1. Create forms.py file

from django import forms  
  
class FormName(forms.Form):  
 name = forms.CharField()  
 email = forms.EmailField()  
 text = forms.CharField(widget=forms.Textarea)

2. Create views (app folder)

from django.shortcuts import render  
from . import forms ### might be also from basicapp import forms  
  
def index(request): ### just for index page  
 return render(request, 'basicapp/index.html')  
  
def form\_name\_view(request):  
 form = forms.FormName() ### Importing previously created class of form from form.py  
 return render(request, 'basicapp/form\_page.html', {'form' : form})

{‘form’ : form} → the same as contex parameter, in this case passing form variable (object of FormName class)

3. Views

In this example views.py from project folder:

from django.contrib import admin  
from django.urls import path  
from basicapp import views ### imported  
  
urlpatterns = [   
 url(r'^$', views.index, name='index'), ###index page  
 url(r'^formpage/', views.form\_name\_view, name='form\_name'), ###page with form  
 url('admin/', admin.site.urls),  
]

4. Templates

{{form}} → object of form class (point 2 – def form\_name\_view)

{%crsf\_token%} → mandatory; data sending protection

<!DOCTYPE html>  
<html lang="en">  
<head>  
 <meta charset="UTF-8">  
 <title>Forms</title>  
 <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css" integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T" crossorigin="anonymous">  
</head>  
<body>  
 <h1>Fill the form!</h1>  
 <div class="container">  
 <form method="post">  
 {{form.as\_p}} <!-- >as\_p -> each form field will be displayed as paragraph -->  
 {%csrf\_token%} <!-- if not working put it under form {{form}} tag -->  
 <input type="submit" class="btn btn-primary" value="Submit">  
 </form>  
 </div>  
</body>  
</html>

5. Form validation (app/forms.py)

from django import forms  
from django.core import validators  
  
### Custom validators  
def check\_for\_z(value): ### 'value' inform that the function acts like validator  
 if value[0].lower() != 'z':  
 raise forms.ValidationError('NEEDS TO START WITH Z')  
  
class FormName(forms.Form):  
 name = forms.CharField(validators=[check\_for\_z]) ### check\_for\_z without ()  
 email = forms.EmailField()  
 text = forms.CharField(widget=forms.Textarea)  
  
 #Validation (hidden fields)  
 botcatcher = forms.CharField(required=False,  
 widget=forms.HiddenInput,  
 validators=[validators.MaxLengthValidator(0)]) ### zero length for bot validator