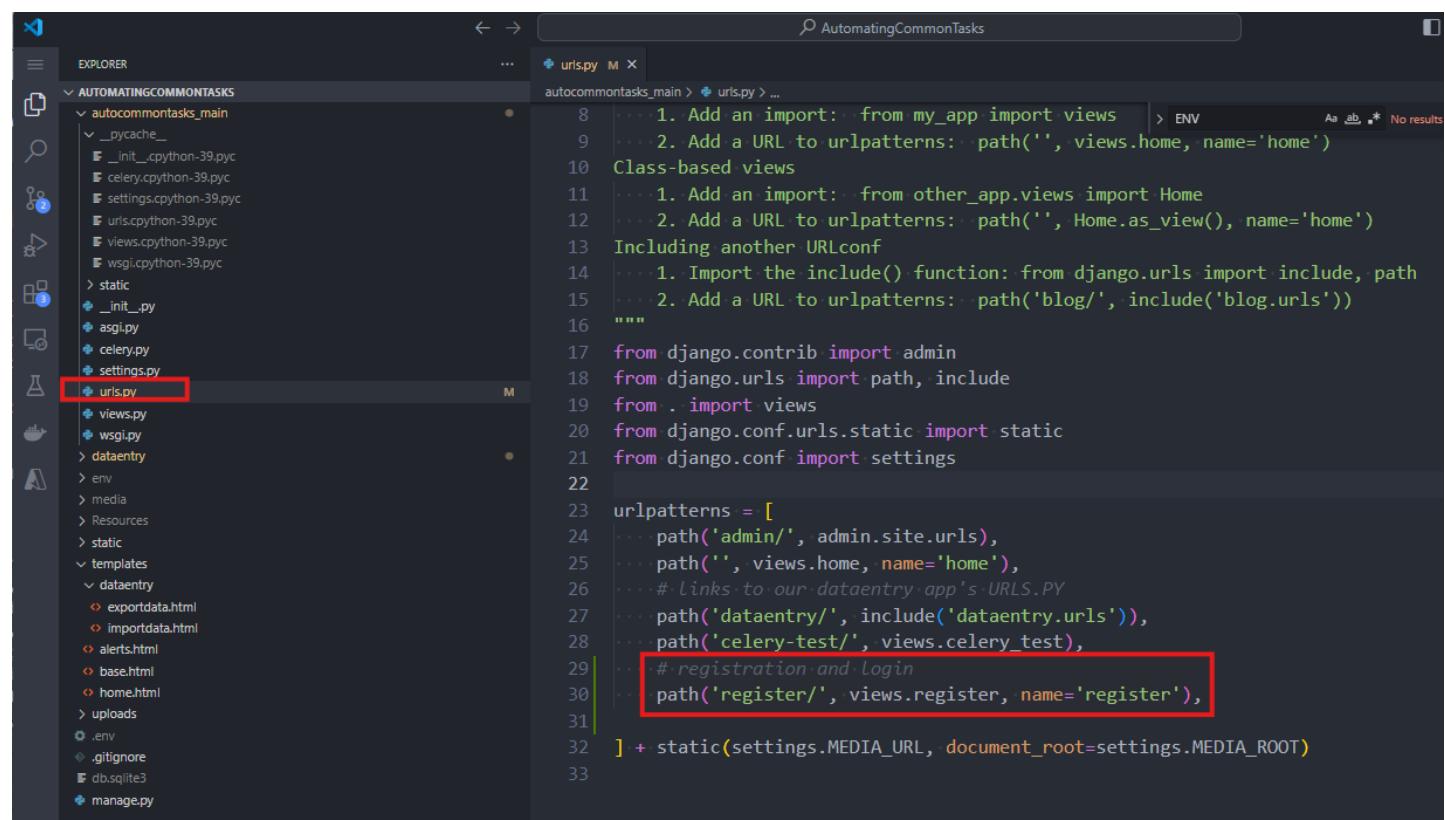


## Topic: Registration Module: Part 14

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`.

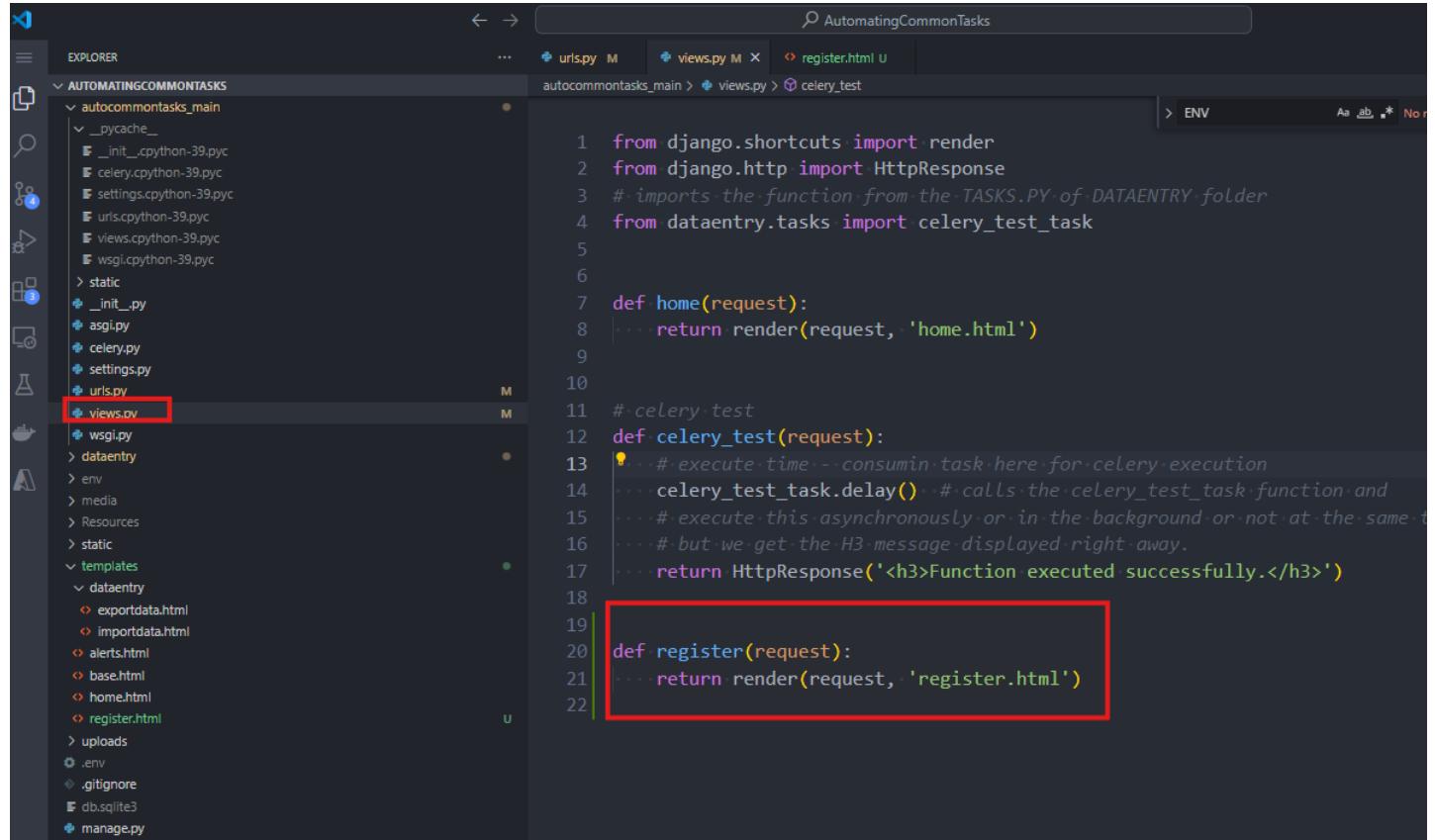


```
from django.contrib import admin
from django.urls import path, include
from . import views
from django.conf.urls.static import static
from django.conf import settings

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'),
]
```

The line `path('register/', views.register, name='register'),` is highlighted with a red box.

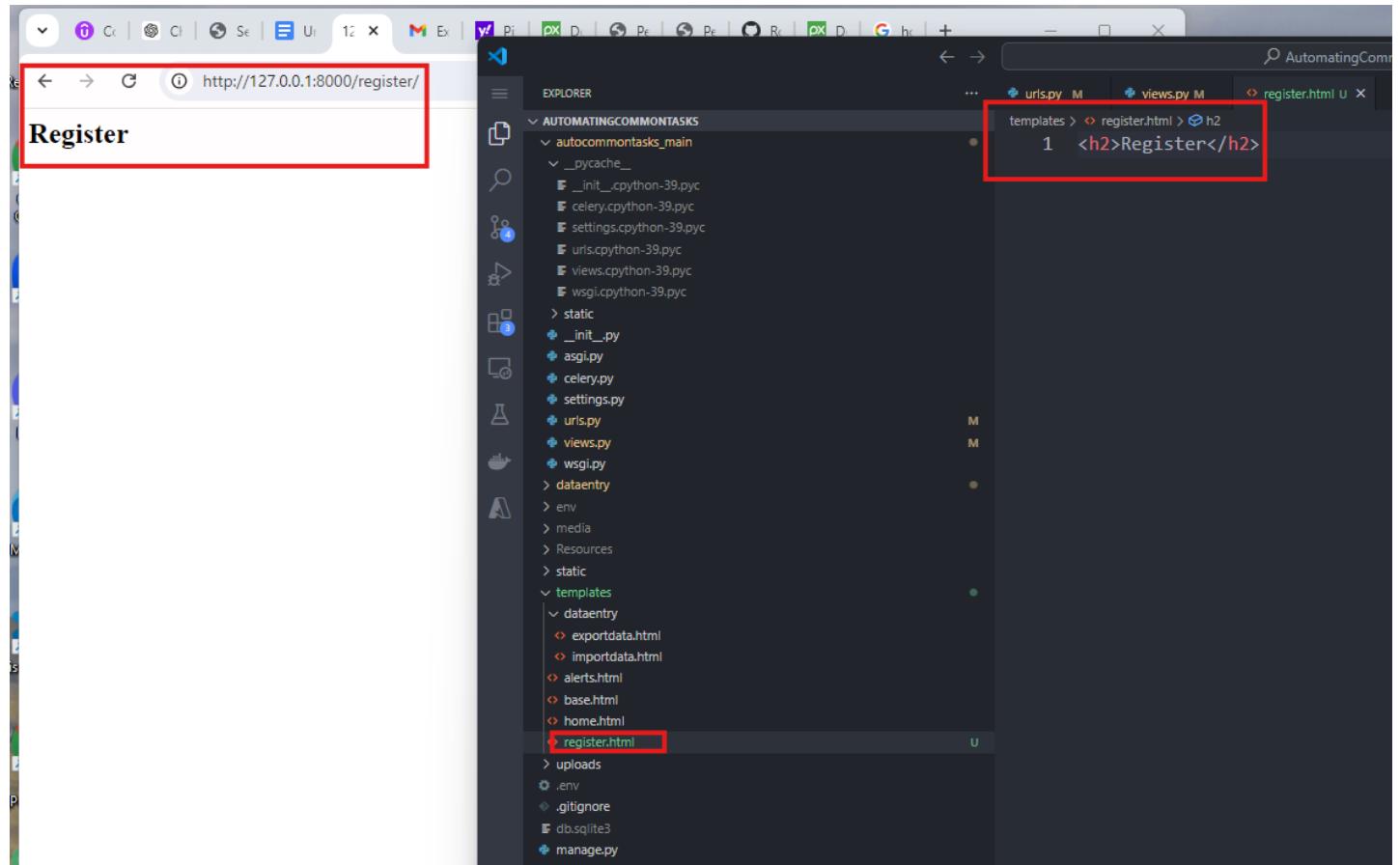
2. Create the REGISTER function in VIEWS.PY:



VS Code interface showing the file structure of the project. The 'EXPLORER' sidebar shows the project tree with 'AUTOMATINGCOMMONTASKS' as the root. Under 'AUTOMATINGCOMMONTASKS', there are subfolders like 'autocommontasks\_main', 'static', 'templates', and 'dataentry'. The 'templates' folder contains files like 'exportdata.html', 'importdata.html', 'alerts.html', 'base.html', 'home.html', and 'register.html'. The 'dataentry' folder contains 'views.py'. The 'views.py' file is open in the editor, showing Python code. A red box highlights the following code block:

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

3. Create the REGISTER.HTML in TEMPLATES.



The browser window shows the URL <http://127.0.0.1:8000/register/> and the page content 'Register'. The VS Code interface shows the 'EXPLORER' sidebar with the 'templates' folder open. Inside 'templates', there is a 'register.html' file. A red box highlights the content of this file:

```
1 <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 %}
```

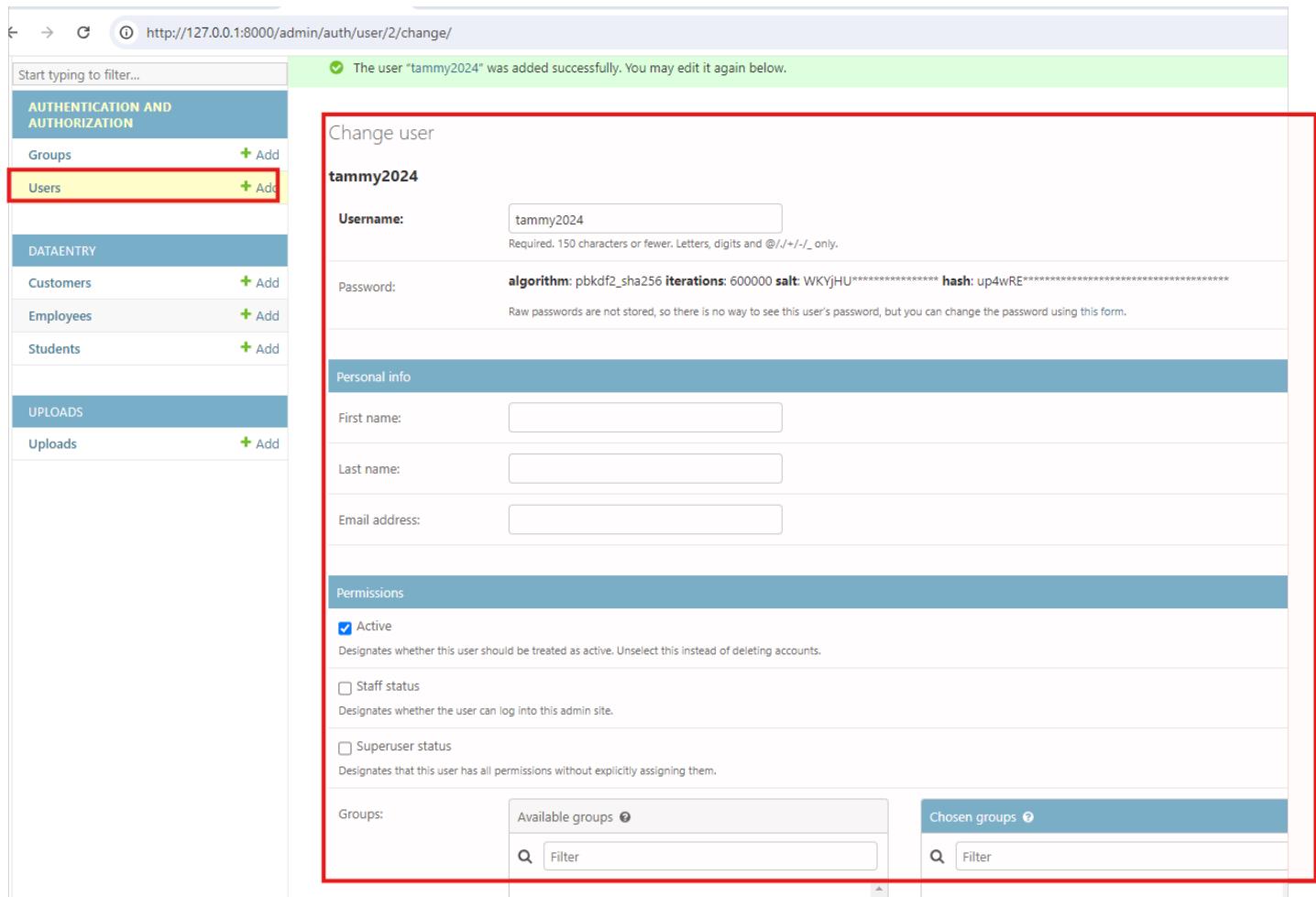
##### 5. We update our BASE.HTML to link our webpages to REGISTER.HTML

Login</a> &ampnbsp<a href="{% url 'register' %}" class="float-right btn btn-primary">Register</a>." data-bbox="45 81 952 260"/>

```
  settings.py
  urls.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
  .env
  .gitignore
  db.sqlite3
  manage.py
```

```
19
20      <div class="container">
21          <!-- navigation-bar -->
22          <nav class="navbar navbar-light bg-light">
23              <div class="container">
24                  <a class="navbar-brand" href="{% url 'home' %}">
25                      
27                      Automate The Common Boring Stuffs
28                  </a>
29              </div>
30          </nav>
31      </div>
32
33  </div>
34
35
36  {% 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.



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

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

The browser window shows the registration form with fields for Email address, Username, Password, and Password confirmation. The code editor shows the `forms.py` file with the following code:

```

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

8. In our VIEWS.PY, we write:

The code editor shows the `views.py` file with the following code:

```

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

The browser window shows the registration form. The code editor shows the `register.html` file with the following code:

```

{% extends 'base.html' %}

{% block content %}

<form action="">

{{ form.as_p }}

</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 %}
    
```

The screenshot shows a web browser window with a registration form and a code editor window side-by-side.

**Web Browser (Left):**

- Address bar: `http://127.0.0.1:8000/register/`
- Page title: Automate The Common Boring Stuffs
- Form fields:
  - Email address:
  - Username:  Required. 150 characters or fewer. Letters, digits and @/./+/-/\_ only.
  - Password:
  - >Password confirmation:  Enter the same password as before, for verification.
- Validation messages:
  - 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.
- Page footer: Copyright © 2024 Automate The Common Boring Stuffs | By [Rosolie Lim](#)

**Code Editor (Right):**

- File: `register.html`
- Content:

```
1  {%extends 'base.html' %}\n2  {% block content %}\n3\n4  <form action=""\n5  | ... {{ form.as_p }}\n6  </form>\n7\n8  {% endblock %}
```
- File Explorer:
  - `__pycache__`
  - `autocommentasks_main`
  - `static`
  - `templates`
  - `dataentry`
  - `base.html`
  - `register.html` (highlighted with a red box)

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:

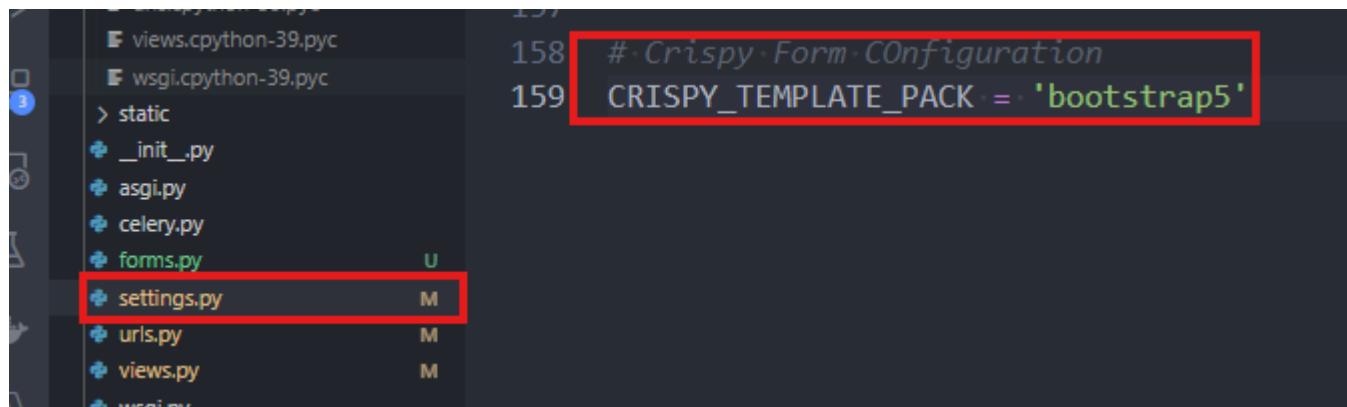
```
CRISPY_TEMPLATE_PACK = 'uni_form'
```

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`

```
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 ]
```



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 ]
```

13. Update our REGISTER.HTML and reload:

The screenshot shows a browser window on the left displaying a registration form for 'Automate The Common Boring Stuffs'. The form includes fields for Email address, Username\*, Password\*, and Password confirmation\*. Below the password fields is a note: 'Required. 150 characters or fewer. Letters, digits and @/./+/-/\_ only.' and a password strength checker. The code editor on the right shows the 'register.html' template file. The code is as follows:

```

1  {%extends 'base.html' %} 
2  {% block content %} 
3 
4  {% load crispy_forms_tags%} 
5 
6  <form action=""> 
7  ... {{ form | crispy }} 
8  </form> 
9 
10  {% endblock %} 

```

The 'register.html' file is highlighted in the Explorer sidebar of the code editor.

#### 14. Further updating our REGISTER.HTML.

The screenshot shows a code editor displaying the 'register.html' template. The code is as follows:

```

1  {%extends 'base.html' %} 
2  {% block content %} 
3 
4  {% load crispy_forms_tags%} 
5 
6  <div class="container mt-5" style="max-width: 600px;"> 
7  ... <h2 class="text-center">Register</h2> 
8  ... <form action=""> 
9  ... ... {{ form | crispy }} 
10 ... ... <input type="submit" value="Register" class="btn btn-primary"> 
11 ... ... </form> 
12  ... </div> 
13 
14  {% endblock %} 

```

The entire structure of the form (div, h2, form, input) is highlighted with a red box.

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 [Rosolie 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\*

Code in 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

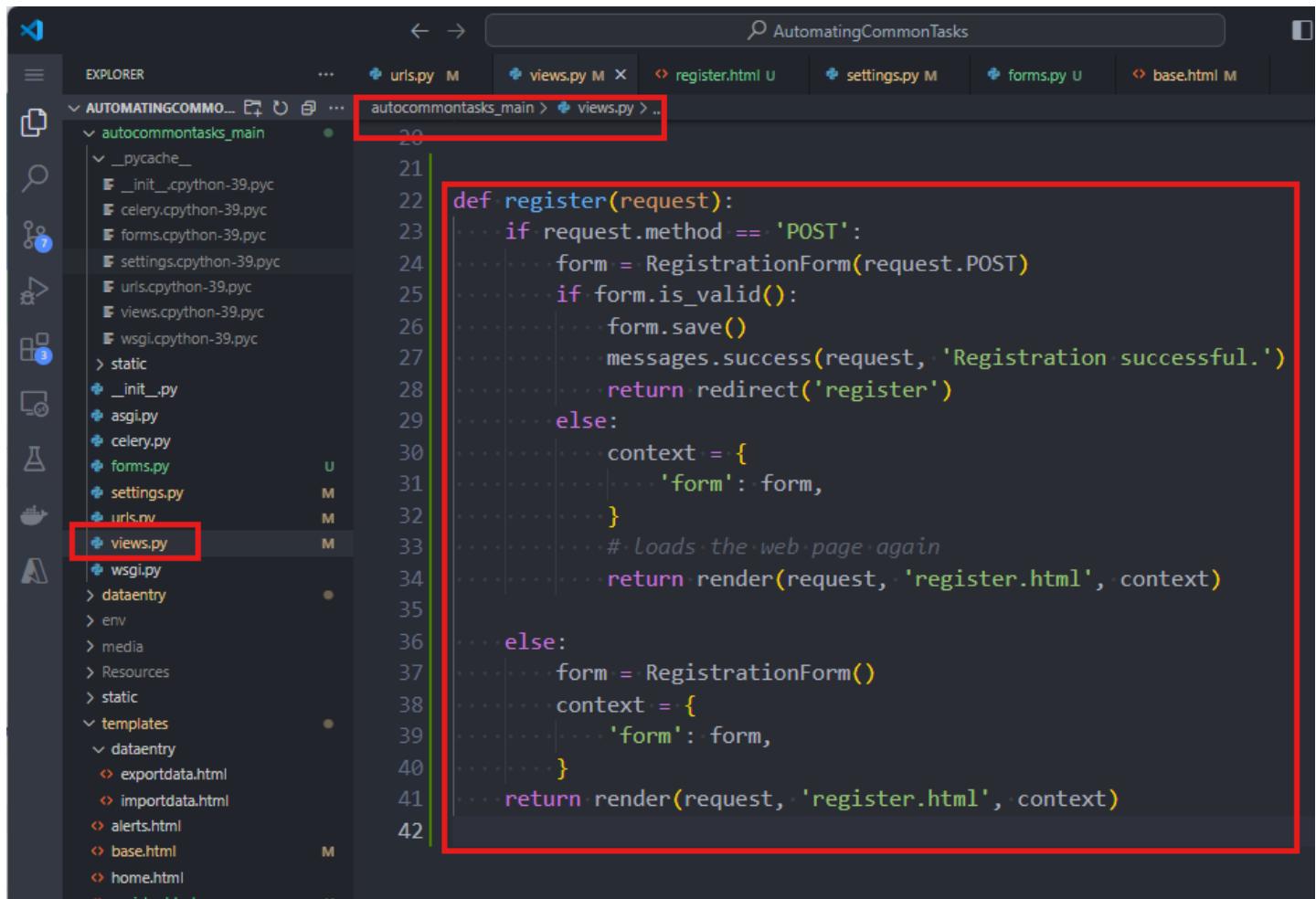
```

16. Our VIEWS.PY as:

```

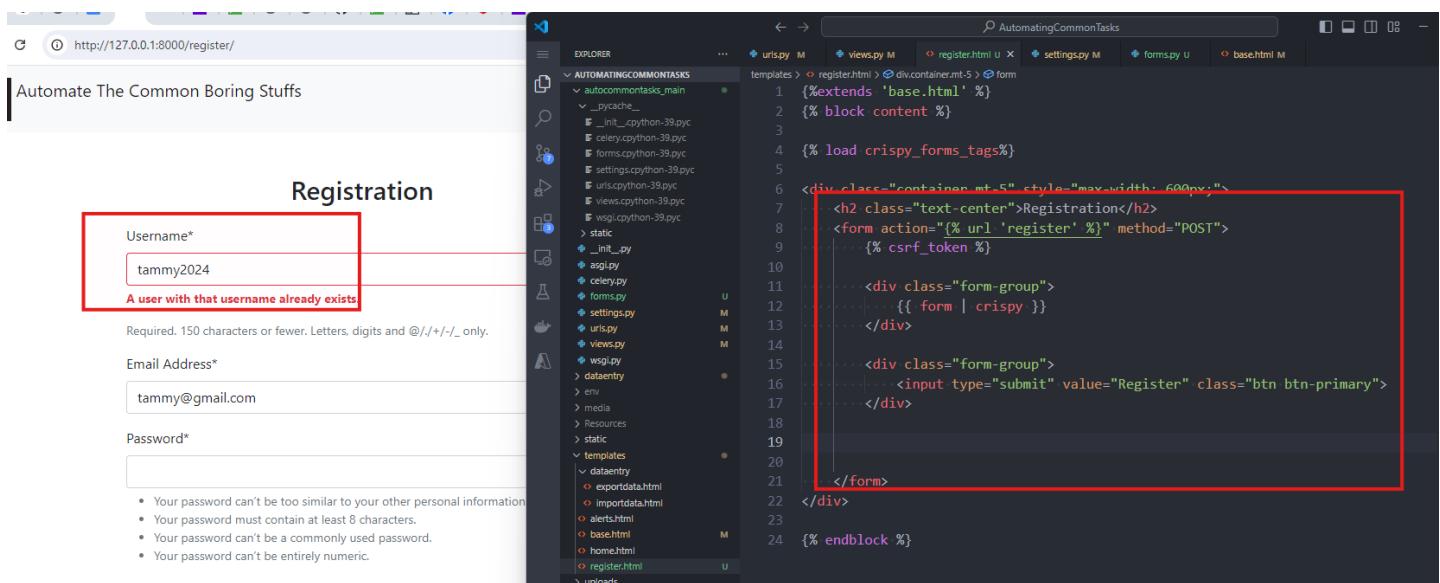
autocommontasks_main > views.py > register
1  from django.shortcuts import render, redirect
2  from django.http import HttpResponseRedirect
3  # imports the function from the TASKS.PY of DATA
4  from dataentry.tasks import celery_test_task
5  from .forms import RegistrationForm
6  from django.contrib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

```



```
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:



Automate The Common Boring Stuffs

## Registration

Username\*  
tammy2024  
A user with that username already exists

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

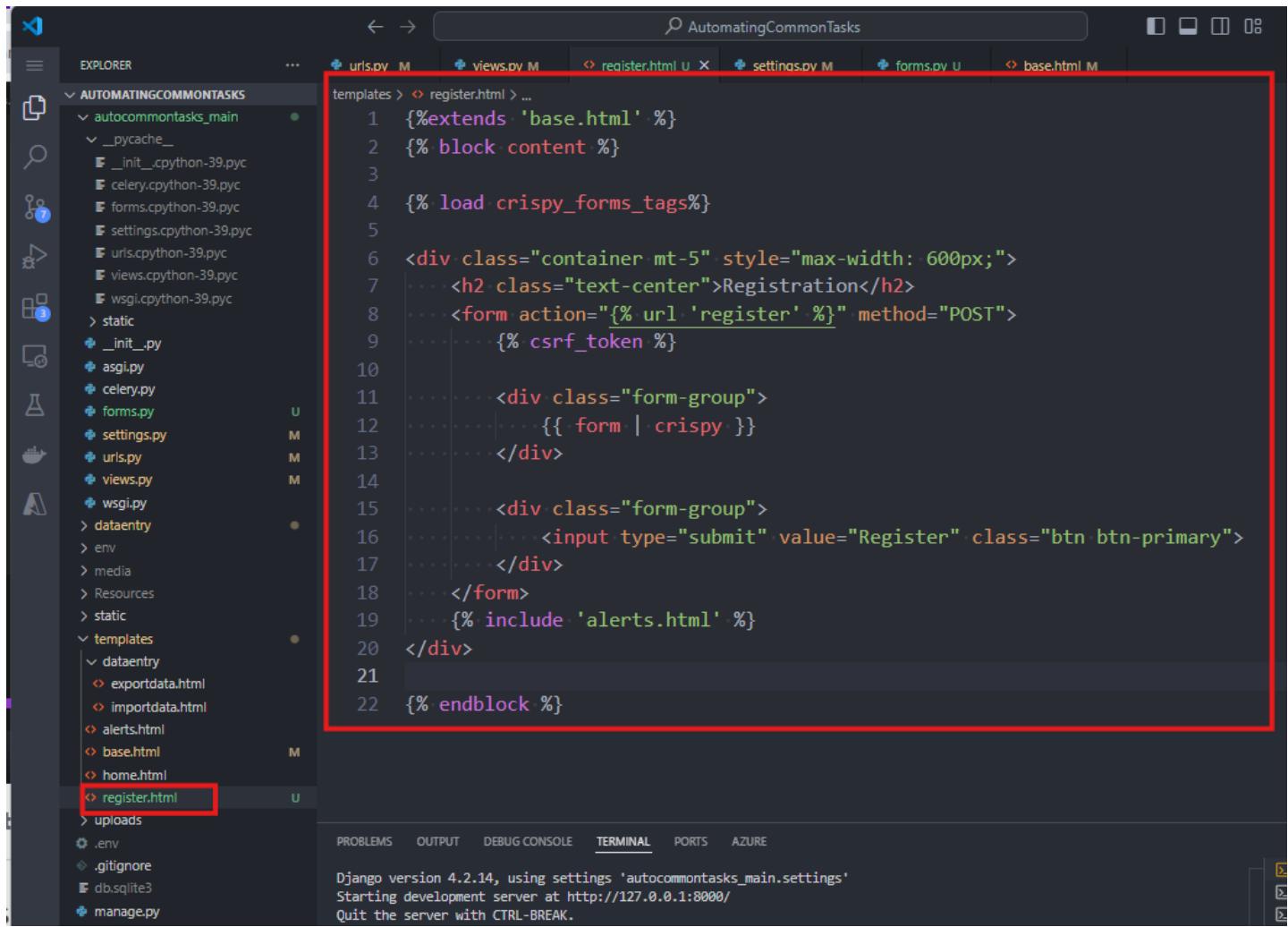
Email Address\*  
tammy@gmail.com

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\*

```
{% 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>  
{% endblock %}
```

17. To include our message alert for successful registration.

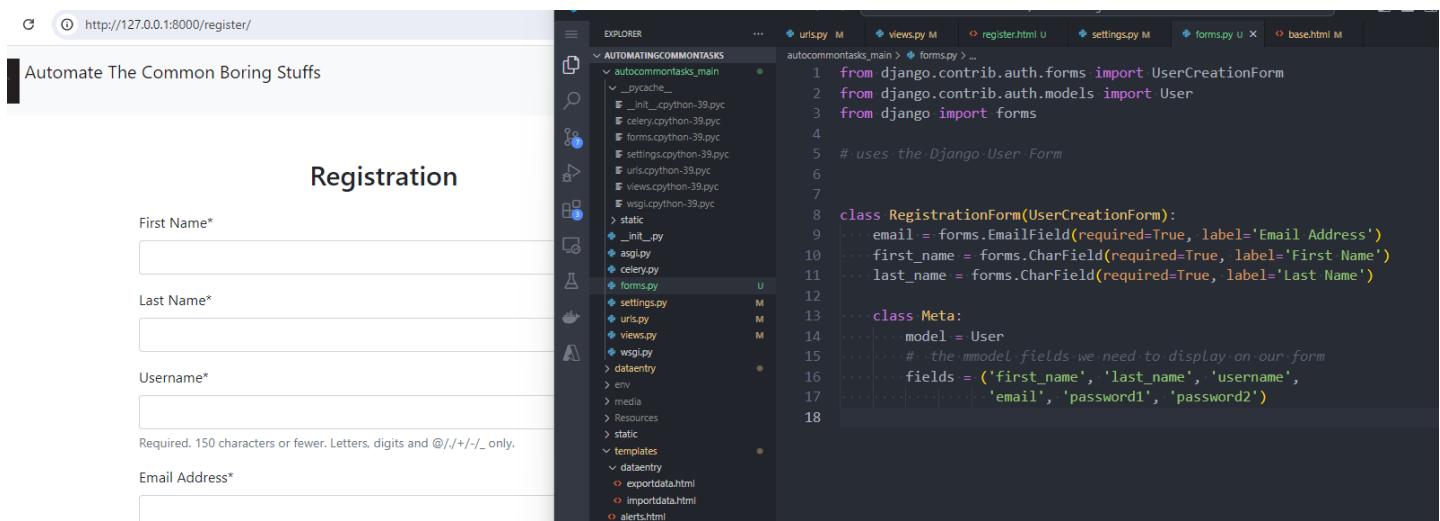


```

1  {% extends 'base.html' %} 
2  {% block content %} 
3 
4  {% load crispy_forms_tags%} 
5 
6  <div class="container mt-5" style="max-width: 600px;"> 
7      <h2 class="text-center">Registration</h2> 
8      <form action="{% url 'register' %}" method="POST"> 
9          {% csrf_token %} 
10 
11         <div class="form-group"> 
12             {{ form | crispy }} 
13         </div> 
14 
15         <div class="form-group"> 
16             <input type="submit" value="Register" class="btn btn-primary" /> 
17         </div> 
18     </form> 
19     {% include 'alerts.html' %} 
20 </div> 
21 
22 {% endblock %} 

```

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



```

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     first_name = forms.CharField(required=True, label='First Name') 
11     last_name = forms.CharField(required=True, label='Last Name') 
12 
13     class Meta: 
14         model = User 
15         # the mmodel fields we need to display on our form 
16         fields = ('first_name', 'last_name', 'username', 
17                   'email', 'password1', 'password2') 

```

19.