

USER REGISTRATION

Up until this point we have only created users from the command line and were logging into our accounts using the django admin dashboard. This is obviously not ideal and does not work if we want people to be able to create accounts from the front end of our site. This tutorial will show you how to create a sign up page where users can create an account.

The Django logo, featuring the word "django" in a bold, lowercase, sans-serif font. The background is split diagonally from the top-left to the bottom-right. The upper-left triangle is dark blue, and the lower-right triangle is black. The logo is positioned in the white triangular area in the center-right of the image.

INSTALLING CRISPY FORMS

The first thing we should do before continuing is install a pip package called crispy forms which does some nice styling of our forms for us.

To do this simply run: `pip install django-crispy-forms` from the command line.

Now that we have installed crispy we need to add the following line into `settings.py` to configure w framework it will use.

settings.py file

```
CRISPY_TEMPLATE_PACK="bootstrap4"
```

This can go anywhere in the file.

settings.py file

```
INSTALLED_APPS = [  
    'django.contrib.admin',  
    'django.contrib.auth',  
    'django.contrib.contenttypes',  
    'django.contrib.sessions',  
    'django.contrib.messages',  
    'django.contrib.staticfiles',  
    'main.apps.MainConfig',  
  
    'crispy_forms',  
]
```

Add
code

CREATING A NEW APPLICATION

We will start by creating a new application that will hold all of the pages and functionality of our user sign up and log in. I will call my application "register".

To create a new app run: `python manage.py startapp`

```
(venv) C:\Users\MarizzaMil\PycharmProjects\mySite>python manage.py startapp register
```

After we have done this we will need to add the app to our project from `settings.py` within our mysite directory.

settings.py file

```
INSTALLED_APPS = [  
    'django.contrib.admin',  
    'django.contrib.auth',  
    'django.contrib.contenttypes',  
    'django.contrib.sessions',  
    'django.contrib.messages',  
    'django.contrib.staticfiles',  
    'main.apps.MainConfig',  
    'register.apps.RegisterConfig',  
]
```

Add
code

CREATING A SIGN UP PAGE

Fortunately for us Django has created a few templates that we can modify to create a sign up page. The first step to start using these is to create a new file inside of our register app called `forms.py`.

The file should contain the following:

forms.py file

```
from django import forms
from django.contrib.auth.forms import UserCreationForm
from django.contrib.auth.models import User

class RegisterForm(UserCreationForm):
    email = forms.EmailField()

    class Meta:
        model = User
        fields = ["username", "email", "password1", "password2"]
```

By doing this we have modified the default sign up page and added an email field. If there are other fields that you would like to add define them as class variables and add them into the "field" list in the order you'd like them to appear in the form.

CREATING THE VIEW

Now time to write the html that will render our sign up page!

We will start by **creating a new templates folder** inside of our register app. Then we will create a folder inside templates called "register". Finally we will create "register.html" in that folder.

register.html file

```
{% extends "main/base.html" %}

{% block title %}Create an Account{% endblock %}
{% load crispy_forms_tags %}

{% block content %}
    <form method="POST" class="form-group">
        {% csrf_token %}
        {{ form|crispy }}
        <button type="submit" class="btn btn-success">Register</button>
    </form>
{% endblock %}
```

CREATING THE VIEW

Now we need to add a function inside the views.py file of our register app to render this page

views.py file

```
from django.shortcuts import render, redirect
from .forms import RegisterForm

# Create your views here.
def register(response):
    if response.method == "POST":
        form = RegisterForm(response.POST)
        if form.is_valid():
            form.save()

            return redirect("/home")
    else:
        form = RegisterForm()

    return render(response, "register/register.html", {"form": form})
```

CREATING THE VIEW

Finally we need to add a url to this page. We will do this from the `urls.py` file inside of the `mysite` directory.

urls.py file

```
from django.contrib import admin
from django.urls import include, path

from register import views as v

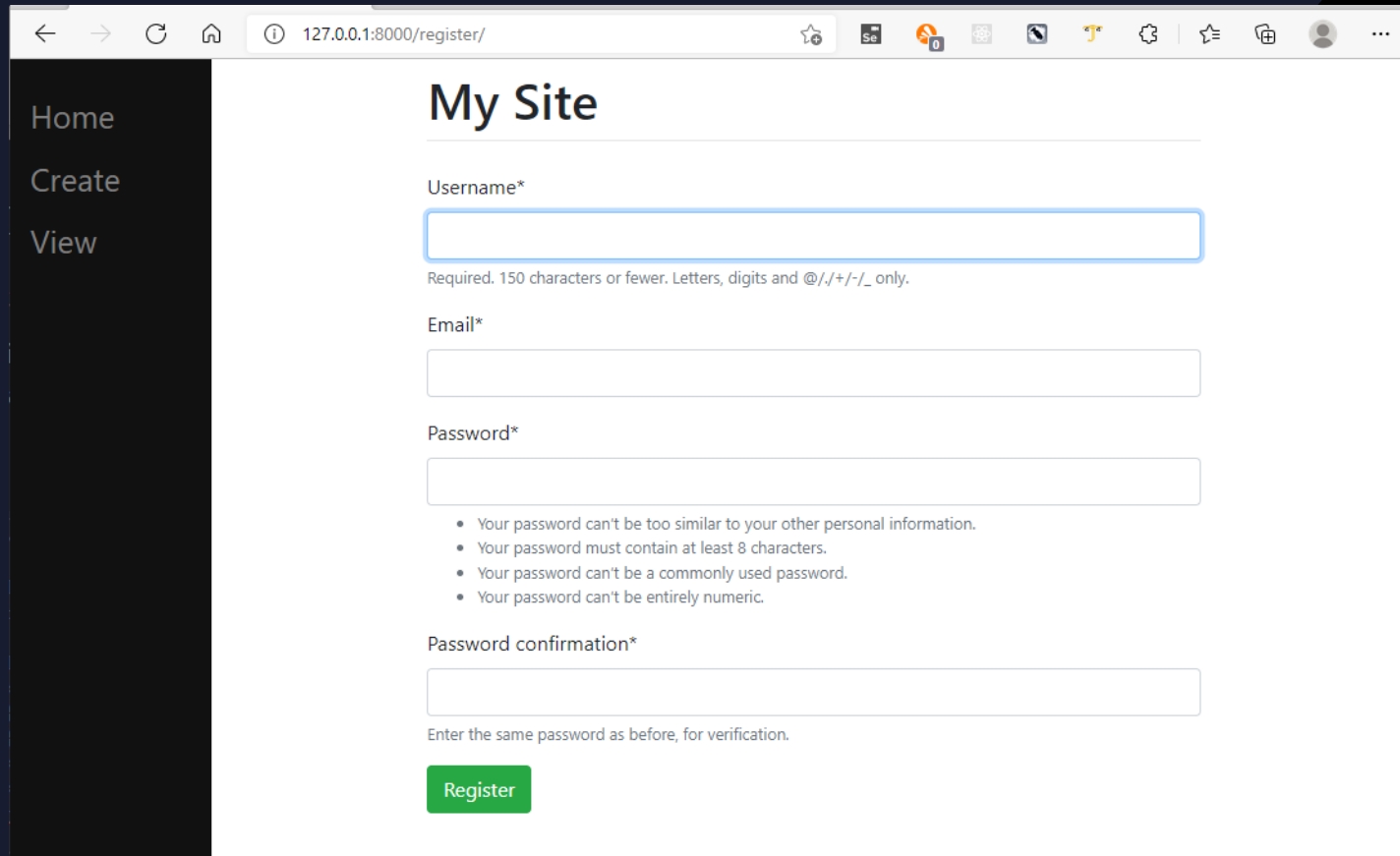
urlpatterns = [
    path('admin/', admin.site.urls),
    path("register/", v.register, name="register"),
    path('', include('main.urls')),
]
```

Add
code



FINAL PRODUCT

If you've done all of the steps outlined above you should now have a functional sign up page that you can access by going to /register.



The screenshot shows a web browser window with the address bar displaying '127.0.0.1:8000/register/'. The page has a dark sidebar on the left with links: 'Home', 'Create', and 'View'. The main content area is titled 'My Site' and contains a registration form. The form includes fields for 'Username*', 'Email*', 'Password*', and 'Password confirmation*'. Below the 'Password*' field, there are four bullet points listing password requirements. At the bottom of the form is a green 'Register' button.

Home
Create
View

My Site

Username*

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

Email*

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.

Register