ঠিক আছে! **Django REST Framework Authentication** তোমাকে একদম **A to Z**, ধাপে ধাপে, beginner → pro লেভেল পর্যন্ত শিখাবো।

🚀 Authentication — A to Z (Full Guide in Bangla)

Authentication মানে হচ্ছে:

> **User কে verify করা—কে API request পাঠাচ্ছে সেটা নিশ্চিত করা।**

DRF এ Authentication + Permission একসাথে কাজ করে:

- **Authentication → তুমি কে?**
- **Permission → তুমি কী করতে পারবে?**

✅ Part–1: DRF Authentication Basic Concept

DRF এ default authentication classes:

1. **SessionAuthentication**
2. **BasicAuthentication**

🔹 1. Session Authentication

- Django login system ব্যবহার করে।
- Browser-based app (HTML forms) এ বেশি ব্যবহৃত।
- JWT লাগে না।

```
REST_FRAMEWORK = {
    "DEFAULT_AUTHENTICATION_CLASSES": [
        "rest_framework.authentication.SessionAuthentication",
    ]
}
```

🔹 2. Basic Authentication

- Request header এ username/password পাঠায়।

- Production এ safe না, mostly testing.

❗ But real project এ সবাই ব্যবহার করে →

🔥 Token Authentication

🔥 JWT Authentication

এগুলোই তুমি শিখবে।

🟦 PART–2: DRF Token Authentication

Step 1: TokenAuth install

```
pip install djangorestframework
pip install djangorestframework-simplejwt
```

Step 2: settings.py এ যুক্ত করো

```
REST_FRAMEWORK = {
    "DEFAULT_AUTHENTICATION_CLASSES": [
        "rest_framework.authentication.TokenAuthentication",
    ]
}
```

Step 3: Token তৈরি করার জন্য signal

models.py এ user তৈরি হলেই token auto তৈরি হবে।

```python
from django.conf import settings
from django.db.models.signals import post_save
from rest_framework.authtoken.models import Token
from django.dispatch import receiver

@receiver(post_save, sender=settings.AUTH_USER_MODEL)
def create_auth_token(sender, instance=None, created=False, **kwargs):
    if created:
        Token.objects.create(user=instance)
```

Step 4: Login API → Token return করবে

```python
from rest_framework.authtoken.views import obtain_auth_token

urlpatterns = [
    path('api/login/', obtain_auth_token),
]
```

Step 5: Protected API

```python
from rest_framework.permissions import IsAuthenticated
from rest_framework.views import APIView
from rest_framework.response import Response

class ProfileView(APIView):
    permission_classes = [IsAuthenticated]

    def get(self, request):
        return Response({"user": request.user.username})
```

Step 6: Request Example

Header

```
Authorization: Token your_token_here
```

TokenAuth clear 😎

🟦 PART–3: JWT Authentication (Most Popular)

সবচেয়ে ব্যবহার হওয়া system
→ Mobile App → SPA → React/Vue → Android → iOS → সবজায়গায় JWT

Step 1: Install JWT

```
pip install djangorestframework-simplejwt
```

Step 2: settings.py এ JWT যোগ করো

```
REST_FRAMEWORK = {
    "DEFAULT_AUTHENTICATION_CLASSES": [
        "rest_framework_simplejwt.authentication.JWTAuthentication",
    ]
}
```

Step 3: URLs

```
from rest_framework_simplejwt.views import (
    TokenObtainPairView,
    TokenRefreshView,
)

urlpatterns = [
    path('api/token/', TokenObtainPairView.as_view()),  # login
    path('api/token/refresh/', TokenRefreshView.as_view()),  # refresh
]
```

Step 4: Login → JWT Access + Refresh token return

POST → /api/token/

Request:

```
{
  "username": "admin",
  "password": "1234"
}
```

Response:

```
{
  "access": "eyJhbGciOiJIUzI1...",
  "refresh": "eyJhbGciOiJIUzI1..."
}
```

Step 5: Protected API

```
class Dashboard(APIView):
    permission_classes = [IsAuthenticated]

    def get(self, request):
        return Response({"msg": "Welcome", "user": request.user.username})
```

Step 6: Auth Header পাঠাতে হবে

```
Authorization: Bearer <ACCESS_TOKEN>
```

🔥 JWT কেন সবাই ব্যবহার করে?

Feature	Explanation
Stateless	Server memory লাগে না
Mobile-friendly	App এ easy integration
Secure	Signature + expiry
Fast	DB query লাগে না

🟧 PART–4: Custom Authentication

তুমি **নিজের authentication system** বানাতে পারো।

Example: API Key, Device ID, Email Token ইত্যাদি।

Example: Custom Header Auth (X-API-KEY)

Step 1: Custom class

```
from rest_framework.authentication import BaseAuthentication
from rest_framework.exceptions import AuthenticationFailed

class APIKeyAuthentication(BaseAuthentication):
    def authenticate(self, request):
        key = request.headers.get('X-API-KEY')

        if key != "secret-key-123":
            raise AuthenticationFailed("Invalid API Key")

        return (None, None)
```

Step 2: View এ use করো

```
class MyView(APIView):
    authentication_classes = [APIKeyAuthentication]

    def get(self, request):
        return Response({"status": "OK"})
```

🟩 PART–5: Full Signup + Login System

তুমি চাইলে আমি complete signup → login → JWT → email verify → password reset সব full code দিয়ে দেবো।

🟦 PART–6: Summary (A to Z)

1️⃣ Basic Auth

2️⃣ Session Auth

3️⃣ Token Auth

4️⃣ JWT Auth (Most important)

5️⃣ Custom Auth

6️⃣ Protected routes (IsAuthenticated)

7️⃣ Permissions + Throttling

8️⃣ Signup/Login full workflow

👍 Next Step

তুমি বলো:

👉 **JWT দিয়ে full signup + login system বানাতে চাও?**
👉 নাকি **TokenAuth project structure** শিখতে চাও?

👉 নাকি **Authentication + Permission + Role-based access** full tutorial চাও?

আমি তোমাকে step-by-step build করিয়ে দেবো।

🔥 ঠিক আছে! এবার তোমাকে Authentication + Permission + Role-Based Access Control (RBAC) এর A to Z Full Tutorial শিখাবো — একদম Beginner → Advanced Level, DRF Standard Code Structure সহ।

🚀 Django REST Framework — Authentication + Permission + Role-Based Access (Full Guide)

এটা পড়লে তুমি যেকোনো বড় project (E-Learning, Job Portal, E-commerce, Admin Panel) এর auth system বানাতে পারবে।

📌 PART–1: Authentication (Who are you?)

DRF এ Authentication মানে হলো → **User কে চিনতে পারা**।

সাধারণত তিনটা Authentication বেশি ব্যবহৃত হয়:

1️⃣ **Session Authentication**
2️⃣ **Token Authentication**
3️⃣ **JWT Authentication (Most Important)**

👉 আমরা JWT দিয়ে সম্পূর্ণ RBAC বানাবো।

📌 PART–2: Permission (Are you allowed?)

Authentication যদি বলে:

> "তুমি কে?"

Permission বলে:

> "তুমি কি করতে পারো?"

DRF এ built-in permission:

- **AllowAny** → সবার জন্য
- **IsAuthenticated** → শুধু logged-in
- **IsAdminUser** → শুধু admin
- **IsAuthenticatedOrReadOnly** → GET allowed, others only for authenticated

---

📌 PART–3: Role-Based Access Control (RBAC)

RBAC মানে:

> **User role অনুযায়ী access control**

Most common roles:

- Admin
- Staff / Moderator
- Customer / Student
- Seller / Teacher
- Guest

আমরা build করবো:

- Admin
- Teacher
- Student

🎯 Final Goal

তুমি এই 3 টি API বানাতে পারবে:

API	Role Allowed
`/admin-dashboard/`	Admin only
`/teacher-panel/`	Admin + Teacher
`/student-dashboard/`	Student + Teacher + Admin

🟦 PART–4: Start Project (Code)

Step 1: Install

```
pip install django djangorestframework djangorestframework-simplejwt
```

Step 2: settings.py

```
INSTALLED_APPS = [
    'rest_framework',
    'rest_framework_simplejwt',
    'users',
]

REST_FRAMEWORK = {
    'DEFAULT_AUTHENTICATION_CLASSES': [
        'rest_framework_simplejwt.authentication.JWTAuthentication',
    ],
    'DEFAULT_PERMISSION_CLASSES': [
        'rest_framework.permissions.IsAuthenticated',
    ]
}
```

🟧 PART–5: Create Custom User Model (with Roles)

Step 1: models.py → role field যুক্ত করি

```
from django.contrib.auth.models import AbstractUser
from django.db import models

class User(AbstractUser):
```

```
    ROLE_CHOICES = (
        ('admin', 'Admin'),
        ('teacher', 'Teacher'),
        ('student', 'Student'),
    )

    role = models.CharField(max_length=10, choices=ROLE_CHOICES,
default='student')

    def __str__(self):
        return f"{self.username} ({self.role})"
```

🟦 PART–6: JWT Authentication URLs

users/urls.py:

```
from django.urls import path
from rest_framework_simplejwt.views import TokenObtainPairView,
TokenRefreshView

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

🟩 PART–7: Permissions (RBAC Logic)

এখন আমরা custom permission বানাবো।

🔥 1. Admin-only Permission

users/permissions.py

```
from rest_framework.permissions import BasePermission

class IsAdmin(BasePermission):
    def has_permission(self, request, view):
        return request.user.is_authenticated and request.user.role ==
'admin'
```

🔥 2. Teacher OR Admin

```
class IsTeacherOrAdmin(BasePermission):
    def has_permission(self, request, view):
        return request.user.is_authenticated and (
            request.user.role == 'teacher' or request.user.role == 'admin'
        )
```

🔥 3. Student Only

```
class IsStudent(BasePermission):
    def has_permission(self, request, view):
        return request.user.is_authenticated and request.user.role ==
'student'
```

🟦 PART–8: Views with Role-Based Access

---

users/views.py

```python
from rest_framework.views import APIView
from rest_framework.response import Response
from .permissions import IsAdmin, IsTeacherOrAdmin, IsStudent

class AdminDashboard(APIView):
    permission_classes = [IsAdmin]

    def get(self, request):
        return Response({"msg": "Welcome Admin!"})


class TeacherPanel(APIView):
    permission_classes = [IsTeacherOrAdmin]

    def get(self, request):
        return Response({"msg": "Teacher or Admin can access"})


class StudentDashboard(APIView):
    permission_classes = [IsStudent]

    def get(self, request):
        return Response({"msg": "Welcome Student!"})
```

🟧 PART–9: URL Setup

users/urls.py

```python
urlpatterns = [
    path('login/', TokenObtainPairView.as_view()),
    path('admin-dashboard/', AdminDashboard.as_view()),
    path('teacher-panel/', TeacherPanel.as_view()),
    path('student-dashboard/', StudentDashboard.as_view()),
]
```

🟥 PART–10: Full Workflow (How it works)

---

1. Student login → access student-dashboard OK

Admin dashboard → Forbidden

2. Teacher login →

Teacher panel access OK
Admin dashboard → Forbidden
Student dashboard → Forbidden

3. Admin login →

Everything allowed (Full access)

🟩 PART–11: BONUS — Add Object Level Permission

— প্রশ্ন: শুধু তার নিজের course modify করতে পারবে?
— কোনো teacher অন্য teacher এর course edit করতে পারবে না?

Example:

```
class IsOwner(BasePermission):
    def has_object_permission(self, request, view, obj):
        return obj.owner == request.user
```

🟦 PART–12: Bonus — Group Based RBAC (Django Groups)

Roles model এ না রেখে তুমি Django Group ব্যবহার করতে পারো:

- Admin group
- Teacher group
- Student group

```
request.user.groups.filter(name='Teacher').exists()
```

🎁 Want Full Project Code?

আমি তোমাকে দিতে পারি:

✅ Full Authentication (JWT)
✅ Signup + Login + Logout
✅ Role-based panel
✅ Admin-only actions
✅ Teacher-only actions
❗ Student-specific dashboard
🔥 Group-based RBAC version
🔥 Permission mixin version
🔥 Class-based + ViewSet সবকিছু

যদি চাও, বলো:

👉 **“Give me full project folder structure + code”**
অথবা
👉 **“Give me signup + login + RBAC with serializers”**

আমি full ready code দিয়ে দেবো।
