

Topic: Registration Module: Part 14

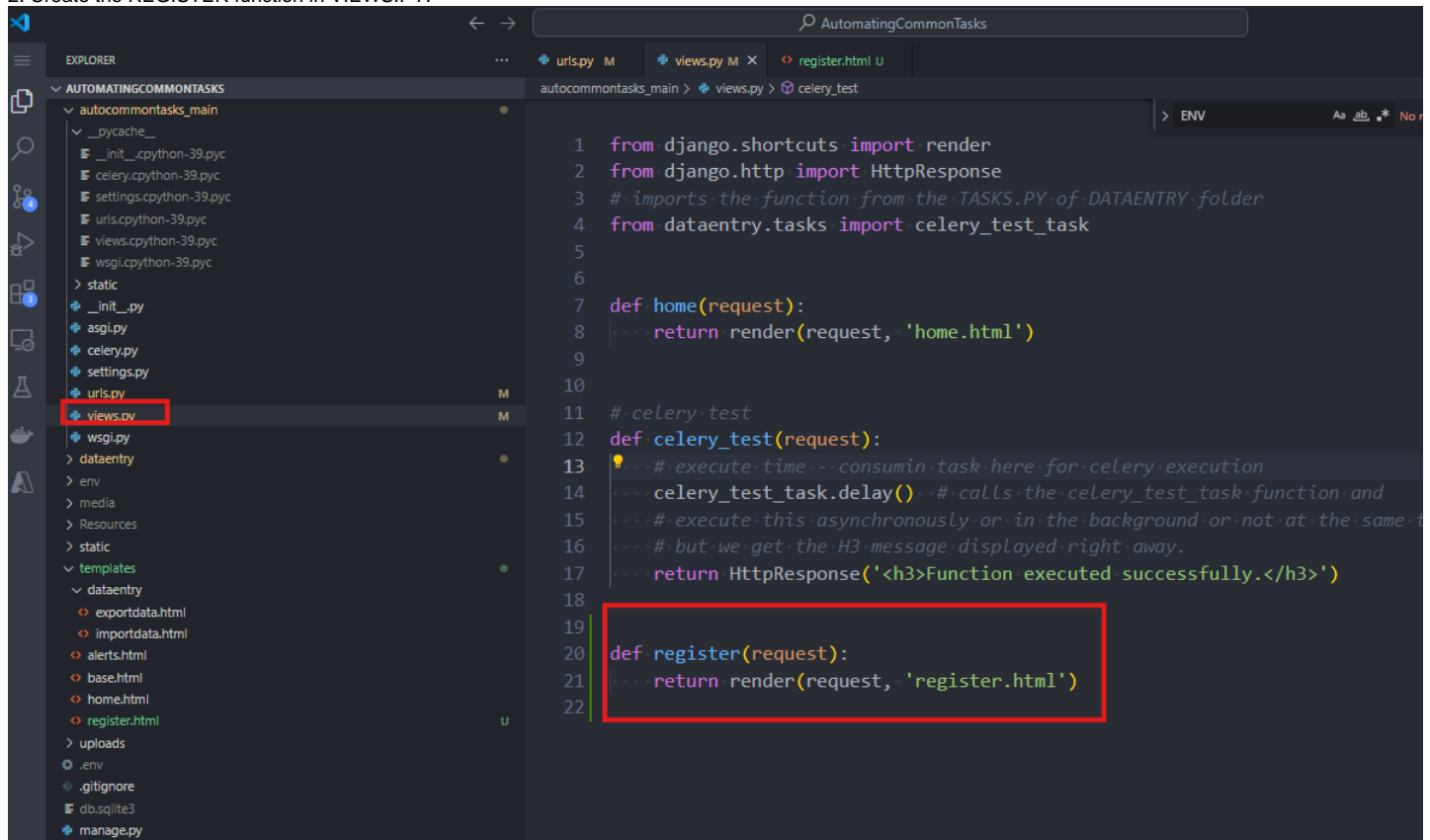
Speaker: Udemy Instructor Rathan Kumar / Notebook: Django: Automating Common Tasks



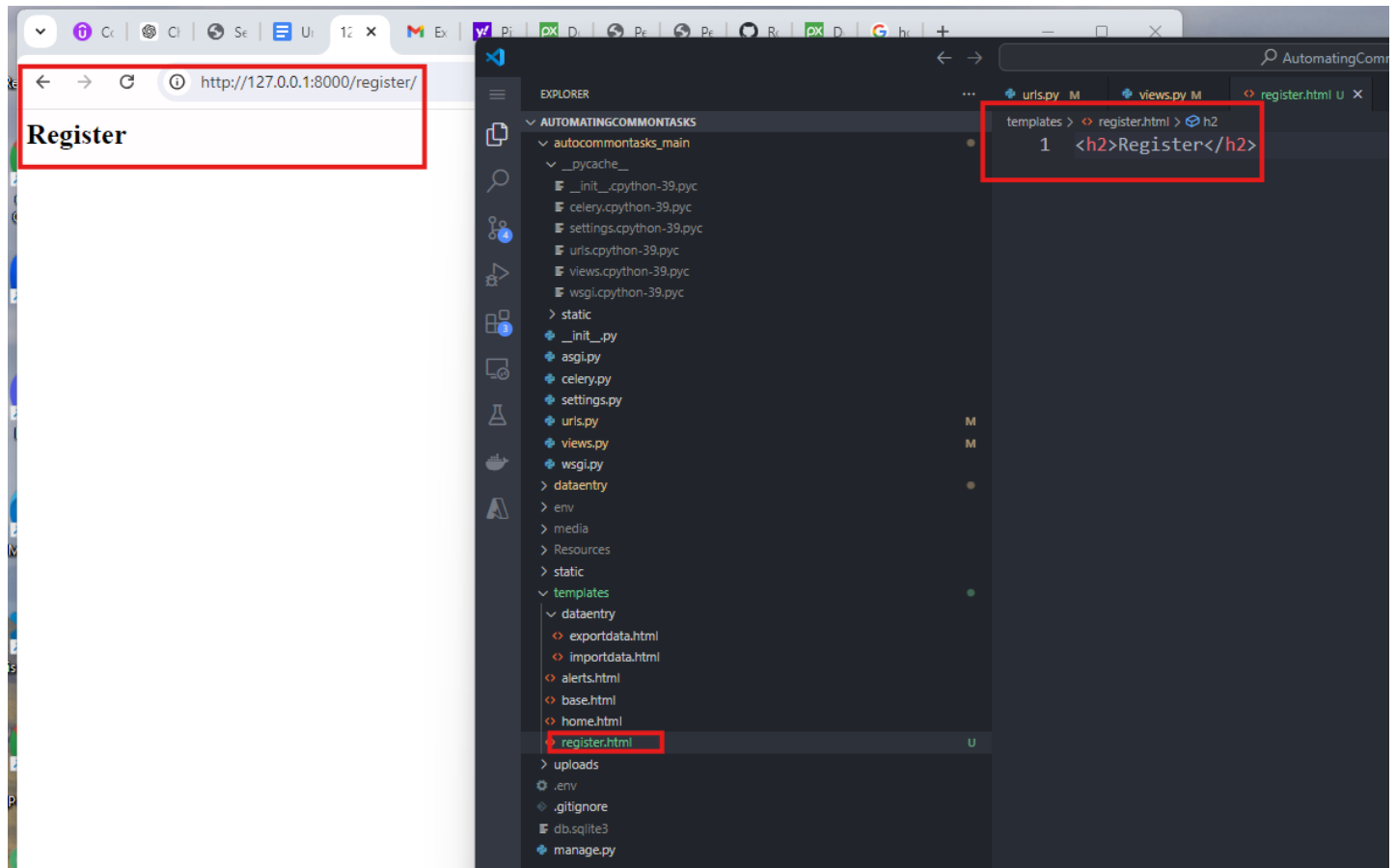
1. We need a URL pattern like `http://127.0.0.1:8000/register/`, so in the main project's `URLS.PY`.

```
8  ... 1. Add an import: from my_app import views
9  ... 2. Add a URL to urlpatterns: path('', views.home, name='home')
10 Class-based views
11 ... 1. Add an import: from other_app.views import Home
12 ... 2. Add a URL to urlpatterns: path('', Home.as_view(), name='home')
13 Including another URLconf
14 ... 1. Import the include() function: from django.urls import include, path
15 ... 2. Add a URL to urlpatterns: path('blog/', include('blog.urls'))
16 """
17 from django.contrib import admin
18 from django.urls import path, include
19 from . import views
20 from django.conf.urls.static import static
21 from django.conf import settings
22
23 urlpatterns = [
24     path('admin/', admin.site.urls),
25     path('', views.home, name='home'),
26     # Links to our dataentry app's URLS.PY
27     path('dataentry/', include('dataentry.urls')),
28     path('celery-test/', views.celery_test),
29     # registration and login
30     path('register/', views.register, name='register'),
31
32 ] + static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)
33
```

2. Create the REGISTER function in VIEWS.PY:



3. Create the REGISTER.HTML in TEMPLATES.



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

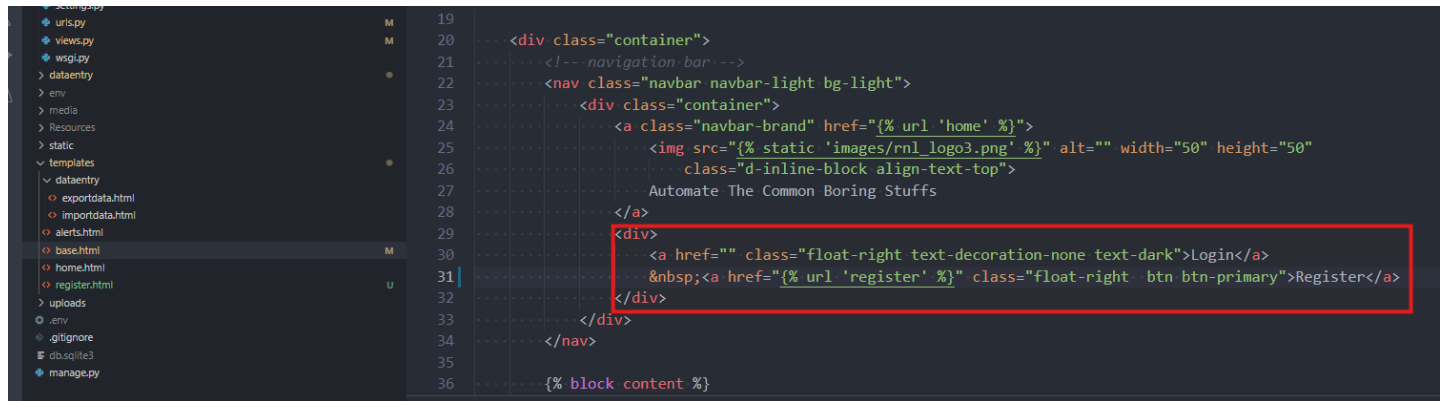
```
{%extends 'base.html' %}

{% block content %}

<we add our unique code here>
```

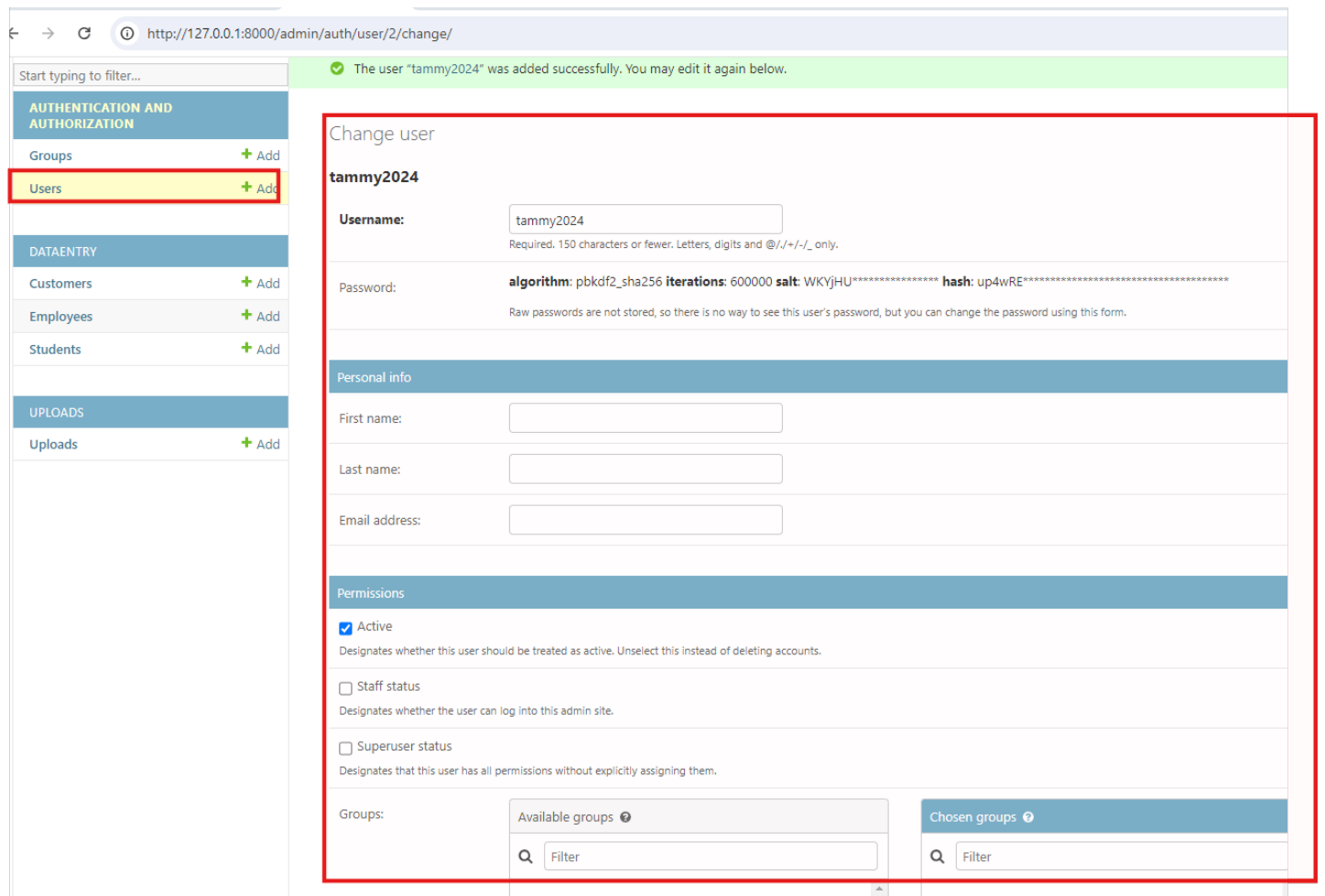
```
{% endblock %}
```

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



```
19 ... <div class="container">
20 ... <!-- navigation bar -->
21 ... <nav class="navbar navbar-light bg-light">
22 ... <div class="container">
23 ... <a class="navbar-brand" href="{% url 'home' %}">
24 ... 
26 ... Automate The Common Boring Stuffs
27 ... </a>
28 ... <div>
29 ... <a href="" class="float-right text-decoration-none text-dark">Login</a>
30 ... &nbsp;<a href="{% url 'register' %}" class="float-right btn btn-primary">Register</a>
31 ... </div>
32 ... </div>
33 ... </nav>
34 ... {% block content %}
35 ...
36 ...
```

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/

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

Start typing to filter...

AUTHENTICATION AND AUTHORIZATION

- Groups + Add
- Users + Add**

DATAENTRY

- Customers + Add
- Employees + Add
- Students + Add

UPLOADS

- Uploads + Add

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.

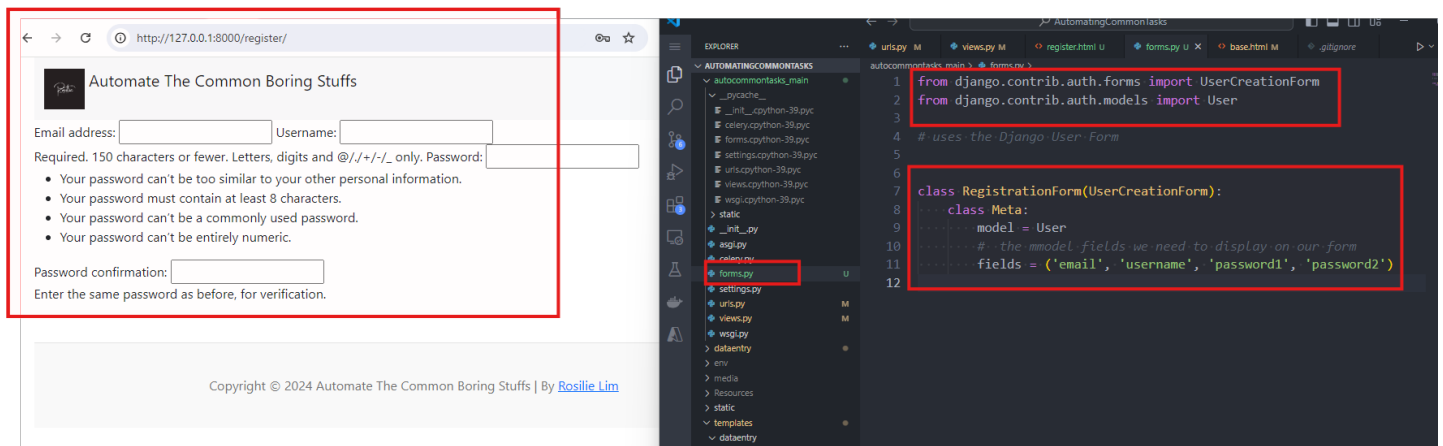
Groups:

Available groups **?**

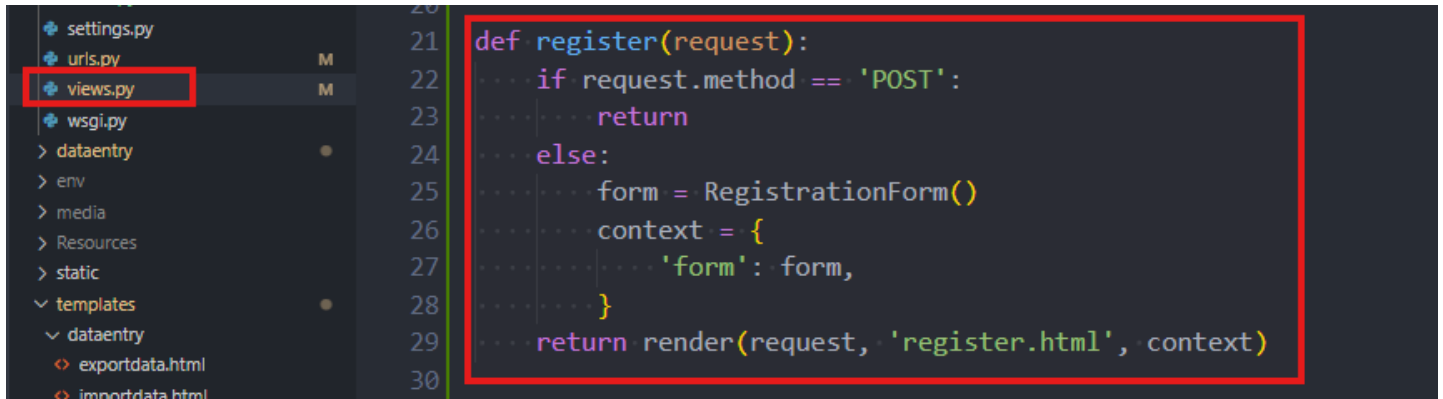
Chosen groups **?**

Filter

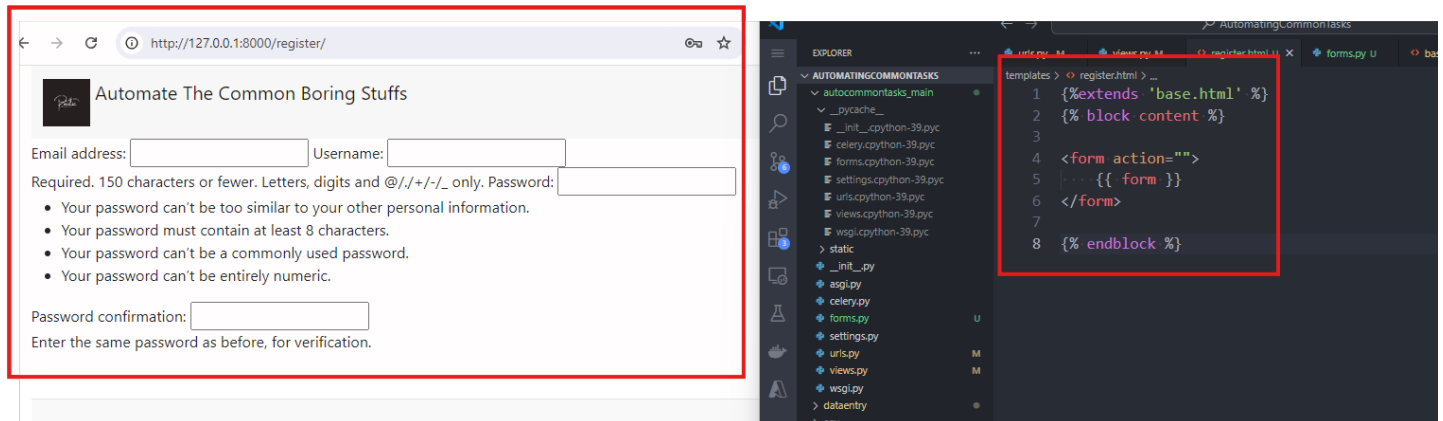
7. Create a new file, FORMS.PY, in your main project directory.



8. In our VIEWS.PY, we write:



9. In our REGISTER.HTML, we write



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 Stuff

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 Stuff | By [Rosilie Lim](#)

```
EXPLORER
AUTOMATINGCOMMONTASKS
  autocommontasks_main
    __pycache__
    _init_.cpython-39.pyc
    celery.cpython-39.pyc
    forms.cpython-39.pyc
    settings.cpython-39.pyc
    uris.cpython-39.pyc
    views.cpython-39.pyc
    wsgi.cpython-39.pyc
  static
  _init_.py
  asgi.py
  celery.py
  forms.py
  settings.py
  uris.py
  views.py
  wsgi.py
  dataentry
  env
  media
  Resources
  static
  templates
    dataentry
      exportdata.html
      importdata.html
      alerts.html
      base.html
      home.html
      register.html
    uploads
```

```
templates > register.html > ...
1 {% extends 'base.html' %}
2 {% block content %}
3
4 <form action="">
5   <{{ form.as_p }}>
6 </form>
7
8 {% 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-templatePacName` 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).

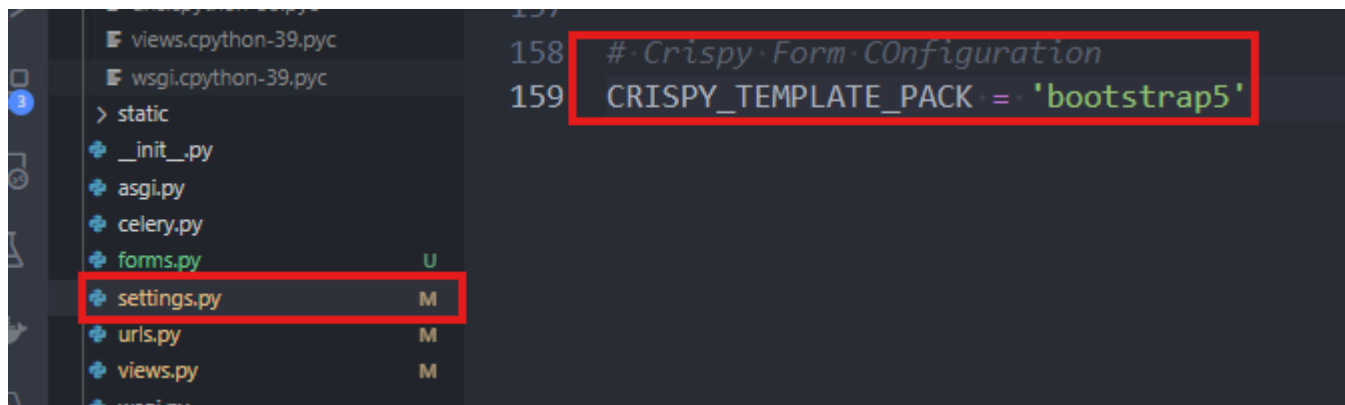
Install Django Crispy Form: `$ pip install django-crispy-forms`

In your `SETTINGS.PY`, register this in your `INSTALLED_APPS`

```

36
37 INSTALLED_APPS = [
38     'django.contrib.admin',
39     'django.contrib.auth',
40     'django.contrib.contenttypes',
41     'django.contrib.sessions',
42     'django.contrib.messages',
43     'django.contrib.staticfiles',
44     'dataentry',
45     'uploads',
46     'crispy_forms',
47 ]

```



```

158 # Crispy Form Configuration
159 CRISPY_TEMPLATE_PACK = 'bootstrap5'

```

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

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

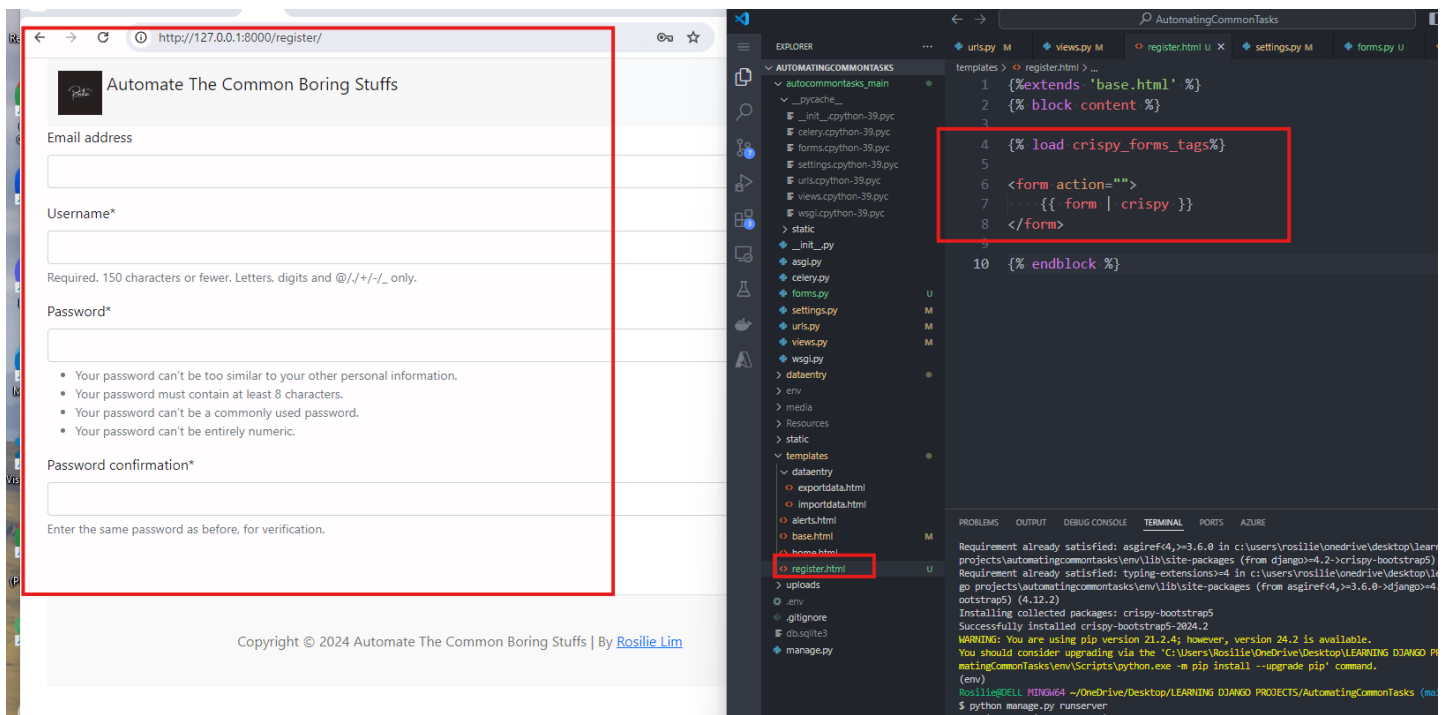
Register this in your INSTALLED_APPS:

```

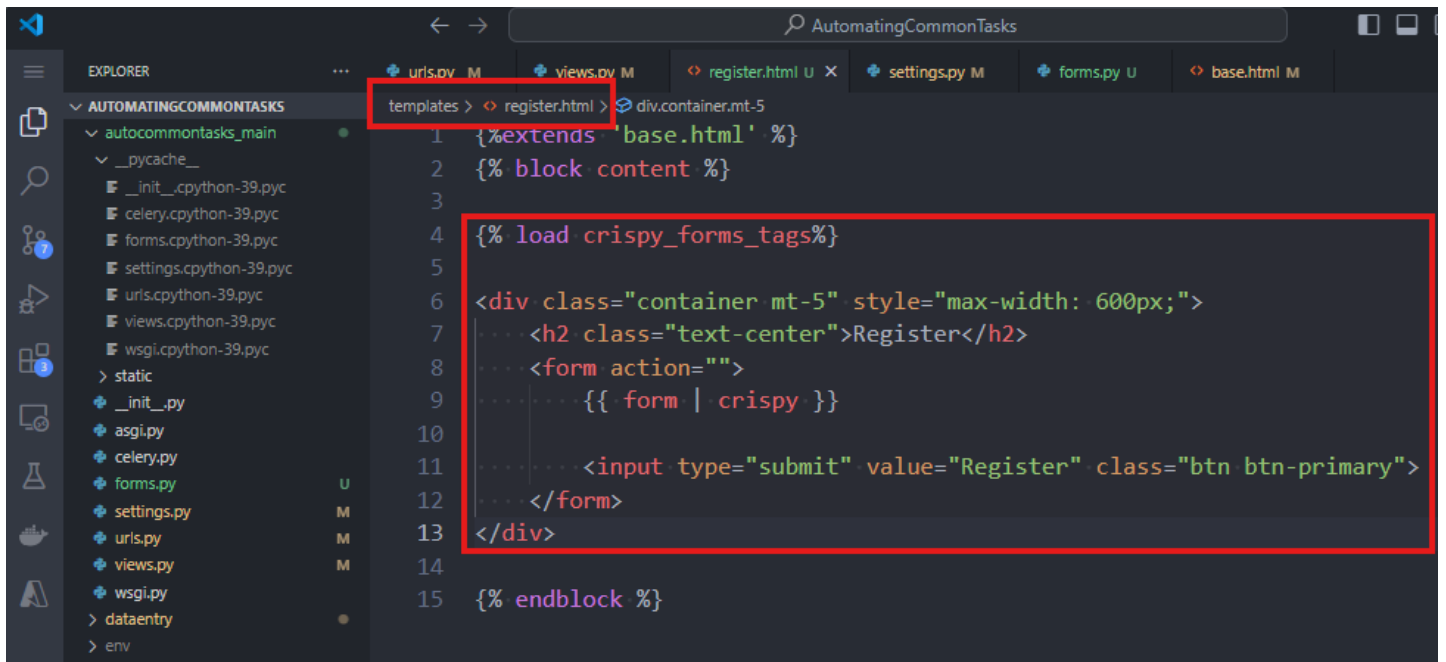
35 # Application definition
36
37 INSTALLED_APPS = [
38     'django.contrib.admin',
39     'django.contrib.auth',
40     'django.contrib.contenttypes',
41     'django.contrib.sessions',
42     'django.contrib.messages',
43     'django.contrib.staticfiles',
44     'dataentry',
45     'uploads',
46     'crispy_forms',
47     'crispy_bootstrap5',
48 ]
49

```

13. Update our REGISTER.HTML and reload:



14. Further updating our REGISTER.HTML.



→ ↻ 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*

EXPLORER

AUTOMATINGCOMMONTASKS

autocommontasks_main

__init__.py

celery.py

forms.py

settings.py

uris.py

views.py

wsgi.py

dataentry

env

autocommontasks_main > forms.py

1 from django.contrib.auth.forms import UserCreationForm

2 from django.contrib.auth.models import User

3 from django import forms

4

5 # uses the Django User Form

6

7

8 class RegistrationForm(UserCreationForm):

9 email = forms.EmailField(required=True, label='Email Address')

10

11 class Meta:

12 model = User

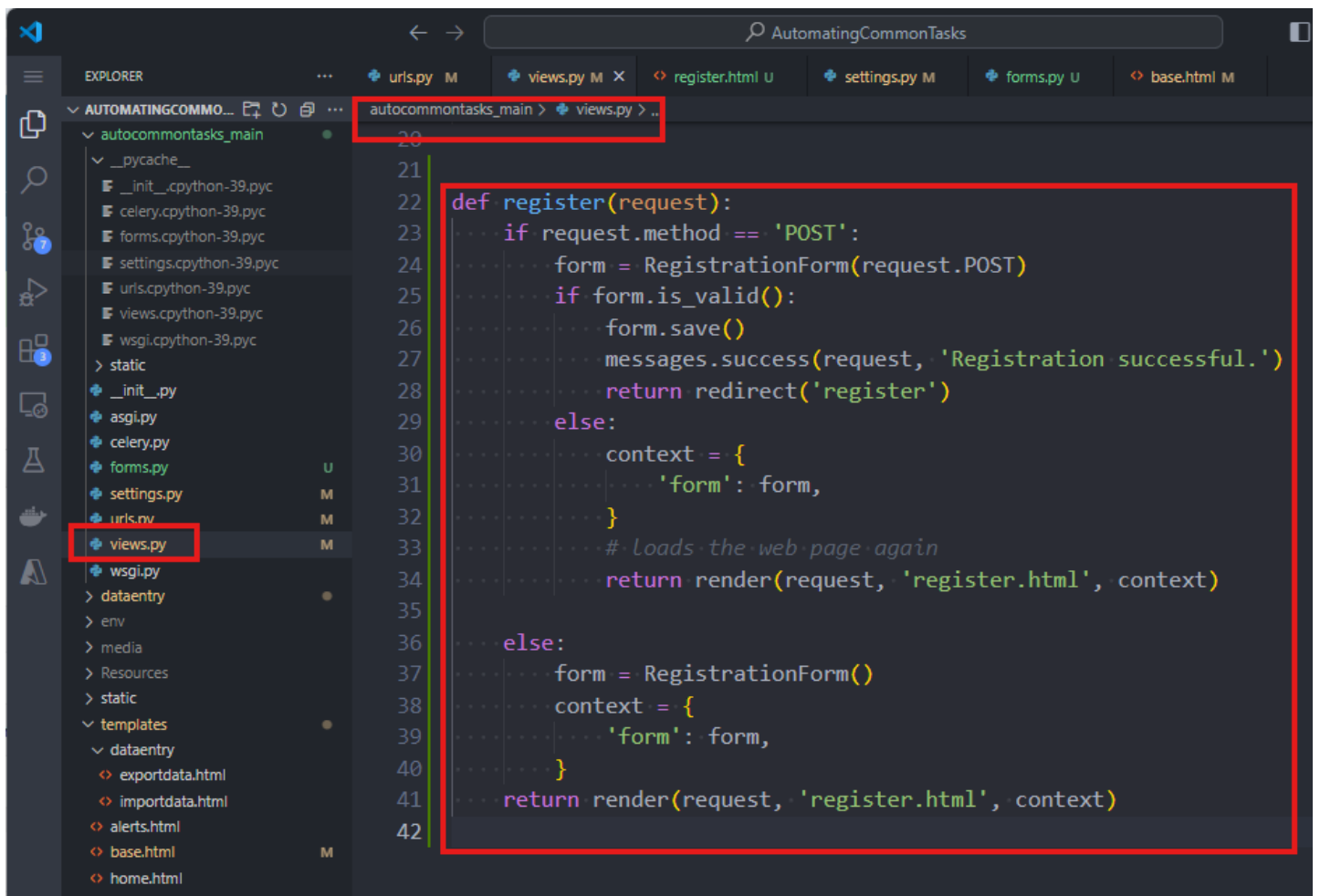
13 # the model fields we need to display on our form

14 fields = ('email', 'username', 'password1', 'password2')

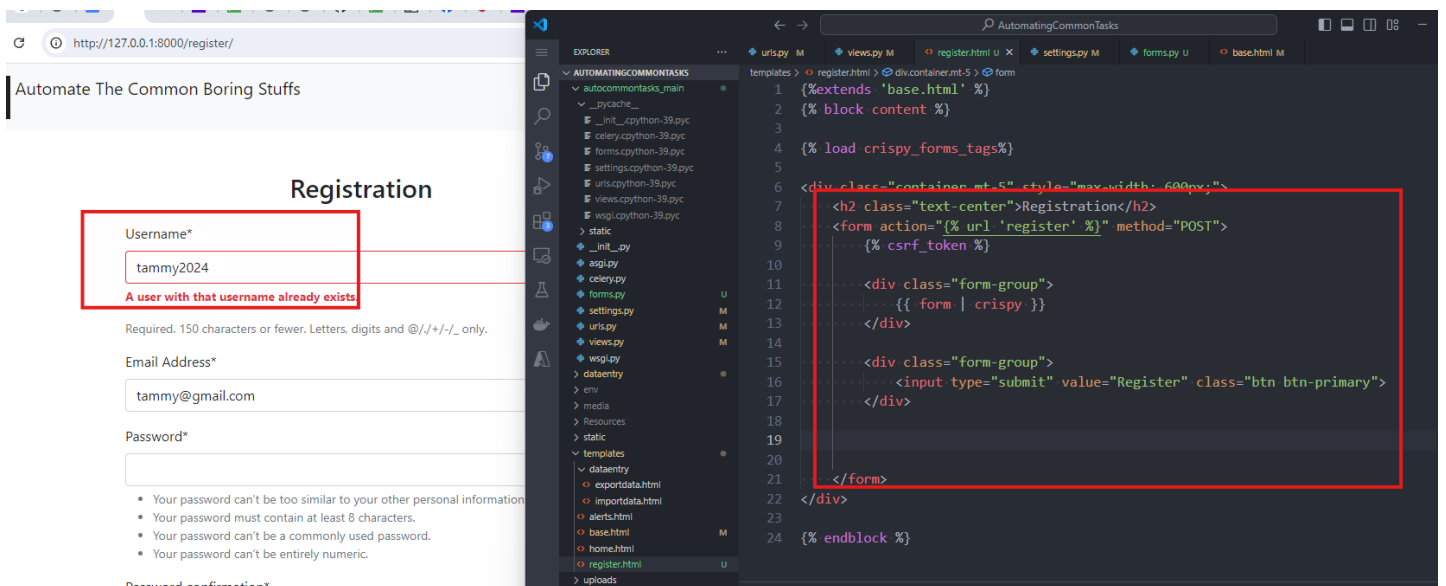
15

16. Our VIEWS.PY as:

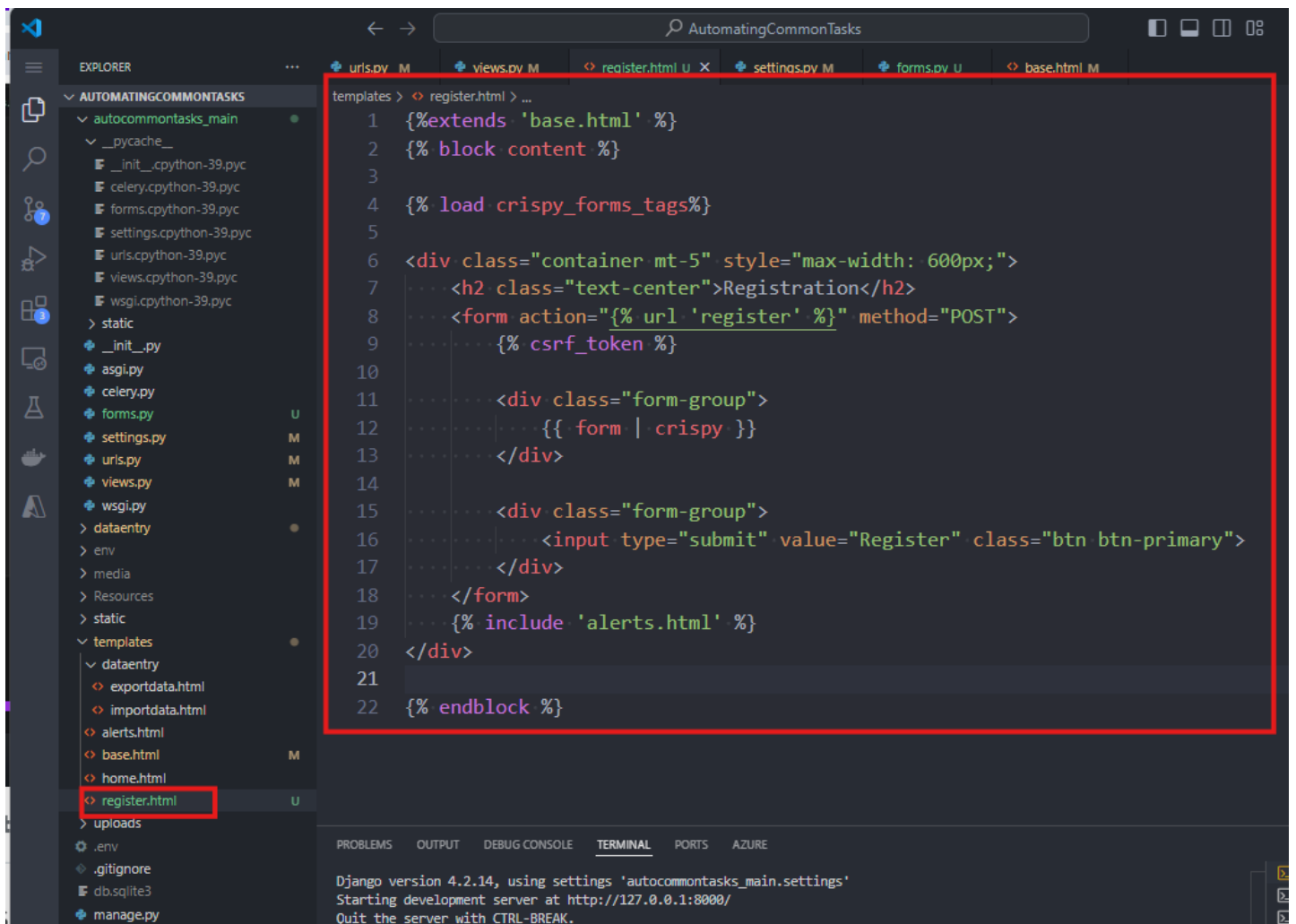
```
autocommontasks_main > views.py > register
1 from django.shortcuts import render, redirect
2 from django.http import HttpResponse
3 # imports the function from the TASKS.PY of DATAENTRY
4 from dataentry.tasks import celery_test_task
5 from .forms import RegistrationForm
6 from django.contrib import messages
```



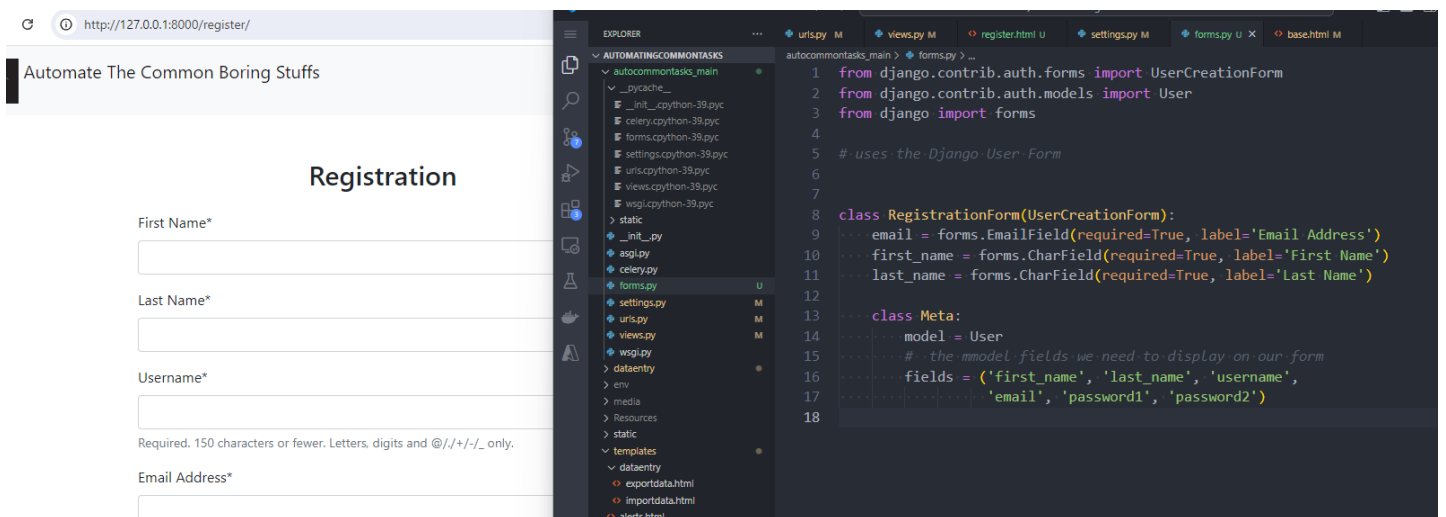
17. Testing the registration form with an existing user:



17. To include our message alert for successful registration.



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



19.