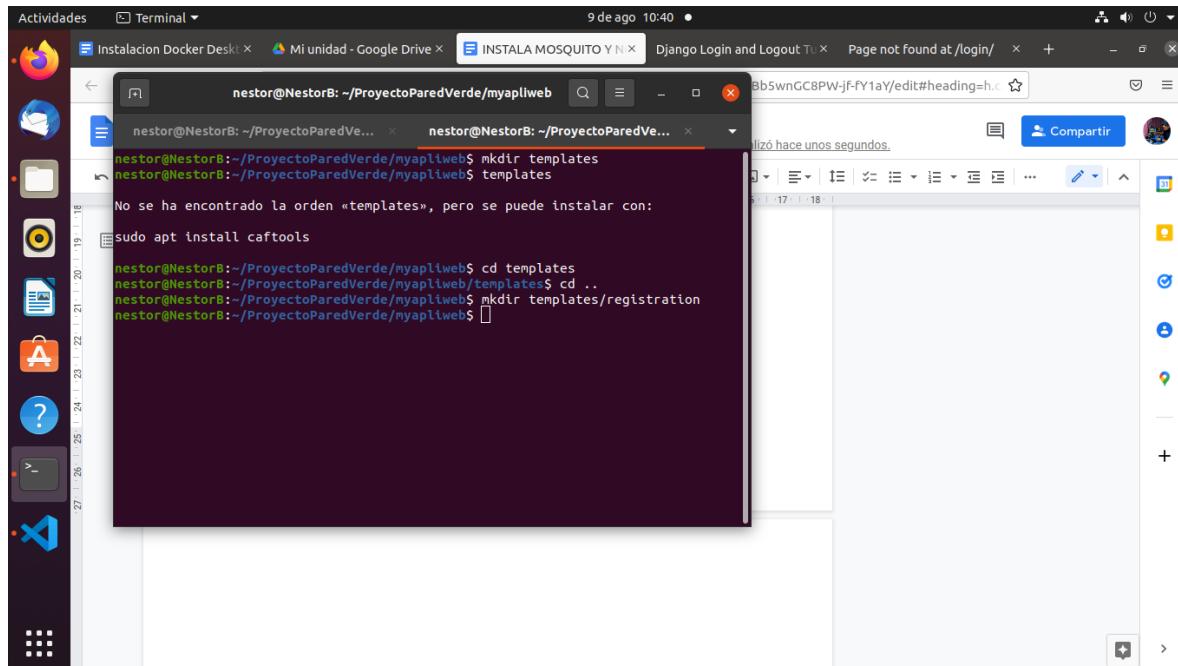


Login Page

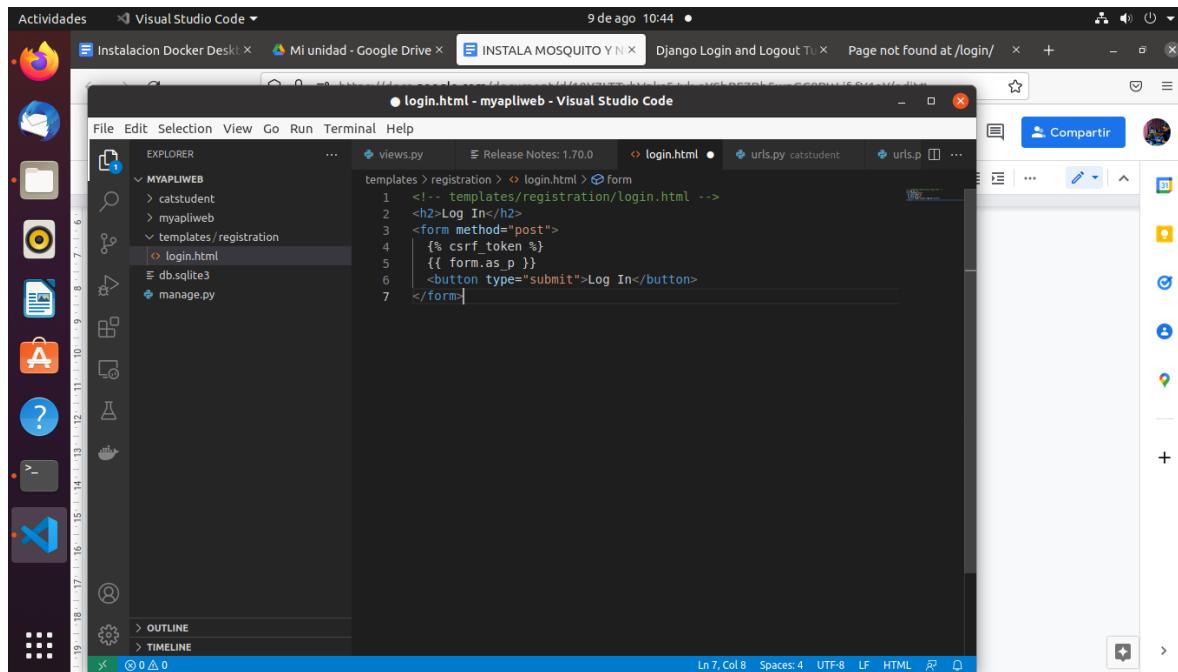
vamos poniendo nuestra pagina de logeo

mkdir templates/registration



después abrimos visual studio y creamos dentro de la carpeta un documento llamado login.html para agregar lo siguiente.

```
<!-- templates/registration/login.html -->
<h2>Log In</h2>
<form method="post">
    {% csrf_token %}
    {{ form.as_p }}
    <button type="submit">Log In</button>
</form>
```

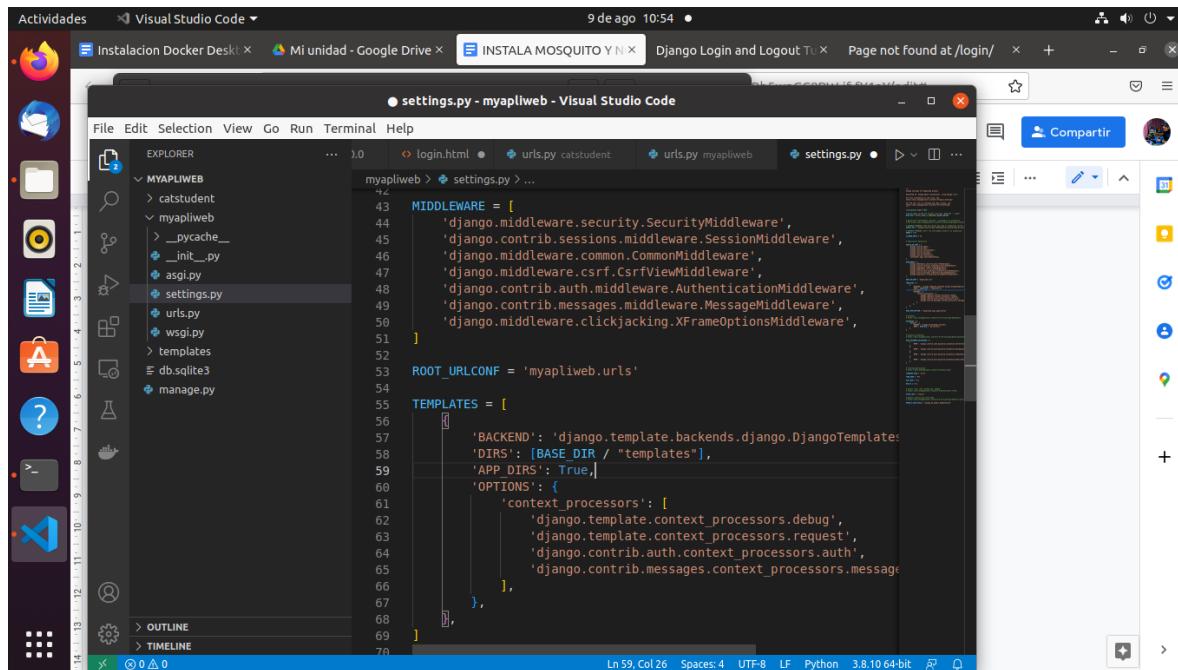


The screenshot shows the Visual Studio Code interface with multiple tabs open. The active tab is 'login.html - myapiweb - Visual Studio Code'. The code editor displays the following HTML template:

```
<!-- templates/registration/login.html -->
<h2>Log In</h2>
<form method="post">
    {% csrf_token %}
    {{ form.as_p }}
    <button type="submit">Log In</button>
</form>
```

The left sidebar shows the project structure with files like 'views.py', 'urls.py', 'catstudent', 'myapiweb', 'db.sqlite3', and 'manage.py'. The bottom status bar indicates 'Ln 7, Col 8' and 'UTF-8'.

Con esto ya tenemos nuestro post y aseguramos la seguridad de entremos a settings.py y dentro de esta buscamos TEMPLATES y específicamente el de "DIRS []" y dentro de los corchetes agragamos la siguiente linea > BASE_DIR / "templates"



The screenshot shows the Visual Studio Code interface with multiple tabs open. The active tab is 'settings.py - myapiweb - Visual Studio Code'. The code editor displays the following Python configuration:

```
MIDDLEWARE = [
    'django.middleware.security.SecurityMiddleware',
    'django.contrib.sessions.middleware.SessionMiddleware',
    'django.middleware.common.CommonMiddleware',
    'django.middleware.csrf.CsrfViewMiddleware',
    'django.contrib.auth.middleware.AuthenticationMiddleware',
    'django.contrib.messages.middleware.MessageMiddleware',
    'django.middleware.clickjacking.XFrameOptionsMiddleware',
]

ROOT_URLCONF = 'myapiweb.urls'

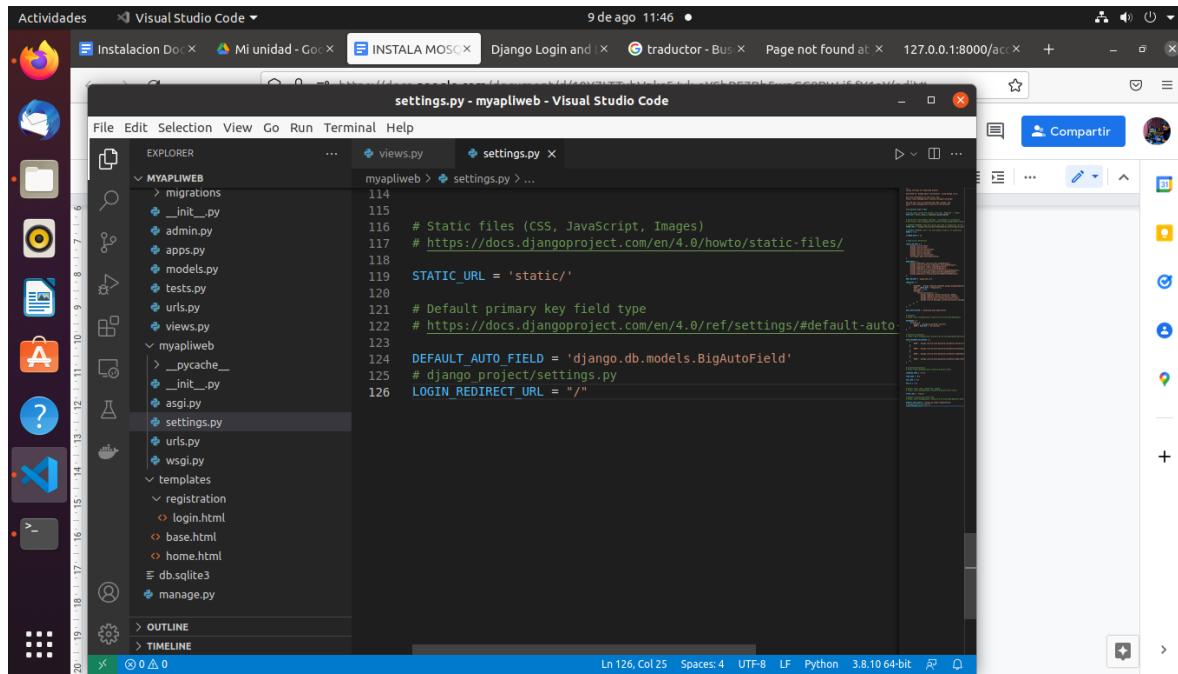
TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [BASE_DIR / "templates"],
        'APP_DIRS': True,
        'OPTIONS': {
            'context_processors': [
                'django.template.context_processors.debug',
                'django.template.context_processors.request',
                'django.contrib.auth.context_processors.auth',
                'django.contrib.messages.context_processors.messages',
            ],
        },
    },
]
```

The left sidebar shows the project structure with files like 'views.py', 'urls.py', 'catstudent', 'myapiweb', 'db.sqlite3', and 'manage.py'. The bottom status bar indicates 'Ln 59, Col 26' and 'UTF-8'.

Nuestra funcionalidad de inicio de sesión ahora funciona, pero para mejorarla, debemos especificar a dónde redirigir al usuario después de un inicio de sesión exitoso. En otras palabras, una vez que un usuario ha iniciado sesión, ¿dónde debe enviarse en el sitio? Usamos la configuración LOGIN_REDIRECT_URL para especificar esta ruta. En la

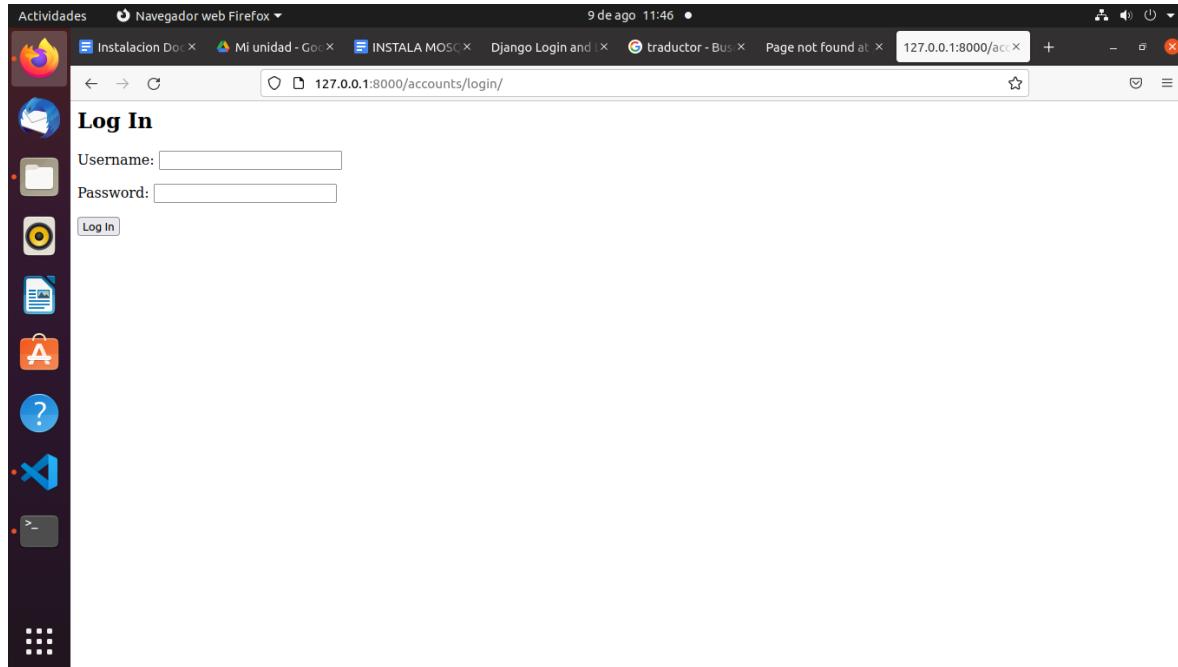
parte inferior del archivo settings.py, agregue lo siguiente para redirigir al usuario a la página de inicio.

¡En realidad hemos terminado en este punto! Si ahora vuelve a iniciar el servidor Django con python manage.py runserver y navega a nuestra página de inicio de sesión en <http://127.0.0.1:8000/accounts/login/> verá lo siguiente.



The screenshot shows the Visual Studio Code interface with the file 'settings.py' open. The code editor displays the following content:

```
114
115
116 # Static files (CSS, JavaScript, Images)
117 # https://docs.djangoproject.com/en/4.0/howto/static-files/
118 STATIC_URL = 'static/'
119
120 # Default primary key field type
121 # https://docs.djangoproject.com/en/4.0/ref/settings/#default-auto-
122
123 DEFAULT_AUTO_FIELD = 'django.db.models.BigAutoField'
124
125 # django_project/settings.py
126 LOGIN_REDIRECT_URL = ''
```



Queremos una página de inicio simple que muestre un mensaje para los usuarios que no hayan iniciado sesión y otro para los usuarios que

Néstor Emmanuel Briones Ramírez
1220100321 GDS0351 Desarrollo de software multi plataforma.

hayan iniciado sesión. Cree dos archivos nuevos con su editor de texto: templates/base.html y templates/home.html. Tenga en cuenta que estos se encuentran dentro de la carpeta de plantillas pero no dentro de templates/registration/ donde Django auth busca por defecto las plantillas de autenticación de usuario.

AGREGAMOS EN EL CMD LO SIGUIENTE EN myapliweb

```
touch templates/base.html
```

```
touch templates/home.html
```

y dentro de cada uno agregamos el HTML

```
<!-- templates/base.html -->
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>{% block title %}Django Auth Tutorial{% endblock %}</title>
</head>
<body>
<main>
  {% block content %}
  {% endblock %}
</main>
</body>
</html>
```

```
<!-- templates/home.html -->
{% extends 'base.html' %}

{% block title %}Home{% endblock %}

{% block content %}
{% if user.is_authenticated %}
  Hi {{ user.username }}!

```

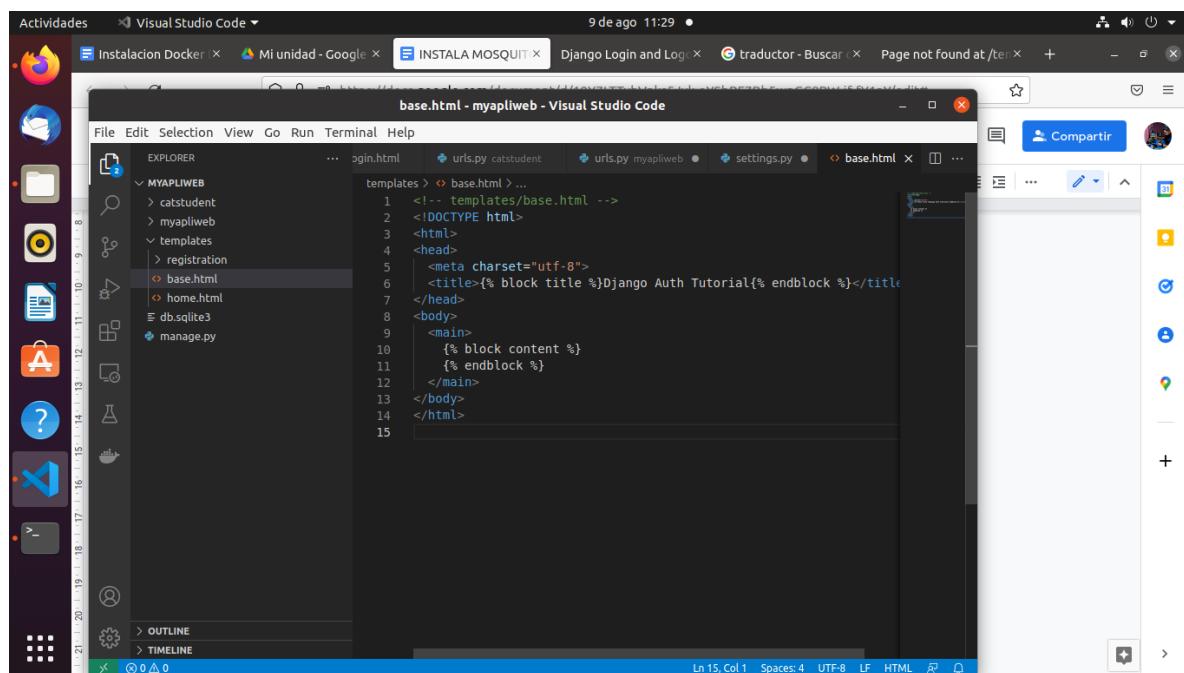
```
{% else %}

<p>You are not logged in</p>

<a href="{% url 'login' %}">Log In</a>

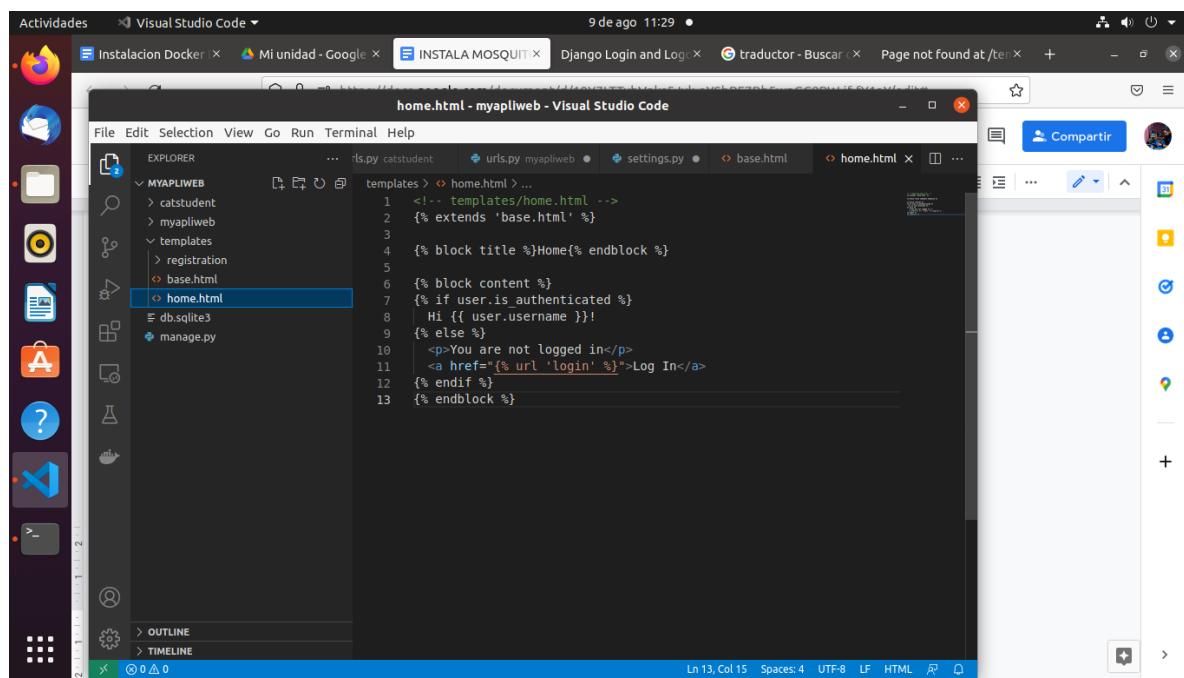
{% endif %}

{% endblock %}
```



A screenshot of the Visual Studio Code interface. The title bar shows "base.html - myapiweb - Visual Studio Code". The code editor displays the following HTML template:

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <title>{% block title %}Django Auth Tutorial{% endblock %}</title>
</head>
<body>
    <main>
        {% block content %}
        {% endblock %}
    </main>
</body>
</html>
```



A screenshot of the Visual Studio Code interface. The title bar shows "home.html - myapiweb - Visual Studio Code". The code editor displays the following HTML template, which extends the base.html template:

```
<!-- templates/home.html -->
{% extends 'base.html' %}

{% block title %}Home{% endblock %}

{% block content %}
    {% if user.is_authenticated %}
        Hi {{ user.username }}!
    {% else %}
        <p>You are not logged in</p>
        <a href="{% url 'login' %}">Log In</a>
    {% endif %}
{% endblock %}
```

Ahora actualice nuestro archivo urls.py para que podamos mostrar la página de inicio. Normalmente preferiría crear una aplicación de páginas dedicadas para este propósito, pero no tenemos que hacerlo y, por simplicidad, simplemente la agregaremos a nuestro archivo django_project/urls.py existente. Asegúrese de importar TemplateView en la tercera línea y luego agregue un patrón de URL en la ruta ''.

```
# django_project/urls.py

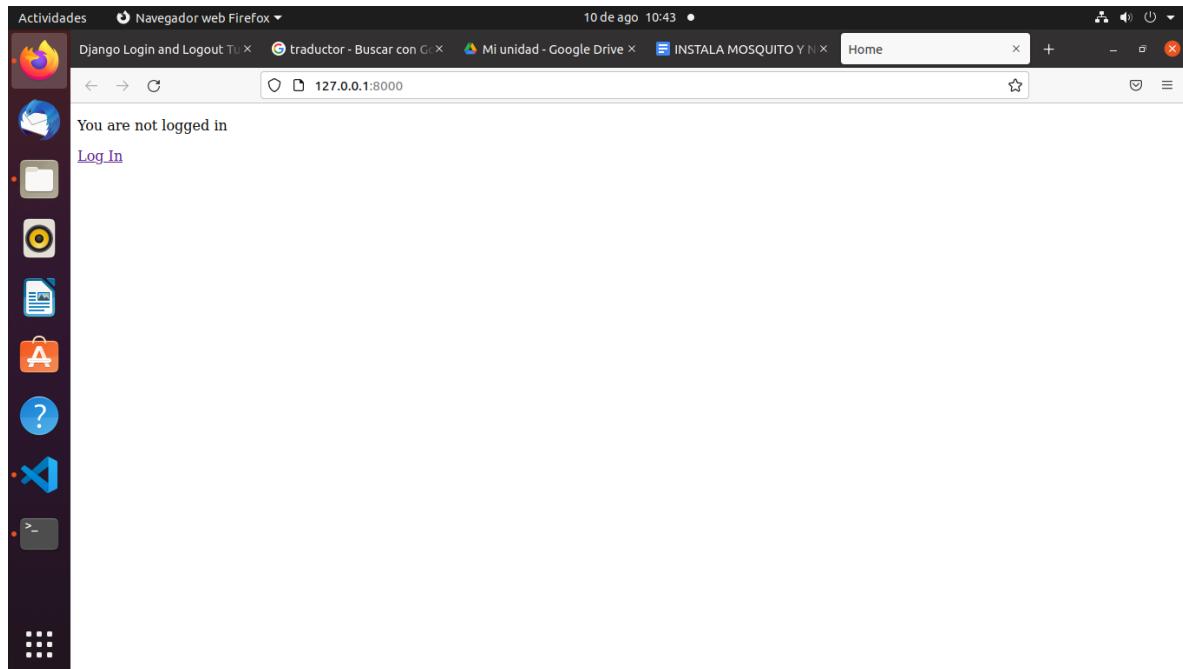
from django.contrib import admin

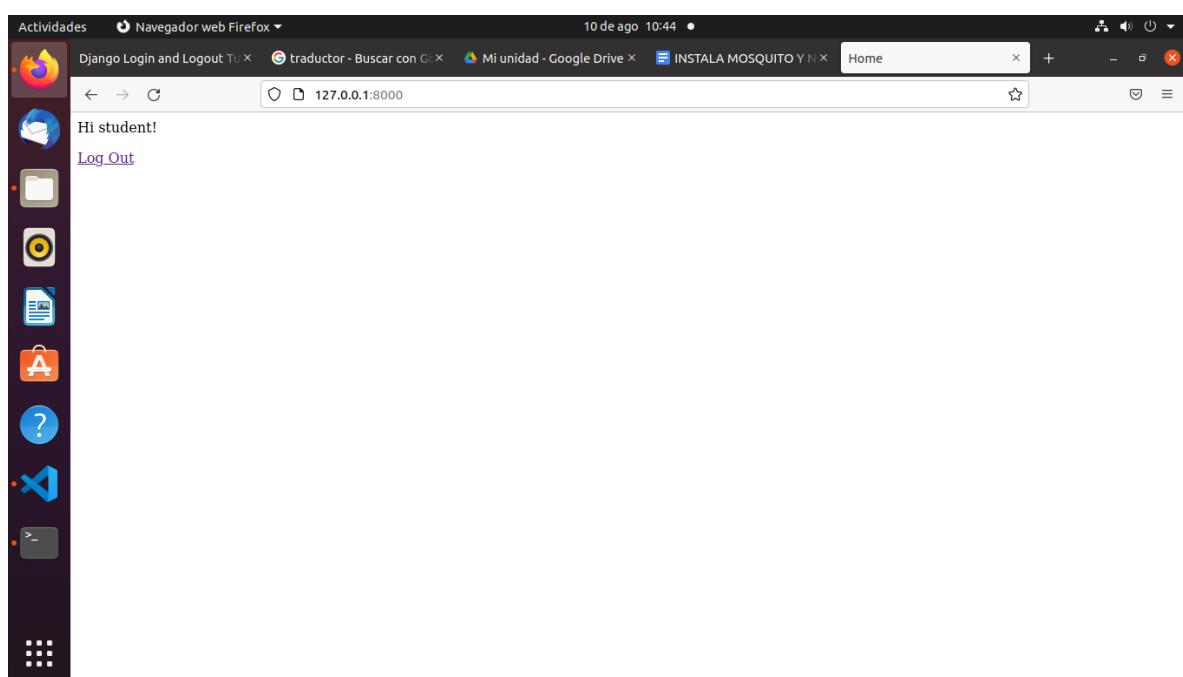
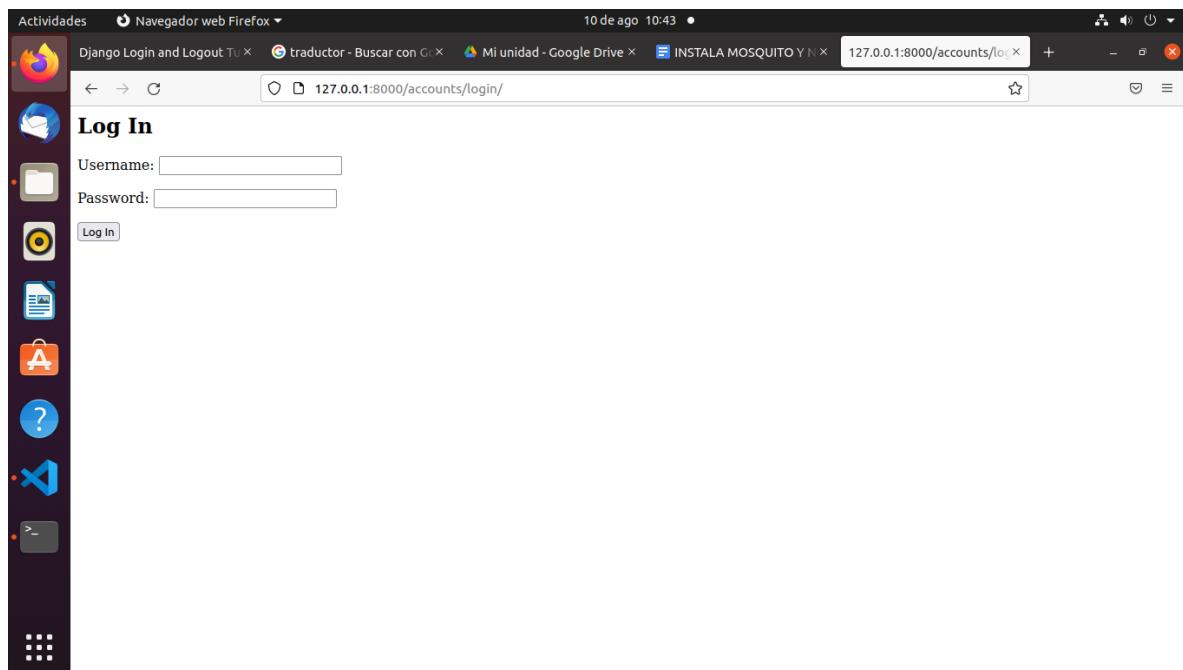
from django.urls import path, include

from django.views.generic.base import TemplateView # new

urlpatterns = [
    path('admin/', admin.site.urls),
    path('accounts/', include('django.contrib.auth.urls')),
    path('', TemplateView.as_view(template_name='home.html'),
name='home'), # new
]
```

Hemos terminado. Si vuelve a iniciar el servidor Django con python manage.py runserver y navega a la página de inicio en <http://127.0.0.1:8000/> verá lo siguiente:



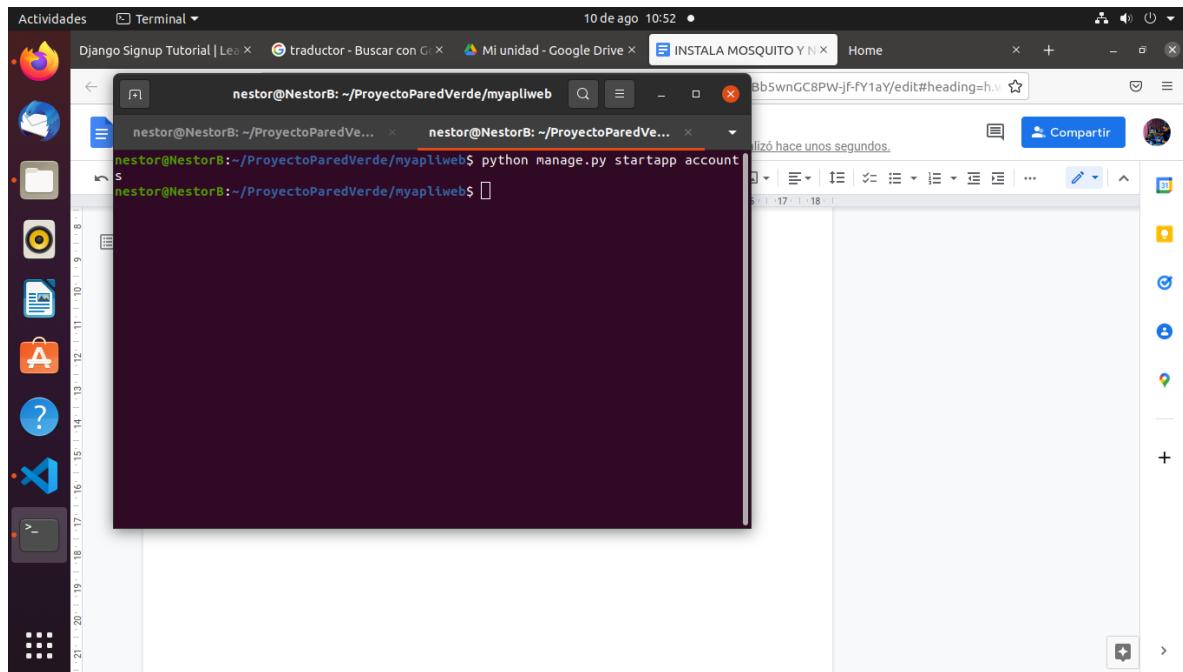


Django Signup Tutorial

Users app

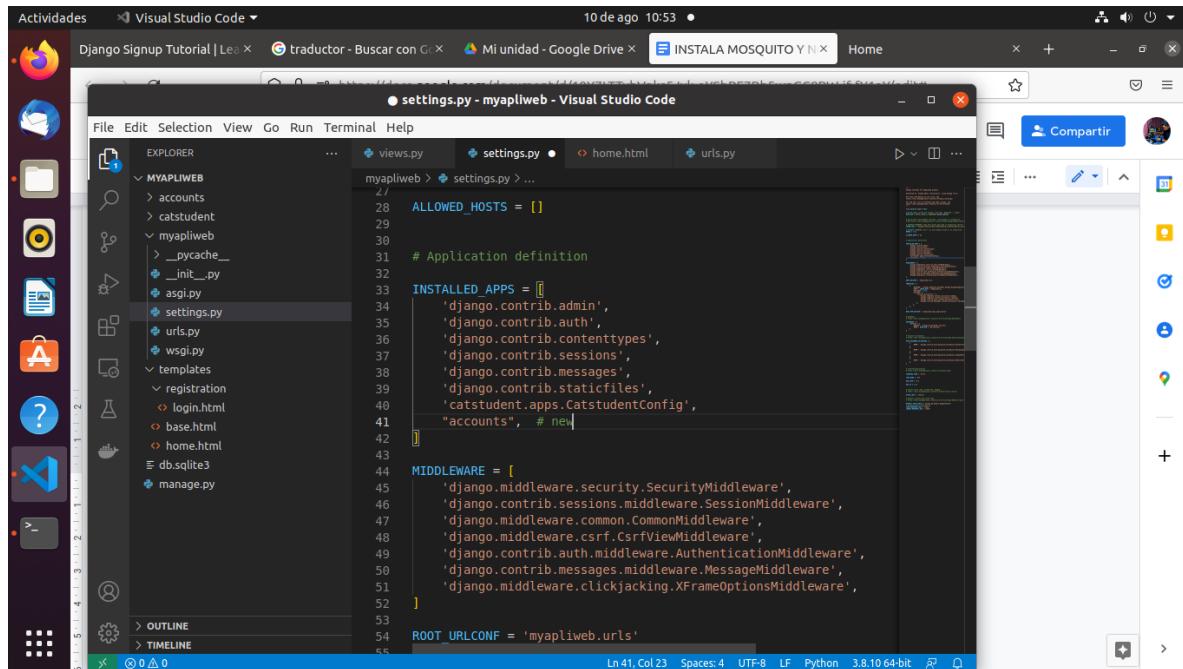
Dado que estamos creando nuestra propia vista y URL para el registro, necesitamos crear una aplicación dedicada. Llámémoslo cuentas.

```
(accounts) $ python manage.py startapp accounts
```



Asegúrate de agregar la nueva aplicación a la configuración `INSTALLED_APPS` en nuestro archivo `django_project/settings.py`:

```
"accounts", # new
```



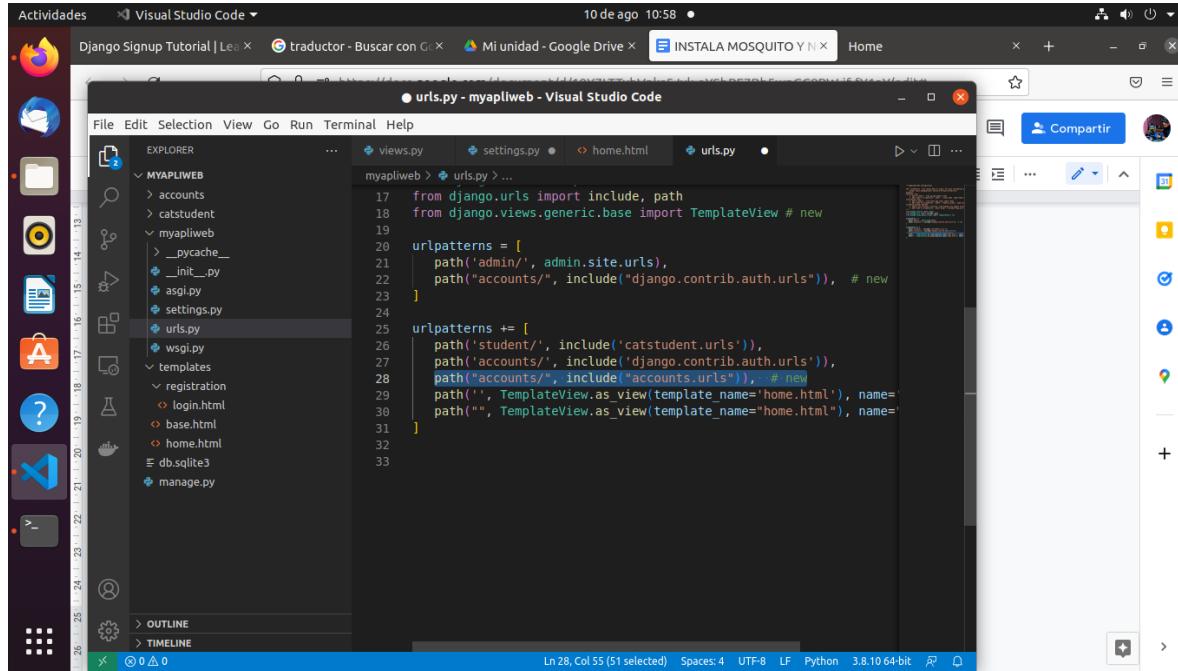
Luego agregue una URL de nivel de proyecto para la aplicación de cuentas arriba de nuestra aplicación de autenticación de Django incluida. Django buscará patrones de URL de arriba a abajo, de modo que cuando vea una ruta de URL dentro de nuestra aplicación de cuentas que coincida con una en la aplicación de autenticación integrada, elegirá primero la ruta de cuentas.

Néstor Emmanuel Briones Ramírez

1220100321 GDS0351 Desarrollo de software multi plataforma.

agregamos lo siguiente asi en django_project/urls.py

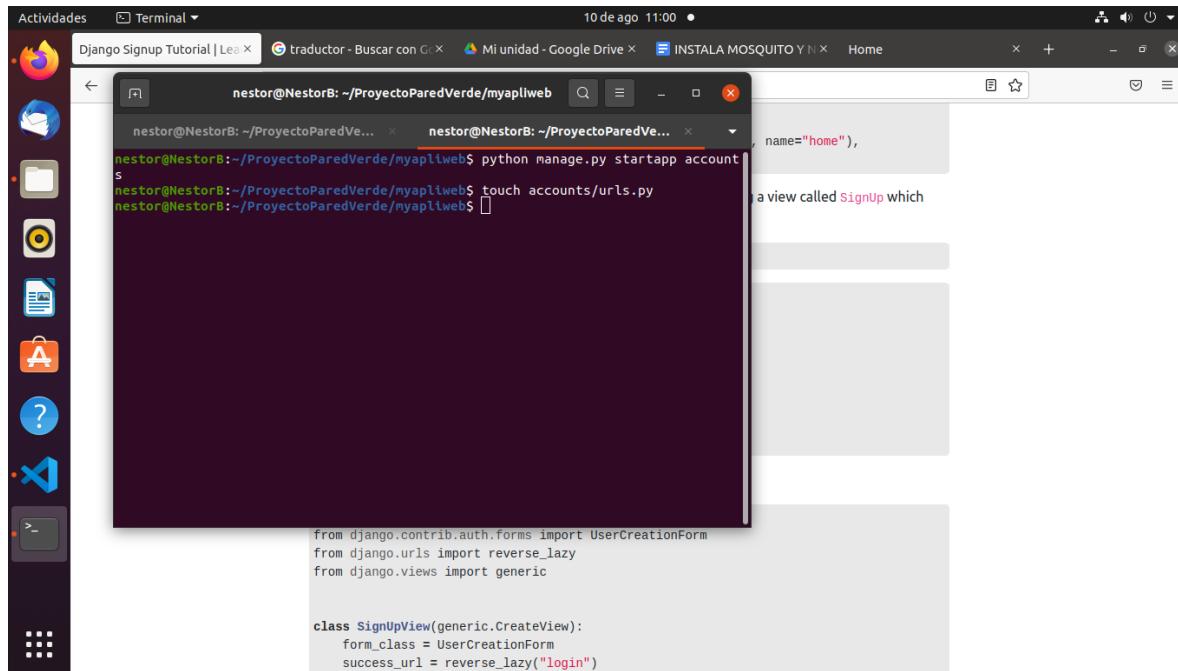
```
path("accounts/", include("accounts.urls")), # new
```



```
myapiweb > urls.py > ...
17 from django.urls import include, path
18 from django.views.generic.base import TemplateView # new
19
20 urlpatterns = [
21     path('admin/', admin.site.urls),
22     path('accounts/', include("django.contrib.auth.urls")), # new
23 ]
24
25 urlpatterns += [
26     path('student/', include('catstudent.urls')),
27     path('accounts/', include("django.contrib.auth.urls")),
28     path('accounts/', include("accounts.urls")), # new
29     path('', TemplateView.as_view(template_name='home.html'), name='home'),
30     path("", TemplateView.as_view(template_name="home.html"), name="home")
31 ]
32
33
```

Cree un nuevo archivo de URL en nuestra aplicación de cuentas. Tenga en cuenta que estamos importando una vista llamada SignUp que implementaremos en la siguiente sección.

```
touch accounts/urls.py
```



```
nestor@NestorB:~/ProyectoParedVerde/myapiweb$ python manage.py startapp accounts
nestor@NestorB:~/ProyectoParedVerde/myapiweb$ touch accounts/urls.py
nestor@NestorB:~/ProyectoParedVerde/myapiweb$
```

```
from django.contrib.auth.forms import UserCreationForm
from django.urls import reverse_lazy
from django.views import generic

class SignUpView(generic.CreateView):
    form_class = UserCreationForm
    success_url = reverse_lazy("login")
```

agregamos el siguiente texto

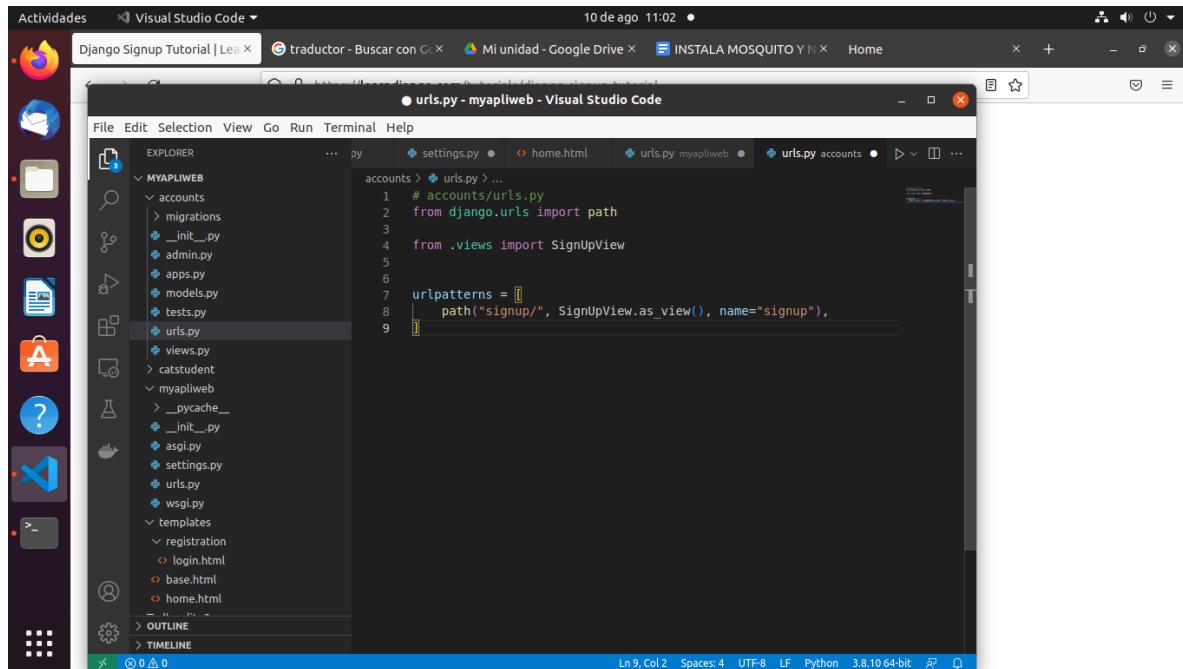
Néstor Emmanuel Briones Ramírez
1220100321 GDS0351 Desarrollo de software multi plataforma.

```
# accounts/urls.py

from django.urls import path

from .views import SignUpView

urlpatterns = [
    path("signup/", SignUpView.as_view(), name="signup"),
]
```



Ahora para el archivo views.py:

```
# accounts/views.py

from django.contrib.auth.forms import UserCreationForm
from django.urls import reverse_lazy
from django.views import generic

class SignUpView(generic.CreateView):
    form_class = UserCreationForm
    success_url = reverse_lazy("login")
    template_name = "registration/signup.html"
```

Bueno último paso. Cree una nueva plantilla templates/registration/signup.html y complétala con este código que se parece casi exactamente al que usamos para login.html.

```
<!-- templates/registration/signup.html -->
```

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

```
{% block title %}Sign Up{% endblock %}
```

```
{% block content %}
```

```
    <h2>Sign up</h2>
```

```
    <form method="post">
```

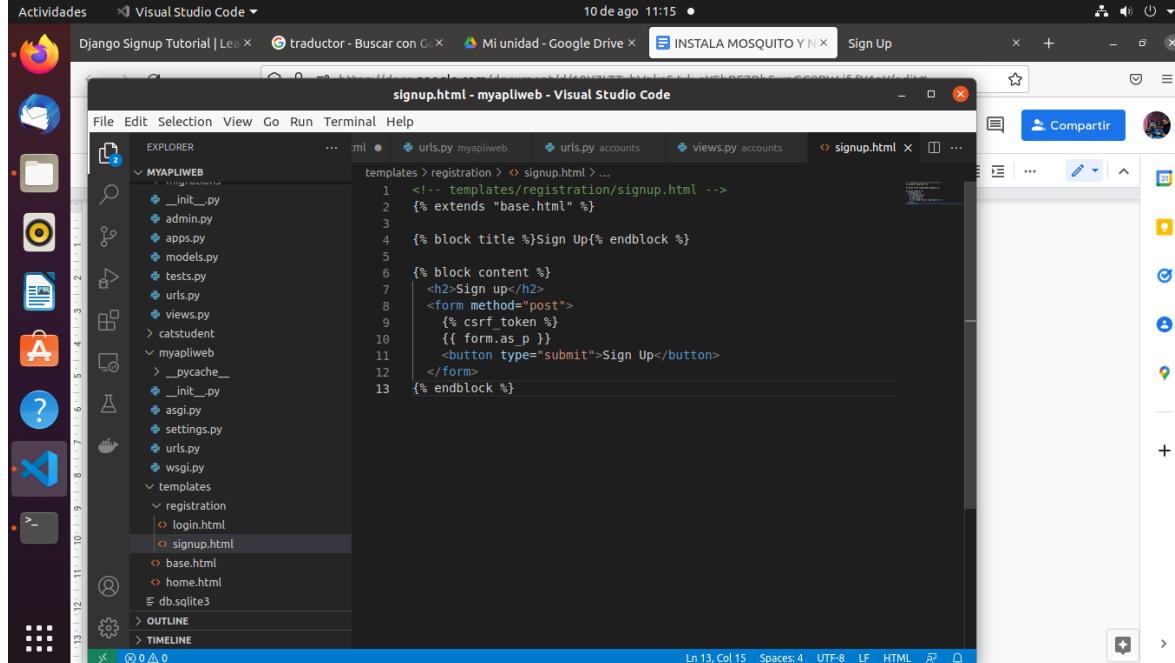
```
        {% csrf_token %}
```

```
        {{ form.as_p }}
```

```
        <button type="submit">Sign Up</button>
```

```
    </form>
```

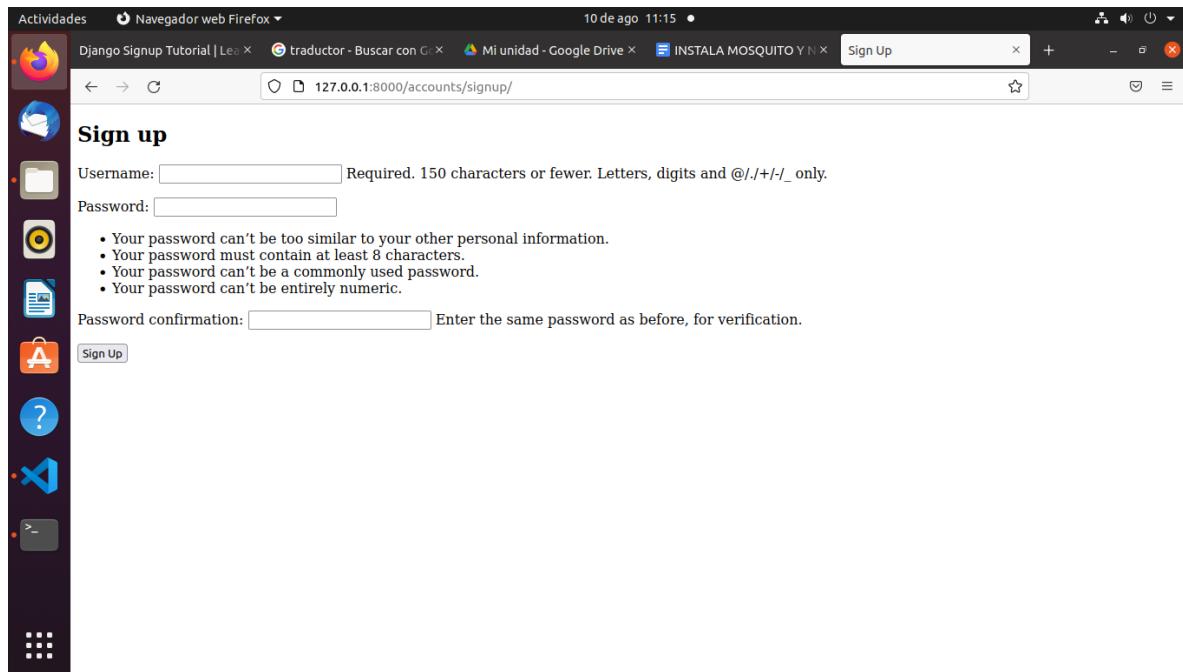
```
{% endblock %}
```



```
File Edit Selection View Go Run Terminal Help
EXPLORER
MYAPIWEB
templates > registration > signup.html > ...
1  <!-- templates/registration/signup.html -->
2  {% extends "base.html" %}
3
4  {% block title %}Sign Up{% endblock %}
5
6  {% block content %}
7      <h2>Sign up</h2>
8      <form method="post">
9          {% csrf_token %}
10         {{ form.as_p }}
11         <button type="submit">Sign Up</button>
12     </form>
13  {% endblock %}
```

¡Y hemos terminado! Para confirmar que todo funciona, activa nuestro servidor local con python manage.py runserver y navega hasta c.

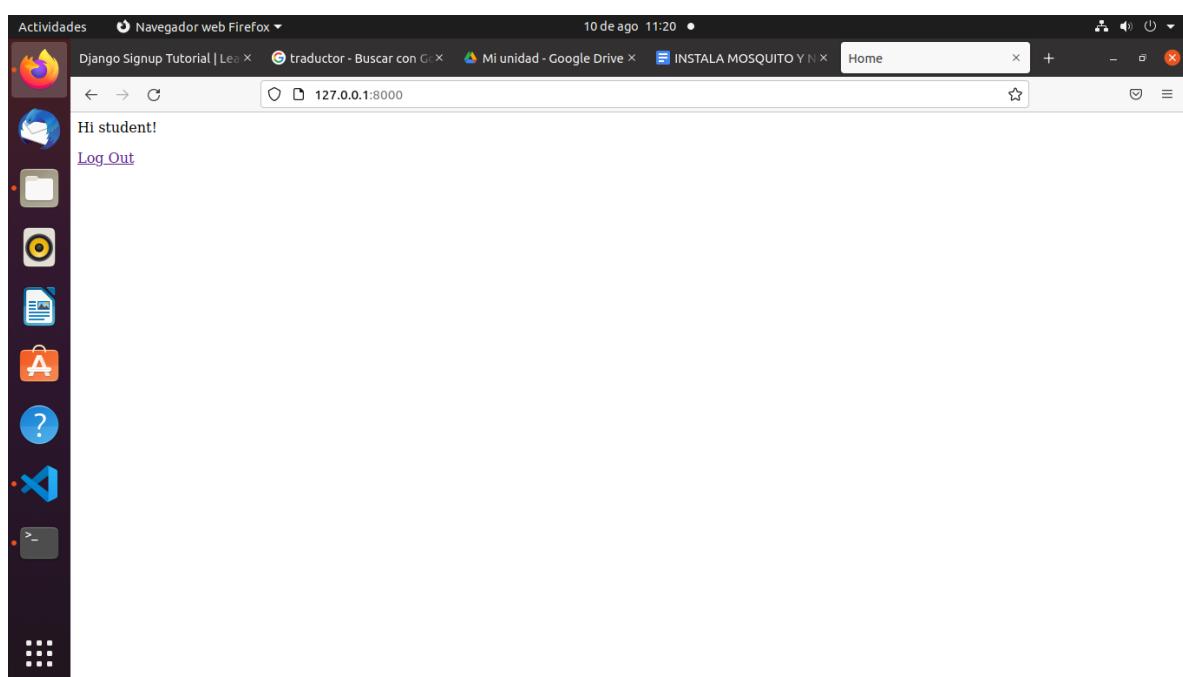
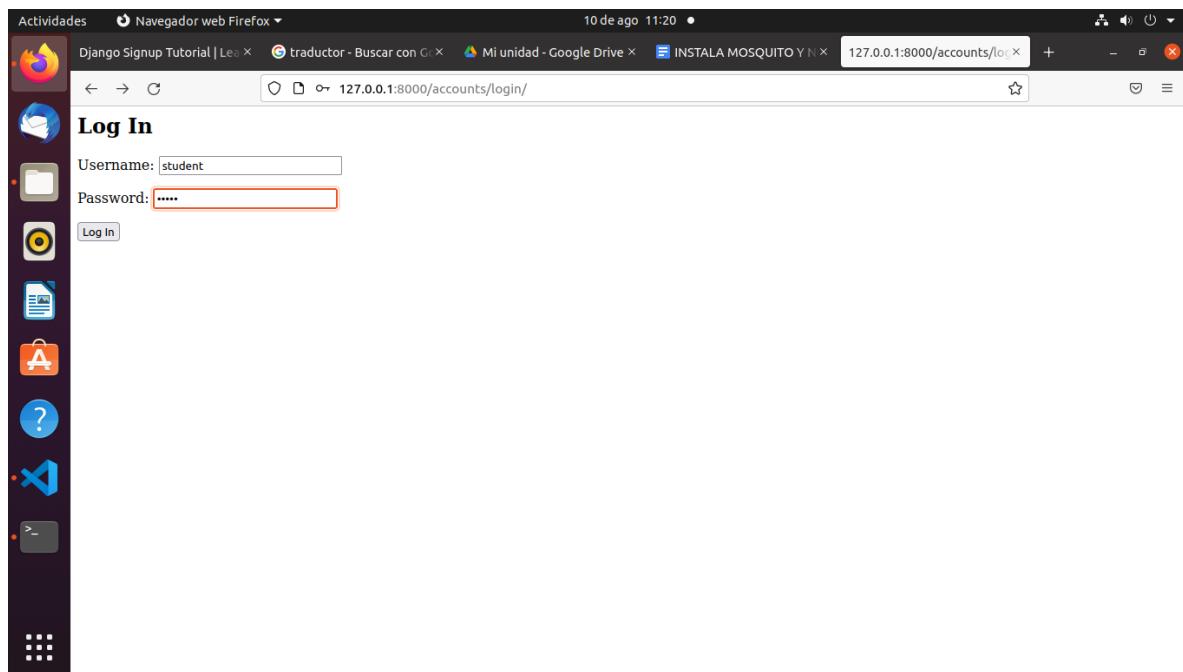
<http://127.0.0.1:8000/accounts/signup/>



El texto adicional con consejos sobre nombres de usuario y contraseñas proviene de Django. También podemos personalizar eso, pero requiere un poco más de trabajo y está más allá del alcance de este tutorial.

Regístrese para obtener una nueva cuenta y presione el botón "Registrarse". Se le redirigirá a la página de inicio de sesión <http://127.0.0.1:8000/accounts/login/> donde podrá iniciar sesión con su nueva cuenta.

Y luego, después de un inicio de sesión exitoso, será redirigido a la página de inicio con un "Hola, nombre de usuario" personalizado. saludo.



Tutorial de restablecimiento de contraseña de Django

Aplicación de autenticación de Django

Lo que queremos es una página de restablecimiento de contraseña donde el usuario pueda ingresar su dirección de correo electrónico y recibir un correo electrónico criptográficamente seguro con un enlace único a una página de restablecimiento. Afortunadamente, Django nos tiene cubiertos.

Si recuerda el conjunto completo de vistas y URL proporcionadas por la aplicación de autenticación de Django, ya hay varias para restablecer una contraseña.

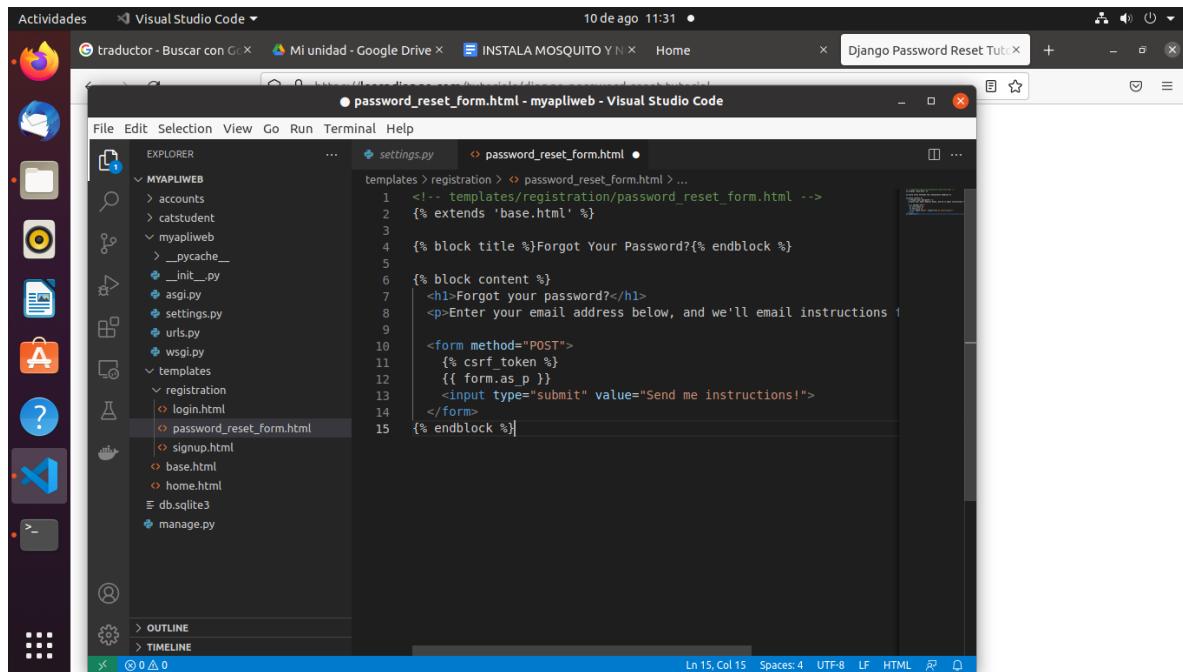
SMTP Server

Para configurar esto, actualice nuestro archivo `django_project/settings.py` agregando las siguientes dos líneas en la parte inferior debajo de nuestras URL de redirección.

Password Reset Form

La plantilla predeterminada para el restablecimiento de contraseña se encuentra en `templates/registration/password_reset_form.html`. Podemos personalizarlo creando nuestro propio archivo `password_reset_form.html`:

```
touch templates/registration/password_reset_form.html  
agregamos el siguiente código en nuestro nuevo HTML.  
<!-- templates/registration/password_reset_form.html -->  
{% extends 'base.html' %}  
  
{% block title %}Forgot Your Password?{% endblock %}  
  
{% block content %}  
    <h1>Forgot your password?</h1>  
    <p>Enter your email address below, and we'll email instructions for  
    setting a new one.</p>  
  
<form method="POST">  
    {% csrf_token %}  
    {{ form.as_p }}  
    <input type="submit" value="Send me instructions!">  
</form>  
{% endblock %}
```



```
<!-- templates/registration/password_reset_form.html -->
{% extends 'base.html' %}



# Forgot your password?



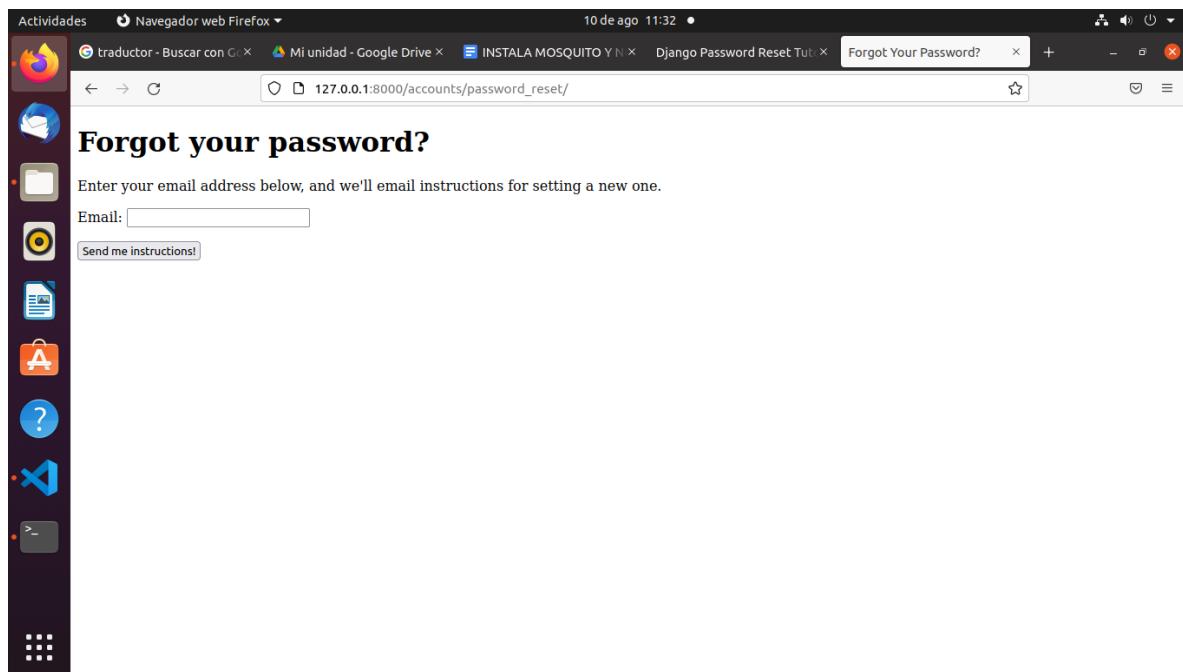
Enter your email address below, and we'll email instructions



<form method="POST">
    {% csrf_token %}
    {{ form.as_p }}
    <input type="submit" value="Send me instructions!">
</form>

{% endblock %}
```

Si actualiza la página en http://127.0.0.1:8000/accounts/password_reset/ podrá ver nuestra nueva actualización:



Ahora continúe e ingrese la dirección de correo electrónico que coincide con un usuario real que ha creado. Luego haga clic en el botón para enviarlo.

Tras el envío exitoso, somos redirigidos a la página de restablecimiento de contraseña, que también es fea. Vamos a cambiarlo.

La plantilla predeterminada se encuentra en templates/registration/password_reset_done.html. Entonces, como antes, en su editor de texto cree un nuevo archivo de plantilla templates/registration/password_reset_done.html y agregue el siguiente código:

```
<!-- templates/registration/password_reset_done.html -->

{% extends "base.html" %}

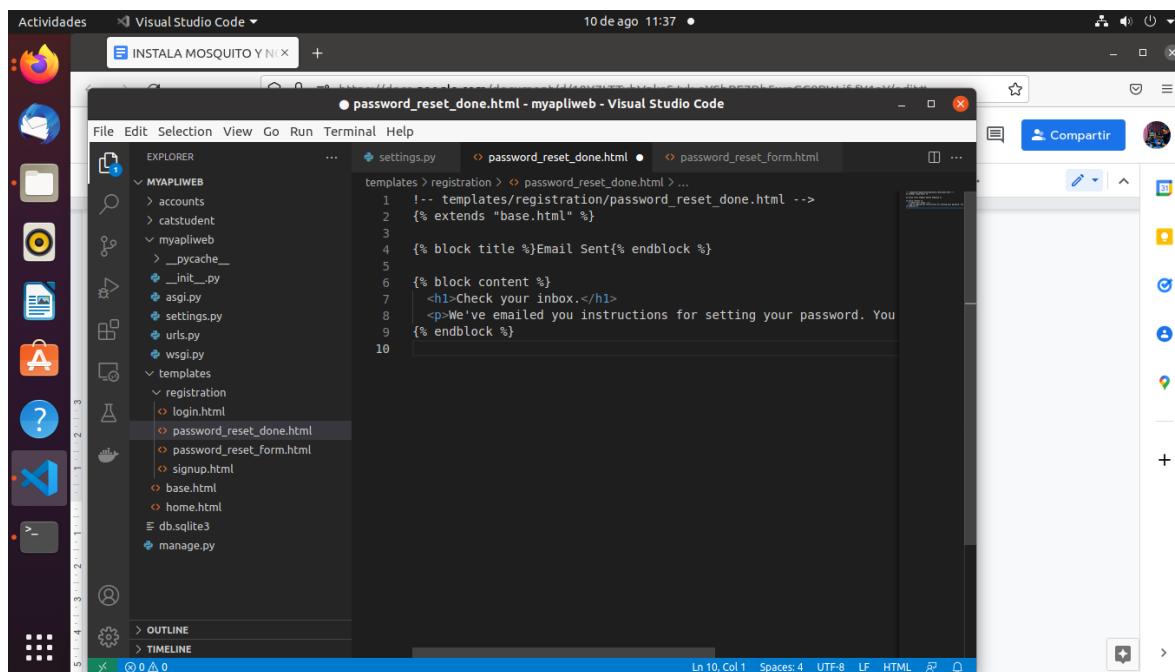
{% block title %}Email Sent{% endblock %}

{% block content %}

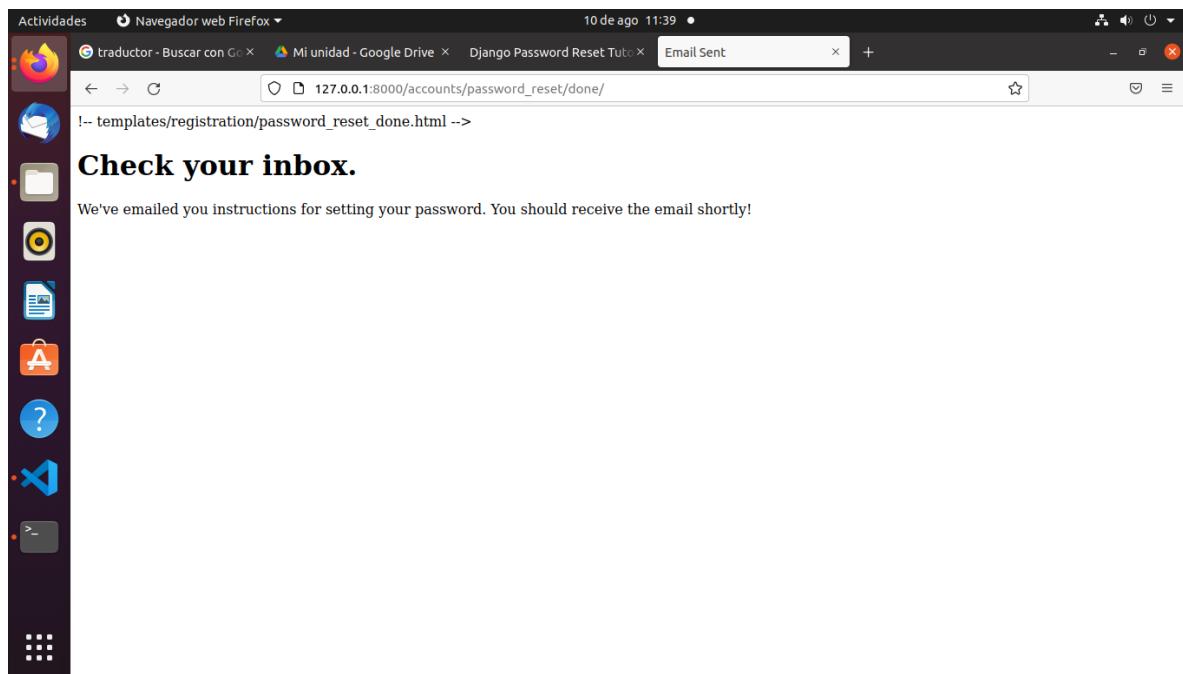
<h1>Check your inbox.</h1>

<p>We've emailed you instructions for setting your password. You
should receive the email shortly!</p>

{% endblock %}
```

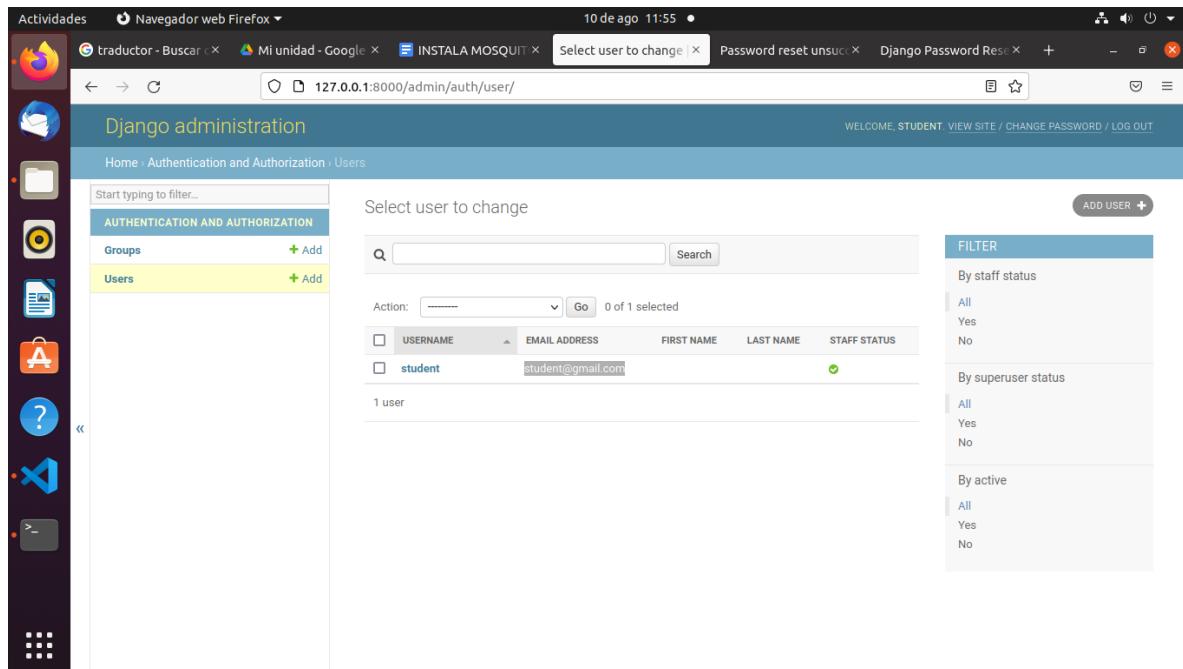


Si actualiza la página de restablecimiento de contraseña en http://127.0.0.1:8000/accounts/password_reset/done/, podemos ver nuestra nueva página.



Restablecimiento de contraseña Confirmar

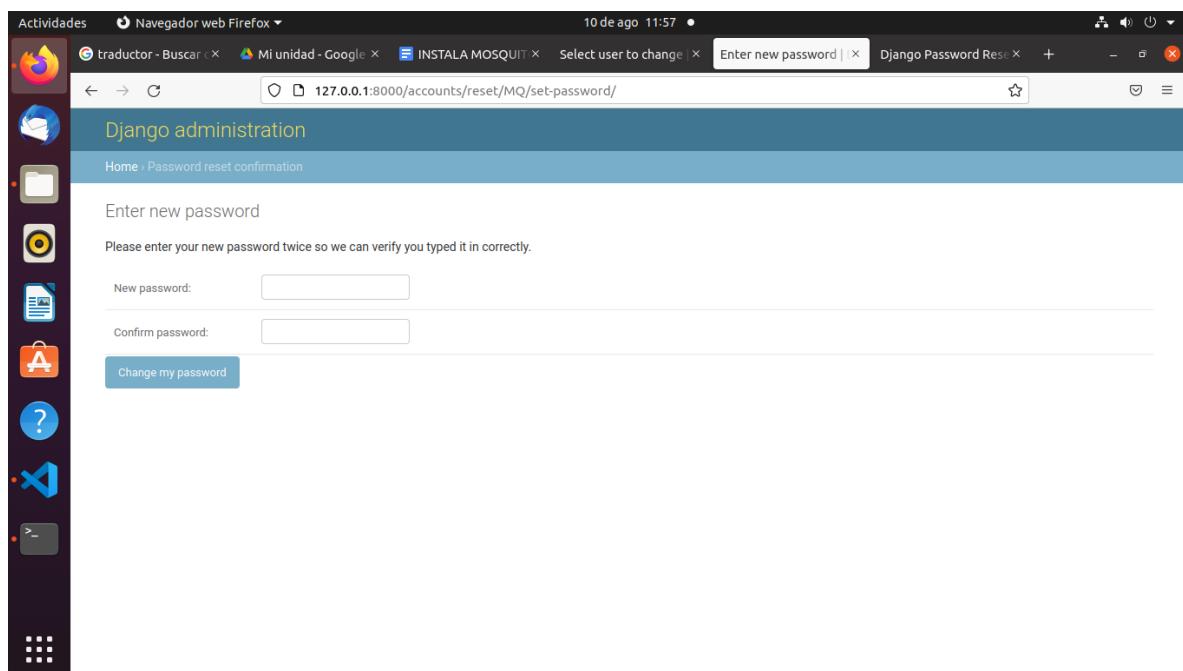
This contains Django's default language which we can customize. But the important section for now is the URL included. In the email above, mine is <http://127.0.0.1:8000/accounts/reset/MQ/aa1v2k-8ab2c9597a4f6cc754e3dc5baaf3c77f/>. Copy and paste yours into your browser and you'll be automatically routed to the *Password reset confirmation* page.



The screenshot shows a Visual Studio Code window with two tabs open. The left tab is 'manage.py' and the right tab is 'password_reset_form.html'. The right tab displays a log file titled '20220810-165252-139808433279424.log' from 'myapliweb'. The log content is as follows:

```
Content-Type: text/plain; charset="utf-8"
MIME-Version: 1.0
Content-Transfer-Encoding: 8bit
Subject: Password reset on 127.0.0.1:8000
From: webmaster@localhost
To: student@gmail.com
Date: Wed, 10 Aug 2022 16:52:52 -0000
Message-ID: <166015037203.32732.4409187934302870975@NestorB>

You're receiving this email because you requested a password reset
Please go to the following page and choose a new password:
http://127.0.0.1:8000/accounts/reset/MQ/b9y9k4-c85eb8e482dc5a778f7/
Your username, in case you've forgotten: student
Thanks for using our site!
The 127.0.0.1:8000 team
```



feo, no? Vamos a crear una nueva plantilla con nuestros pasos familiares. En su editor de texto, cree la nueva plantilla llamada templates/registration/password_reset_confirm.html e ingrese este nuevo código:

```
<!-- templates/registration/password_reset_confirm.html -->
{% extends "base.html" %}

{% block title %}Enter new password{% endblock %}
```

```
{% block content %}

{% if validlink %}

<h1>Set a new password!</h1>

<form method="POST">

    {% csrf_token %}

    {{ form.as_p }}

    <input type="submit" value="Change my password">

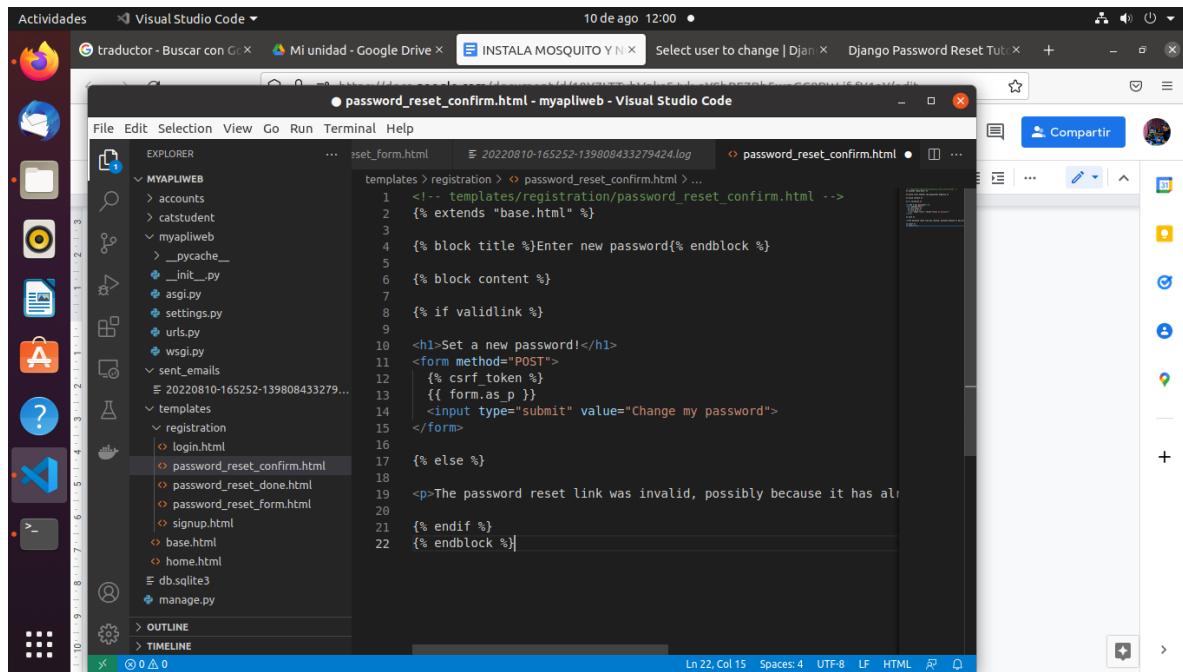
</form>

{% else %}

<p>The password reset link was invalid, possibly because it has already been used. Please request a new password reset.</p>

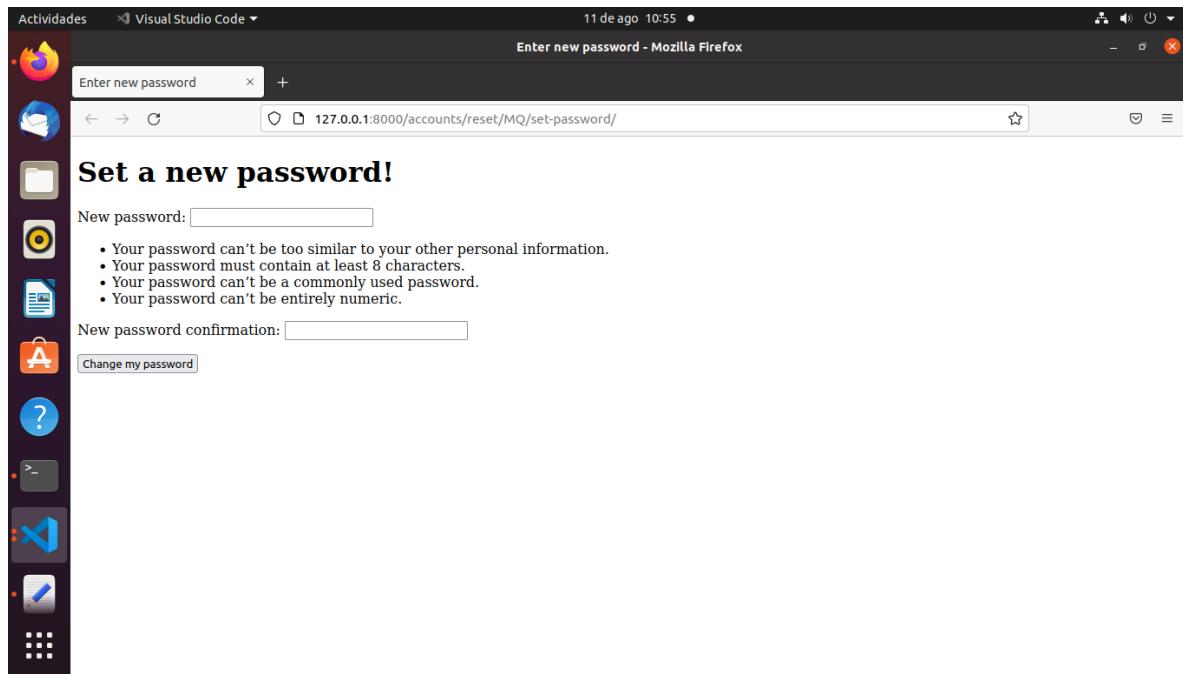
{% endif %}

{% endblock %}
```



Actualice la página en <http://127.0.0.1:8000/accounts/reset/Mg/set-password/> y verá nuestra nueva plantilla.

4btAqrQvgysGzJV



Para personalizar esta página, crearemos un nuevo archivo llamado `password_reset_complete.html` e ingresaremos el siguiente código:

```
<!-- templates/registration/password_reset_complete.html -->

{% extends 'base.html' %}

{% block title %}Password reset complete{% endblock %}

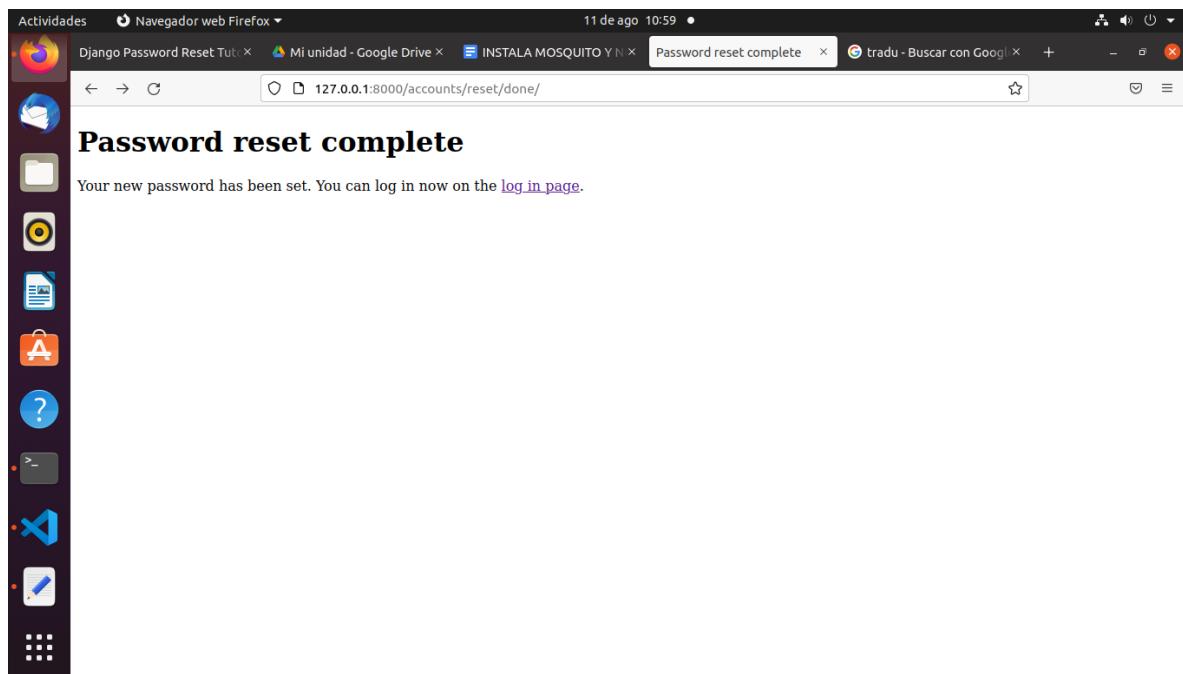
{% block content %}

<h1>Password reset complete</h1>

<p>Your new password has been set. You can log in now on the <a href="{% url 'login' %}">log in page</a>. </p>

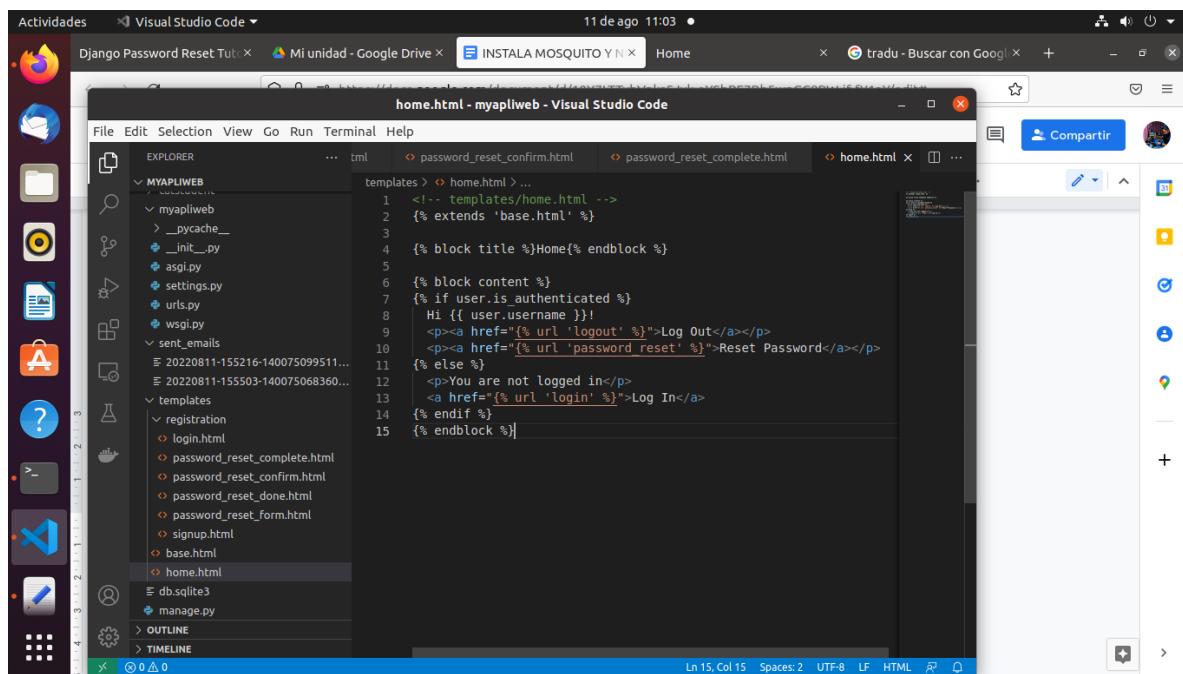
{% endblock %}

probamos con la url http://127.0.0.1:8000/accounts/reset/done/
```

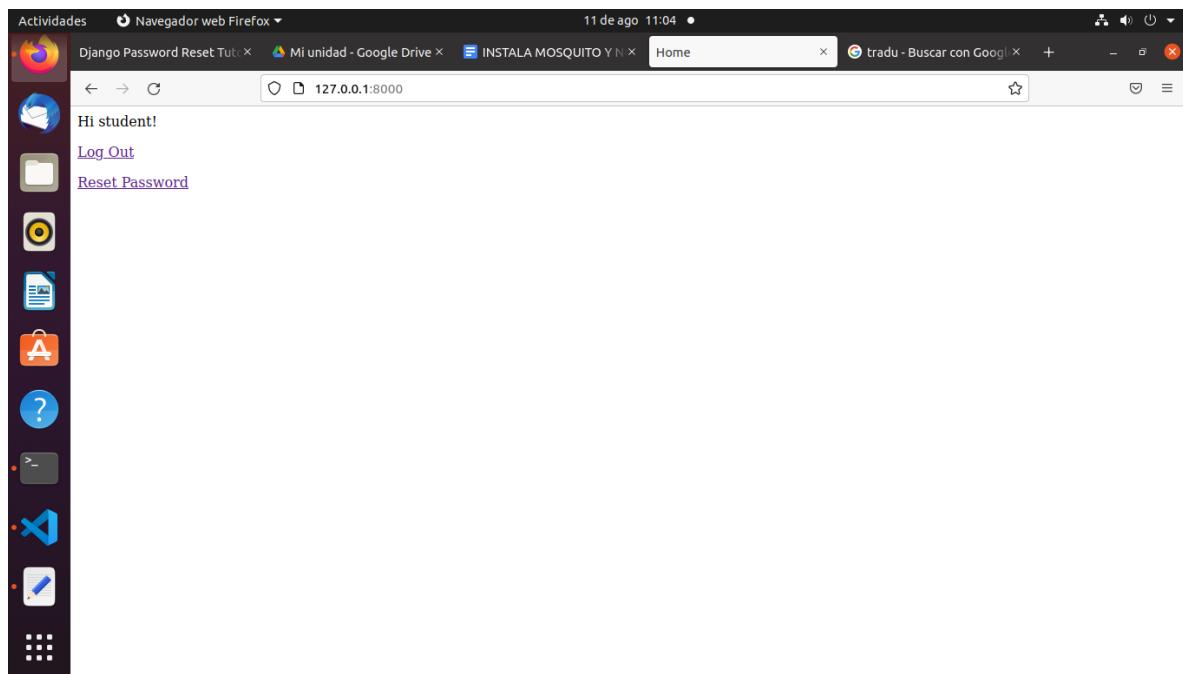


Agreguemos el enlace de restablecimiento de contraseña a la página de inicio ahora para que los usuarios registrados lo vean. Podemos usar la etiqueta integrada `{% url 'password_reset' %}`. Aquí está el código.

```
<p><a href="{% url 'password_reset' %}">Reset Password</a></p>
```



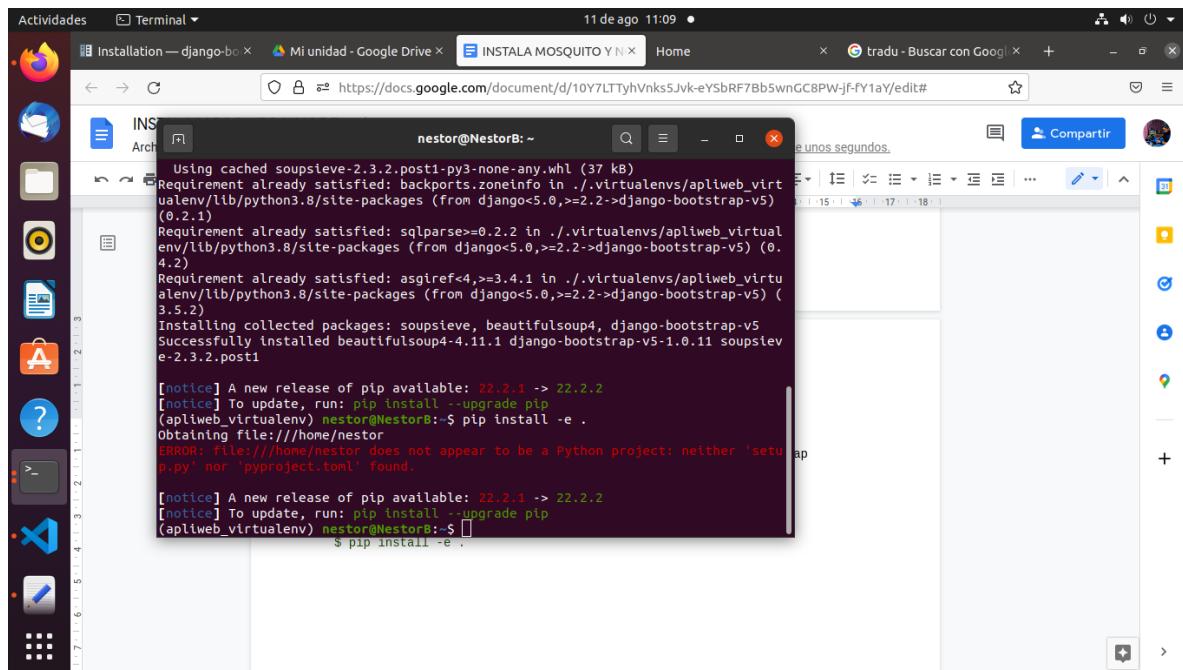
<http://127.0.0.1:8000/>



DAR ESTILOS A LAS PAGINAS CREADAS

usamo los siguientes comando para actualizar o instalar bootstrap en django.

```
$ pip install django-bootstrap-v5  
$ pip install -e .
```



En su proyecto, debe agregar django-bootstrap-v5 a sus `requisitos.txt`.

Néstor Emmanuel Briones Ramírez

1220100321 GDS0351 Desarrollo de software multi plataforma.

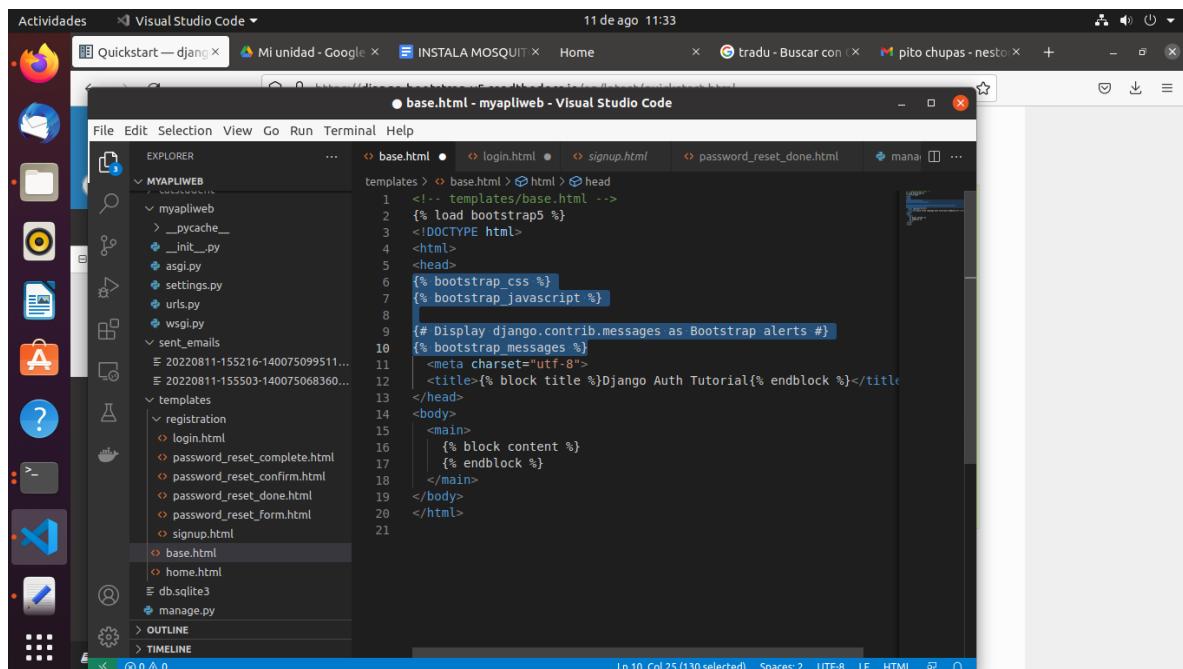
Asegúrese de usar virtualenv si desarrolla proyectos de python.

Agregue a INSTALLED_APPS en su configuración.py:

```
'arranque5',
```

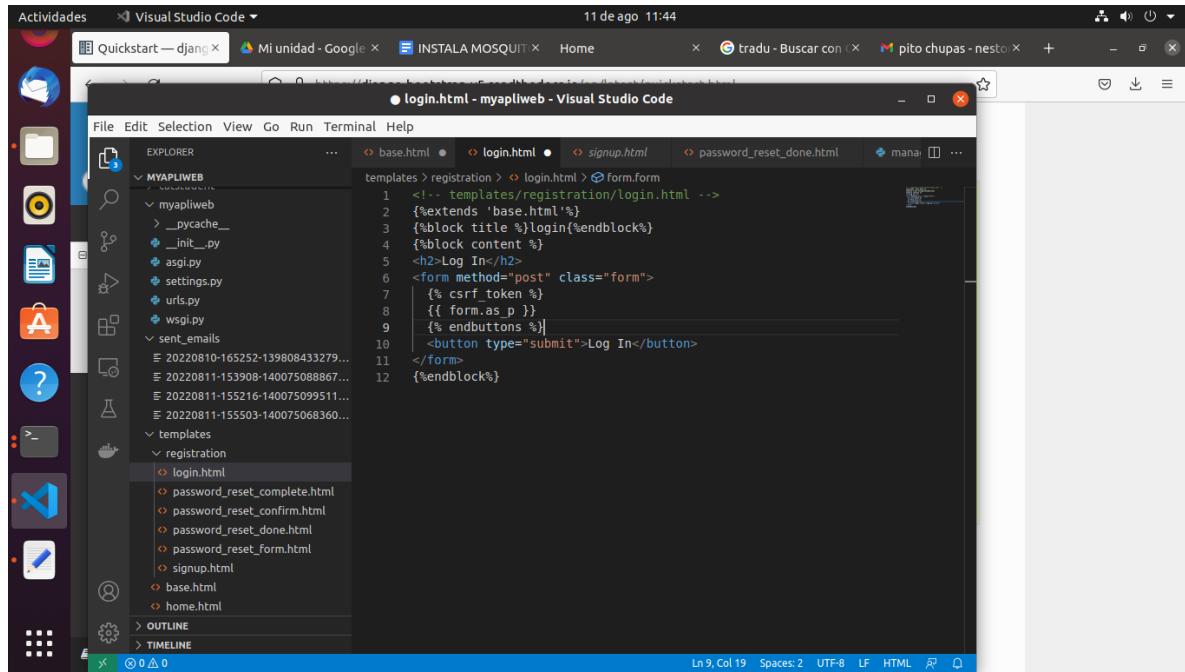
Después de la instalación, el inicio rápido lo ayudará a usar django-bootstrap-v5.

En nuestro documento BASE.HTML vamos a modificarlo de la siguiente maner.



```
File Edit Selection View Go Run Terminal Help
EXPLORER
MYAPIWEB
  myapiweb
    __pycache__
    __init__.py
    asgi.py
    settings.py
    urls.py
    wsgi.py
  sent_emails
    20220811-155216-140075099511...
    20220811-155503-140075068360...
  templates
    registration
      login.html
      password_reset_complete.html
      password_reset_confirm.html
      password_reset_done.html
      password_reset_form.html
      signup.html
    base.html
    home.html
    db.sqlite3
    manage.py
  OUTLINE
  TIMELINE
Ln 10, Col 25 (130 selected) Spaces: 2 UTF-8 LF HTML ⚡ 0
```

y en el documento login.HTML también vamos a agregar lo siguientes de la siguientes maner.



The screenshot shows the Visual Studio Code interface with the file 'login.html' open in the center editor pane. The code is a Django template for a login form:

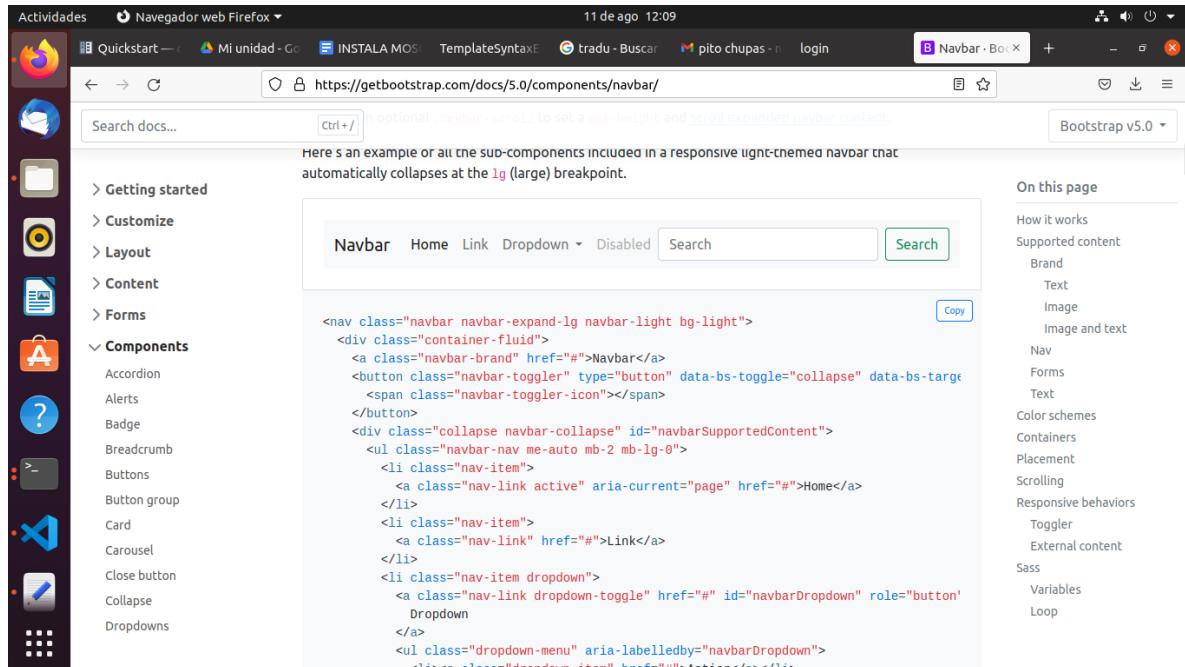
```
<!-- templates/registration/login.html -->
{% extends 'base.html' %}
{% block title %}Login{% endblock %}
{% block content %}
<h2>Log In</h2>
<form method="post" class="form">
    {% csrf_token %}
    {{ form.as_p }}
    <button type="submit">Log In</button>
</form>
{% endblock%}
```

The left sidebar shows the project structure under 'MYAPIWEB' with files like __init__.py, asgi.py, settings.py, urls.py, wsgi.py, and sent_emails.py. The 'templates' folder contains registration/login.html and other password reset-related files. The bottom status bar shows 'Ln 9, Col 19 Spaces: 2 UTF-8 LF HTML'.

ahora para ver si podemos implementar código de bootstrap lo que haremos es que vamos a buscar un navbar en la página. en el siguiente link

<https://getbootstrap.com/docs/5.0/components/navbar/>

agregaremos este código



The screenshot shows a Firefox browser window with the URL 'https://getbootstrap.com/docs/5.0/components/navbar/' in the address bar. The page displays the Bootstrap v5.0 navbar documentation. On the left, there's a sidebar with navigation links like 'Getting started', 'Customize', 'Layout', 'Content', 'Forms', and 'Components' (which is expanded to show Accordion, Alerts, Badge, Breadcrumb, Buttons, Button group, Card, Carousel, Close button, Collapse, and Dropdowns). The main content area shows a responsive navbar example with a search bar and a dropdown menu. Below the example is the HTML code for the navbar:

```
<nav class="navbar navbar-expand-lg navbar-light bg-light">
  <div class="container-fluid">
    <a class="navbar-brand" href="#">Navbar</a>
    <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarSupportedContent">
      <span class="navbar-toggler-icon"></span>
    </button>
    <div class="collapse navbar-collapse" id="navbarSupportedContent">
      <ul class="navbar-nav me-auto mb-2 mb-lg-0">
        <li class="nav-item">
          <a class="nav-link active" aria-current="page" href="#">Home</a>
        </li>
        <li class="nav-item">
          <a class="nav-link" href="#">Link</a>
        </li>
        <li class="nav-item dropdown">
          <a class="nav-link dropdown-toggle" href="#" id="navbarDropdown" role="button" data-bs-toggle="dropdown" data-bs-target="#navbarDropdown">
            Dropdown
          </a>
          <ul class="dropdown-menu" aria-labelledby="navbarDropdown">
            <li><a class="dropdown-item" href="#">Action</a></li>
          </ul>
        </li>
      </ul>
    </div>
  </div>
</nav>
```

<nav class="navbar navbar-expand-lg navbar-light bg-light">

```
<div class="container-fluid">

    <a class="navbar-brand" href="#">Navbar</a>

    <button class="navbar-toggler" type="button" data-bs-
        toggle="collapse" data-bs-target="#navbarSupportedContent" aria-
        controls="navbarSupportedContent" aria-expanded="false" aria-
        label="Toggle navigation">

        <span class="navbar-toggler-icon"></span>
    </button>

    <div class="collapse navbar-collapse" id="navbarSupportedContent">

        <ul class="navbar-nav me-auto mb-2 mb-lg-0">

            <li class="nav-item">
                <a class="nav-link active" aria-current="page"
                    href="#">Home</a>
            </li>

            <li class="nav-item">
                <a class="nav-link" href="#">Link</a>
            </li>

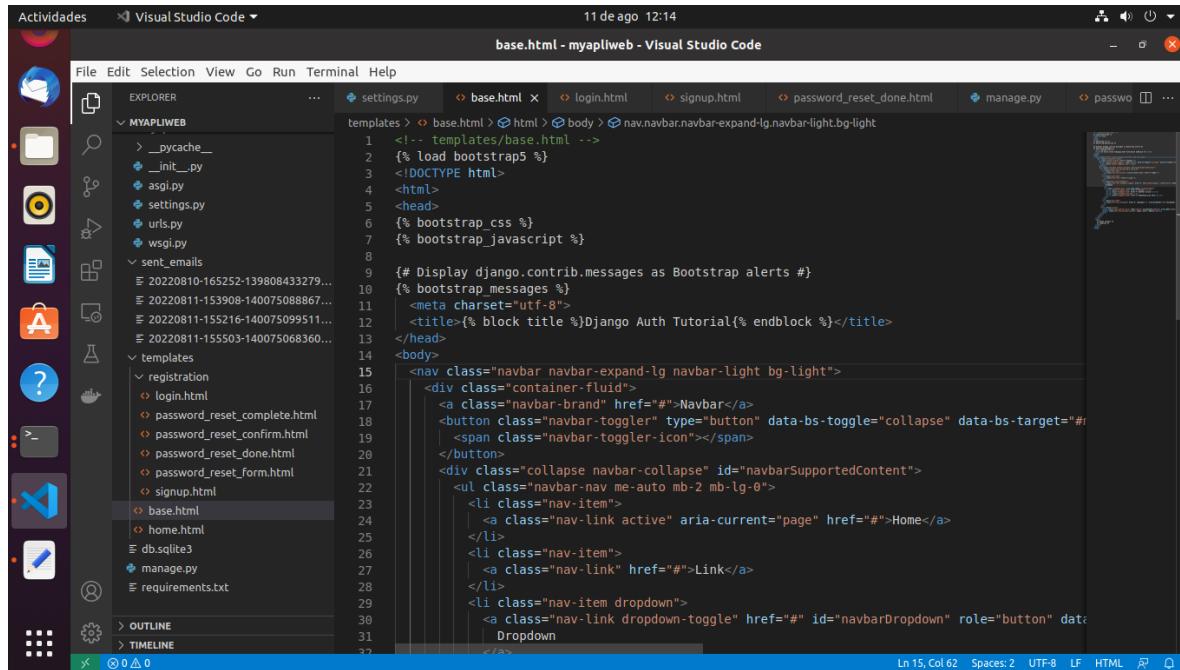
            <li class="nav-item dropdown">
                <a class="nav-link dropdown-toggle" href="#"
                    id="navbarDropdown" role="button" data-bs-toggle="dropdown" aria-
                    expanded="false">
                    Dropdown
                </a>
                <ul class="dropdown-menu" aria-labelledby="navbarDropdown">
                    <li><a class="dropdown-item" href="#">Action</a></li>
                    <li><a class="dropdown-item" href="#">Another
                        action</a></li>
                    <li><hr class="dropdown-divider"></li>
                    <li><a class="dropdown-item" href="#">Something else
                        here</a></li>
                </ul>
            </li>

            <li class="nav-item">
```

```
<a class="nav-link disabled" href="#" tabindex="-1" aria-  
disabled="true">Disabled</a>  
  
</li>  
  
</ul>  
  
<form class="d-flex">  
  
    <input class="form-control me-2" type="search"  
placeholder="Search" aria-label="Search">  
  
    <button class="btn btn-outline-success"  
type="submit">Search</button>  
  
</form>  
  
</div>  
  
</div>  
  
</nav>
```

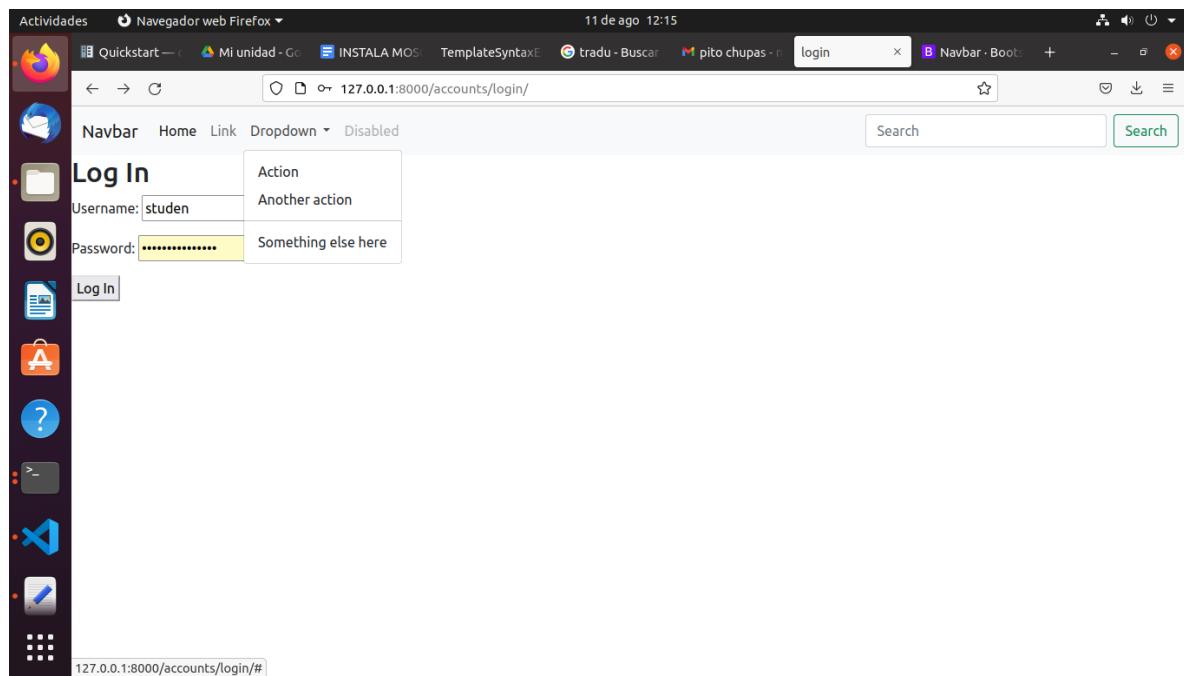
Todo este código se agrega al documento base.html después del `<body>` y refrescamos para ver los resultados.

```
<title>{% block title %}Django Auth Tutorial{% endblock %}</title>  
</head>  
<body>
```



refrescamos nuestra página y se tiene que ver así.

Néstor Emmanuel Briones Ramírez
1220100321 GDS0351 Desarrollo de software multi plataforma.



así podemos agregar estilos y muchas mas posibilidades de acción.