**Simple Steps to Create a REST API Using Django**

**REST Framework in Win10**

**1st Download & Install : for database software**

[**https://www.postgresql.org/ftp/pgadmin/pgadmin4/v9.1/windows/**](https://www.postgresql.org/ftp/pgadmin/pgadmin4/v9.1/windows/) **- Setup,**

**2nd Create new database, Username & Password**

**3rd Run in Background**

**Download Python Form Office Site**

**https://www.python.org/ftp/python/3.13.2/python-3.13.2-amd64.exe**

**Install File & Restart OS Windows**

**Download VS Code & Install : Open VS Code**

*Run : python –version*

*Run : django-admin –version*

**If Not Present, Then you need to install all packages**

*Run :* pip freeze List Show packages || Not Available then install all packages

*Run :* **pip install -r requirements.txt (if Existing Project then Run important)**

*Run :* pip install django

*Run :* pip install djangorestframework

*Run : pip install django\_rest\_framework*

*Run :* pip install psycopg2

*Run :* python manage.py createsuperuser | **for developer id**

***If Security Error Show Then*** *Run : Set-ExecutionPolicy Unrestricted -Scope Process*

**Create main project Folder (anything name)**

*Run :* python -m venv venv (virtue Env cerate )

**1st Create Main Project :** *Run :* **django-admin startproject app**

**Cd app**

**2nd Create Project Module or app :** *Run :***python manage.py startapp crud\_api**

**Under app Folder (api-name-folder or app name)**

*Run : cd* **api-name-folder**

*Run :* venv\Scripts\activate (active)

GoTo main app folder decretory, open **setting.py** and goto section config #

**Application definition- install apps name**

Under add 2 line simple.

'crud\_api',

    'rest\_framework',

save & exit;

After go to database setting: use default

 'ENGINE': 'django.db.backends.postgresql',

        'NAME':'api\_health',

        'USER':'postgres',

        'PASSWORD':'admin',

        'HOST': 'localhost',  # Change if using a remote DB

        'PORT': '5432',  # Default PostgreSQL port

After database credentials add save & exit;

Open **urls.py** And add this, basically api folder root

from django.contrib import admin

from django.urls import path,include

urlpatterns = [

    path('admin/', admin.site.urls),

    path("api/", include("crud\_api.urls")),

]

After save & exit; All setting are done in main project.

**Now Open Api Folder or app folder**

**Open apps.py and view code**

from django.apps import AppConfig

class CrudApiConfig(AppConfig):

    default\_auto\_field = 'django.db.models.BigAutoField'

    name = 'crud\_api'

After save & exit;

**open models.py and Create**

from django.db import models

# Create your models here.

class Person(models.Model):

    name=models.CharField(max\_length=50)

    age=models.IntegerField(default=18)

    city=models.CharField(max\_length=50)

    def \_\_str\_\_(self):

        return self.name

After save & exit;

**Create a black serializers.py page …**

**open serializers.py and edit**

from rest\_framework import serializers

from .models import Person

class PersonSerializer(serializers.ModelSerializer):

    class Meta:

        model=Person

        fields='\_\_all\_\_'

After save & exit;

**open views.py and edit**

from django.shortcuts import render

from rest\_framework.response import Response

from rest\_framework.decorators import api\_view

from .models import Person

from .serializers import PersonSerializer

from rest\_framework import status

from django.db import transaction

# Create your views here.

@api\_view()

def view\_dtl(request):

    return Response({'success': 409, 'message': 'api'})

@api\_view(['GET', 'POST', 'PUT', 'PATCH', 'DELETE'])

def view\_person(request):

    # GET method to retrieve all persons

    if request.method == 'GET':

        person\_obj = Person.objects.all()

        serializer = PersonSerializer(person\_obj, many=True)

        return Response({'msg': 'Successfully retrieved data', 'data': serializer.data}, status=status.HTTP\_200\_OK)

    # POST method to create a new person

    elif request.method == 'POST':

        serializer = PersonSerializer(data=request.data)

        if serializer.is\_valid():

            serializer.save()

            return Response({'msg': 'Person created successfully', 'data': serializer.data}, status=status.HTTP\_201\_CREATED)

        return Response(serializer.errors, status=status.HTTP\_400\_BAD\_REQUEST)

    # PUT method to update a person (full update)

    elif request.method == 'PUT':

        person\_obj = Person.objects.get(pk=request.data.get('id'))

        serializer = PersonSerializer(person\_obj, data=request.data)

        if serializer.is\_valid():

            serializer.save()

            return Response({'msg': 'Person updated successfully', 'data': serializer.data}, status=status.HTTP\_200\_OK)

        return Response(serializer.errors, status=status.HTTP\_400\_BAD\_REQUEST)

    # PATCH method to partially update a person

    elif request.method == 'PATCH':

        person\_obj = Person.objects.get(pk=request.data.get('id'))

        serializer = PersonSerializer(person\_obj, data=request.data, partial=True)

        if serializer.is\_valid():

            serializer.save()

            return Response({'msg': 'Person updated successfully', 'data': serializer.data}, status=status.HTTP\_200\_OK)

        return Response(serializer.errors, status=status.HTTP\_400\_BAD\_REQUEST)

    # DELETE method to delete a person

    elif request.method == 'DELETE':

        person\_obj = Person.objects.get(pk=request.data.get('id'))

        person\_obj.delete()

        return Response({'msg': 'Person deleted successfully'}, status=status.HTTP\_204\_NO\_CONTENT)

    return Response({'msg': 'Invalid request method'}, status=status.HTTP\_405\_METHOD\_NOT\_ALLOWED)

After save & exit;

**Create urls.py and edit**

from django.urls import path

from .views import \*

urlpatterns = [

    path('', view\_dtl, name='dtl'),

    path('person/',view\_person, name='person'),

]

… After save & exit;

**Open admin.py and edit (if You Want Show in Django Admin Panel, Otherwise)**

from django.contrib import admin

from .models import Person

# Register your models here.

admin.site.register(Person)

After save & exit;

cd app

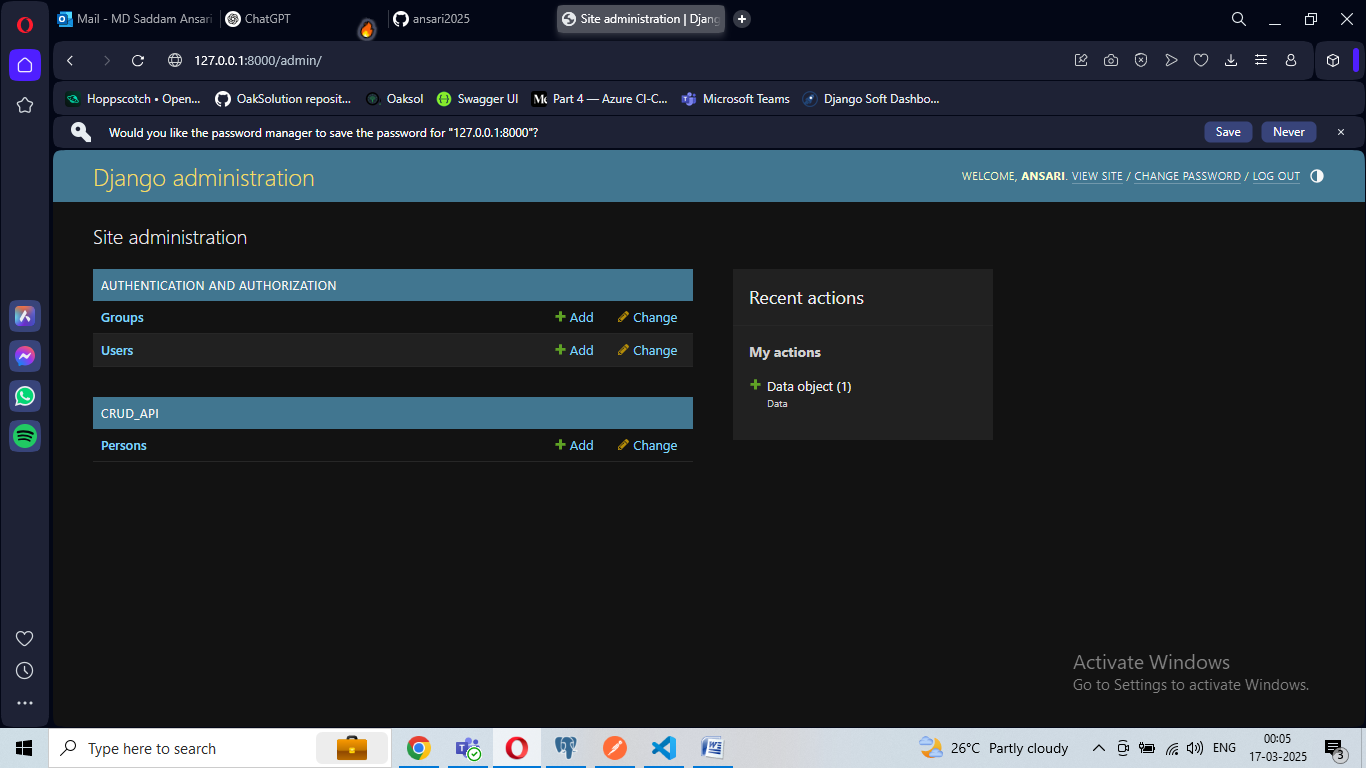
python manage.py makemigrations

python manage.py migrate **every work after**

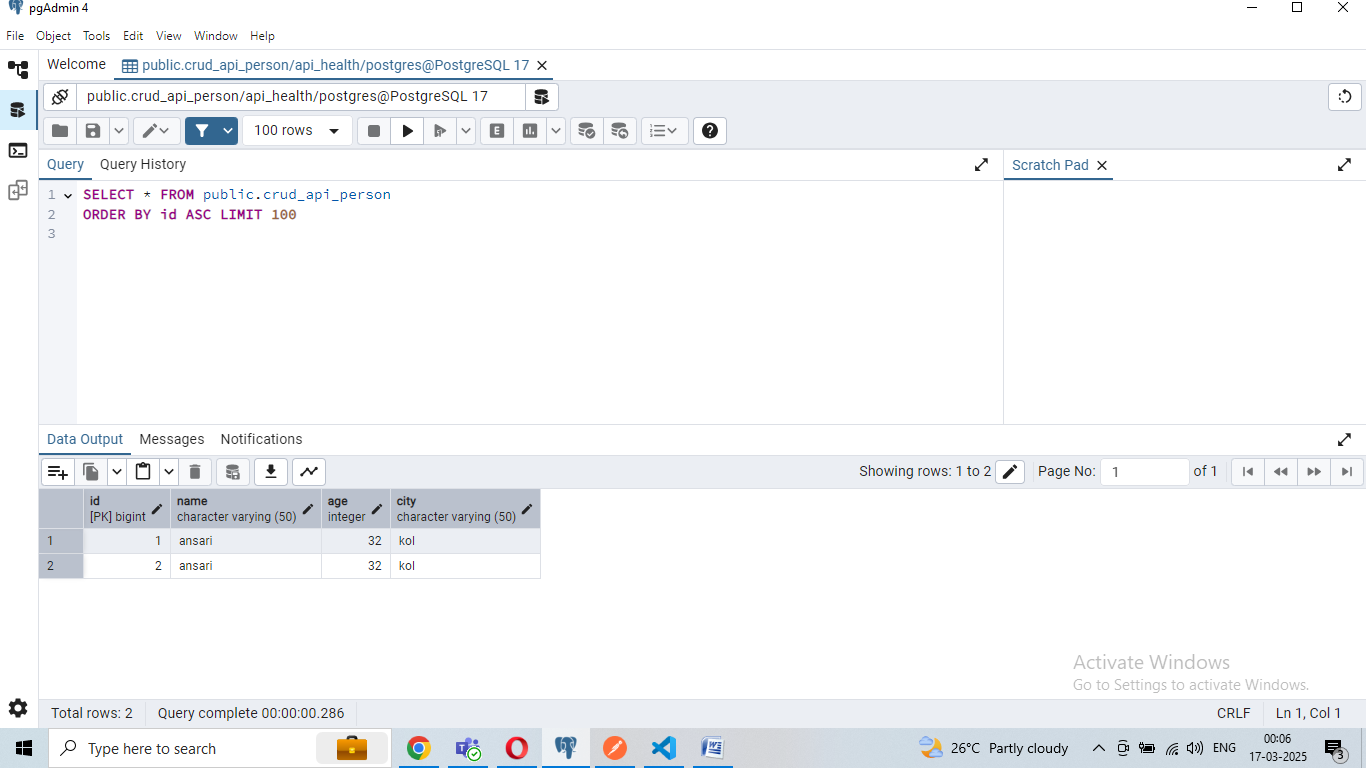
python manage.py runserver 8000 – **Get web address with custom port**

**after run open browser**

**http://127.0.0.1:8000/admin/**

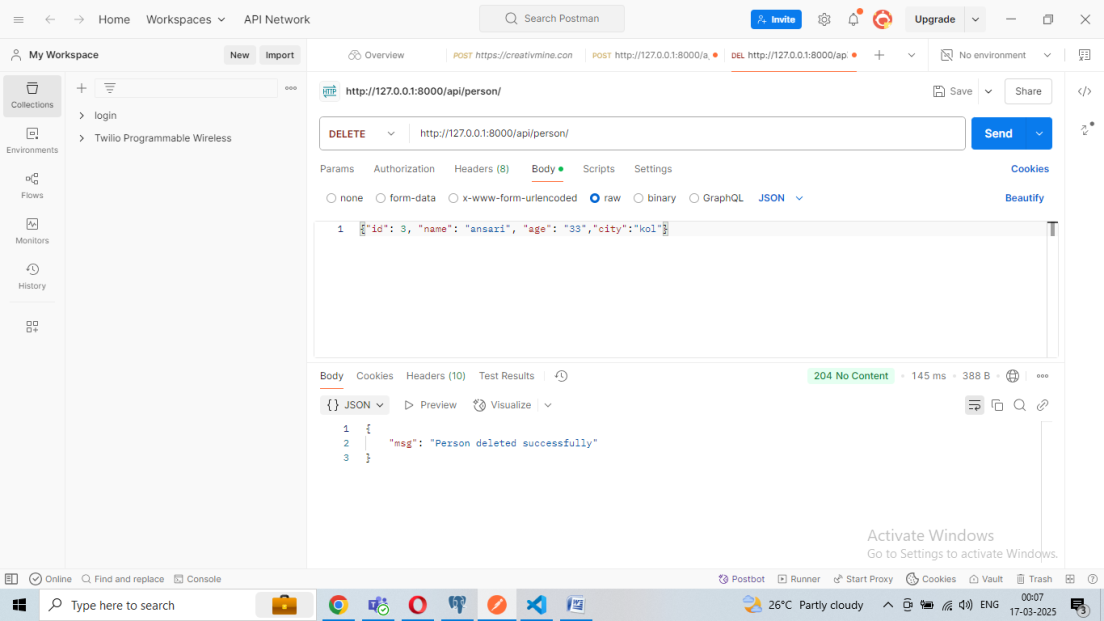
****

**Admin view mode**

****

**Show data in table under database.**

**Url for CRUD API :** <http://127.0.0.1:8000/api/person/>

****

**Now You Can Add , Edit ,Fetch & Delete Same URL , Only Need to Change method**

**Sample Format :** {"id": 3, "name": "ansari", "age": "33","city":"kol"}

**Ref : https://medium.com/@sinturana07/django-post-put-get-delete-requests-example-rest-apis-chapter-19-c449b6260b2a**