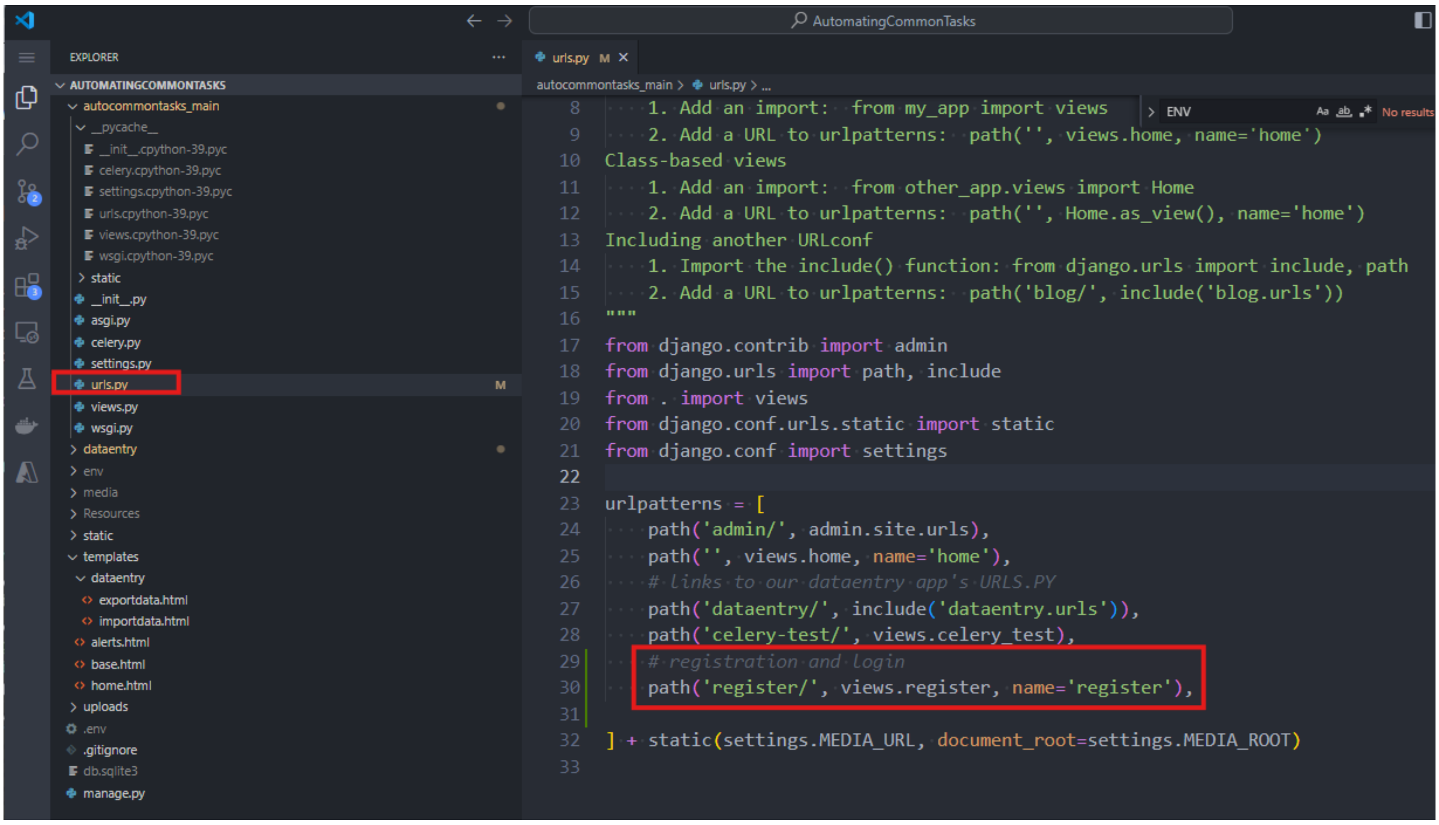
Topic: Registration Module: Part 14

Speaker: Udemy Instructor Rathan Kumar | Notebook: Django: Automating Common Tasks

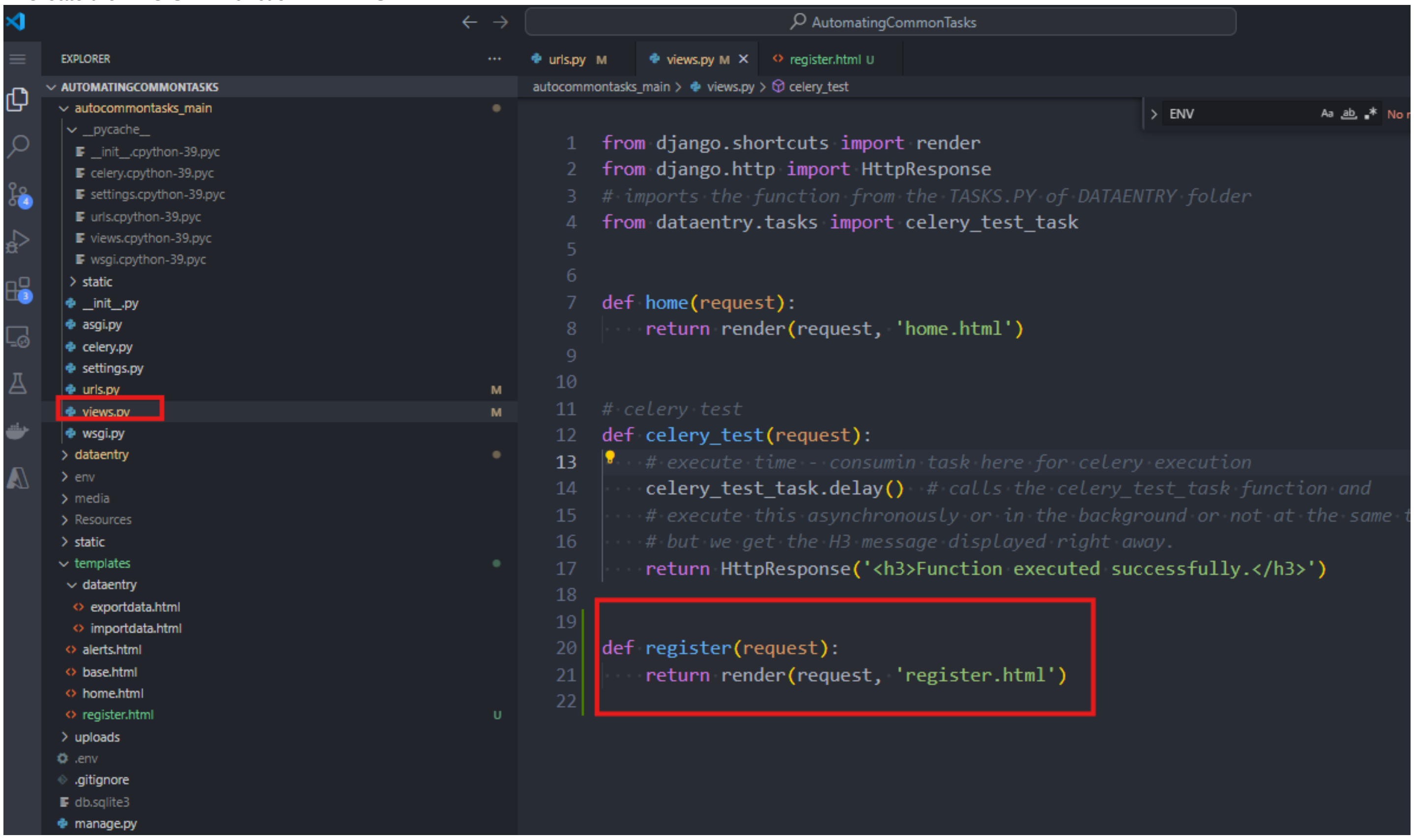
---

1. We need a URL pattern like http://127.0.0.1:8000/register/, so in the main project's URLS.PY.



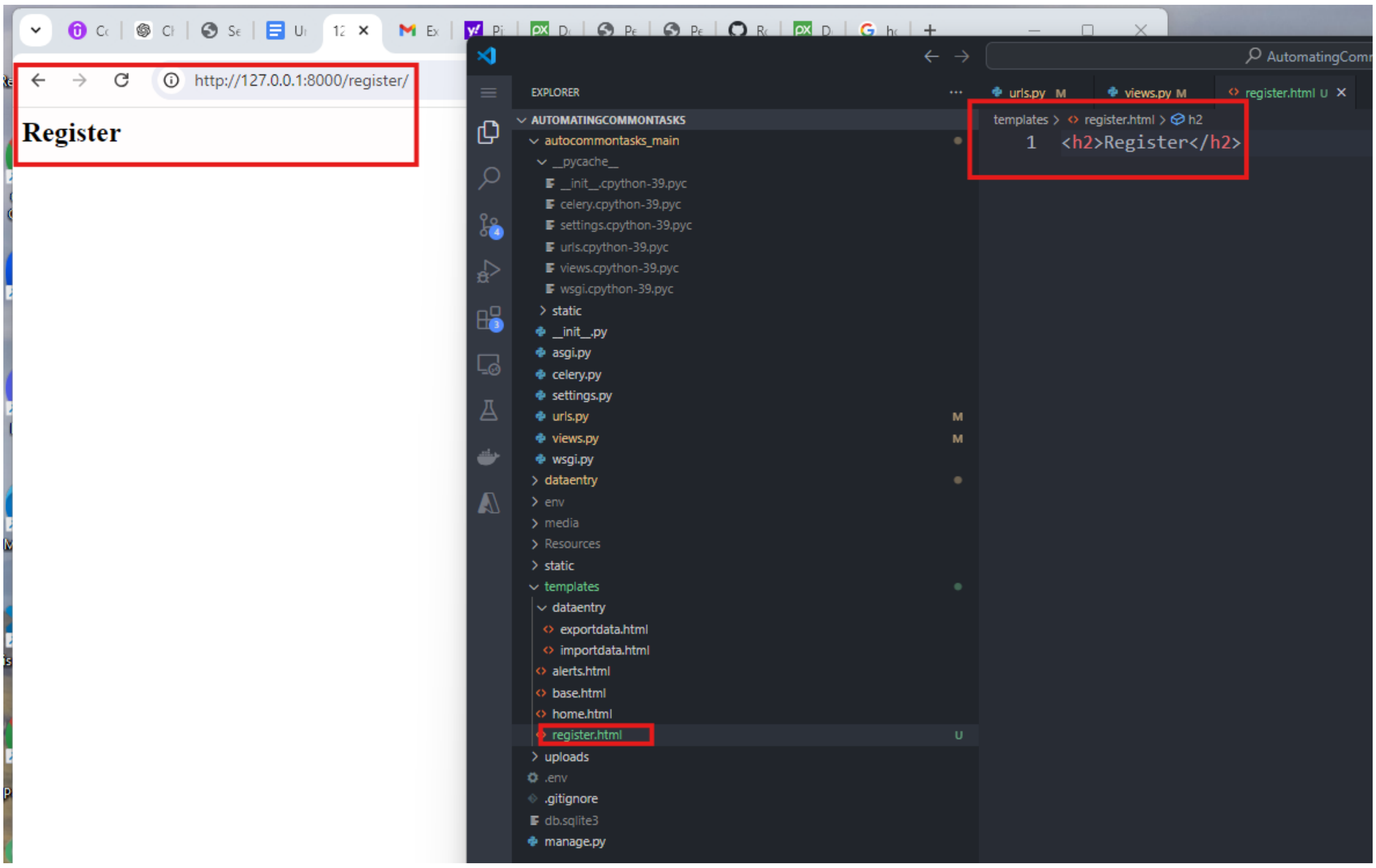
```python
    1. Add an import:  from my_app import views
    2. Add a URL to urlpatterns:  path('', views.home, name='home')
Class-based views
    1. Add an import:  from other_app.views import Home
    2. Add a URL to urlpatterns:  path('', Home.as_view(), name='home')
Including another URLconf
    1. Import the include() function: from django.urls import include, path
    2. Add a URL to urlpatterns:  path('blog/', include('blog.urls'))
"""
from django.contrib import admin
from django.urls import path, include
from . import views
from django.conf.urls.static import static
from django.conf import settings

urlpatterns = [
    path('admin/', admin.site.urls),
    path('', views.home, name='home'),
    # links to our dataentry app's URLS.PY
    path('dataentry/', include('dataentry.urls')),
    path('celery-test/', views.celery_test),
    # registration and login
    path('register/', views.register, name='register'),

] + static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)
```

2. Create the REGISTER function in VIEWS.PY:



```python
from django.shortcuts import render
from django.http import HttpResponse
# imports the function from the TASKS.PY of DATAENTRY folder
from dataentry.tasks import celery_test_task


def home(request):
    return render(request, 'home.html')


# celery test
def celery_test(request):
    # execute time - consumin task here for celery execution
    celery_test_task.delay()  # calls the celery_test_task function and
    # execute this asynchronously or in the background or not at the same t
    # but we get the H3 message displayed right away.
    return HttpResponse('<h3>Function executed successfully.</h3>')


def register(request):
    return render(request, 'register.html')
```

3. Create the REGISTER.HTML in TEMPLATES.



http://127.0.0.1:8000/register/

Register

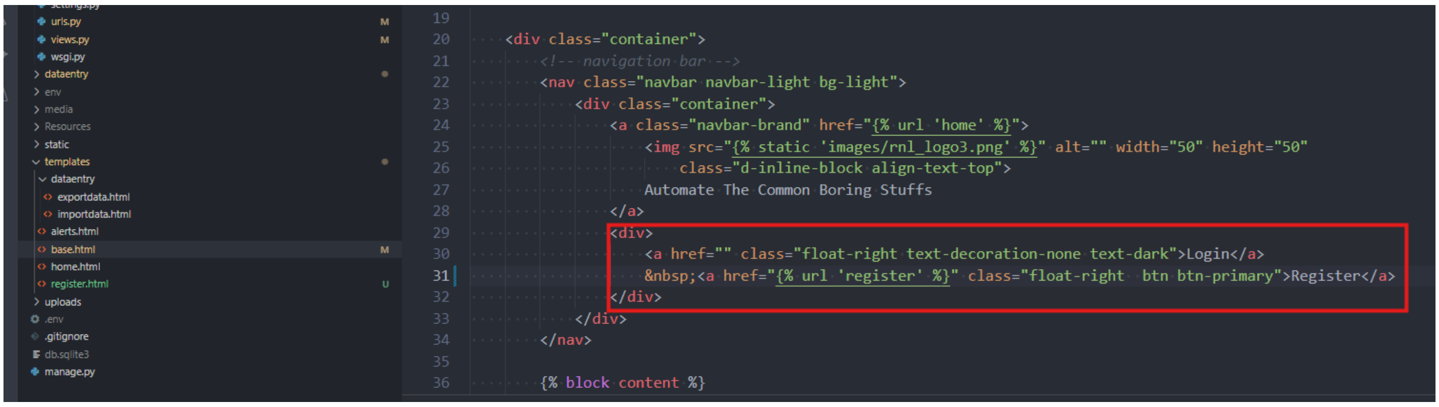
```html
<h2>Register</h2>
```

4. We update our REGISTER.HTML to include our Django tags.

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

{% block content %}

<we add our unique code here>
```

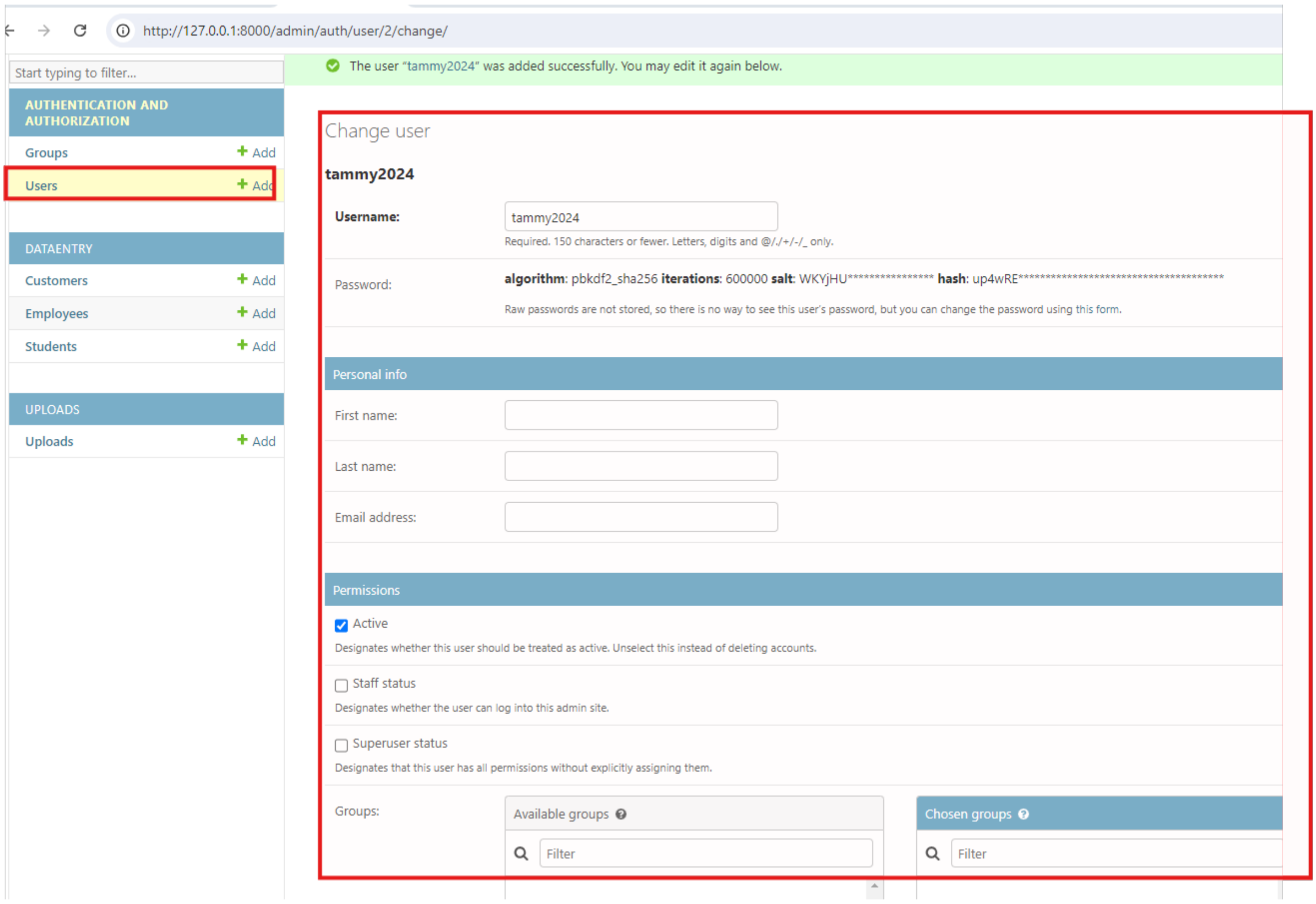
```
{% endblock %}
```

5. We update our BASE.HTML to link our webpages to REGISTER.HTML



```
    <div class="container">
        <!-- navigation bar -->
        <nav class="navbar navbar-light bg-light">
            <div class="container">
                <a class="navbar-brand" href="{% url 'home' %}">
                    <img src="{% static 'images/rnl_logo3.png' %}" alt="" width="50" height="50"
                        class="d-inline-block align-text-top">
                    Automate The Common Boring Stuffs
                </a>
                <div>
                    <a href="" class="float-right text-decoration-none text-dark">Login</a>
                    &nbsp;<a href="{% url 'register' %}" class="float-right  btn btn-primary">Register</a>
                </div>
            </div>
        </nav>

        {% block content %}
```

6. To create the registration form, we can manually create the UI form or automate this using DJANGO MODEL FORM (similar to our Admin Panel User Registration) WITH CRISPY TO FORMAT IT.


http://127.0.0.1:8000/admin/auth/user/2/change/
Start typing to filter...
AUTHENTICATION AND AUTHORIZATION
Groups + Add
Users + Add
DATAENTRY
Customers + Add
Employees + Add
Students + Add
UPLOADS
Uploads + Add

The user "tammy2024" was added successfully. You may edit it again below.

Change user

tammy2024

Username: tammy2024

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

Password: **algorithm**: pbkdf2_sha256 **iterations**: 600000 **salt**: WKYjHU************** **hash**: up4wRE**************************************

Raw passwords are not stored, so there is no way to see this user's password, but you can change the password using this form.

Personal info

First name:

Last name:

Email address:

Permissions

[✓] Active

Designates whether this user should be treated as active. Unselect this instead of deleting accounts.

[ ] Staff status

Designates whether the user can log into this admin site.

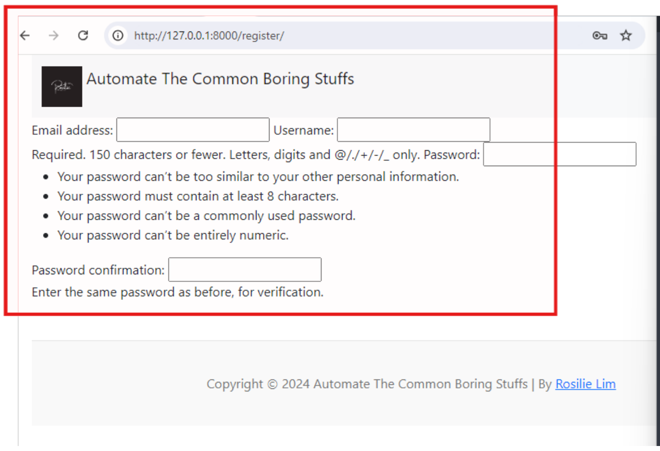
[ ] Superuser status

Designates that this user has all permissions without explicitly assigning them.

Groups: Available groups | Chosen groups

Filter | Filter

7. Create a new file, FORMS.PY, in your main project directory.


http://127.0.0.1:8000/register/

Automate The Common Boring Stuffs

Email address: Username:

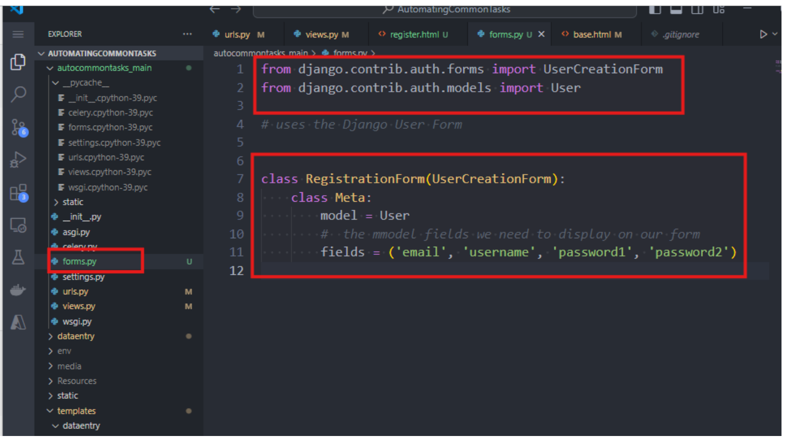
Required. 150 characters or fewer. Letters, digits and @/./+/-/_ only. Password:

- 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.

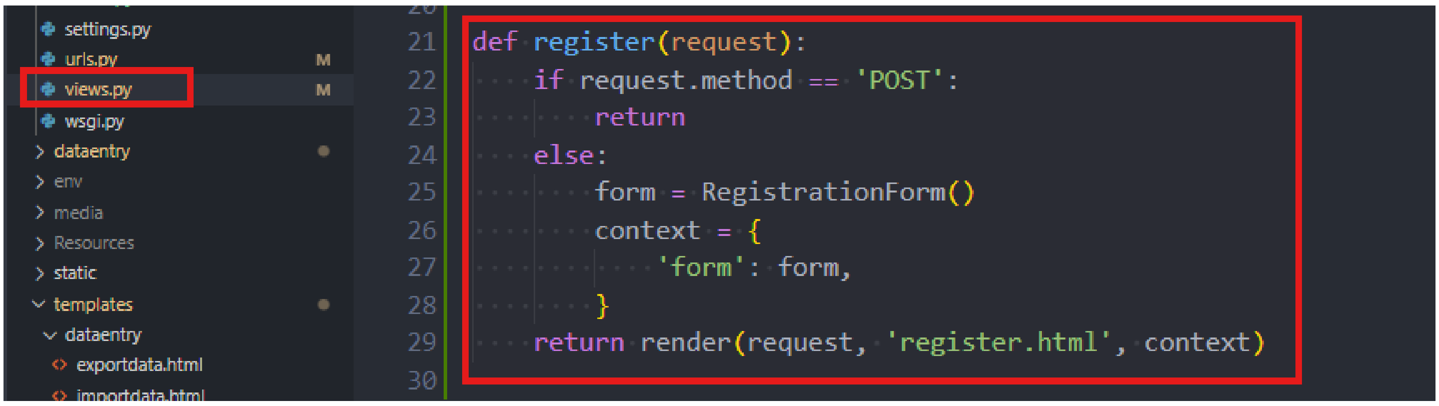
Password confirmation:

Enter the same password as before, for verification.

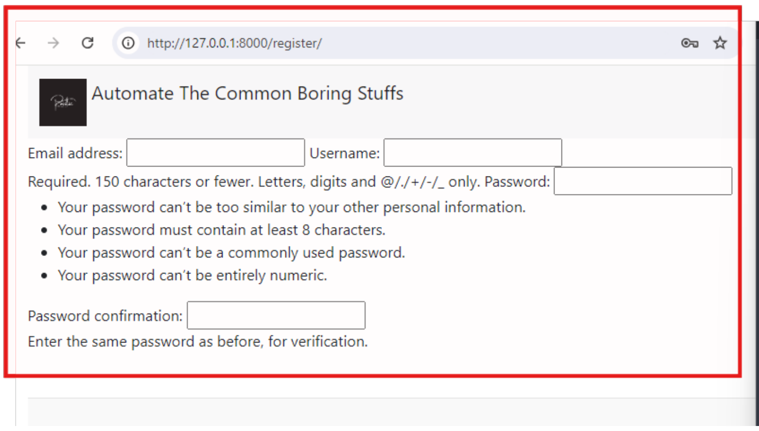
Copyright © 2024 Automate The Common Boring Stuffs | By Rosilie Lim


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

# uses the Django User Form


class RegistrationForm(UserCreationForm):
    class Meta:
        model = User
        #  the mmodel fields we need to display on our form
        fields = ('email', 'username', 'password1', 'password2')
```

8. In our VIEWS.PY, we write:


```python
def register(request):
    if request.method == 'POST':
        return
    else:
        form = RegistrationForm()
        context = {
            'form': form,
        }
    return render(request, 'register.html', context)
```

9. In our REGISTER.HTML, we write


http://127.0.0.1:8000/register/

Automate The Common Boring Stuffs

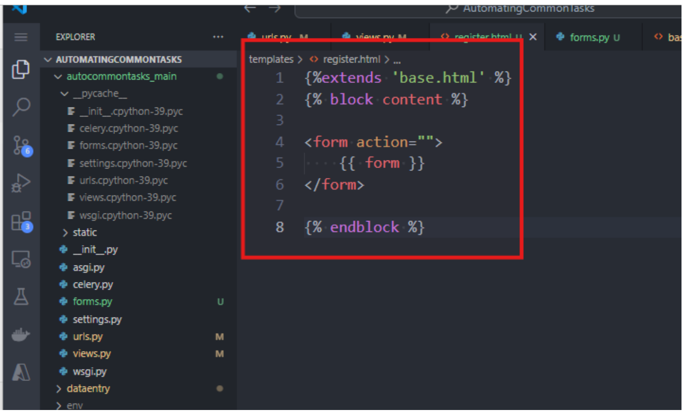
Email address: Username:

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.

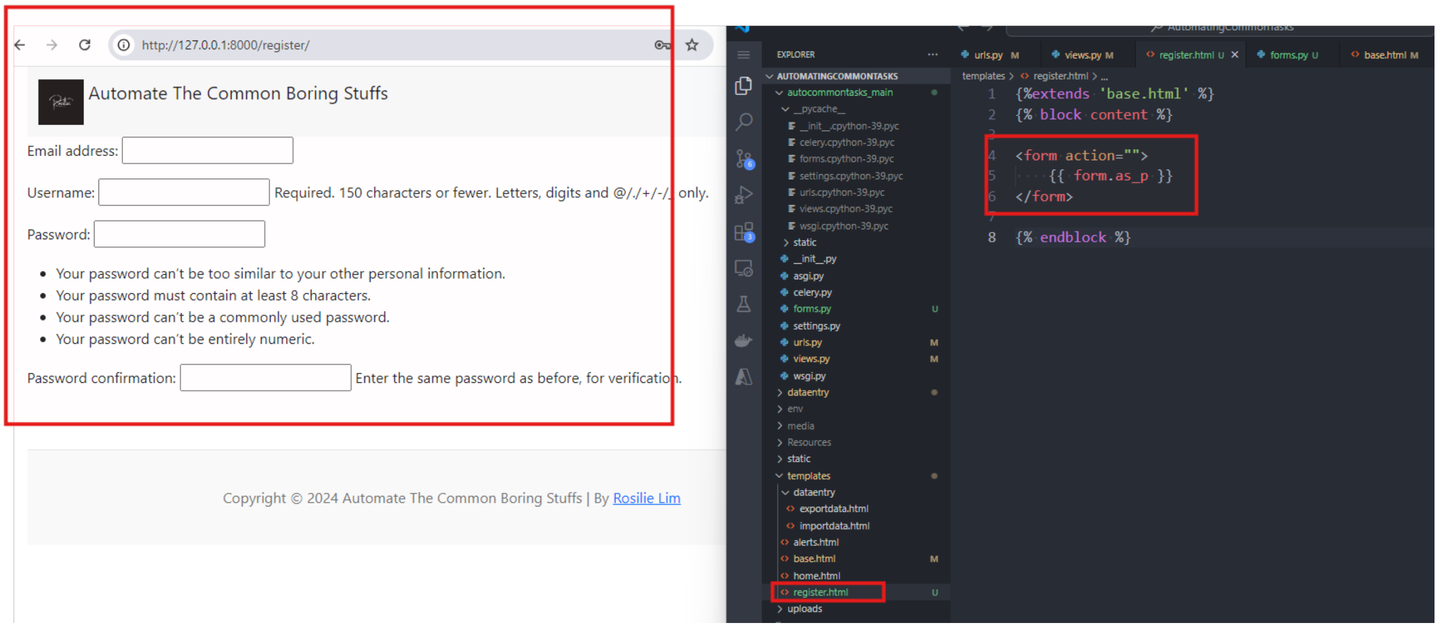
Password confirmation:

Enter the same password as before, for verification.


```html
{%extends 'base.html' %}
{% block content %}

<form action="">
    {{ form }}
</form>

{% endblock %}
```

10. If we use the conventional way to set up the form, it shall be like this:

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

{% block content %}

<form action="">

  {{ form.as_p }}

</form>

{% endblock %}
```



Automate The Common Boring Stuffs

Email address:

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.

Copyright © 2024 Automate The Common Boring Stuffs | By Rosilie Lim

```
{%extends 'base.html' %}
{% block content %}

<form action="">
    {{ form.as_p }}
</form>

{% endblock %}
```

11. To format this form, we can use the CRISPY FORM. See this for the CRISPY FORM DOCUMENTATION.

Installing django-crispy-forms

Install latest stable version into your python environment using pip:

```
pip install django-crispy-forms
```

If you want to install development version (unstable), you can do so doing:

```
pip install git+git://github.com/django-crispy-forms/django-crispy-forms.git@main#egg
```

Or, if you'd like to install the development version as a git repository (so you can `git pull` updates), use the `-e` flag with `pip install`, like so:

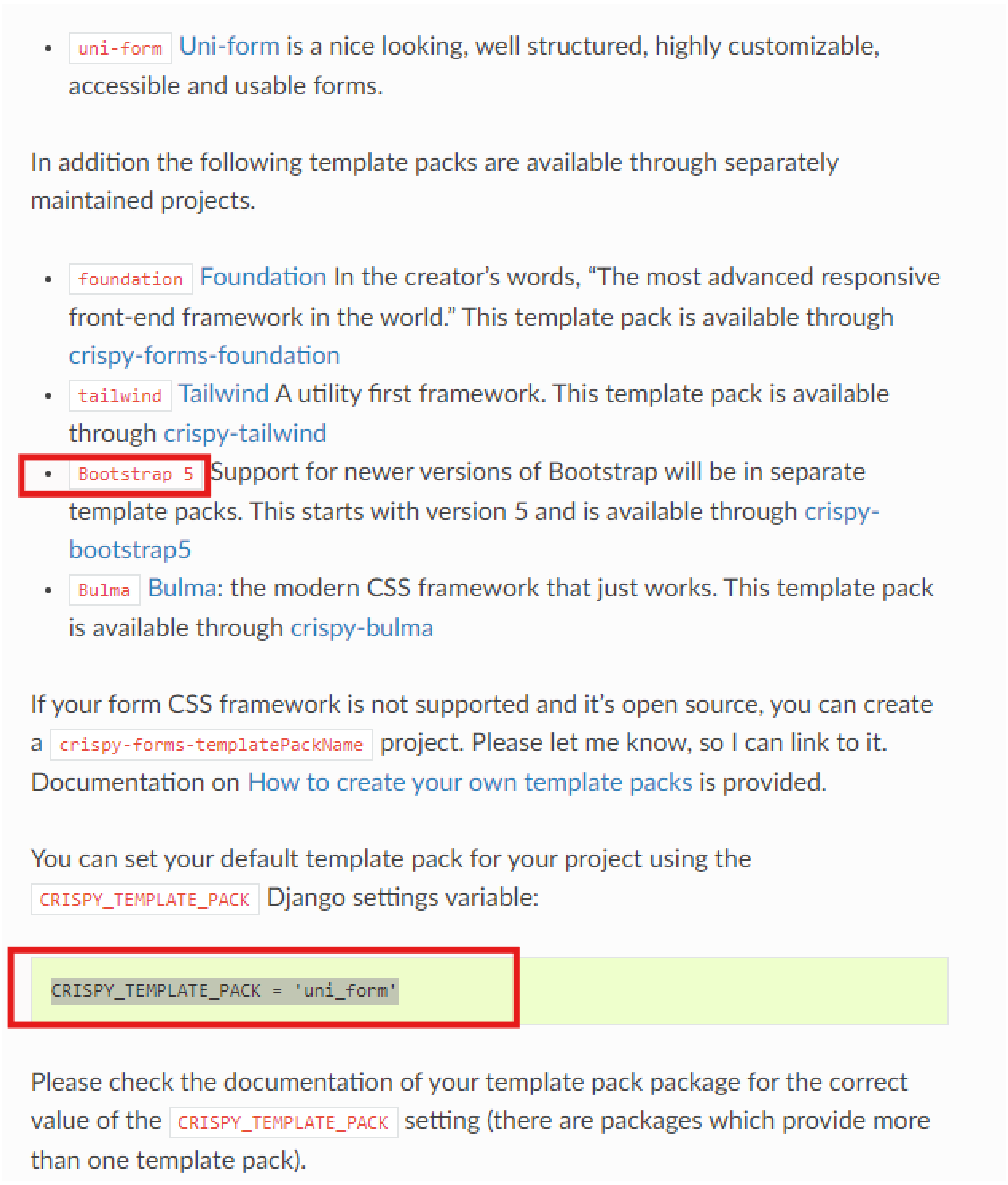
```
pip install -e git+git://github.com/django-crispy-forms/django-crispy-forms.git@main#
```

Once installed add `crispy_forms` to your `INSTALLED_APPS` in settings.py:

```
INSTALLED_APPS = (
    ...
    'crispy_forms',
)
```

In production environments, always activate Django template cache loader. This is available since Django 1.2 and what it does is basically load templates once, then cache the result for every subsequent render. This leads to a significant performance improvement. To see how to set it up refer to the fabulous Django docs page.

Template packs



- `uni-form` Uni-form is a nice looking, well structured, highly customizable, accessible and usable forms.

In addition the following template packs are available through separately maintained projects.

- `foundation` Foundation In the creator's words, "The most advanced responsive front-end framework in the world." This template pack is available through crispy-forms-foundation
- `tailwind` Tailwind A utility first framework. This template pack is available through crispy-tailwind
- `Bootstrap 5` Support for newer versions of Bootstrap will be in separate template packs. This starts with version 5 and is available through crispy-bootstrap5
- `Bulma` Bulma: the modern CSS framework that just works. This template pack is available through crispy-bulma

If your form CSS framework is not supported and it's open source, you can create a `crispy-forms-templatePackName` project. Please let me know, so I can link to it. Documentation on How to create your own template packs is provided.

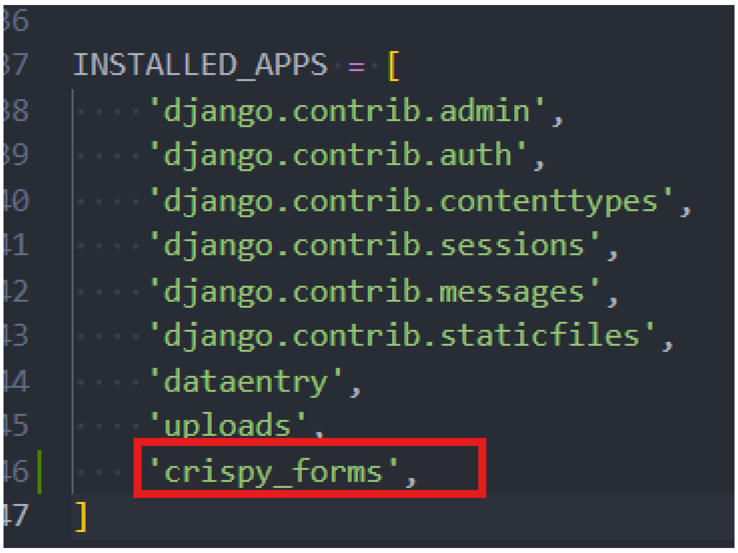
You can set your default template pack for your project using the `CRISPY_TEMPLATE_PACK` Django settings variable:

```
CRISPY_TEMPLATE_PACK = 'uni_form'
```

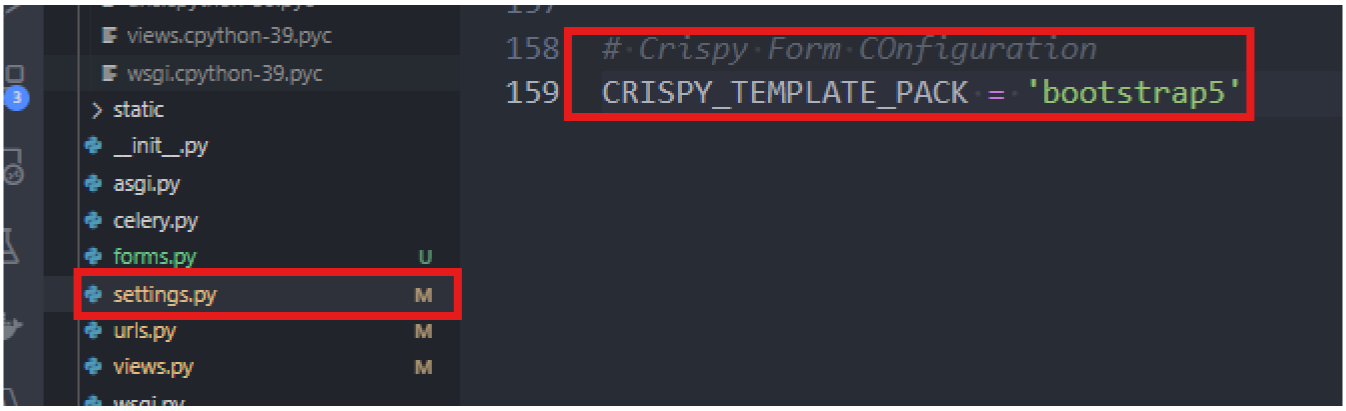
Please check the documentation of your template pack package for the correct value of the `CRISPY_TEMPLATE_PACK` setting (there are packages which provide more than one template pack).

Install Django Crispy Form: `$ pip install django-crispy-forms`

In your SETTINGS.PY, register this in your INSTALLED_APPS



```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'dataentry',
    'uploads',
    'crispy_forms',
]
```

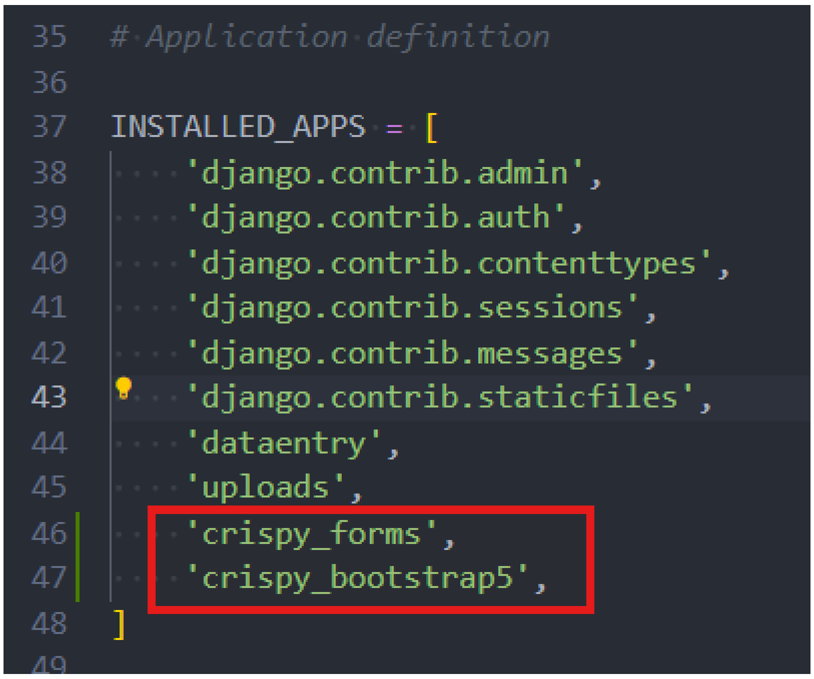


```
# Crispy Form COnfiguration
CRISPY_TEMPLATE_PACK = 'bootstrap5'
```

12. If you check your BASE.HTML the bootstrap we are using is version 5, that is why our CRISPY_TEMPALTE_PACK='bootstrap5' and so we need to install this as well.

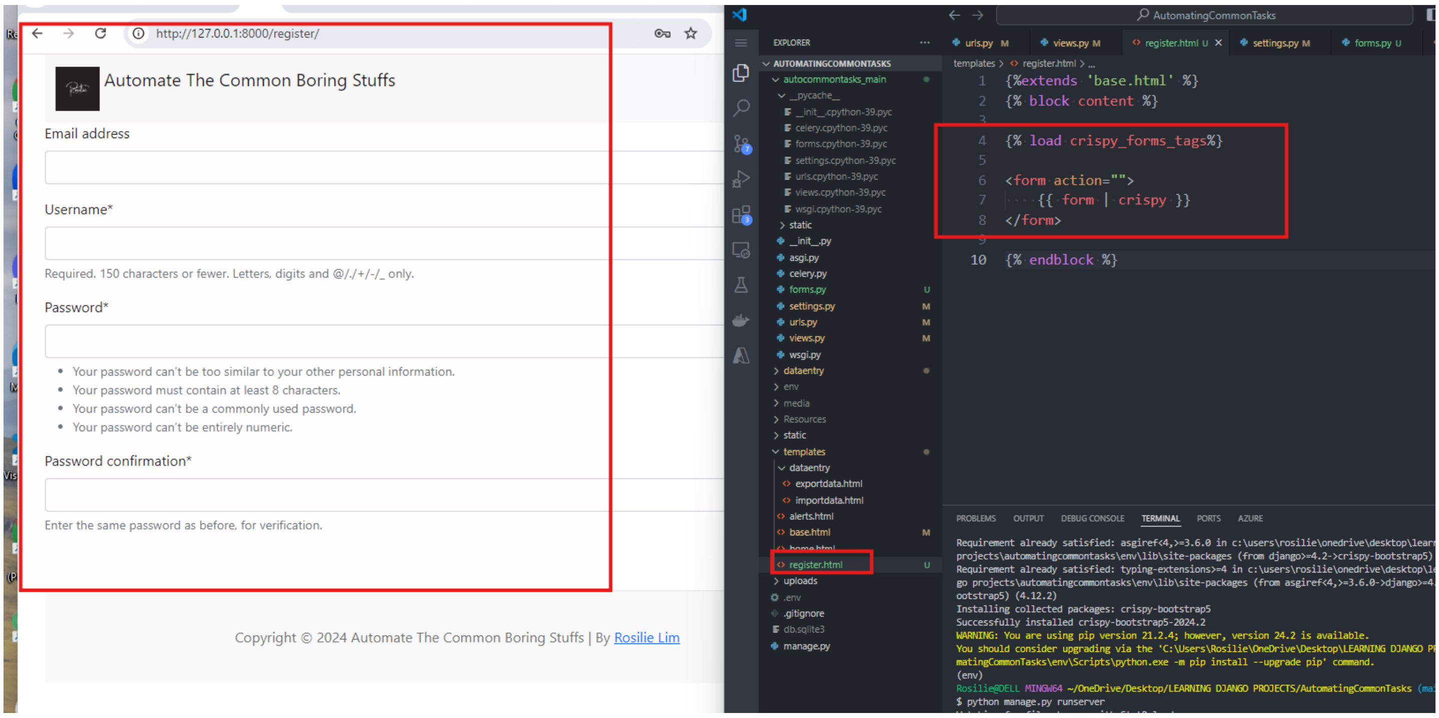
```
$ pip install  crispy-bootstrap5
```

Register this in your INSTALLED_APPS:



```
# Application definition

INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'dataentry',
    'uploads',
    'crispy_forms',
    'crispy_bootstrap5',
]
```

13. Update our REGISTER.HTML and reload:



http://127.0.0.1:8000/register/

Automate The Common Boring Stuffs

Email address

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.

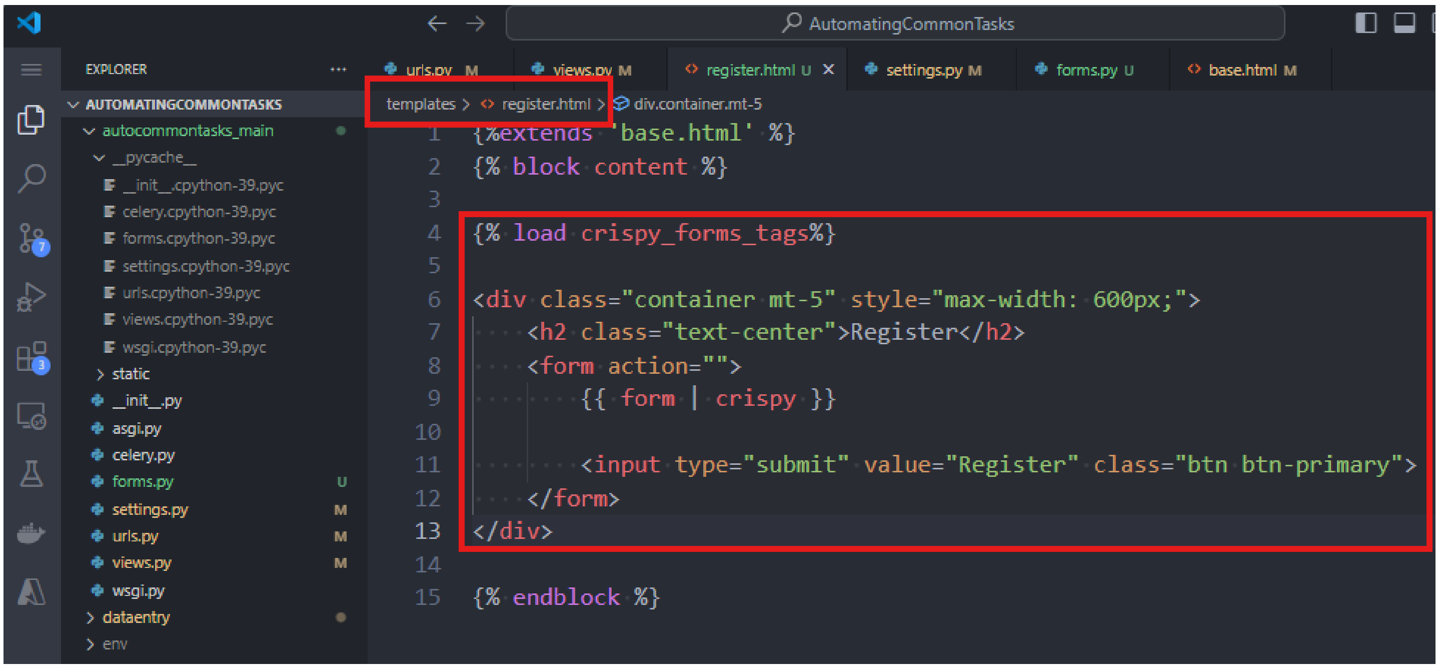
Copyright © 2024 Automate The Common Boring Stuffs | By Rosilie Lim

templates > register.html > ...

```
{%extends 'base.html' %}
{% block content %}

{% load crispy_forms_tags%}

<form action="">
    {{ form | crispy }}
</form>

{% endblock %}
```

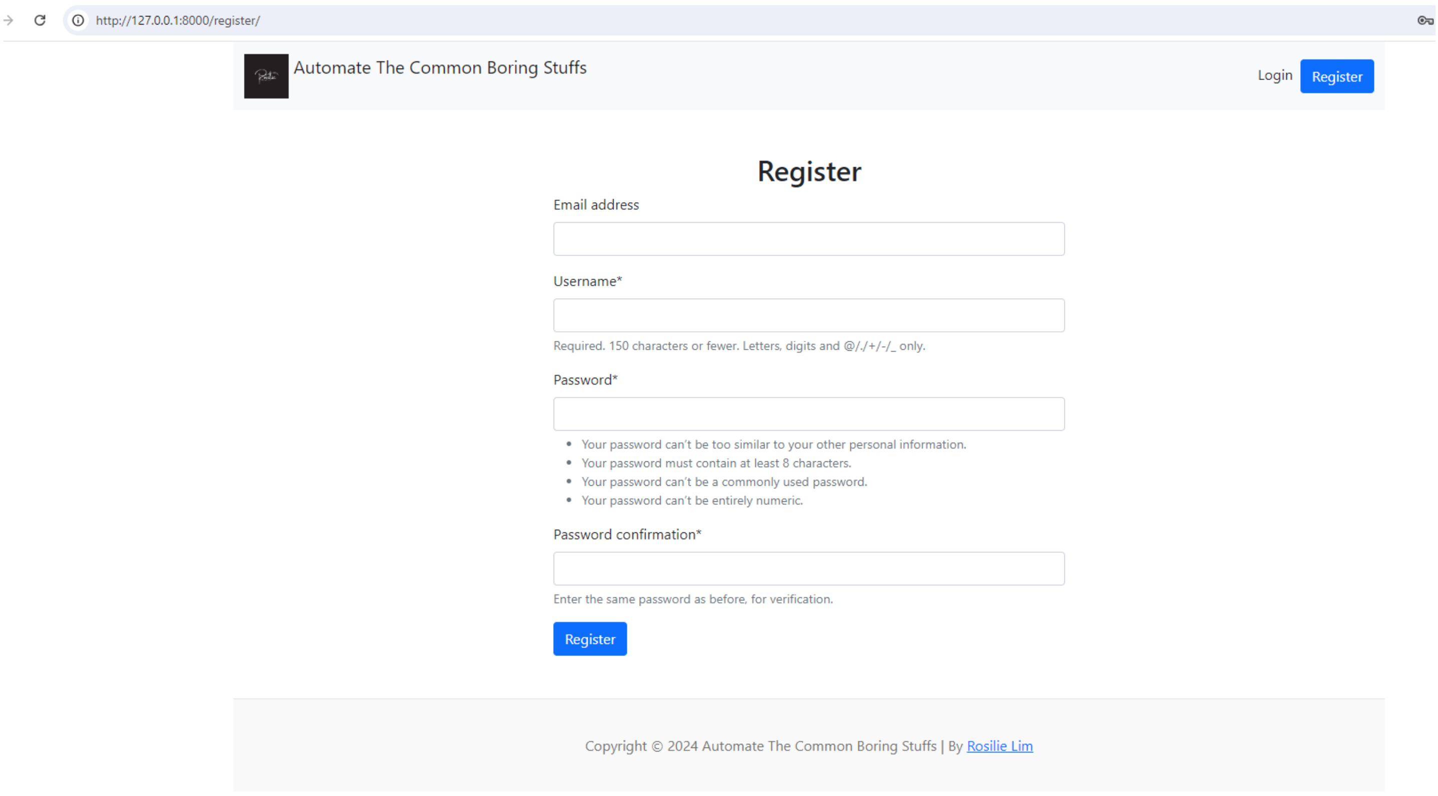
```
Requirement already satisfied: asgiref<4,>=3.6.0 in c:\users\rosilie\onedrive\desktop\learn
projects\automatingcommontasks\env\lib\site-packages (from django>=4.2->crispy-bootstrap5)
Requirement already satisfied: typing-extensions>=4 in c:\users\rosilie\onedrive\desktop\le
go projects\automatingcommontasks\env\lib\site-packages (from asgiref<4,>=3.6.0->django>=4.
ootstrap5) (4.12.2)
Installing collected packages: crispy-bootstrap5
Successfully installed crispy-bootstrap5-2024.2
WARNING: You are using pip version 21.2.4; however, version 24.2 is available.
You should consider upgrading via the 'C:\Users\Rosilie\OneDrive\Desktop\LEARNING DJANGO PR
matingCommonTasks\env\Scripts\python.exe -m pip install --upgrade pip' command.
(env)
Rosilie@DELL MINGW64 ~/OneDrive/Desktop/LEARNING DJANGO PROJECTS/AutomatingCommonTasks (ma
$ python manage.py runserver
```

14. Further updating our REGISTER.HTML.



templates > register.html > div.container.mt-5

```
{%extends 'base.html' %}
{% block content %}

{% load crispy_forms_tags%}

<div class="container mt-5" style="max-width: 600px;">
    <h2 class="text-center">Register</h2>
    <form action="">
        {{ form | crispy }}

        <input type="submit" value="Register" class="btn btn-primary">
    </form>
</div>

{% endblock %}
```



http://127.0.0.1:8000/register/

Automate The Common Boring Stuffs

Login Register

Register

Email address

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.

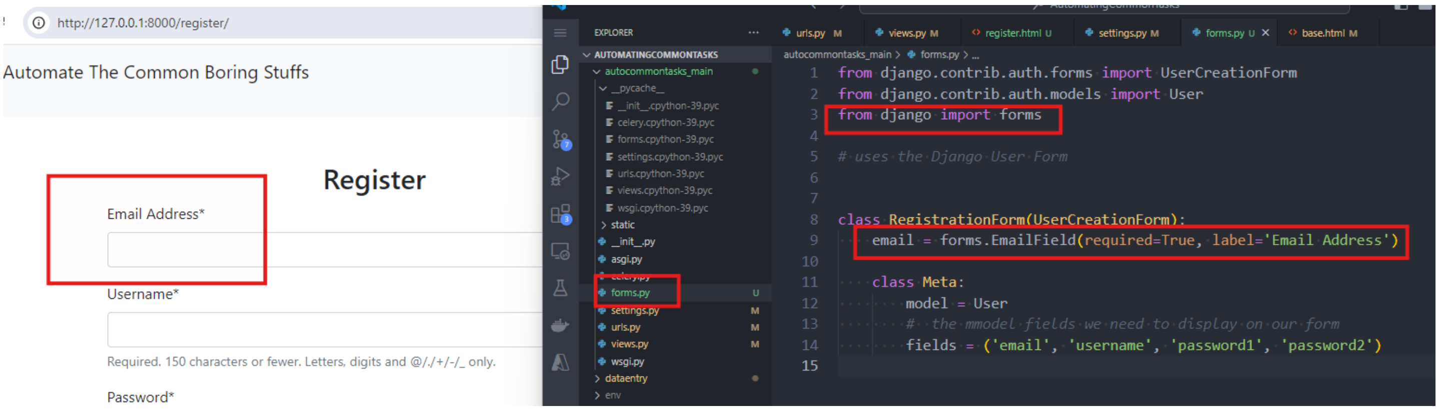
Password confirmation*

Enter the same password as before, for verification.

Register

Copyright © 2024 Automate The Common Boring Stuffs | By Rosilie Lim

15. In the default User Form, the email is an optional field. We can make this as a required field by updating our FORMS.PY



http://127.0.0.1:8000/register/

Automate The Common Boring Stuffs

Register

Email Address*

Username*

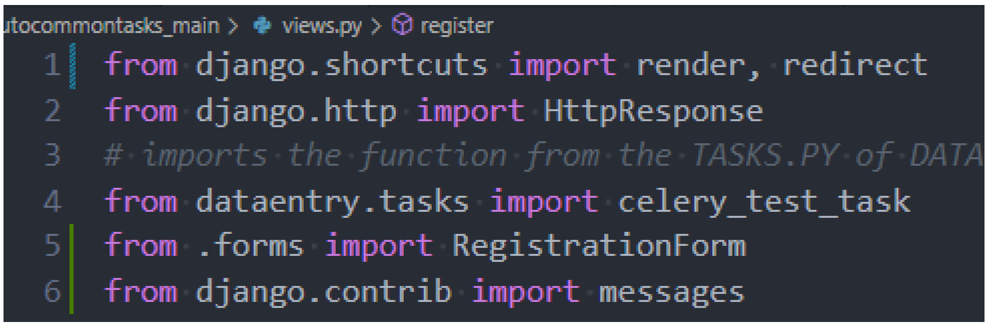
Required. 150 characters or fewer. Letters, digits and @/./+/-/_ only.

Password*

autocommontasks_main > forms.py > ...

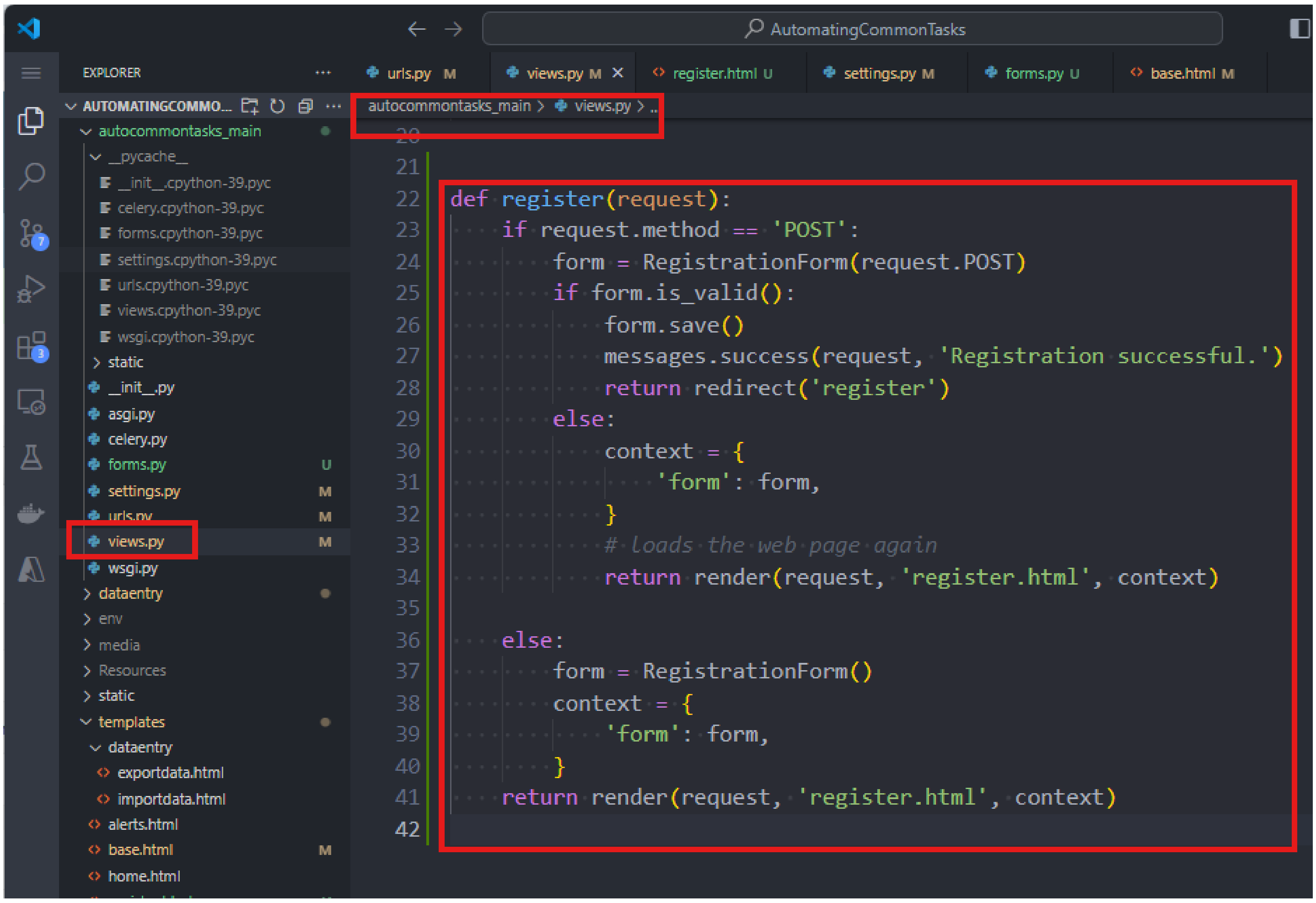
```python
from django.contrib.auth.forms import UserCreationForm
from django.contrib.auth.models import User
from django import forms

# uses the Django User Form


class RegistrationForm(UserCreationForm):
    email = forms.EmailField(required=True, label='Email Address')

    class Meta:
        model = User
        #  the mmodel fields we need to display on our form
        fields = ('email', 'username', 'password1', 'password2')
```

16. Our VIEWS.PY as:



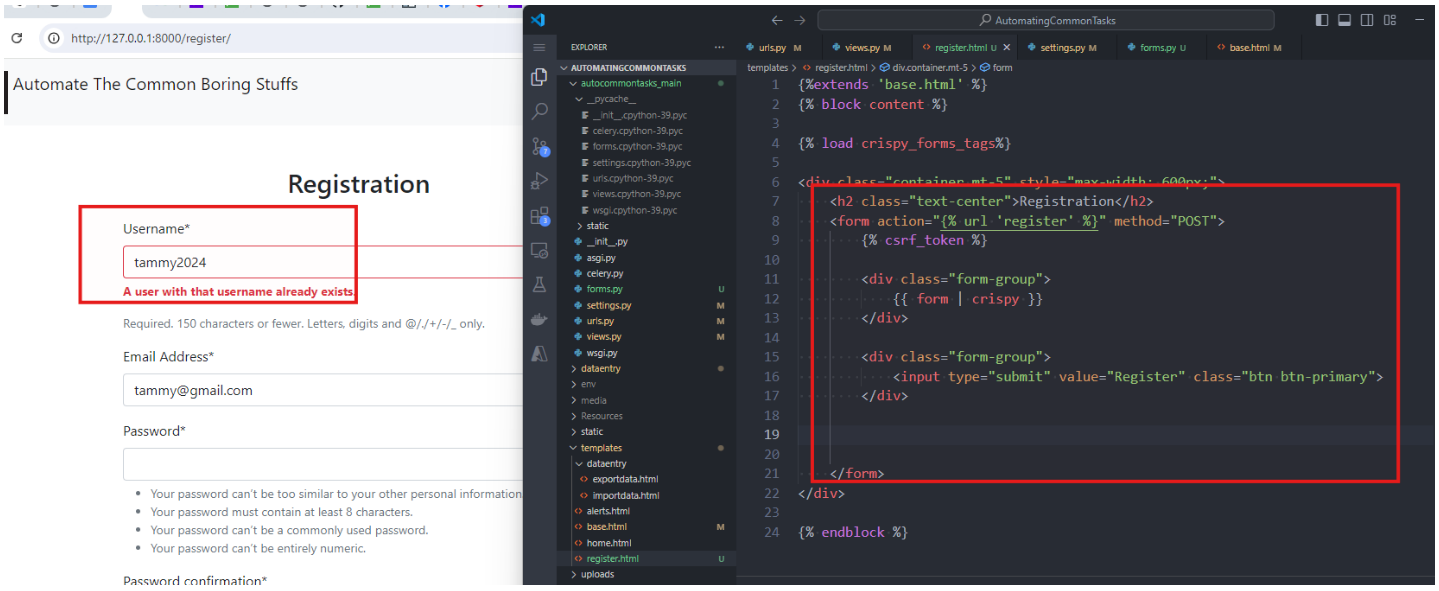
utocommontasks_main > views.py > register

```python
from django.shortcuts import render, redirect
from django.http import HttpResponse
# imports the function from the TASKS.PY of DATA
from dataentry.tasks import celery_test_task
from .forms import RegistrationForm
from django.contrib import messages
```



```python
def register(request):
    if request.method == 'POST':
        form = RegistrationForm(request.POST)
        if form.is_valid():
            form.save()
            messages.success(request, 'Registration successful.')
            return redirect('register')
        else:
            context = {
                'form': form,
            }
            # loads the web page again
            return render(request, 'register.html', context)

    else:
        form = RegistrationForm()
        context = {
            'form': form,
        }
    return render(request, 'register.html', context)
```

17. Testing the registration form with an existing user:



http://127.0.0.1:8000/register/

Automate The Common Boring Stuffs

Registration

Username*

tammy2024

A user with that username already exists.

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

Email Address*

tammy@gmail.com

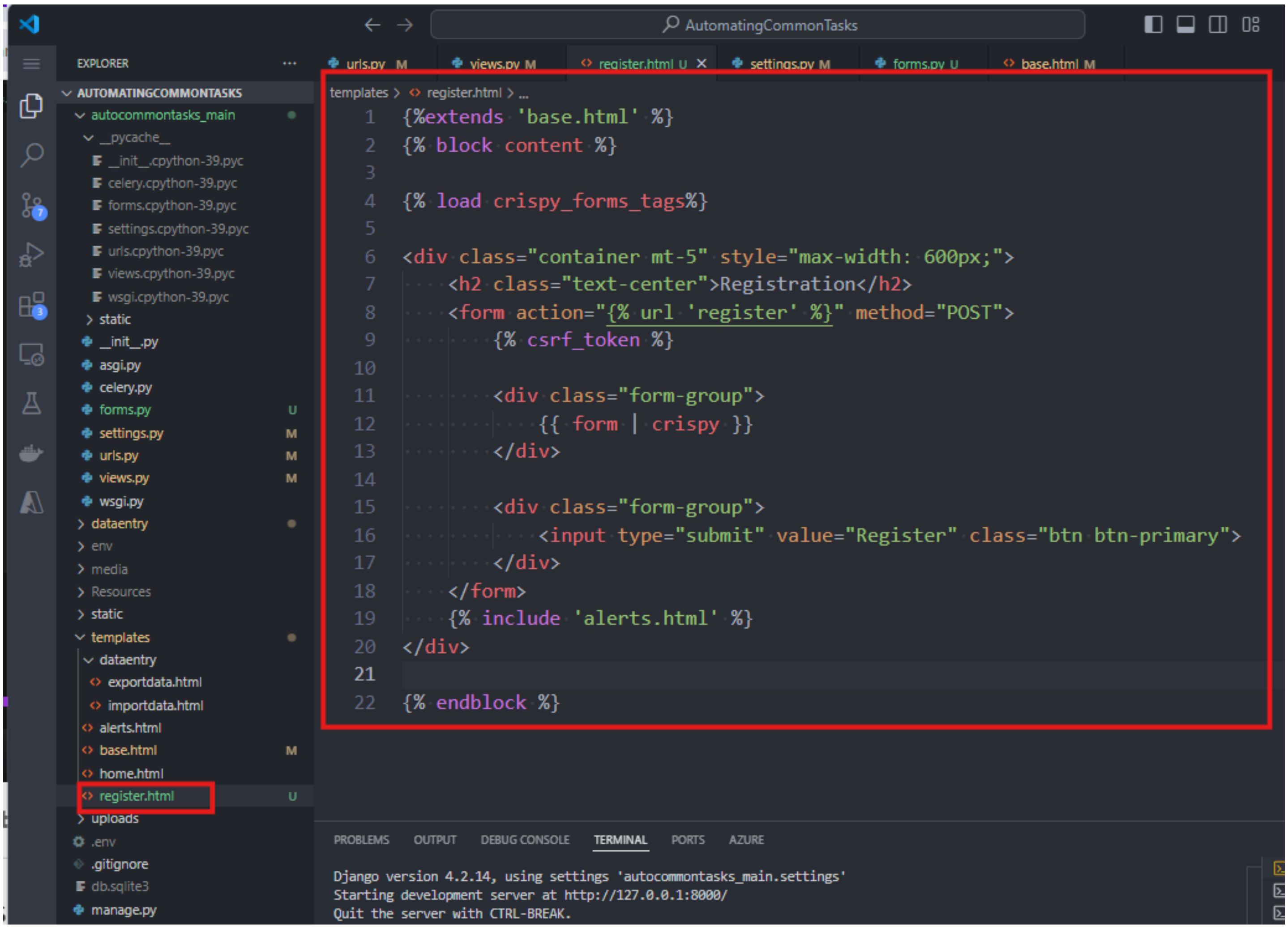
Password*

- 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*

```html
{%extends 'base.html' %}
{% block content %}

{% load crispy_forms_tags%}

<div class="container mt-5" style="max-width: 600px;">
    <h2 class="text-center">Registration</h2>
    <form action="{% url 'register' %}" method="POST">
        {% csrf_token %}

        <div class="form-group">
            {{ form | crispy }}
        </div>

        <div class="form-group">
            <input type="submit" value="Register" class="btn btn-primary">
        </div>



    </form>
</div>

{% endblock %}
```

17. To include our message alert for successful registration.

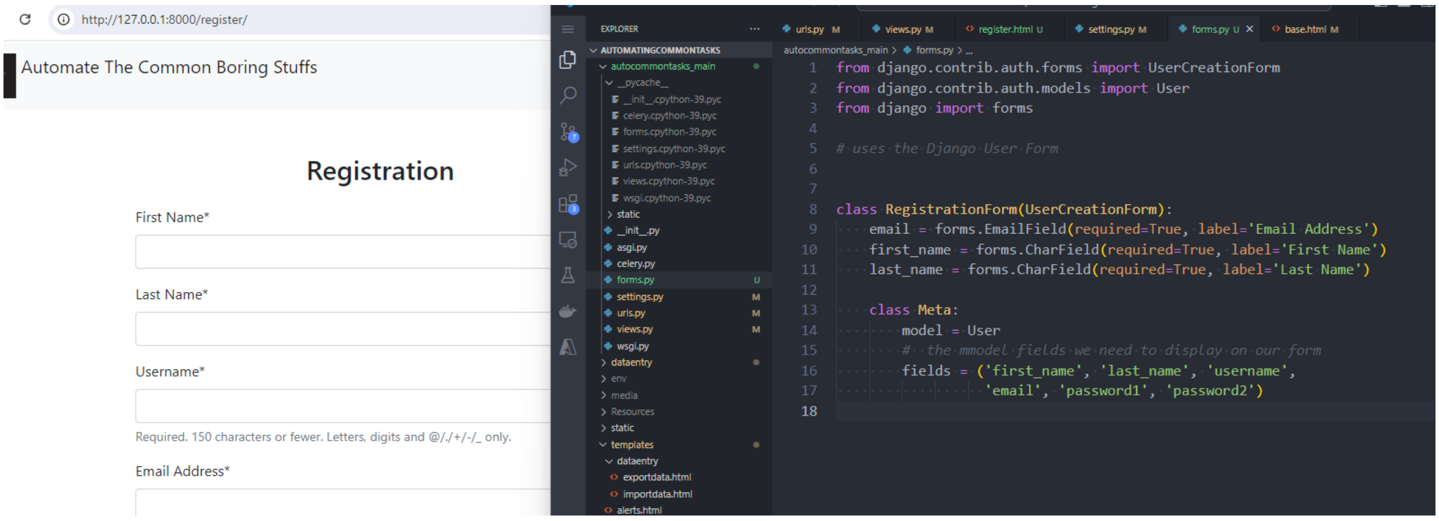


templates > register.html > ...

```
{%extends 'base.html' %}
{% block content %}

{% load crispy_forms_tags%}

<div class="container mt-5" style="max-width: 600px;">
    <h2 class="text-center">Registration</h2>
    <form action="{% url 'register' %}" method="POST">
        {% csrf_token %}

        <div class="form-group">
            {{ form | crispy }}
        </div>

        <div class="form-group">
            <input type="submit" value="Register" class="btn btn-primary">
        </div>
    </form>
    {% include 'alerts.html' %}
</div>

{% endblock %}
```

```
Django version 4.2.14, using settings 'autocommontasks_main.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

18. I updated the form to include the firstname and the lastname in the registration.



http://127.0.0.1:8000/register/

Automate The Common Boring Stuffs

Registration

First Name*

Last Name*

Username*

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

Email Address*

autocommontasks_main > forms.py > ...

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

# uses the Django User Form


class RegistrationForm(UserCreationForm):
    email = forms.EmailField(required=True, label='Email Address')
    first_name = forms.CharField(required=True, label='First Name')
    last_name = forms.CharField(required=True, label='Last Name')

    class Meta:
        model = User
        #  the mmodel fields we need to display on our form
        fields = ('first_name', 'last_name', 'username',
                  'email', 'password1', 'password2')
```

19.