrap5

Configuring Django settings:

- Add **'crispy_forms'** and **crispy_bootstrap5** to the **INSTALLED_APPS** in settings.py file,

```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'employeeapp.apps.EmployeeappConfig',
    'crispy_forms', # add this one
    'crispy_bootstrap5', # add this one
]
```

- Next we need to add **bootstrap5** template pack variables like below,

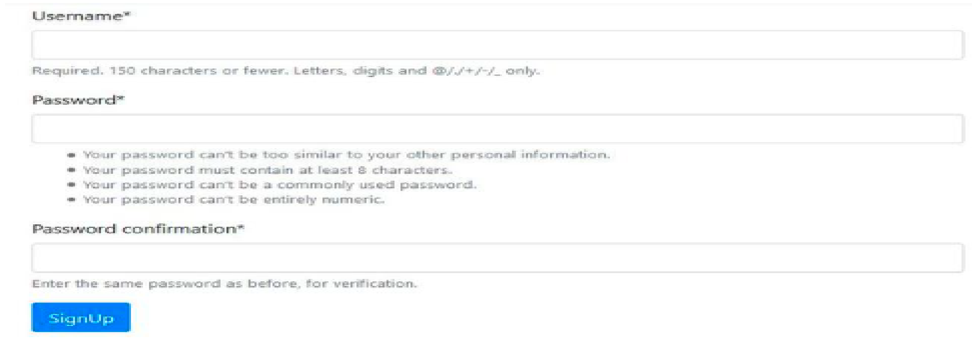
```
CRISPY_ALLOWED_TEMPLATE_PACKS = "bootstrap5"

CRISPY_TEMPLATE_PACK = 'bootstrap5'
```

- Till now we have configured settings needed for django-crispy-forms.

Using django-crispy-forms in Django templates:

- First, we need to load **django-crispy-forms** tags in our django template.
- For loading django-crispy-forms tags, add {% load crispy_forms_filters %} on the top of your django template or {% load crispy_forms_tags %} on the top of your django template
- Now to style any form with django-crispy-forms , replace {{ form }} with **{{ form|crispy }}** or **{{form.fieldName|as_crispy_field }}**
- Now you have successfully styled your form with django-crispy-forms.
- Now you can see changes in your Django form by running server like **python manage.py runserver**



Username*

Required. 150 characters or fewer. Letters, digits and @/./+/-/_ only.

Password*

- Your password can't be too similar to your other personal information.
- Your password must contain at least 8 characters.
- Your password can't be a commonly used password.
- Your password can't be entirely numeric.

Password confirmation*

Enter the same password as before, for verification.

models.py

```python
from django.db import models

class Position(models.Model):
    title = models.CharField(max_length=50)

    def __str__(self):
        return self.title

class Employee(models.Model):
    fullname = models.CharField(max_length=100, help_text='enter your full name')
    emp_code = models.CharField(max_length=3)
    mobile = models.BigIntegerField(unique=True)
    position = models.ForeignKey(Position, on_delete=models.CASCADE)

    def __str__(self):
        return self.fullname
```

admin.py

```python
from django.contrib import admin
from employeeapp.models import Employee, Position

class EmployeeAdmin(admin.ModelAdmin):
    list_display = ['id', 'fullname', 'mobile', 'emp_code', 'position']

class PositionAdmin(admin.ModelAdmin):
    list_display = ['id', 'title']

admin.site.register(Employee,EmployeeAdmin)
admin.site.register(Position, PositionAdmin)
```

forms.py

```python
from django import forms
from employeeapp.models import Employee

class Employee_ModelForm(forms.ModelForm):
    class Meta:
        model = Employee
        fields = ['fullname', 'mobile', 'emp_code', 'position']
        # fields = '__all__'
        labels = {
            "fullname" : "Full Name",
            "emp_code" : "EMP.CODE"
        }
```

views.py

```python
from django.shortcuts import render,redirect
from django.http import HttpResponse
from employeeapp.models import Employee
from employeeapp.forms import Employee_ModelForm

def EmployeeListView(request):
    employee_list = Employee.objects.all()
    context = {
        "employee_list" : employee_list,
        "message" : "Employee records not available"
    }
    return render(request, 'employee_list.html', context)

def EmployeeCreateView(request):
    if request.method=='POST':
        form = Employee_ModelForm(request.POST)
        if form.is_valid():
            form.save()
            return redirect('employee_list')
        else:
            context = {
                'error' : "Employee record not created successfully"
            }
            return render(request, 'employee_create.html', context)
    else:
        form = Employee_ModelForm()
        context = {"form" : form}
        return render(request, 'employee_create.html', context)
```

---

base.html

```html
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>
      {% block title %}      {% endblock %}
    </title>
   <meta charset="utf-8">
   <meta name="viewport" content="width=device-width, initial-scale=1">
   <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@4.6.2/dist/css/bootstrap.min.css">
   <script src="https://cdn.jsdelivr.net/npm/jquery@3.6.3/dist/jquery.slim.min.js"></script>
   <script src="https://cdn.jsdelivr.net/npm/popper.js@1.16.1/dist/umd/popper.min.js"></script>
   <script src="https://cdn.jsdelivr.net/npm/bootstrap@4.6.2/dist/js/bootstrap.bundle.min.js"></script>
    <style>
      #base_h1 {  background-color: green ;   color: white;  padding: 20px 10px;  text-align: center; }
```

```
        form{  width: 500px; border: 5px solid black;  padding: 30px; }
      </style>
    </head>
    <body>
      <div class="container">
        <h1 id="base_h1">Django Crispy Forms Concept</h1>
        {% block body %}

        {% endblock %}
      </div>
    </body>
</html>
```

employee_list.html

```
{% extends 'base.html' %}
{% load crispy_forms_filters %}

{% block title %} Employee List {% endblock %}

{% block body %}
   {% if employee_list %}
     <table class="table">
        <thead>
          <tr>
              <th>Id</th>
              <th>Fullname</th>
              <th>Mobile</th>
              <th>EmpCode</th>
              <th>Position</th>
          </tr>
        </thead>
     {% for employee in employee_list %}
       <tbody>
          <tr>
              <td>{{ employee.id }}</td>
              <td>{{ employee.fullname }}</td>
              <td>{{ employee.mobile }}</td>
              <td>{{ employee.emp_code }}</td>
              <td>{{ employee.position }}</td>
          </tr>
       </tbody>
     {% endfor %}

     </table>
   {% else %}
     <h1>{{ message }}</h1>
   {% endif %}
{% endblock %}
```

employee_create.html

```
{% extends 'base.html' %}
{% load crispy_forms_filters %}
{#{% load crispy_forms_tags %}#}

{% block title %} Employee Form {% endblock %}

{% block body %}
    {% if form %}
   <form method="POST">
     {% csrf_token %}
     <table>
           {# {{ form|crispy }} #}
           {{ form.fullname|as_crispy_field }}
           {{ form.mobile|as_crispy_field }}
           <div class="row">
               <div class="col-md-4">
                   {{ form.emp_code|as_crispy_field }}
               </div>

               <div class="col-md-8">
                   {{ form.position|as_crispy_field }}
               </div>
           </div>
         </table>
         <input type="submit" value="Send">
       </form>
    {% else %}
      <h1>{{ error }}</h1>
    {% endif %}
{% endblock %}
```

urls.py

```
from django.contrib import admin
from django.urls import path
from employeeapp import views
urlpatterns = [
    path('admin/', admin.site.urls),

    path('employee/list/', views.EmployeeListView, name='employee_list'),
    path('employee/create/', views.EmployeeCreateView, name='employee_create'),
]
```

- Next execute the makemigrations and migrate, createsuperuser and runserver commands
- Next Test your urls from browser