

Django Authentication



Rest_auth

Por medio de estas librerías Python vamos a incluir en nuestros proyectos Django la siguiente funcionalidad:

1. El usuario puede registrarse.
2. El usuario puede iniciar sesión.
3. Autenticación basada en token.
4. El usuario puede cerrar la sesión.



Ahora, agreguemos algunos paquetes a nuestro entorno que manejarán nuestra autenticación:

```
$ pipenv install django-rest-auth django-allauth
```

settings.py

Hemos agregado algunos paquetes nuevos, hagamos que nuestra aplicación Django conozca estos paquetes de autenticación y agreguemos algunas funcionalidades también.

```
# ./bookstore_app/settings.py
...
INSTALLED_APPS = [
    ...
    'rest_framework.authtoken',
    'rest_auth',
    'django.contrib.sites',
    'allauth',
    'allauth.account',
    'rest_auth.registration',
]
```



settings.py

Debido a que Django intenta enviar un correo electrónico, por ejemplo, cuando un usuario crea una cuenta de correo electrónico, necesitamos agregar una pequeña configuración EMAIL_BACKEND y SITE_ID al archivo settings.py.

Esto imprime el correo electrónico que se enviará a la consola.

```
# ./bookstore_app/settings.py
```

```
EMAIL_BACKEND = 'django.core.mail.backends.console.EmailBackend'
```

```
SITE_ID = 1
```



urls.py

A continuación, agregaremos las URL del paquete rest_auth al archivo urls.py del proyecto. Esto nos da acceso a una gran cantidad de funcionalidades de autenticación.

1. registration/
2. login/
3. logout/
4. user/
5. password/change/
6. password/reset/
7. password/reset/confirm/

```
# ./bookstore_app/urls.py

urlpatterns = [
    ...
    path("", include('rest_auth.urls')),
    path('registration/', include('rest_auth.registration.urls')),
]
```



settings.py

Después de agregar las URL, ahora establezcamos algunos esquemas de autenticación y políticas de permisos.

Migrate

```
$ python manage.py migrate
```

Now, let's run our migrations:

```
# ./bookstore_app/settings.py

REST_FRAMEWORK = {
    # Authentication Scheme
    'DEFAULT_AUTHENTICATION_CLASSES': (
        'rest_framework.authentication.BasicAuthentication',
        'rest_framework.authentication.SessionAuthentication',
        'rest_framework.authentication.TokenAuthentication',
    ),
    # Permission Policies
    'DEFAULT_PERMISSION_CLASSES': [
        'rest_framework.permissions.IsAuthenticated',
    ]
}
```

Api/views.py

Agreguemos algunas restricciones al conjunto de vistas de bienvenida en el archivo views.py para que los usuarios no autorizados no tengan acceso.

```
# ./bookstore_app/api/views.py

from rest_framework.decorators import api_view, permission_classes
from rest_framework.permissions import IsAuthenticated
from django.http import JsonResponse
from django.views.decorators.csrf import csrf_exempt

@api_view(["GET"])
@csrf_exempt
@permission_classes([IsAuthenticated])
def welcome(request):
    content = {"message": "Welcome to the BookStore!"}
    return JsonResponse(content)
```



EndPoints Auth

POST - <http://127.0.0.1:8000/registration/>

POST - <http://127.0.0.1:8000/login/>

POST - <http://127.0.0.1:8000/logout/>

GET - <http://127.0.0.1:8000/api/welcome>

django



POST - http://127.0.0.1:8000/registration/

GET /registration/

HTTP 405 Method Not Allowed
Allow: POST, OPTIONS
Content-Type: application/json
Vary: Accept

```
{  
  "detail": "Method \"GET\" not allowed."  
}
```

[Raw data](#)[HTML form](#)

Username

Email

Password1

Password2

POST



POST - http://127.0.0.1:8000/registration/

Register

Register

OPTIONS

POST /registration/

HTTP 201 Created

Allow: POST, OPTIONS

Content-Type: application/json

Vary: Accept

```
{  
  "key": "0d015da93d209c92174577670dbf5c4c4ab6c96f"  
}
```

Raw data

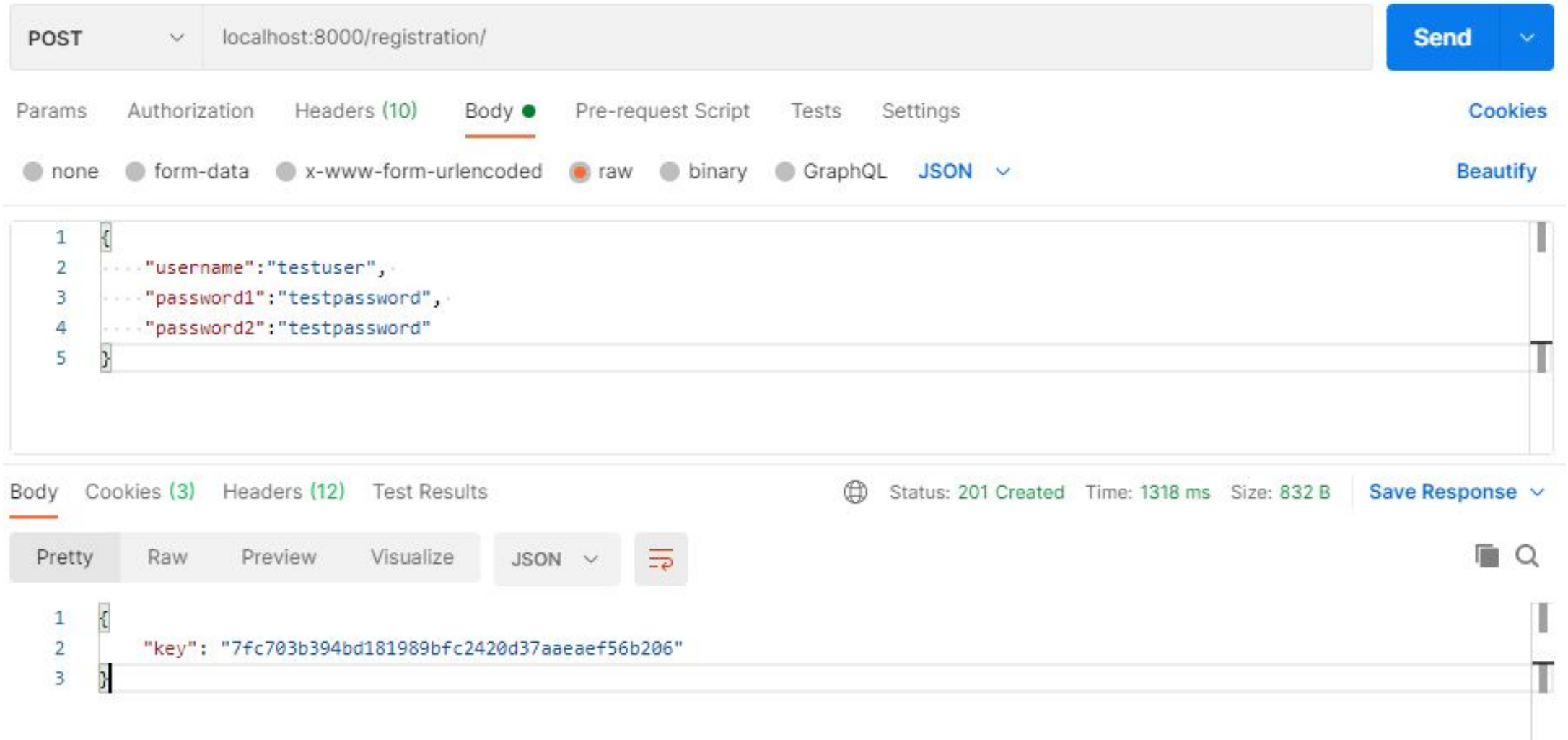
HTML form

Key

0d015da93d209c92174577670dbf5c4c4ab6c96f

POST

POST - http://127.0.0.1:8000/registration/




The screenshot displays a REST client interface with a POST request to `localhost:8000/registration/`. The request body is a JSON object with three fields: `username`, `password1`, and `password2`, all set to `testuser`, `testpassword`, and `testpassword` respectively. The response is a JSON object with a single field `key` containing a long alphanumeric string. The interface includes tabs for Params, Authorization, Headers, Body, Pre-request Script, Tests, Settings, Cookies, and Beautify. The response status is 201 Created, with a time of 1318 ms and a size of 832 B.


POST `localhost:8000/registration/` **Send**

Params Authorization Headers (10) **Body** Pre-request Script Tests Settings Cookies

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL **JSON** [Beautify](#)

```
1 {
2   ... "username": "testuser",
3   ... "password1": "testpassword",
4   ... "password2": "testpassword"
5 }
```

Body Cookies (3) Headers (12) Test Results  Status: 201 Created Time: 1318 ms Size: 832 B [Save Response](#)

Pretty Raw Preview Visualize **JSON** 

```
1 {
2   "key": "7fc703b394bd181989bfc2420d37aaeaf56b206"
3 }
```

POST -

http://127.0.0.1:8000/login/

Login

Check the credentials and return the REST Token if the credentials are valid and authenticated.
Calls Django Auth login method to register User ID in Django session framework

Accept the following POST parameters: username, password
Return the REST Framework Token Object's key.

GET /login/

HTTP 405 Method Not Allowed
Allow: POST, OPTIONS
Content-Type: application/json
Vary: Accept

```
{  
  "detail": "Method \"GET\" not allowed."  
}
```

Username

Email

Password

POST -

http://127.0.0.1:8000/logout/

Logout

Logout

Calls Django logout method and delete the Token object assigned to the current User object.

Accepts/Returns nothing.

GET /logout/

HTTP 405 Method Not Allowed
Allow: GET, POST, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

```
{  
  "detail": "Method \"GET\" not allowed."  
}
```

Media type:

application/json

Content:

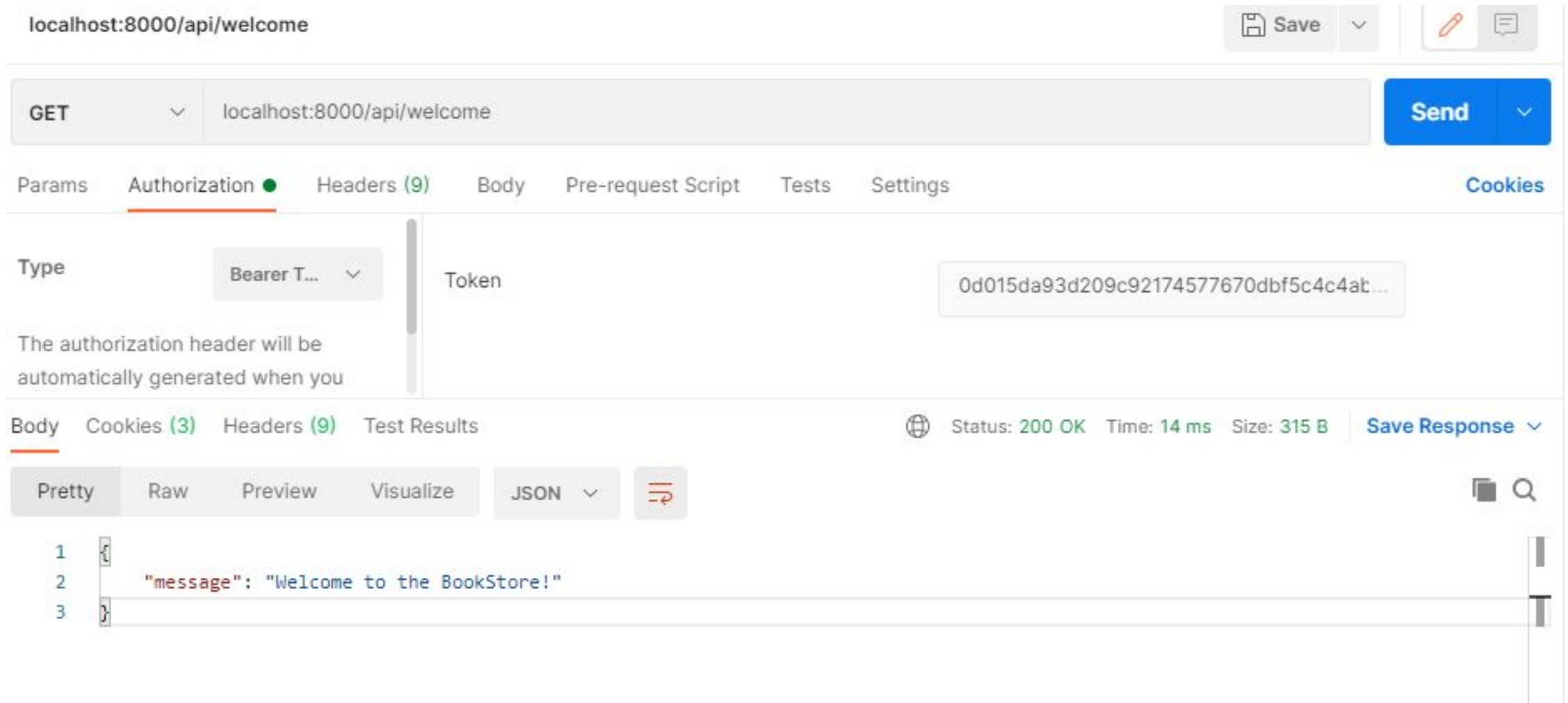
GET - <http://127.0.0.1:8000/api/welcome>



The screenshot shows a JSON viewer interface with a dark theme. At the top, there are three tabs: 'JSON' (which is selected and highlighted in blue), 'Datos sin procesar', and 'Cabeceras'. Below the tabs is a toolbar with five buttons: 'Guardar', 'Copiar', 'Contraer todo', 'Expandir todo', and 'Filtrar JSON' (which includes a funnel icon). The main area displays the JSON response: `message: "Welcome to the BookStore!"`. The text is color-coded: 'message:' is blue, and the string value is in pink.



GET - http://127.0.0.1:8000/api/welcome



localhost:8000/api/welcome

Save

GET localhost:8000/api/welcome Send

Params Authorization Headers (9) Body Pre-request Script Tests Settings Cookies

Type Bearer T... Token 0d015da93d209c92174577670dbf5c4c4at...

The authorization header will be automatically generated when you

Body Cookies (3) Headers (9) Test Results Status: 200 OK Time: 14 ms Size: 315 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "message": "Welcome to the BookStore!"
3 }
```



```
Terminal: Local x +
[15/May/2021 13:46:14] "GET /login/ HTTP/1.1" 405 8033
[15/May/2021 13:46:14] "GET /static/rest_framework/js/jquery-3.5.1.min.js HTTP/1.1" 304 0
[15/May/2021 13:46:14] "GET /static/rest_framework/js/bootstrap.min.js HTTP/1.1" 304 0
[15/May/2021 13:46:14] "GET /static/rest_framework/js/ajax-form.js HTTP/1.1" 304 0
[15/May/2021 13:46:14] "GET /static/rest_framework/js/csrf.js HTTP/1.1" 304 0
[15/May/2021 13:46:14] "GET /static/rest_framework/js/prettify-min.js HTTP/1.1" 304 0
[15/May/2021 13:46:14] "GET /static/rest_framework/js/default.js HTTP/1.1" 304 0
Method Not Allowed: /logout/
[15/May/2021 13:46:45] "GET /logout/ HTTP/1.1" 405 7017
[15/May/2021 13:47:42] "GET /api/welcome HTTP/1.1" 200 40
Method Not Allowed: /registration/
[15/May/2021 13:48:47] "GET /registration/ HTTP/1.1" 405 40
[15/May/2021 13:49:59] "POST /registration/ HTTP/1.1" 201 50
Forbidden: /registration/
[15/May/2021 13:52:10] "POST /registration/ HTTP/1.1" 403 58
Forbidden: /registration/
[15/May/2021 13:52:40] "POST /registration/ HTTP/1.1" 403 58
Not Found: /api/welcome/
[15/May/2021 13:54:20] "GET /api/welcome/ HTTP/1.1" 404 3515
[15/May/2021 13:54:41] "GET /api/welcome HTTP/1.1" 200 40
```