

Token Authentication:-

By using token authentication model, user (or) clients are authenticated by using the ^{user} token value. For every user admin person generates a token value and it will be stored inside tokens table. To create the token value inside tokens table, add one default application which name as "rest-framework.auth.token" inside Installed app section in settings.py

For this application purpose, execute make migrations and migrate commands, then only tokens table created inside database

Generating the token value :-

1. create the superuser credentials and login into admin site
2. Admin person can create new users if required any, by click 'Add user'
- Then all user names are stored in users table in admin site
3. To creating the token value for any particular user, then click token table and click add token button, then it displays one dropdown menu which shows all existing users

users 

→ Now select any required user and click save button, the successfully created value for this user

Token	<input type="text"/>	<input type="button" value="Add token"/>
users	<input type="text"/>	<input type="button" value="Add user"/>
emps	<input type="text"/>	<input type="button" value="Add emp"/>

- * Before accessing the rest API urls, a user has to get a token value from the database by sending username & password
- * when we are sending username and password in the form of json request then one built-in view is responsible to get the token value from tokens table by taken user credentials

→ this view name is obtain-auth-token, which is available in

from rest-framework.authToken.Views import obtain_auth_token

* To executing this view we required one url

url(r'gettoken/\$', obtain_auth_token),

* this view is getting the token values from tokens table with respect to username and password

* Now by using this token value we will send Api request to communicate the database

* After obtaining the token value, user has to send this token value with 'Authorization' header to access the restApi.

Eg:- open postman → headers

Enter key as 'Authorization' and value as 'Token token-value'

Q which type of request is required to get the token value from the db?

Ans we are sending 'post' request only to get the token value but not 'GET' request

Q which view is responsible to get the token value from db?

Ans: by using "obtain_auth_token" view available in rest-framework
rest-framework.authToken.Views

Q: How do we get token value from database?

Ans by using ~~username~~ and ~~password~~ 'gettoken' and username & password

Drawbacks of Token Authentication:-

* ~~For~~ for every request, it will check in database for username and password so it create reduce scalability.

project :-

create a project to providing the token authentication to access the apis

step-1: project name : tokenauthenticationpro

step-2: application name : tokenauthenticationapp

step-3: db name : token db

step-4: open settings.py add our app name , rest-framework,
rest-framework-authtoken

```
INSTALLED_APPS = [
```

```
    'django.contrib.auth',
```

```
    'django.contrib.contenttypes',
```

```
    'django.contrib.sessions',
```

```
    'rest_framework',
```

```
]
```

step-5: configure the database in 'settings.py'

step-6: Add (project level security in-s) token authentication settings
by using rest-framework dictionary variable

```
REST_FRAMEWORK = {
```

```
    'DEFAULT_AUTHENTICATION_CLASSES':
```

```
        ('rest_framework.authentication.TokenAuthentication',),
```

```
    'DEFAULT_PERMISSION_CLASSES':
```

```
        ('rest_framework.permissions.IsAuthenticated',),
```

```
}
```

step-7: open models.py

```
from django.models import
```

```
from django.db import models
```

```
class Emp(models.Model):
```

```
empid = models.IntegerField(primary_key=True)
empname = models.CharField(max_length=200)
empsal = models.DecimalField(max_digits=10, decimal_places=2)
created = models.DateTimeField(auto_now_add=True)
modified = models.DateTimeField(auto_now=True)
```

step-8:- open serializers.py

```
from models import Emp
from rest_framework import serializers
class empserializers(serializers.ModelSerializer):
    class Meta:
        model = Emp
        fields = '__all__'
```

step-9:- open views.py

```
from models import Emp
from rest_framework import viewsets
from django import views
from serializers import empserializer
class EmpViewSet(viewsets.ModelViewSet):
    queryset = Emp.objects.all()
    serializer_class = empserializers
    authentication_classes = (SessionAuthentication),
    permission_classes = (IsAdminUser),
```

step-10:- open project level urls

```
from django.contrib import admin
from django.conf.urls import url, include
from rest_framework import auth, views
```

```
urlpatterns = [
```

```
    url('admin/', admin.site.urls),  
    url(r'api/', include('token_authentication_app.urls')),  
    url(r'gettokens/$', views.obtain_auth_token),
```

```
]
```

step-11:-

open app level urls.py

```
from django.conf.urls import url, include  
from .views import EmpViewSet  
from rest_framework.routers import DefaultRouter
```

```
router = DefaultRouter()
```

```
router.register('emp', EmpViewSet)
```

```
urlpatterns = [
```

```
    url(r'', include(routers.urls))
```

```
]
```

step-12: Execute makemigrations, migrate and create superuser and runserver

NOTE:-

→ create one admin user, username: srinivas
password: admin123

→ create another user like active person (not admin) username: harish
password: admin123

→ open postman and access the 'srinivas' related token value

POST	127.0.0.1:8000/api/gettoken
Headers	Body
key	value
username	srinivas
password	admin123

Then we will get similar token value

```
{  
  "token": "fd635786cfd0fgh1239F44"  
}
```

→ select GET method and select url and click on headers

key	value
Authorization	Token fd635786cfd0fgh1239F44

Then we will get successfully data from database

→ Now execute above process with 'naveen' credentials then we are getting message like

```
{  
  "detail": "you do not have permission to perform this action"  
}
```

Bcz, this person is a normal user, but we used permission class as 'IsAdminUser', that's why only admin persons can access APIs