

## 1. Using Django's Built-in Views (`rest_framework.authtoken`)


Django REST Framework (DRF) provides an easy way to set up token-based authentication. Use the `ObtainAuthToken` class to create a login endpoint.

### Steps:

1. Install DRF:

```
bash

pip install djangorestframework
```

 Copy code


2. Add `rest_framework` and `rest_framework.authtoken` to `INSTALLED_APPS`.
3. Create a token-based login API.

### Code Example:

```
python

from rest_framework.authtoken.views import ObtainAuthToken
from rest_framework.authtoken.models import Token
from rest_framework.response import Response


class CustomAuthToken(ObtainAuthToken):
    def post(self, request, *args, **kwargs):
        serializer = self.get_serializer(data=request.data)
        serializer.is_valid(raise_exception=True)
        user = serializer.validated_data['user']
        token, created = Token.objects.get_or_create(user=user)
        return Response({
            'token': token.key,
            'user_id': user.pk,
            'email': user.email,
        })
```

 Copy code



## URLs:

python

 Copy code

```
from django.urls import path
from .views import CustomAuthToken

urlpatterns = [
    path('api/login/', CustomAuthToken.as_view(), name='api_login'),
]
```

## 2. Using Django Sessions


Django's session framework can be used to handle login with session-based authentication.

### Steps:

1. Use the `authenticate` and `login` functions from `django.contrib.auth`.

### Code Example:

python


 Copy code

```
from django.contrib.auth import authenticate, login
from django.http import JsonResponse
from django.views.decorators.csrf import csrf_exempt
import json

@csrf_exempt
def login_view(request):
    if request.method == 'POST':
        data = json.loads(request.body)
        username = data.get('username')
        password = data.get('password')
        user = authenticate(username=username, password=password)
        if user is not None:
            login(request, user)
            return JsonResponse({'message': 'Login successful!'}, status=200)
        else:
            return JsonResponse({'error': 'Invalid credentials'}, status=400)
    return JsonResponse({'error': 'Invalid request method'}, status=405)
```

#### URLs:

python

 Copy code

```
from django.urls import path
from .views import login_view

urlpatterns = [
    path('api/login/', login_view, name='session_login'),
]
```


### 3. Using JWT (JSON Web Token)

JWT is a popular method for stateless authentication.

#### Steps:

1. Install `django-rest-framework-simplejwt`:

bash

 Copy code


```
pip install django-rest-framework-simplejwt
```

2. Update `INSTALLED_APPS` with `rest_framework`.

#### Code Example:

Update settings:


python

 Copy code

```
REST_FRAMEWORK = {
    'DEFAULT_AUTHENTICATION_CLASSES': (
        'rest_framework_simplejwt.authentication.JWTAuthentication',
    ),
}
```

Add the login endpoint:

python

 Copy code


```
from rest_framework_simplejwt.views import TokenObtainPairView, TokenRefreshView

urlpatterns = [
    path('api/token/', TokenObtainPairView.as_view(), name='token_obtain_pair'),
    path('api/token/refresh/', TokenRefreshView.as_view(), name='token_refresh'),
]
```

### Example Request:

POST `/api/token/`


json

 Copy code

```
{
  "username": "your_username",
  "password": "your_password"
}
```

Response:

json

 Copy code


```
{
  "access": "eyJhbGciOiJIUzI1NiIs... ",
  "refresh": "eyJhbGciOiJIUzI1NiIs..."
}
```

## 4. Custom Authentication System

If you need to implement a custom authentication method (e.g., email-based login).

### Code Example:

python

 Copy code

```
from django.contrib.auth import get_user_model, authenticate
from django.http import JsonResponse
from django.views.decorators.csrf import csrf_exempt
import json


User = get_user_model()

@csrf_exempt
def custom_login_view(request):
    if request.method == 'POST':
        data = json.loads(request.body)
        email = data.get('email')
        password = data.get('password')
        try:
            user = User.objects.get(email=email)
        except User.DoesNotExist:
            return JsonResponse({'error': 'User not found'}, status=404)
```

```
        user = authenticate(username=user.username, password=password)
        if user is not None:
            login(request, user)
            return JsonResponse({'message': 'Login successful!'}, status=200)
        return JsonResponse({'error': 'Invalid credentials'}, status=400)
    return JsonResponse({'error': 'Invalid request method'}, status=405)
```

### URLs:

python

 Copy code

```
urlpatterns = [
    path('api/custom-login/', custom_login_view, name='custom_login'),
]
```

## 5. OAuth2 Login (Social Authentication)

Use `django-allauth` or `django-oauth-toolkit` for social login (Google, Facebook, etc.).

### Example Library:

- `django-allauth`
- `django-oauth-toolkit`

## Choosing the Best Approach

- Use **Django sessions** if the application has server-side session management.
- Use **JWT** for a stateless API with client-side token storage.
- Use **OAuth2** for third-party authentication.
- Use **custom methods** for specific business needs.