Tutorial Python

Cara memanggilurldaricmdtanpaharusmembuka browser:

Python 2

Import urllib

urllib.urlopen('http://127.0.0.1:5000')

pada error handle, jika baris line error yg ditunjukkan as di file kita, bisa jadi line errornya ada di library yang kita pakai.

caramelihat version python yg available di anaconda:

conda search python

caramenggunakan python 3.7.0 di anaconda

1. buat environment barudengancara: conda create --name nama\_enviroment python=3.7.0

2. masukke environment tersebutdengancara: conda activate nama\_enviromentatau activate nama\_enviroment.

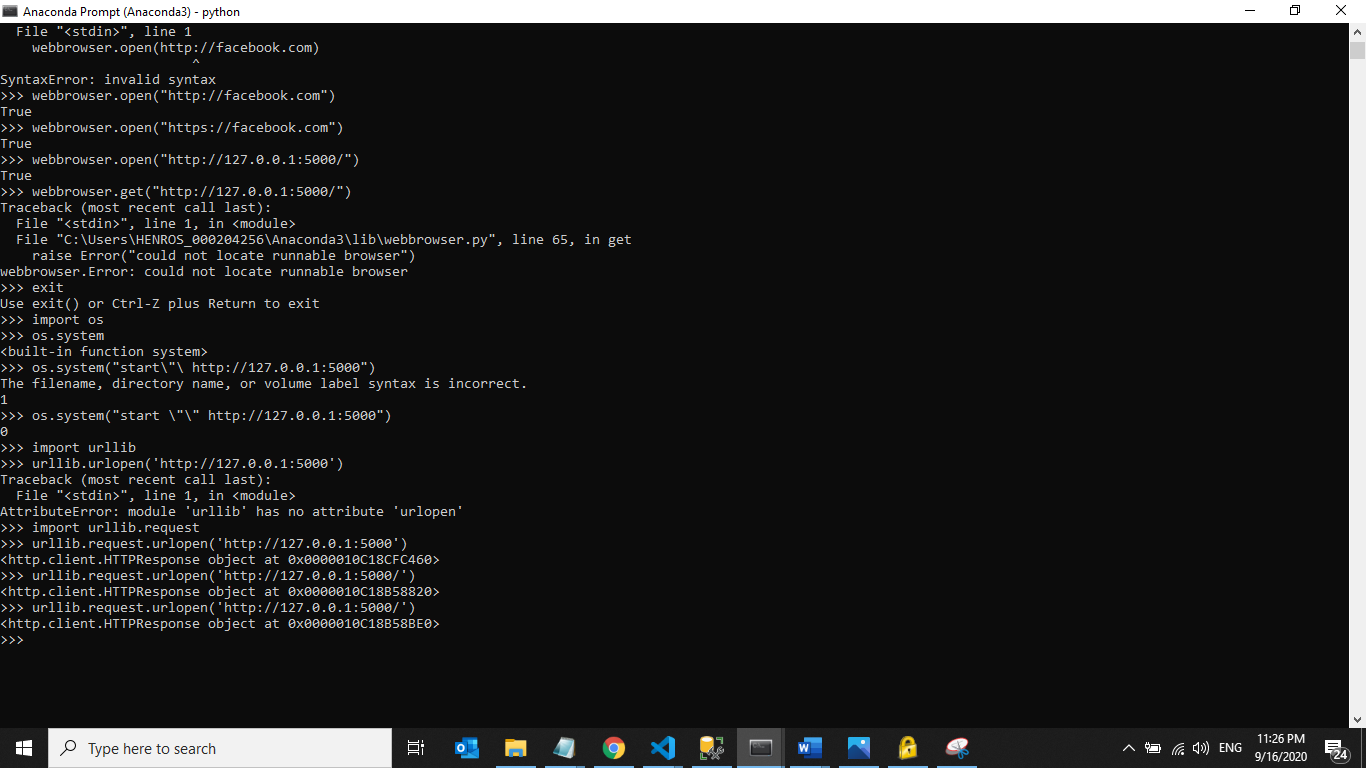
3. setelahmasukkemudian install python 3.7.0 dengancara: conda install python=3.7.0

4. carakeluardarienvironment : deactivate subscribe

python 3

importurllib.request

urllib.request.urlopen('http://127.0.0.1:5000')



Membuat file response untukmenjadi file json di python.

Variable\_response.json()

Contoh:

payload = {'access\_key':'04bfbcf40959b5169e710da06848a577', 'number':['+62 81364037371', '+62 81364037373', '123219302138092183']}

    response = requests.get("http://apilayer.net/api/validate", params = payload)

    return response.json()

menyimpan data kedalam database sql server:

server = '10.0.10.142'

database = 'TELESTATUS\_DEV'

username = 'IT\_SOLWOM'

password = 'Passwd789'

cnxn = pyodbc.connect('DRIVER={ODBC Driver 17 for SQL Server};SERVER='+server+';DATABASE='+database+';UID='+username+';PWD='+ password)

cursor = cnxn.cursor()

query = "INSERT INTO TELESTATUS\_DEV.dbo.tb\_response\_c (no\_telp, status) VALUES ('081364037371', 'valid'"

cursor.execute(query)

cnxn.commit()

perludiperhatikan code terakhiradalahcommit(). Jangansampailupameyertakan commit itu. Sebabkalautidak, nantisaatkitamembuka table darimanajement studio, tidakakanbisamembukanyaatau timeout. karenasebenarnyamasihada proses yang ditungguyaitu command insert kitadari python itubelumdicommit.

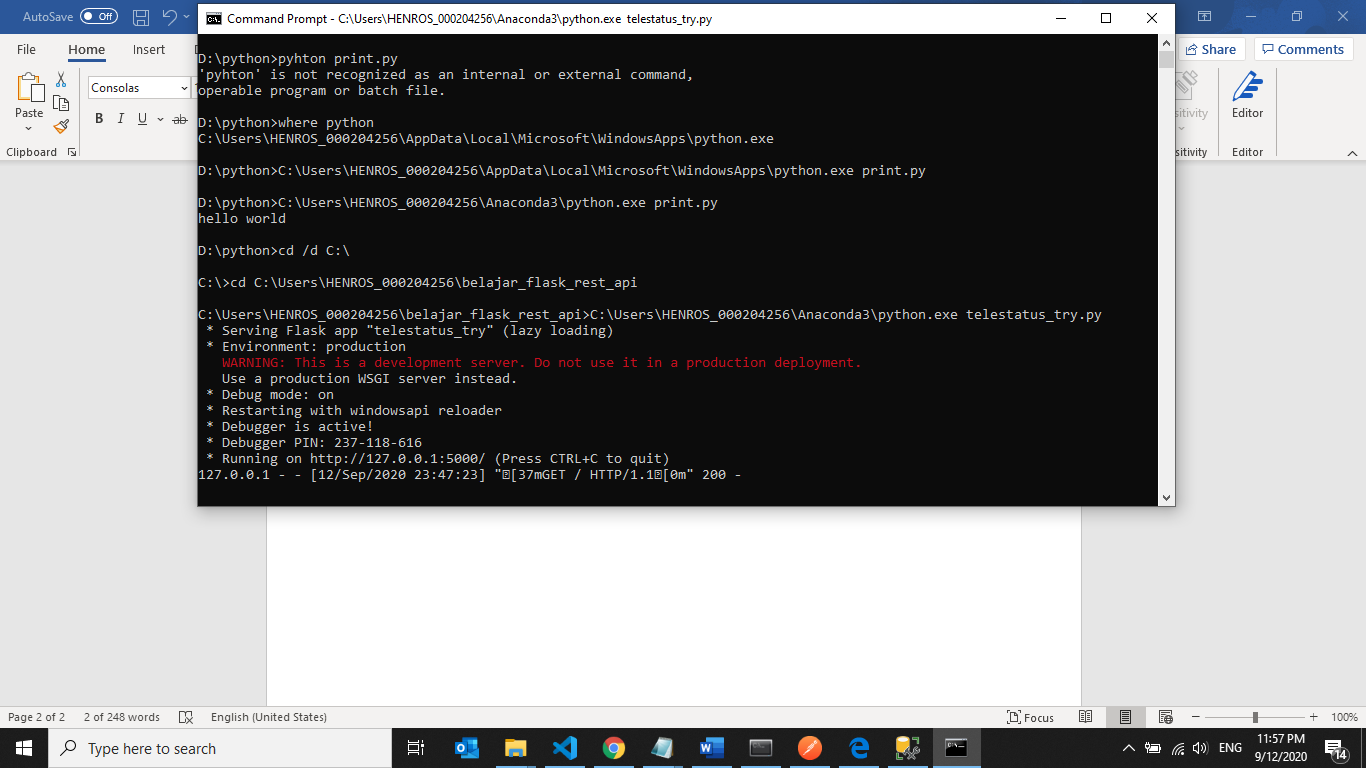
**Penginstalan Python di laptop (windows)**

Jika kitamenginstall python penggunakan anaconda, sebenarnyakitaharusmembuatnya di path environment di windows kita. Namunjikakitabelum setting, makakitabisalangsungmenggunakanperintah python untukmeng execute file python kitadenganmenyertakansemua directory python kita.

Contoh:

Untuk anaconda:

C:\Users\HENROS\_000204256\Anaconda3\python.exe telestatus\_try.py



Bagaimanacaramencari directory python tersebut? Masukketerminal anaconda, kemudianketikkanperintah: where python.

Hasilnyamungkinadabeberapasepertiberiku:

A screenshot of a computer screen

Description automatically generated

Cukuppilihsaja anaconda (yang ditengah).

Cara membuat file exe dari file python.

Kita bisamemakailibrary pyinstaller. Untukmenjalankanperintahnya (**pyinstaller --onefilerun\_test.py**)nanti, untuk laptop iniharusdaridalamcmd anaconda, karena variable environment belum di setting di windows ini.

Function map

Rumus:

map(func, iter)

fun: namasebuah function

iter: parameter ygiterable (bisa di loop). Contoh: list, tuple, dictionary, dll.

contoh:

#fungsi

Def addition(n):

    Return n + n

#iterable

numbers = [1,2,3,4]

#implementasi ke map

result = map(addition, numbers)

#tampilkan

print (list(result))

output:

[2, 4, 6, 8]

Function Lamda

lamdamerupakananinomous function di python.

Lamdabisamemilikilebihdari 1 argument, tapihanyabisamemiliki 1 statement saja.

Contoh:

X = lamda a: a + 10

print (x(5))

output:

15

Y = lamdaa,b,c: a+b+c+d

print (y(1,2,3,4))

output:

10

Decorator

Decorator adalahsebuahfungsiygbergunauntukmembungkusfungsi lain di python. Sama sepertifungsidekorasi, yaitumemodifikasi/mempercantisesuaikeperluan.

Kalau di python, fungsiinimemanggilfungsi lain di dalamparameternya.

def make\_pretty(func):

    def inner():

        print("I got decorated")

        func()

    return inner

def ordinary():

    print("I am ordinary")

kemuidankitajalankan function di atas:

ordinary()

output:

I am ordinary

# Mari kitabuat decorator darifungsi ordinary

pretty = make\_pretty(ordinary)

pretty()

I got decorated

I am ordinary

Pada contoh di atas, make\_pretty() adalahsebuah decorator.

Pada baris perintah pretty = make\_pretty(ordinary), fungsi ordinary didekorasi dan fungsikembaliannyadiberinama pretty.

Di python, kitabisamempersingkatpenulisandengantanda @ diikutidengannama function decoratornya. Kita menempatkannya di atasfungsi yang akandidecorator.

Jadi sepertiini:

@make\_pretty

def ordinary():

print("I am ordinary")

Error Experience:

File "app\_demo\_v\_three.py", line 287, in <module>

telestatus\_check\_v\_one()

File "app\_demo\_v\_three.py", line 161, in telestatus\_check\_v\_one

cursor.execute(query\_seq\_check)

pyodbc.OperationalError: ('08S01', '[08S01] [Microsoft][ODBC Driver 17 for SQL Server]TCP Provider: An existing connection was forcibly closed by the remote host.\r\n (10054) (SQLExecDirectW); [08S01] [Microsoft][ODBC Driver 17 for SQL Server]Communication link failure (10054)')

cara fix:

melakukan looping request. Contoh:

retry\_flag = True

            retry\_count = 0

            while retry\_flag and retry\_count < 5:

                try:

                    cursor.execute(query\_seq\_check)

                    retry\_flag = False

                except:

                    print ("Retry after 1 sec")

                    retry\_count = retry\_count + 1

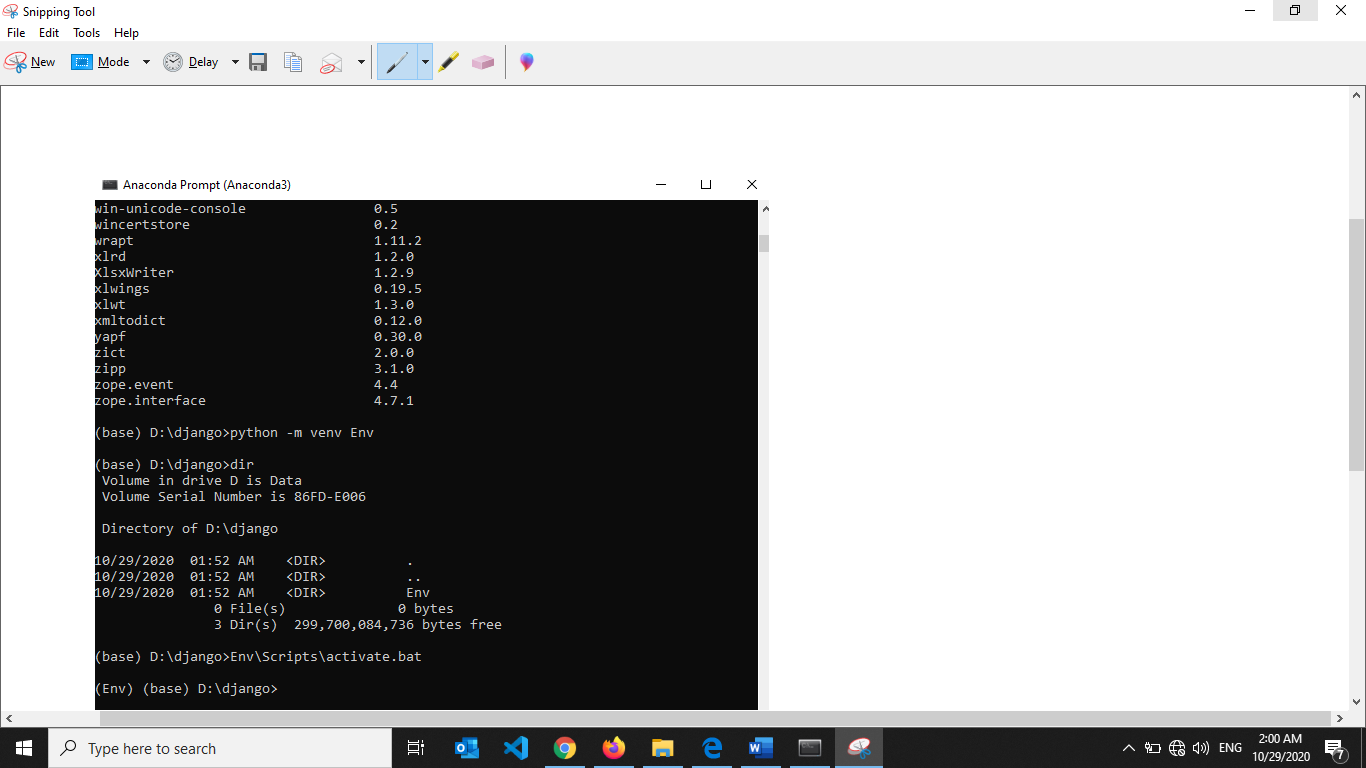
                    time.sleep(1)

caramembuat local environment python

python -m venvnamaEnviroment

caramasukke local enviromenygbaru di buat

namaEnviroment\Scripts\activate.bat



Cara lain masukke virtual environment

env\Scripts\activate

note: 1 enviromentygkitasudahbuat di 1 folder bisakitapakaiuntukbanyak project. Namunlebihbagusmembuat 1 enviromentuntuksetiap project

caramelihat version python yg available di anaconda:

conda search python

caramenggunakan python 3.7.0 di anaconda

1. buat environment barudengancara: conda create --name nama\_enviroment python=3.7.0

2. masukke environment tersebutdengancara: conda activate nama\_enviromentatau activate nama\_enviroment.

3. setelahmasukkemudian install python 3.7.0 dengancara: conda install python=3.7.0

4. carakeluardarienvironment : deactivate subscribe

Cara melihatsemua environment di anaconda:

conda env list

caramenghapussebuah environment di anaconda:

conda env remove --name nama\_enviroment

**tutorial Django**

cara install Django:

setelahmasukke virtual environment,

kemudian install Django dengancara:

pip install Django ,untkversispesifik:

pip3 install Django==2.2.6

kemudian, check apakah Django sudahterinstallataubelumdengancara:

pip list

sedangankanuntuk check versi Django yang terinstal :

django-admin –version

untukmembuat project baru, contohmywebsite:

django-admin startprojectmywebsite

nantiakanadatambah 1 folder dengannamamywebsite

cd kemywebsiteitu. Manage.py itulahygmenjadi server dari Django tersebut.

Kemudiankitabisamenjalankan file tersebutdengancara:

python manage.py runserver.

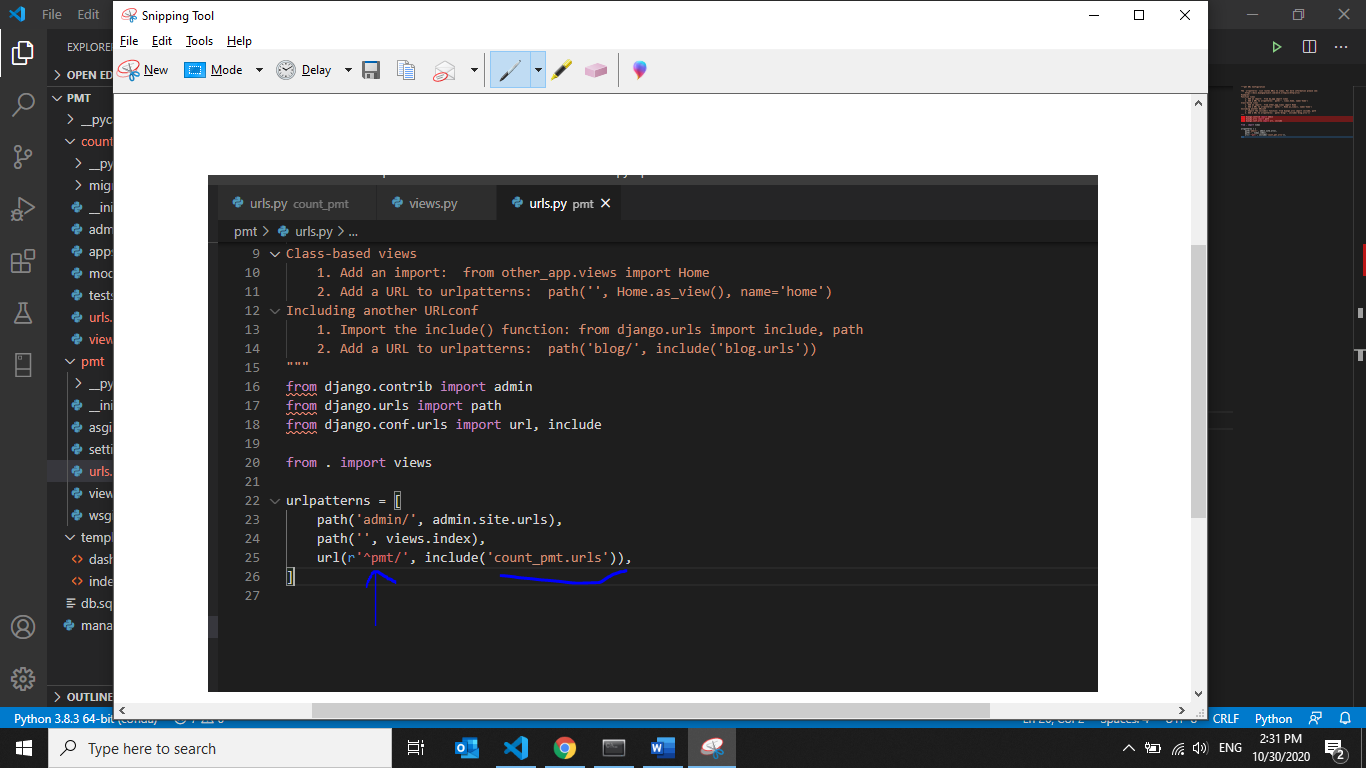
Namunada juga orang ygmenjalankandengancaratanpamengikutkanperintah python di depan:

manage.py runserver

namuncaraini di laptop sayatidakbisa. Sehinggamemakaicarapakaicara :python manage.py runserver.

Ketika kitainginmembuat regex di routing (urls) utama Django, ituharussudahadasambunganlinknya.

Contoh:



Perhatikanulrygditandai di atas. Kita disanabisamenggunakanfungsi regex:

r'^pmt/'

itukarenasudahadaurlygkita include di app(module count\_pmt). Kalautidakada (kitalangsungpanggil function response), kemungkinan link tidakakanjalankitapanggilurlpmttersebut.

Jika inginmembuaturl\_patternsdariurlutama:

Harus ikut / di paling belakang, supaya di diurls apps nyabisatidakmenggunakan /. Hal inipernahmembuataplikasisayatidakjalan.

Contohyg salah:

urlutama:

path('polls', include('polls.urls')),

url app:

path('<int:question\_id>/', views.detail, name='detail'),

2 contohdiatasadalahbersambung. Namunkarenakurangtanda slash (/) antara polls dan <int:….

Makaitutidakakanjalan.

Yang benaritu di urlutamanyaharusditambahtanda slash (/) di paling belakangurlnya.

path('polls/', include('polls.urls')),

adabeberapacarauntuk render tempalate view html kita di Django:

cara 1 denganmenggunakan loader:

from django.shortcuts import render

from django.http import HttpResponse

from .models import Question

from django.template import loader

# Create your views here.

def index(request):

    latest\_question\_list = Question.objects.order\_by('-pub\_date')[:5]

    template = loader.get\_template('polls/index.html')

    context = {

        'latest\_question\_list': latest\_question\_list,

    }

    return HttpResponse(template.render(context, request))

cara 2:

carainilebih simple, kitahanyamenggunakan shortcut render dari module django.shorcuts

from django.shortcuts import render

from .models import Question

def index(request):

    latest\_question\_list = Question.objects.order\_by('-pub\_date')[:5]

    context = {'latest\_question\_list': latest\_question\_list}

    return render(request, 'polls/index.html', context)

caramembuatakses 404 pagenot found di Django

import http404 module darihtpp Django

from django.http import Http404

def detail(request, question\_id):

    try:

        question = Question.objects.get(pk=question\_id)

    except Question.DoesNotExist:

        raise Http404("Question does not exist")

    return render(request, 'polls/detail.html', {'question': question})

penjelasan:

pertamakitaimport Http404 dariDjango.http

di dalamfunctionnyakitabuat handle error dengantry: catch

kita select ketabeldengan id ygkitadapatdariurl dan di assingke variable question. Kemudia variable question kita check apakahhasilnyakosongdengan function Django .DoesNotExist

jikatidakadakitalangsung raise Http404

namunkita juga bisamembuatnyadengancaralebihsingkatmenggunakan shortcuts

from django.shortcuts import get\_object\_or\_404, render

def detail(request, question\_id):

    question = get\_object\_or\_404(Question, pk=question\_id)

    return render(request, 'polls/detail.html', {'question': question})

caramembuaturl di file html tidak hardcoded untukurltemplatenya. Kita bisamenggunakancarasepertiberikut: {% url …%}

contoh hardcoded:

<li><a href="/polls/{{ question.id }}/">{{ question.question\_text }}</a></li>

Contohtidakharcoded:

<li><a href="{% url 'detail' question.id%}">{{ question.question\_text }}</a></li>

`detail` merupakan name ygdiambildari urls.py ygsudahkitatentukan

urlpatterns = [

    # ex: /polls/

    path('', views.index, name='index'),

    # ex: /polls/5/

    path('<int:question\_id>/', views.detail, name='detail'),

]

## **Namespacing URL names**[¶](https://docs.djangoproject.com/en/3.1/intro/tutorial03/#namespacing-url-names)

Kita bisamembuatnyadenganperintah: **app\_name**

Kita tentukan di urls.py

from django.urls import path

from . import views

app\_name = 'polls'

urlpatterns = [

    path('', views.index, name='index'),

    path('<int:question\_id>/', views.detail, name='detail'),

    path('<int:question\_id>/results/', views.results, name='results'),

    path('<int:question\_id>/vote/', views.vote, name='vote'),

]

Kemudian di views.py kitasertakannamespacenya di depanurlnya

Contoh

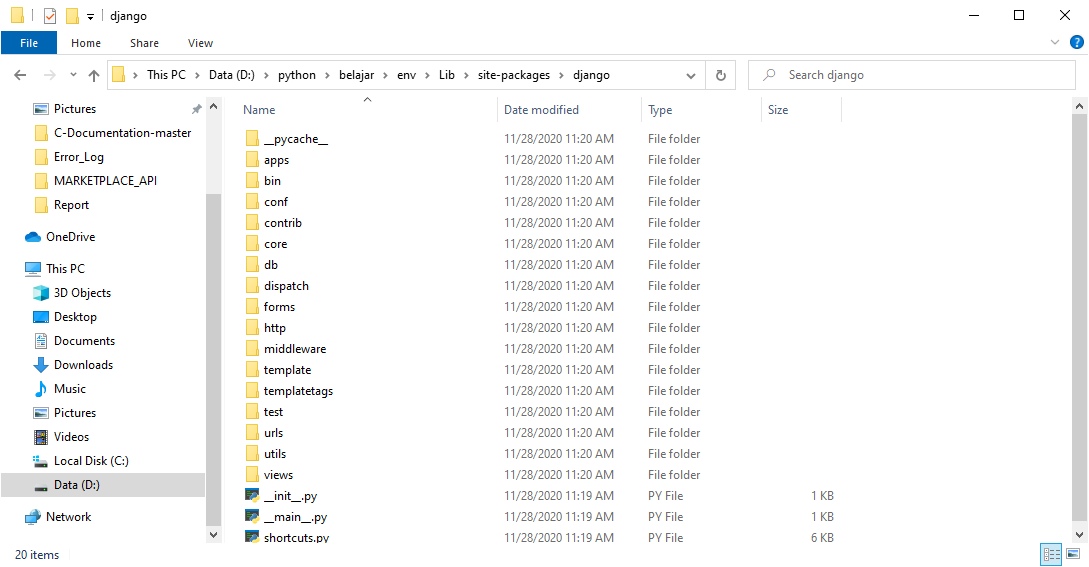
<**li**><**a**href="{%**url**'polls:detail'question.id%}">{{question.question\_text}}</**a**></**li**>

Note: tujuan namespace adalahmenghindari conflict name Ketika kitasudahmemilikibanyak app.

Contohjikaada app Namanya blog, didalam blog ada detail. Makanantiuntukmenujuhalamanyg mana kitamau, urlnyabisa conflict.

mengenai core Django:

ygkita import core darisi Django adalahterdapa di dalam module environment kita



Class dan fungsiygkita import terdapatdidalam folder2 itu.

ContohHttpResponsedari folder(module) http. Contoh import dari script kita:

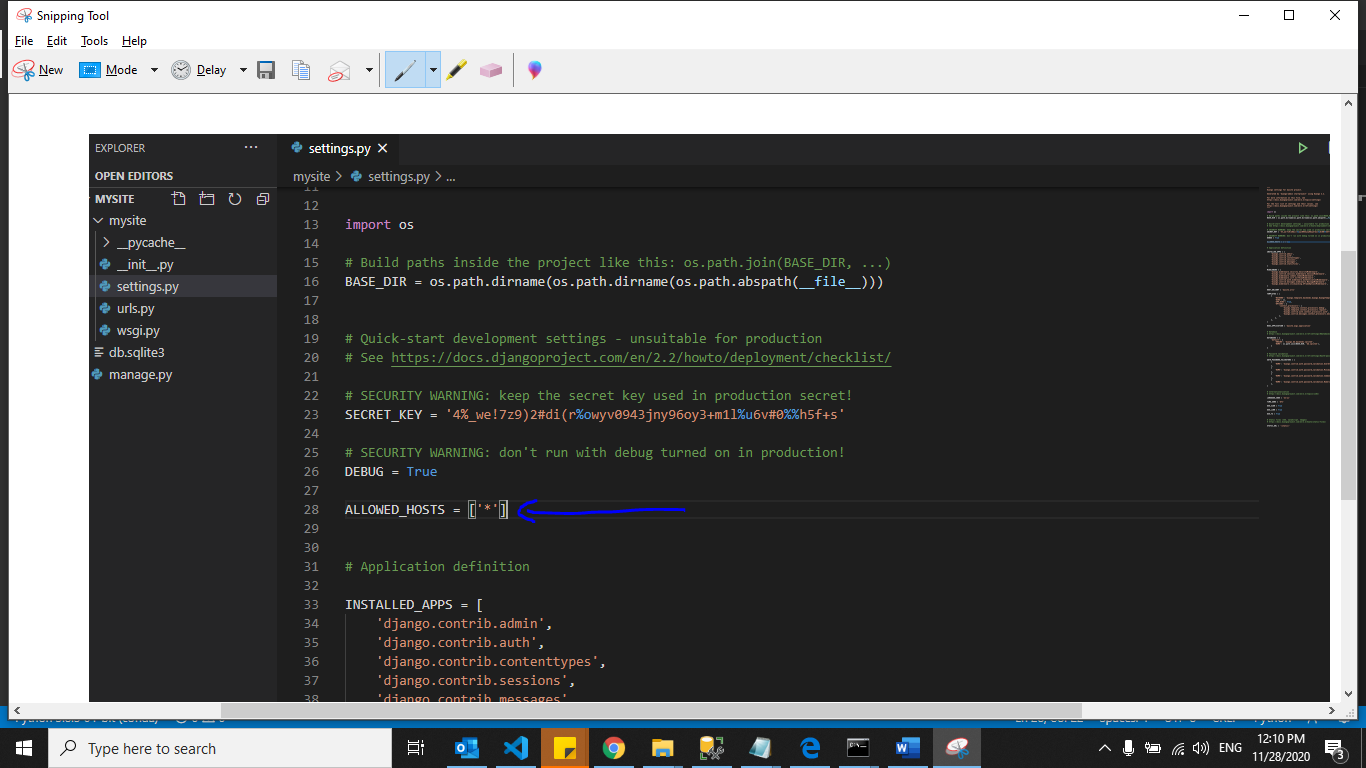
from django.http import HttpResponse

note: djangoadalahnama folder pertamasepertiygkitalihatdiatas

**Cara membuat server Django kitabisadiaksesdari computer lain:**

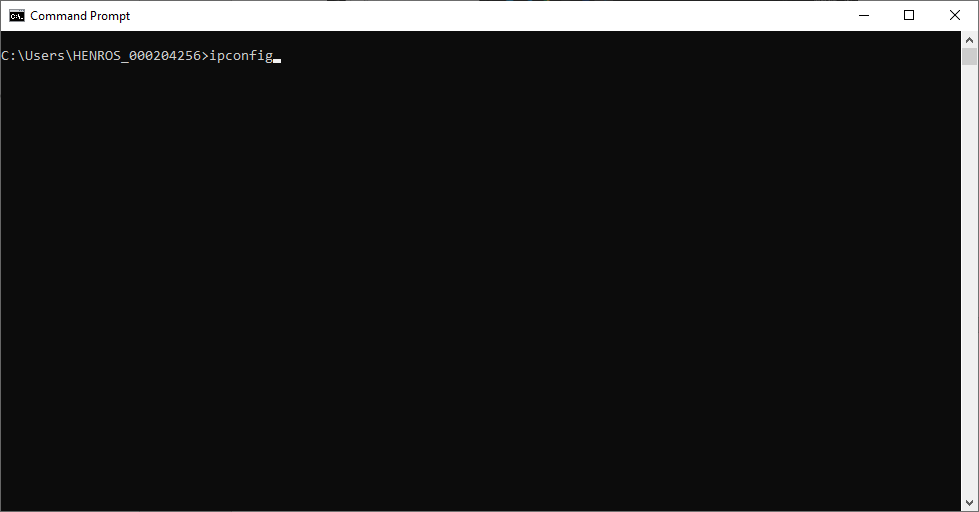
Sebelumkitamengubah(pointing) ipnya, pastikankitaSudahmembuatiptersebut ALLOWOD\_HOST di file settings.py Django:

Root\_folder/settings.py

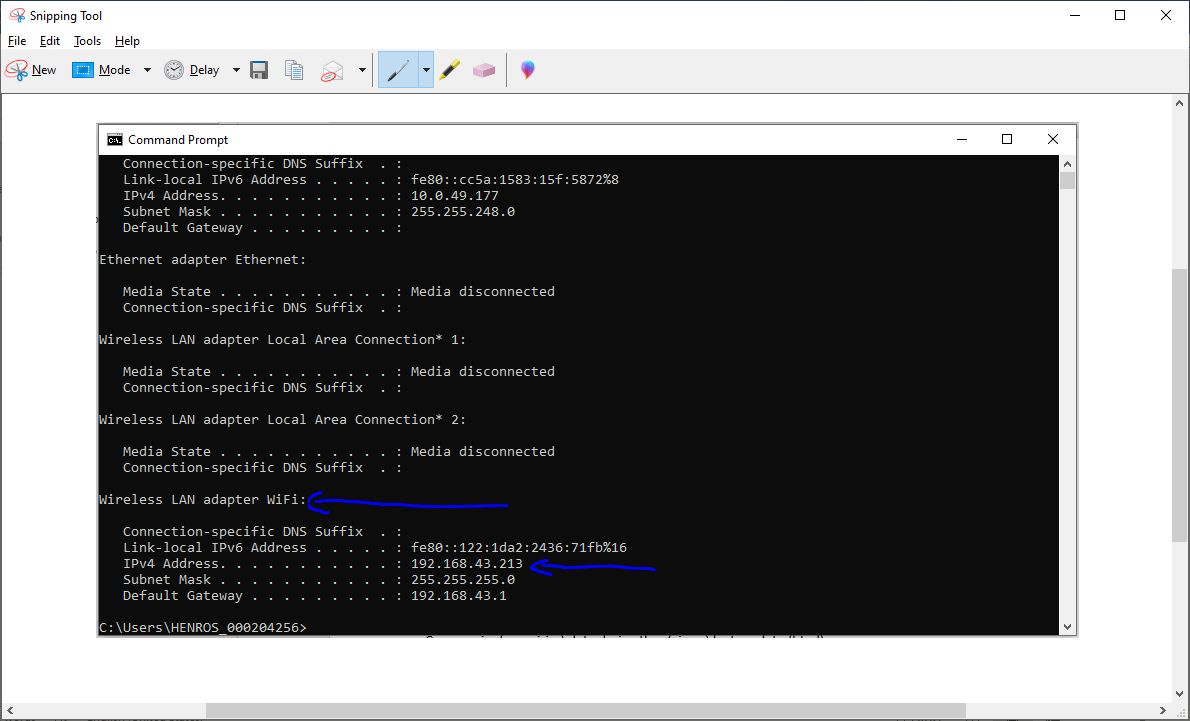


*Note: ‘\*’ :semua host di allow (di ijinkan)*

Caranyaadalahdengankitamenjalankan server dengancaramenyebutkanipkita (public) kemudiannamaportnya. Denganbegitukitaharuscariduluip public kita. Caranyabukacmd ->ketikkan ipconfig



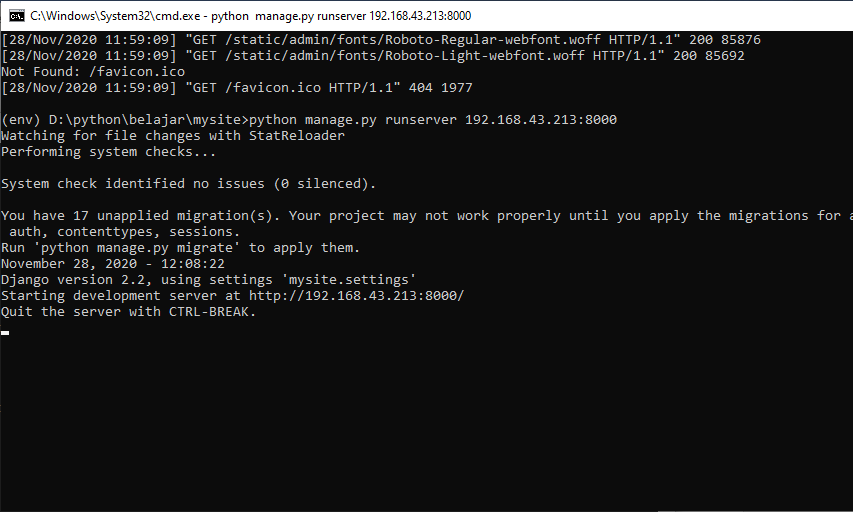
Hasilnyasepertiberikut:



Perhatikanygditandai, yang kitaambiladalahip wireless LAN itu, dan IPv4 address :192.168.43.213

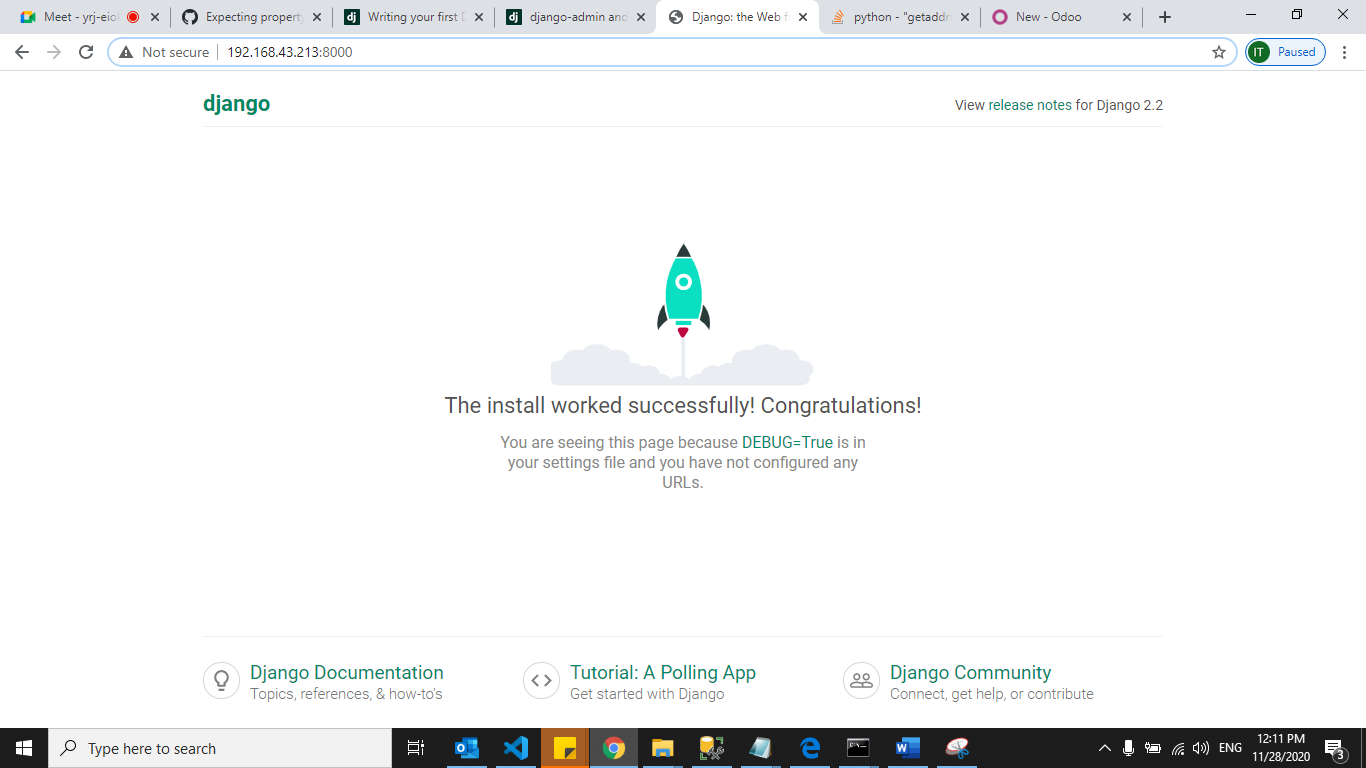
Denganbegitu, kitasudahmendapatkanippublicnya. Makakitabisamenjalankanservernyadengancara:

python manage.py runserver192.168.43.213:8000



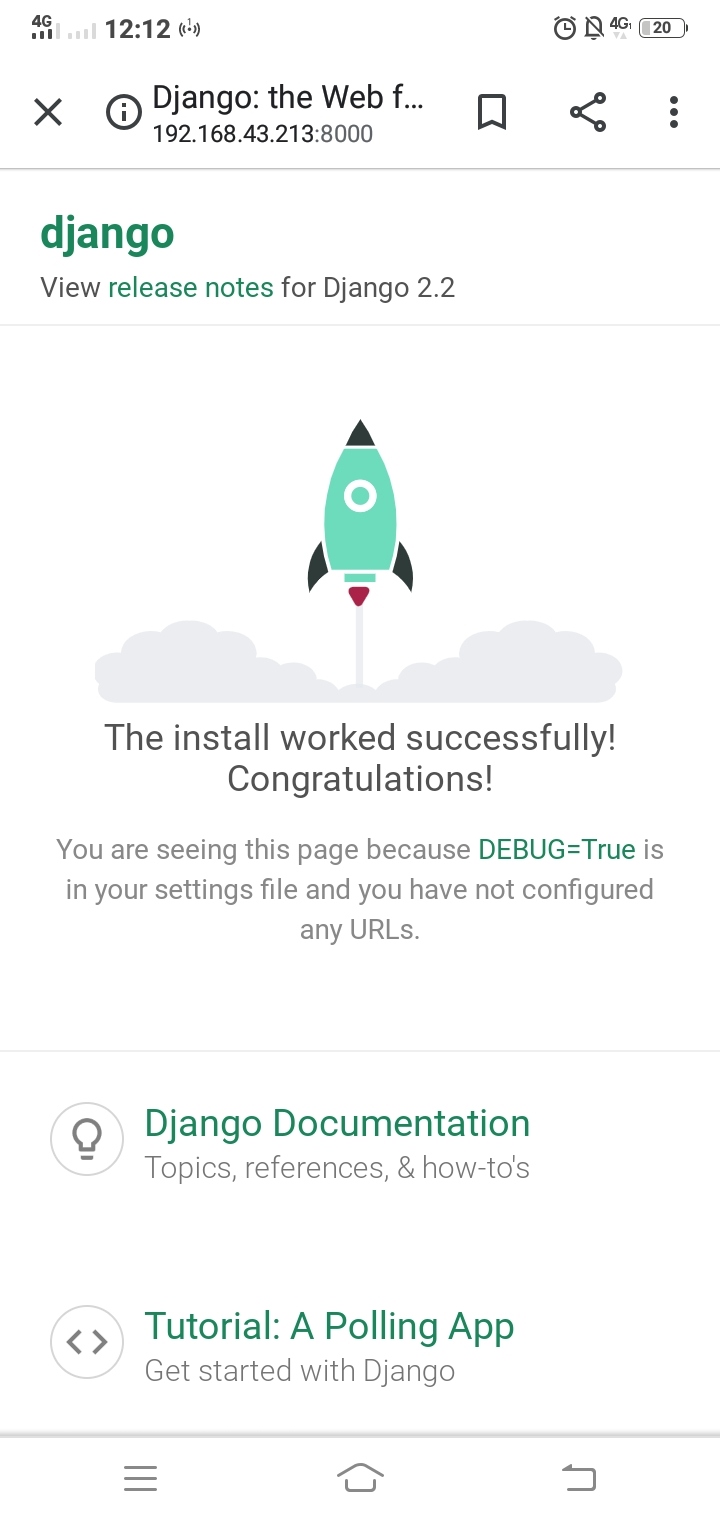
*note :portnyaterserahkita*

jika Sudha jalan,cobajalankan di browser local:



Itutandanyasudahberhasil.

Kemudiancobakita access dari device lain. Contoh di sinisayaakanaksesdari browser hp:



Denganbegitu, carakitasudahberhasil.

Cara parsing(mengirim) data dari python(views) ke template (html).

def index(request):

    context = {

        'judul': 'Count PMT'

    }

    return render(request, 'count\_pmt/index.html', context)

kemudiandipanggil di html dengancara:

{{ judul }}

Contohmembuatperulangan di template (html) di djanggo:

{% for link, name in nav%}

            <li>

                <a href="{{link}}">{{name}}</a>

            </li>

        {% endfor %}

Nav ituadalahhasil parsing dari file .py (views.py)

Documentation Template Tag Django

<https://docs.djangoproject.com/en/3.1/ref/templates/language/#templates>

pembuatancss di Django,

kitacukupletakkan 1 buah file cssnya di folder utamanya.ItuSudahotomatisdipanggilkesetiap app dengancarasepertiinisaja:

<link rel="stylesheet" href="{% static 'css/style.css' %}">

Comment di janggo html adalahsepertiini:

{# #}

Kalautidaksepretiitu, bisaterjadikesalahan. Jika langsungpakai shortcut comment di vs code, bisatidaksesuai.

File setting di Django:

Lokasi: nama\_project/settings.py

Ada beberapahalygbiasakita setting:

Database, installed\_app, sama folder templates

JSONRenderer

Tujuannyauntukmembuatsebuahtampilanmenjaditampilan JSON.

Contohtampilanawal:

from django.shortcuts import render

# Create your views here.

from django.contrib.auth.models import User

from rest\_framework.renderers import JSONRenderer

from rest\_framework.response import Response

from rest\_framework.views import APIView

class UserCountView(APIView):

    """

    A view that returns the count of active users in JSON.

    """

    # renderer\_classes = [JSONRenderer]

    def get(self, request, format=None):

        user\_count = User.objects.all().count()

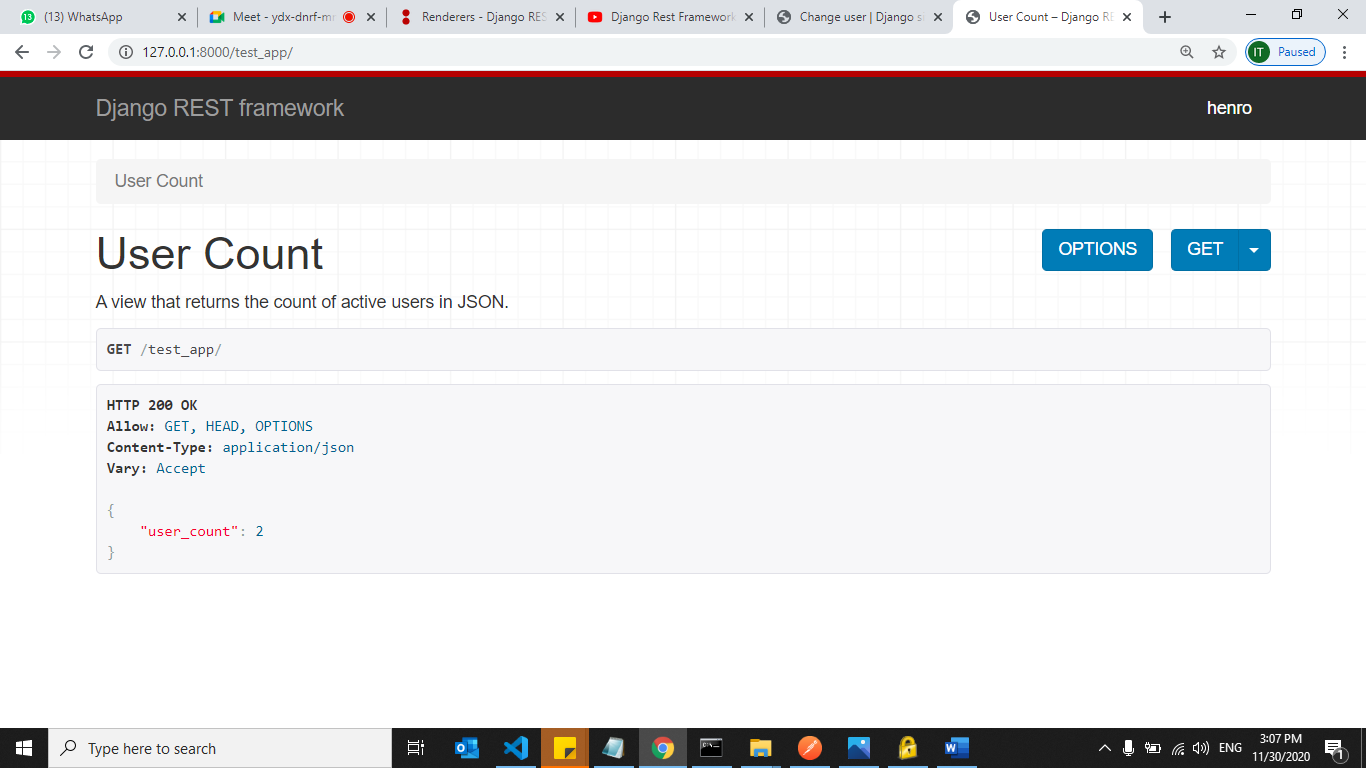
        content = {'user\_count': user\_count}

        print (type(Response(content)))

        print('\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_')

        return Response(content)

kalaukitaperhatikanJSONRenderernyakita remark, jaditampilannyabelum di convert menjadi json. Berikuttampilannya



KalaukitapakaiJSONRenderer, tampilannyamenjadi

from django.shortcuts import render

# Create your views here.

from django.contrib.auth.models import User

from rest\_framework.renderers import JSONRenderer

from rest\_framework.response import Response

from rest\_framework.views import APIView

class UserCountView(APIView):

    """

    A view that returns the count of active users in JSON.

    """

    renderer\_classes = [JSONRenderer]

    def get(self, request, format=None):

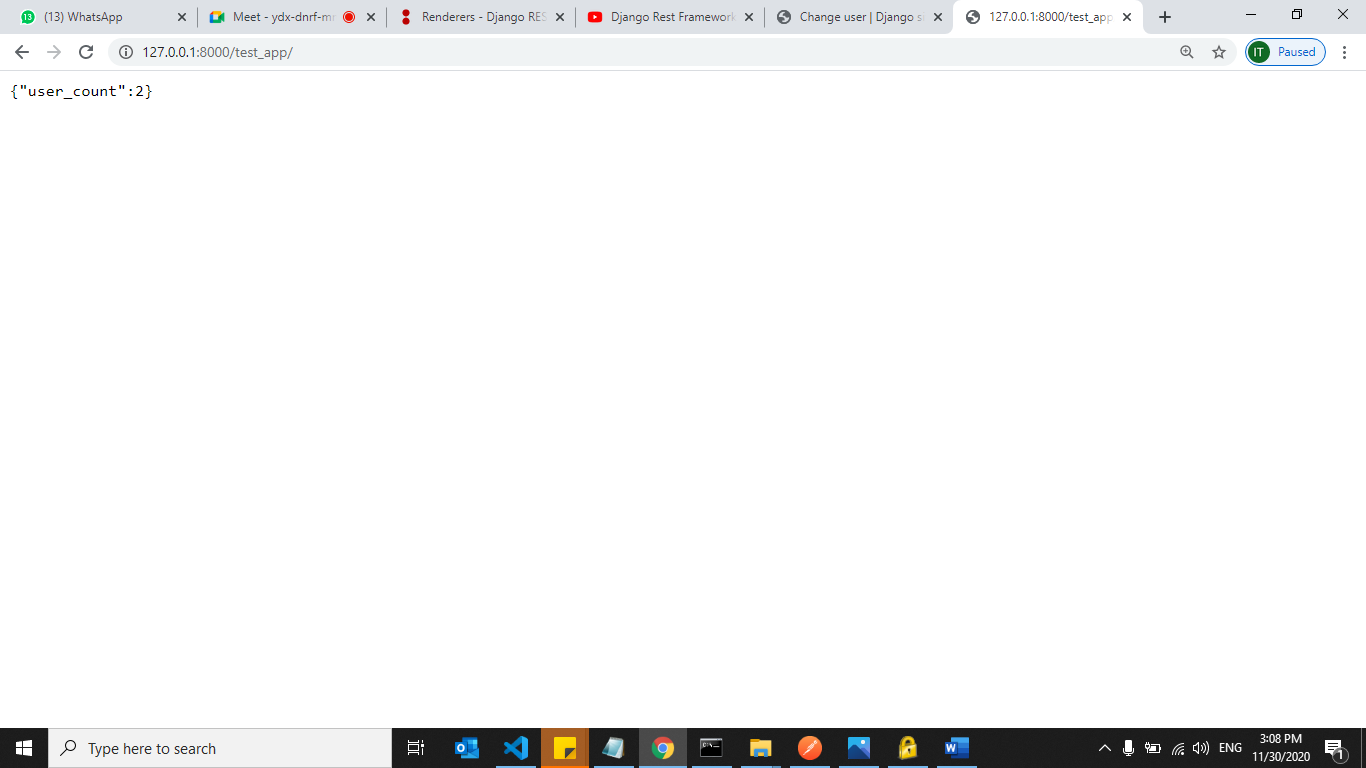
        user\_count = User.objects.all().count()

        content = {'user\_count': user\_count}

        print (type(Response(content)))

        print('\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_')

        return Response(content)

enjadi:

Proses koneksi Django ke database

File setting -> model -> view->templates

Contoh:

Settings.py

DATABASES = {

    'default': {

        'ENGINE': 'django.db.backends.sqlite3',

        'NAME': BASE\_DIR / 'db.sqlite3',

    }

}

Models.py

from django.db import models

# Create your models here.

class Post(models.Model):

    judul = models.CharField(max\_length=200)

    body = models.TextField()

    def \_\_str\_\_(self):

        return "{}".format(self.judul)

views.py

from django.shortcuts import render

# Create your views here.

from .models import Post

def index(request):

    posts = Post.objects.all()

    context = {

        'Title': 'Blog',

        'Heading': 'Blog di mywebsite',

        'Posts': posts

    }

    return render(request, 'blog/index.html', context)

ygterakhiradalah file htmlnyaatauygkitasebutsebagai template. Untktampil html samadengan html yang lain pada umumnya.

Cara membuatkoneksidari Django ke database sql server

1. pip install sql\_server.pyodbc

Sql server saatinitidakterintegrasilangsungdengan Django, sehinggakitaharusmengistallsebuah library yang bernama: **django-mssql-backend ,iniharus di install, kalaungganantibisabikin error Ketika menjalankan server**

Install library tersebutdengancara:

2. pip install django-mssql-backend

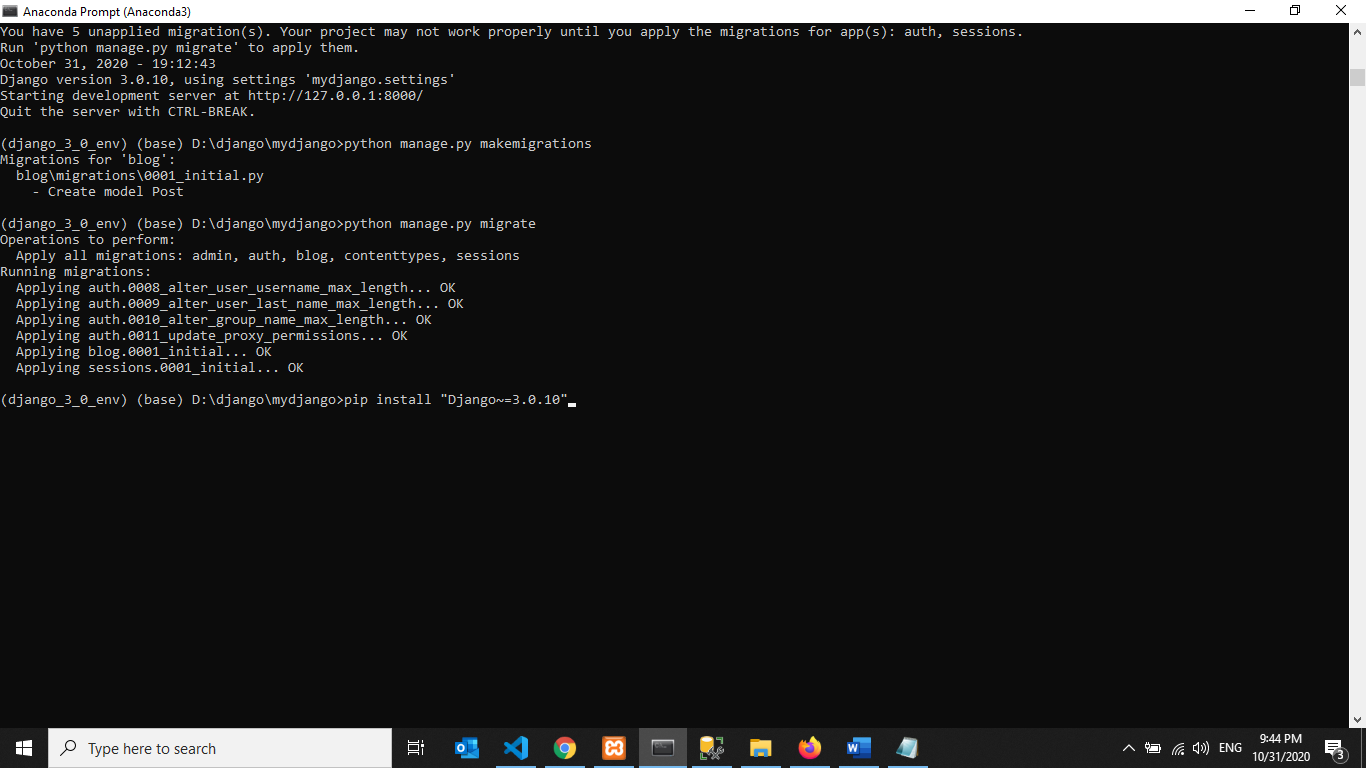
note:

library tersebuthanyasupport(bisadipakai) sampaiversi Django 3.0.10

oleh karenaitupastikankitamenginstall Django versitersebut.

Cara installnya:

pip install "Django~=3.0.10"



Setelah library di atassudahterinstall, kemudiankitaatursettingannya di folder utamanya (main project): folder\_project/settings.py

Setting sepertiini:

DATABASES = {

    'default': {

        'NAME': 'Product',

        'ENGINE': 'sql\_server.pyodbc',

        'HOST': '(localdb)\\MSSQLLocalDB',

        'USER': '',

        'PASSWORD': '',

        'OPTIONS': {

            'driver': 'ODBC Driver 17 for SQL Server',

            'unicode\_results': True,

        },

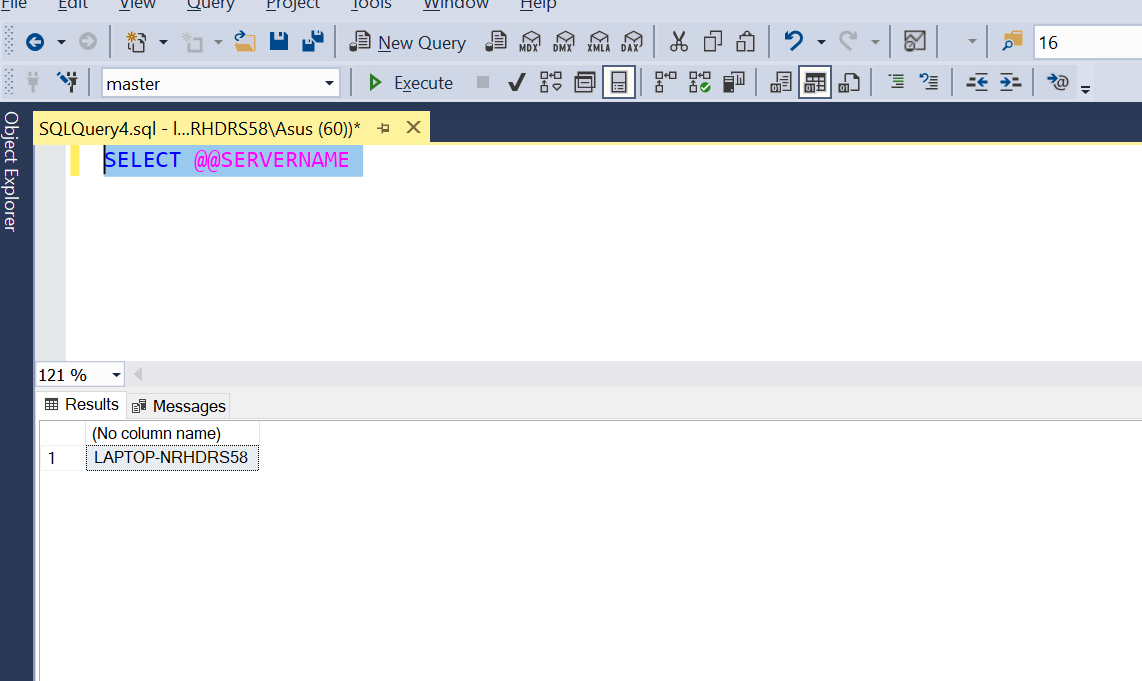
    }

}

Settinganituuntuk database local saya. Nantiuntuk server atursesuai engine masing-masing.

HOST itu , kalua dilocal adalah servername local kita. Cara melihatnya dari ssms adalah jalankan perintah berikut:

SELECT @@SERVERNAME



**Cara mengkoneksikan dengan mysql**

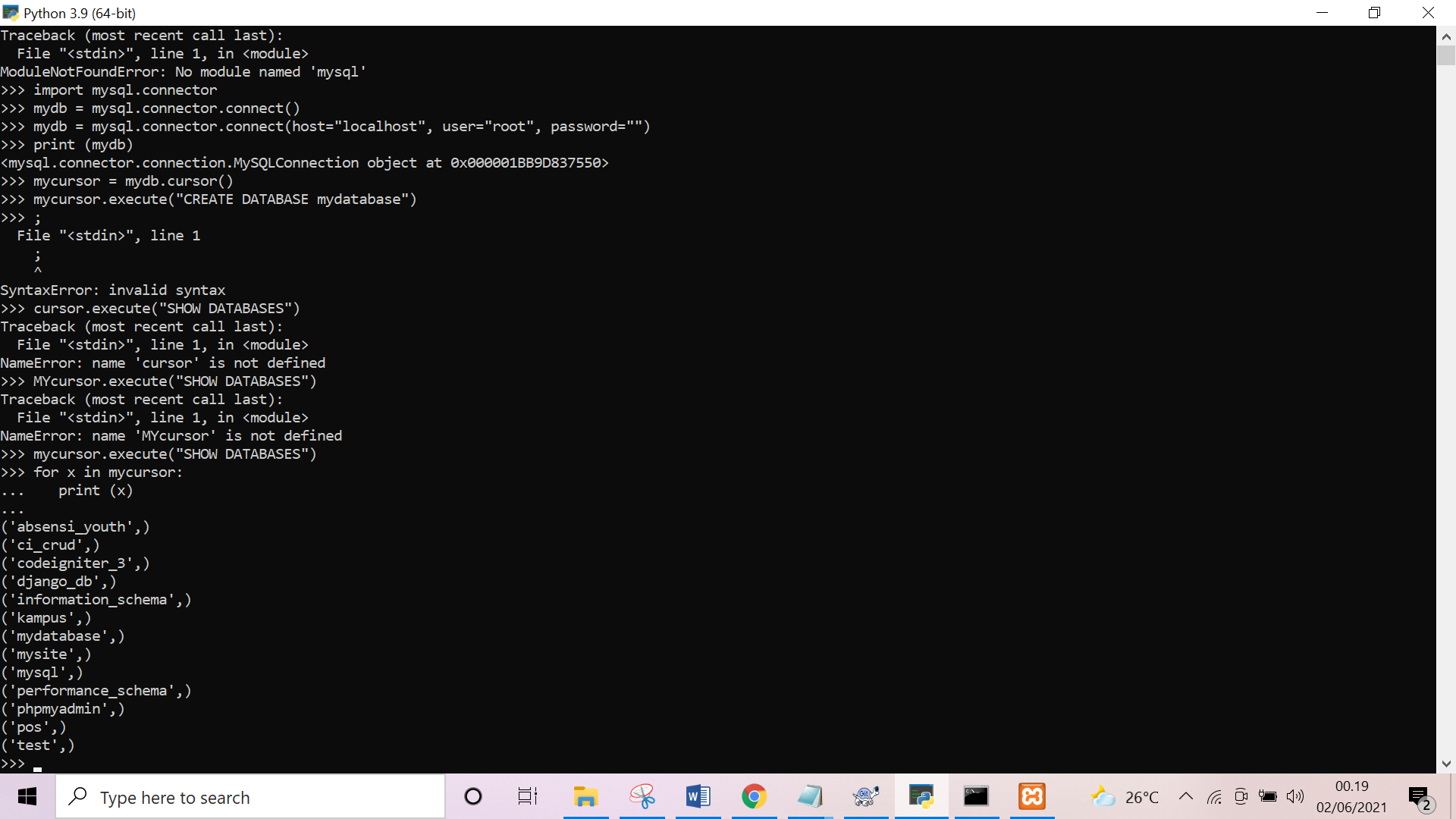
Install library mysql connector:

pip install mysql-connector

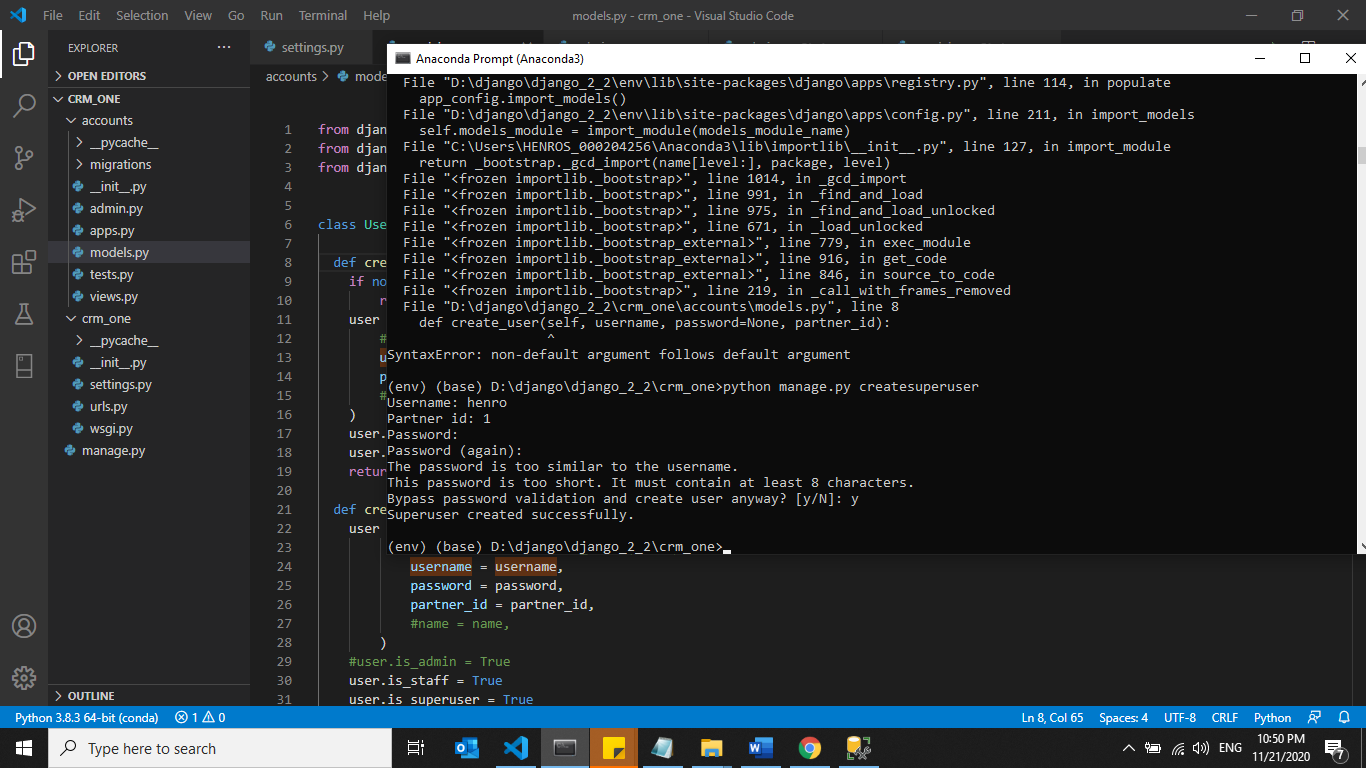
setelah terinstall, berikut cara membuatnya:

import mysql.connector  
  
mydb = mysql.connector.connect(  
  host="localhost",  
  user="yourusername",  
  password="yourpassword"  
)  
  
print(mydb)

sedangkan untuk mencobanya sudah berhasil connect atau tidak, kita bisa mengexecute perintah sql, seperti berikut:  
mycursor = mydb.cursor()  
mycursor.execute("CREATE DATABASE mydatabase")

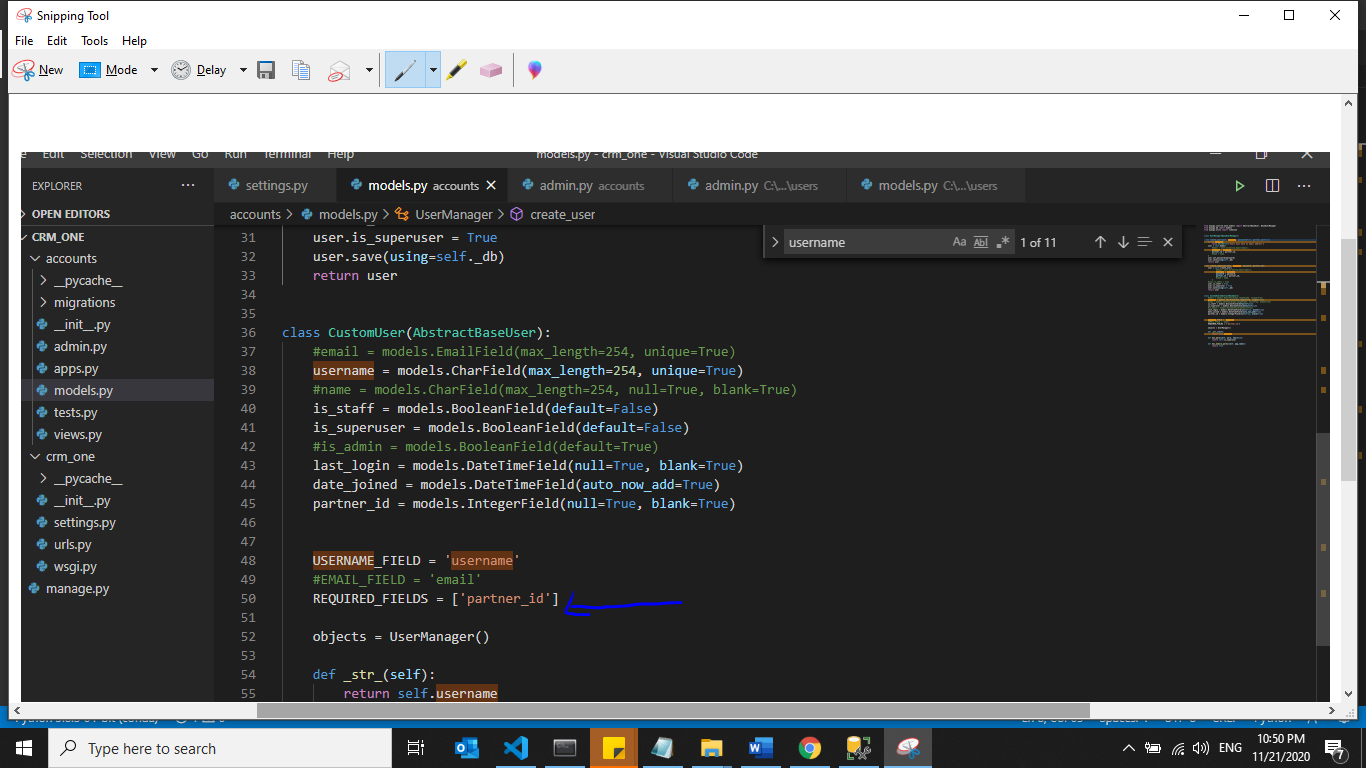


Untukmenambah list ygharusdiisisaat create super dari terminal:

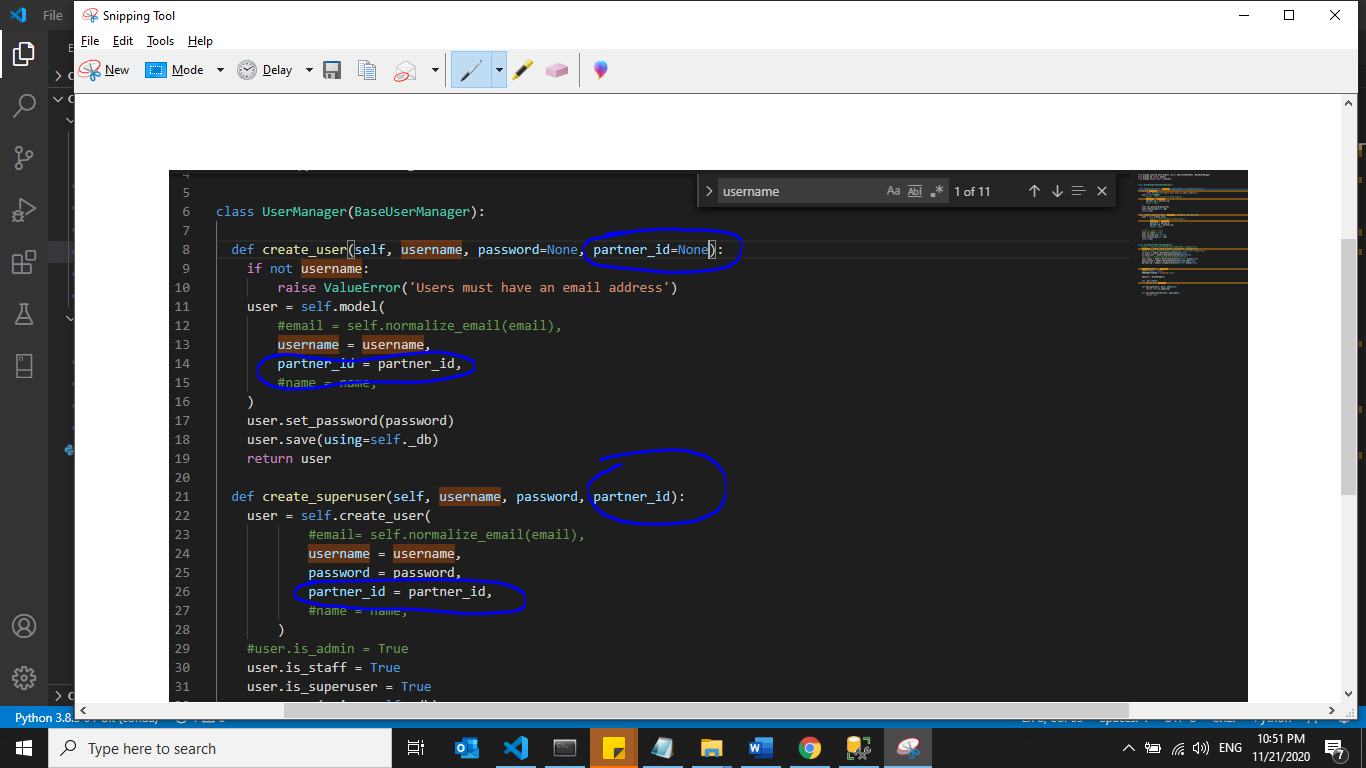


Berikutcaranya:

1.



2.



Cara membuatvalidasi di field form API:

Di class serialezerstambahkan code spertiini:

def validate\_NAMA\_PARTNER(self, value):

        """

        Check that the blog post is about Django.

        """

        if 'django' not in value.lower():

            raise serializers.ValidationError("Blog post is not about Django")

        return value

cara custom user model di Django

contohkitatambah 1 folder accounts untuk handle custom user model:

from django.db import models

from django.contrib.auth.models import AbstractUser

# Create your models here.

class CustomUser(AbstractUser):

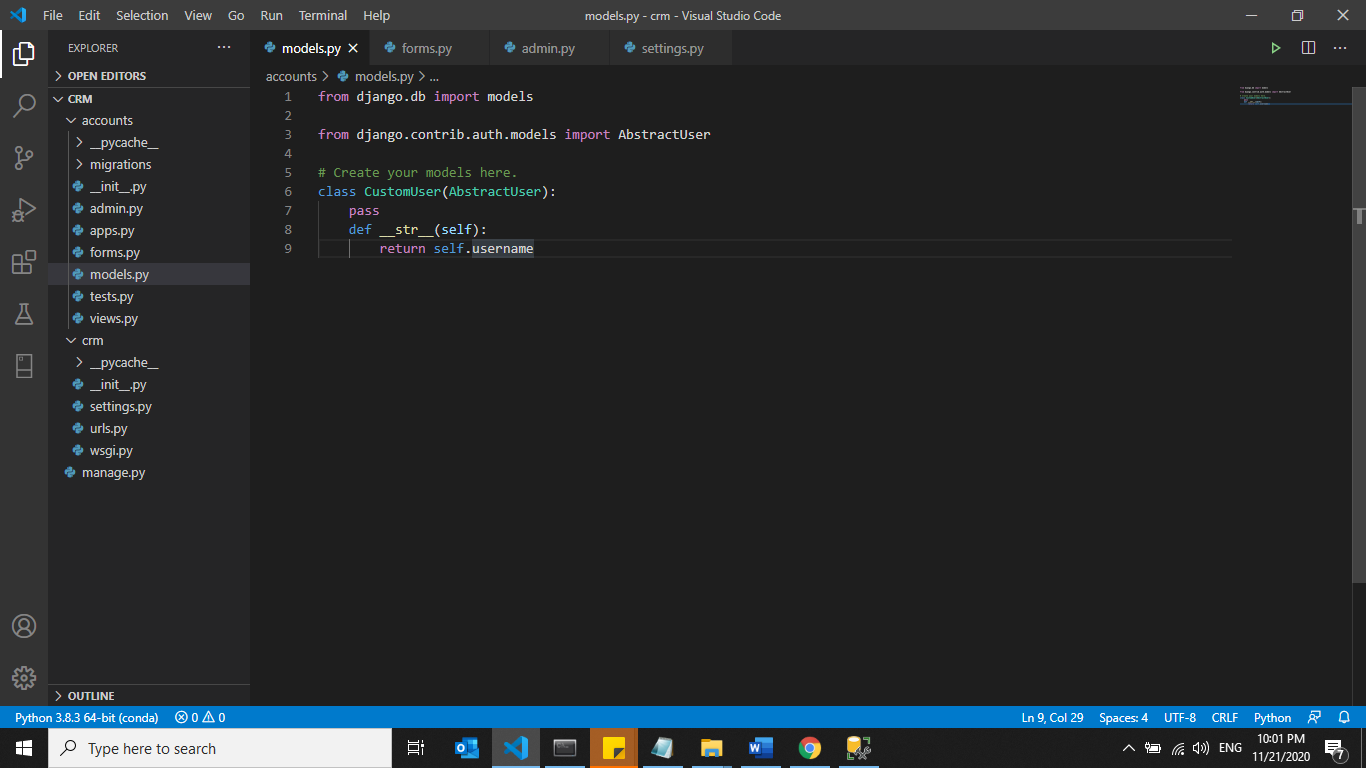
    pass

    def \_\_str\_\_(self):

        return self.username

berikutcontoh di project

models.py



Forms.py

from django import forms

from django.contrib.auth.forms import UserCreationForm, UserChangeForm

from .models import CustomUser

class CustomUserCreationForm(UserCreationForm):

    class Meta:

        model = CustomUser

        fields = ('username', 'email')

class CustomUserChangeForm(UserChangeForm):

    class Meta:

        model = CustomUser

        fields = ('username', 'email')

di settings.py

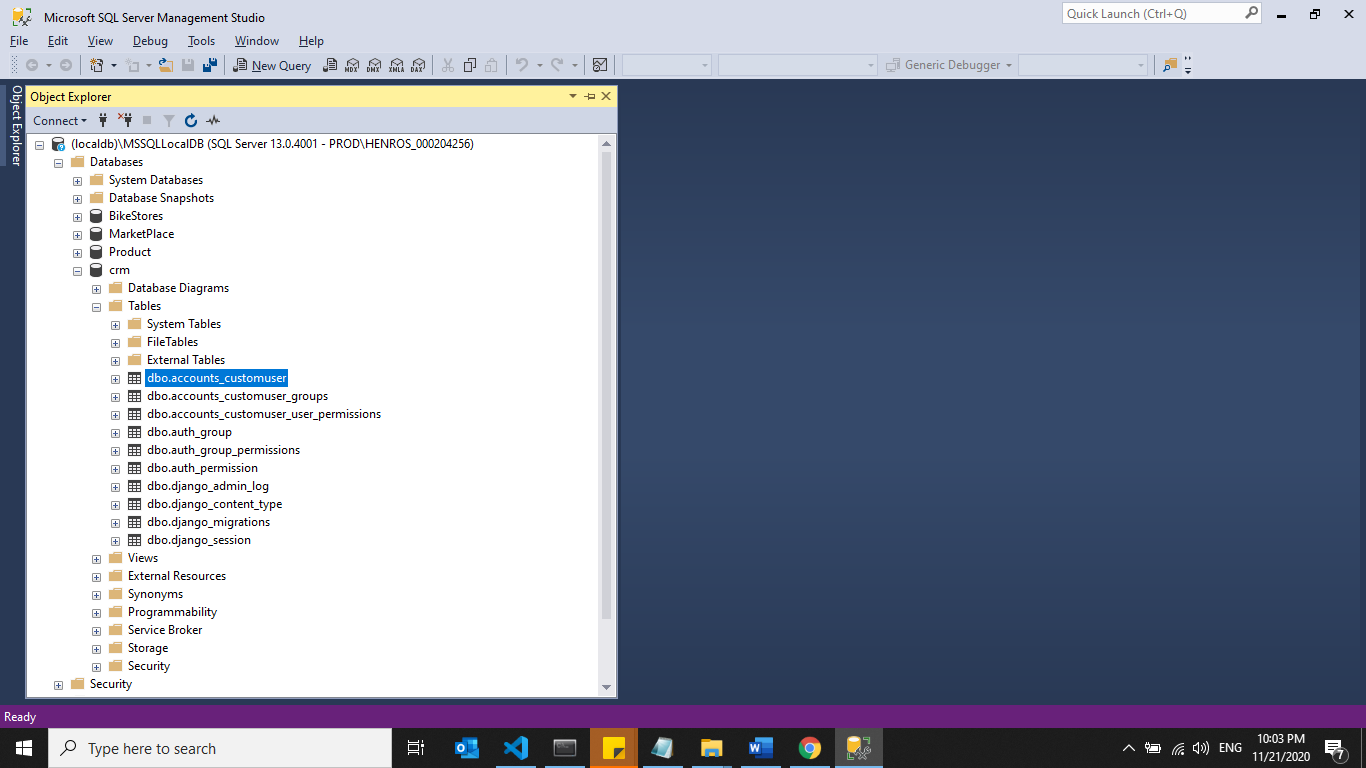
tambahkaninstalledapp di setting nama app kita, yaitu accounts.

Kemudiantambahkan juga script berikut:

AUTH\_USER\_MODEL = 'accounts.CustomUser'

Kemudianmakemigrations dan migrate

Hasilnya di dabatasesepertiberikut:



NAMA\_PARTNER ituadalahnama field.

https://www.django-rest-framework.org/api-guide/serializers/#field-level-validation

Cara membuatauthentikasi token dengan library simple\_jwt di Django:

halygperlu di ketahuiketikaimplementasi simple jwt di django:

di file setting, tambahkan:

REST\_FRAMEWORK = {

# 'DEFAULT\_PAGINATION\_CLASS': 'rest\_framework.pagination.PageNumberPagination',

# 'PAGE\_SIZE': 10,

'DEFAULT\_AUTHENTICATION\_CLASSES': ('rest\_framework\_simplejwt.authentication.JWTAuthentication',)

}

kemudiantambahkan juga di file ygsamauntukmengatur masa berlaku token:

from datetime import timedelta

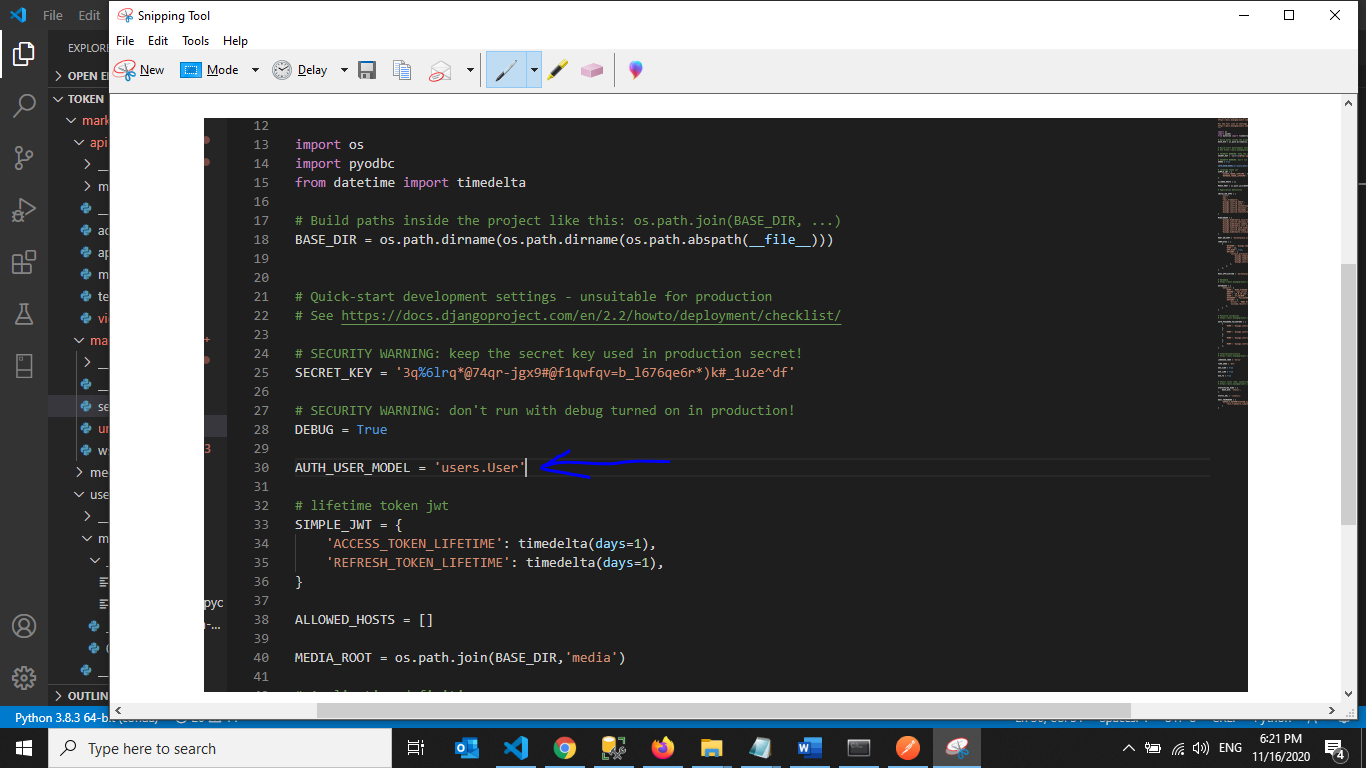
SIMPLE\_JWT = {

'ACCESS\_TOKEN\_LIFETIME': timedelta(days=9999),

'REFRESH\_TOKEN\_LIFETIME': timedelta(days=9999),

}

Ini juga janganlupa di settingannya



Jika kitasudahpernah migrate sebelumnya, jikaingin update tabel, mungkinhapustabeluser\_usernyajigamemungkinkan, hapus file migrate (0001\_initial\_migrations) di aplikasidjangonyasekaliancachenyajikaperlu.

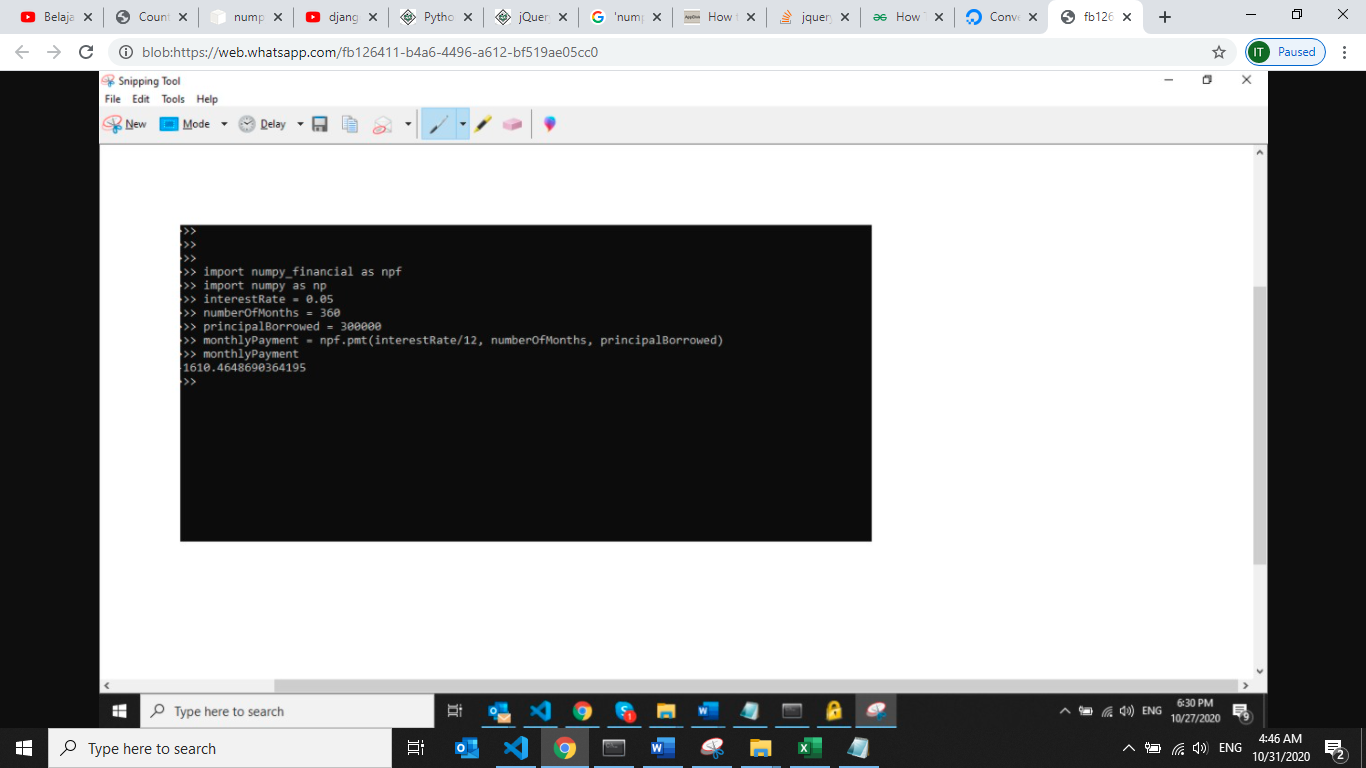
Pmt di python.

Kita bisamenggunakna library: numpy-financial

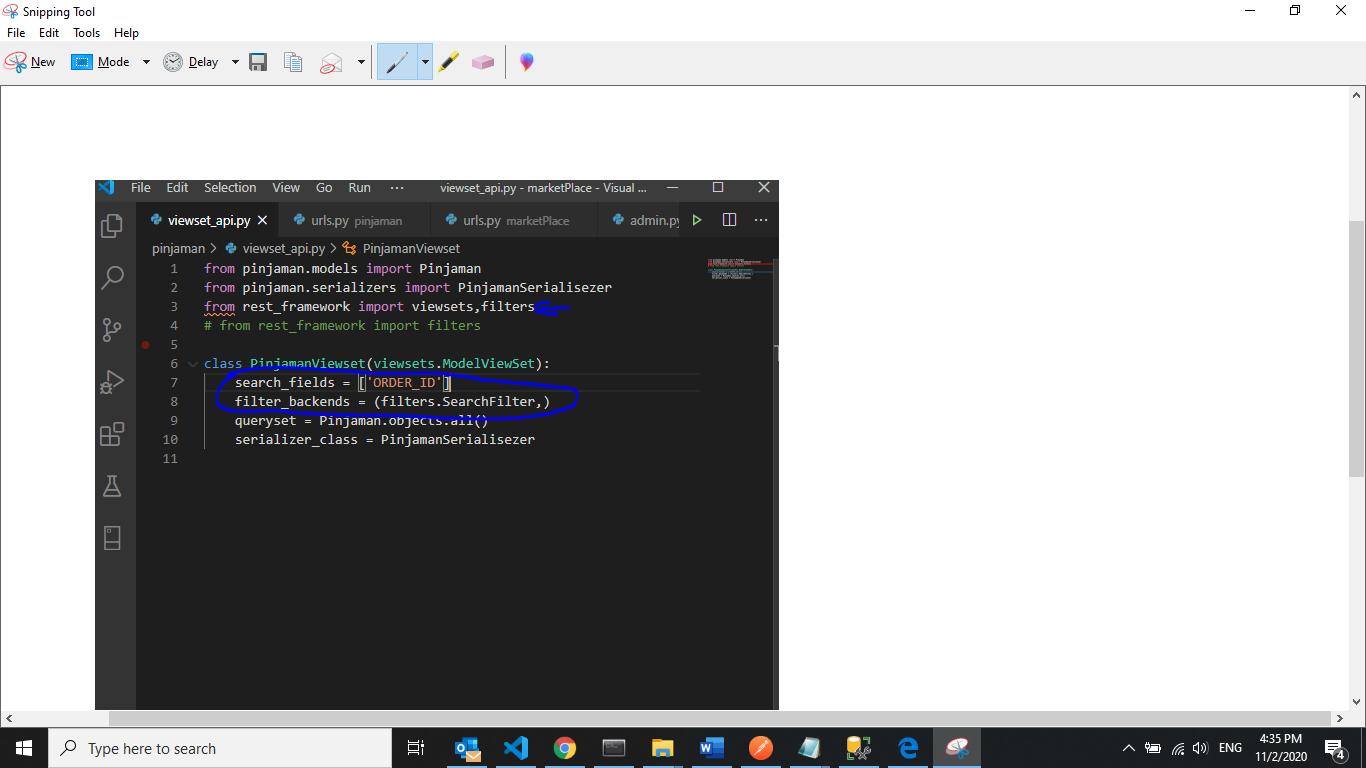
Cara install:

pip install numpy-financial

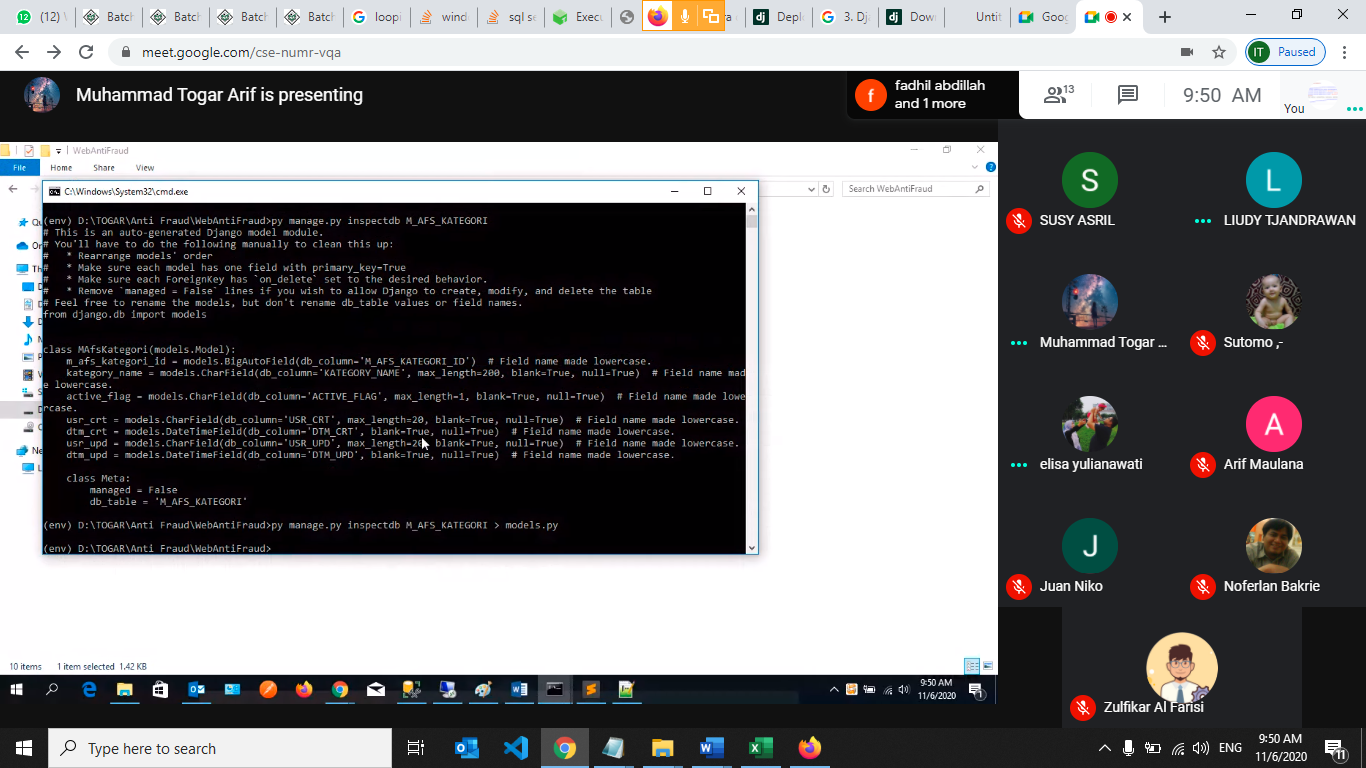
carapakai:



Cara menambah filter search data api Django:

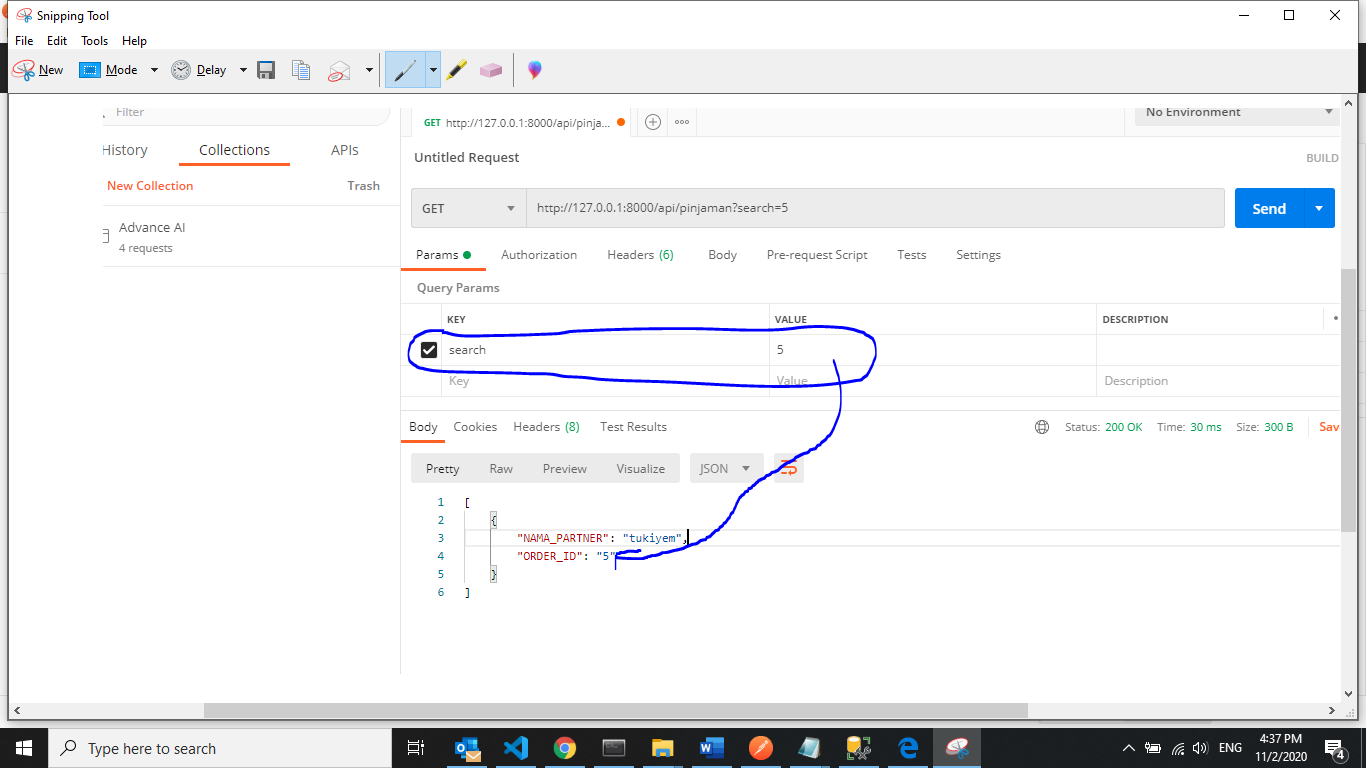


Cara membuat model otomatis di Django, biarnggaketik 1 demi 1



Nanti key ygkitapakaiadalah: search, kemudianvaluenya value dari field yang kitamasukkan di search\_fieldsdiatas.

Contoh:



Cara ganti style field bawaan Django:

classMyForm(forms.Form):

myfield = forms.CharField(widget=forms.TextInput(attrs={'class' : 'myfieldclass'}))

or

classMyForm(forms.ModelForm):

classMeta:

model = MyModel

def\_\_init\_\_(self, \*args, \*\*kwargs):

super(MyForm, self).\_\_init\_\_(\*args, \*\*kwargs)

self.fields['myfield'].widget.attrs.update({'class' : 'myfieldclass'})

or

classMyForm(forms.ModelForm):

classMeta:

model = MyModel

widgets = {

'myfield': forms.TextInput(attrs={'class': 'myfieldclass'}),

}

--- EDIT ---  
The above is the easiest change to make to original question's code that accomplishes what was asked. It also keeps you from repeating yourself if you reuse the form in other places; your classes or other attributes just work if you use the Django's as\_table/as\_ul/as\_p form methods. If you need full control for a completely custom rendering, this is [clearly documented](https://docs.djangoproject.com/en/2.2/topics/forms/#working-with-form-templates)

-- EDIT 2 ---  
Added a newer way to specify widget and attrs for a ModelForm.

Ataubisa juga pakaiini:

This can be done using a custom template filter. Consider rendering your form this way:

<formaction="/contact/"method="post">

{{ form.non\_field\_errors }}

<divclass="fieldWrapper">

{{ form.subject.errors }}

{{ form.subject.label\_tag }}

{{ form.subject }}

<spanclass="helptext">{{ form.subject.help\_text }}</span>

</div>

</form>

form.subject is an instance of [BoundField](https://docs.djangoproject.com/en/dev/ref/forms/api/#django.forms.BoundField) which has the as\_widget() method.

You can create a custom filter addclass in **my\_app/templatetags/myfilters.py**:

fromdjangoimport template

register = template.Library()

@register.filter(name='addclass')

defaddclass(value, arg):

returnvalue.as\_widget(attrs={'class': arg})

And then apply your filter:

{% load myfilters %}

<formaction="/contact/"method="post">

{{ form.non\_field\_errors }}

<divclass="fieldWrapper">

{{ form.subject.errors }}

{{ form.subject.label\_tag }}

{{ form.subject|addclass:'MyClass' }}

<spanclass="helptext">{{ form.subject.help\_text }}</span>

</div>

</form>

form.subjects will then be rendered with the MyClass CSS class.

Contohperulangandalamperulangandengan 1 baris:

['s'+str(x)+y for x in range(1,11) for y in 'ab']

Hasil:

['s1a', 's1b', 's2a', 's2b', 's3a', 's3b', 's4a', 's4b', 's5a', 's5b', 's6a', 's6b', 's7a', 's7b', 's8a', 's8b', 's9a', 's9b', 's10a', 's10b']

Cara upload gambar python native

source = "D:\\Project\\marketplace\\add\_script.PNG"

        f1 = open(source, 'rb')

        if source:

            print (123)

            # strip leading path from file name to avoid

            # directory traversal attacks

            fn = os.path.basename("add\_script.PNG")

            open('D:\\' + fn, 'wb').write(f1.read())

            message = 'The file "' + fn + '" was uploaded successfully'

        else:

            message = 'No file was uploaded'

        #### data real

        img\_ATTACHMENT1 = request.FILES["ATTACHMENT1"]

        img\_ATTACHMENT2 = request.FILES["ATTACHMENT2"]

        img\_ATTACHMENT3 = request.FILES["ATTACHMENT3"]

        img\_ATTACHMENT4 = request.FILES["ATTACHMENT4"]

        img\_ATTACHMENT5 = request.FILES["ATTACHMENT5"]

        images = [img\_ATTACHMENT1, img\_ATTACHMENT2, img\_ATTACHMENT3, img\_ATTACHMENT4, img\_ATTACHMENT5]

        # for image in images:

        #     file\_name = str(image)

        #     path = default\_storage.save("attachment1\%s"%(file\_name), ContentFile(image.read()))

        #     tmp\_file = os.path.join(settings.MEDIA\_ROOT, path)

        images\_dict = {

            'attachment1': img\_ATTACHMENT1,

            'attachment2': img\_ATTACHMENT2,

            'attachment3': img\_ATTACHMENT3,

            'attachment4': img\_ATTACHMENT4,

            'attachment5': img\_ATTACHMENT5

            }

        for image in images\_dict:

            file\_name = str(images\_dict[image])

            path = default\_storage.save("%s"%(image)+"\\"+"%s"%(file\_name), ContentFile(images\_dict[image].read()))

            tmp\_file = os.path.join(settings.MEDIA\_ROOT, path)

upload gambarke directory ygkitatentukansendir (bukandari setting djanggo). Konsepnyadisinisaya move, pakai library shutil

from django.shortcuts import render

from django.db import connection

from rest\_framework.response import Response

from rest\_framework.views import APIView

from rest\_framework import status

###########

### library upload image

###########

from django.http import HttpResponse

from django.core.files.storage import default\_storage

from django.core.files.base import ContentFile

from django.conf import settings

import cgi, os

import cgitb; cgitb.enable()

import os, sys

import shutil

from rest\_framework.permissions import IsAuthenticated

###########

### library upload image

###########

# Create your views here.

import cgi, os

import cgitb; cgitb.enable()

class Pinjaman(APIView):

    # permission\_classes = (IsAuthenticated,)

    def get(self, request):

        res = {'get': 'get'}

        return Response(res)

    def post(self,request):

        cursor = connection.cursor()

        ###########

        ### upload image

        ###########

        #### data test

        source = "D:\\Project\\marketplace\\add\_script.PNG"

        f1 = open(source, 'rb')

        if source:

            print (123)

            # strip leading path from file name to avoid

            # directory traversal attacks

            fn = os.path.basename("add\_script.PNG")

            open('D:\\' + fn, 'wb').write(f1.read())

            message = 'The file "' + fn + '" was uploaded successfully'

        else:

            message = 'No file was uploaded'

        #### data real

        img\_ATTACHMENT1 = request.FILES["ATTACHMENT1"]

        img\_ATTACHMENT2 = request.FILES["ATTACHMENT2"]

        img\_ATTACHMENT3 = request.FILES["ATTACHMENT3"]

        img\_ATTACHMENT4 = request.FILES["ATTACHMENT4"]

        img\_ATTACHMENT5 = request.FILES["ATTACHMENT5"]

        images = [img\_ATTACHMENT1, img\_ATTACHMENT2, img\_ATTACHMENT3, img\_ATTACHMENT4, img\_ATTACHMENT5]

        print (request.POST)

        content = {'test': 'test'}

        piID\_PARTNER = request.POST.get('ID\_PARTNER')

        piORDER\_ID = request.POST.get('ORDER\_ID')

        piORDER\_DT = request.POST.get('ORDER\_DT')

        piTRANSACTION\_ID = request.POST.get('TRANSACTION\_ID')

        piPRODUCT\_TYPE = request.POST.get('PRODUCT\_TYPE\_ID')

        piFINANCE\_AMOUNT = request.POST.get('FINANCE\_AMOUNT')

        piDP\_AMOUNT = request.POST.get('DP\_AMOUNT')

        piTENOR = request.POST.get('TENOR')

        piEST\_CICILAN = request.POST.get('EST\_CICILAN')

        piFULL\_NAME = request.POST.get('FULL\_NAME')

        piID\_NO = request.POST.get('ID\_NO')

        piBIRTH\_PLACE = request.POST.get('BIRTH\_PLACE')

        piBIRTH\_DT = request.POST.get('BIRTH\_DT')

        piGENDER = request.POST.get('GENDER')

        piMOTHER\_MAIDEN\_NAME = request.POST.get('MOTHER\_MAIDEN\_NAME')

        piPHONE\_NUMBER = request.POST.get('PHONE\_NUMBER')

        piCITY = request.POST.get('CITY')

        piADDR = request.POST.get('ADDR')

        piPROFESSION = request.POST.get('PROFESSION')

        piINDUSTRY\_TYPE = request.POST.get('INDUSTRY\_TYPE')

        piINCOME = request.POST.get('INCOME')

        piNPWP\_NO = request.POST.get('NPWP\_NO')

        piJOB\_ADDR = request.POST.get('JOB\_ADDR')

        piJOB\_CITY = request.POST.get('JOB\_CITY')

        piEMERGENCY\_CNTCT\_NAME = request.POST.get('EMERGENCY\_CNTCT\_NAME')

        piPHONE\_NUMBER\_EMER\_CNTCT = request.POST.get('PHONE\_NUMBER\_EMER\_CNTCT')

        piMANUFACTURING\_YEAR = request.POST.get('MANUFACTURING\_YEAR')

        piID\_JAMINAN = request.POST.get('ID\_JAMINAN')

        piNAMA\_JAMINAN = request.POST.get('NAMA\_JAMINAN')

        piMERK = request.POST.get('MERK')

        piMODEL = request.POST.get('MODEL')

        piTYPE = request.POST.get('TYPE')

        piATTACHMENT1 = request.POST.get('ATTACHMENT1')

        piATTACHMENT2 = request.POST.get('ATTACHMENT2')

        piATTACHMENT3 = request.POST.get('ATTACHMENT3')

        piATTACHMENT4 = request.POST.get('ATTACHMENT4')

        piATTACHMENT5 = request.POST.get('ATTACHMENT5')

        # piSOURCE\_DATA = request.POST.get('SOURCE\_DATA')

        # piDTM\_CRT = request.POST.get('DTM\_CRT')

        # piUSR\_CRT = request.POST.get('USR\_CRT')

        # piDTM\_UPD = request.POST.get('DTM\_UPD')

        # piUSR\_UPD = request.POST.get('USR\_UPD')

        print (type(int(piORDER\_ID)))

        print ('------------')

        piFINANCE\_AMOUNT = int(piFINANCE\_AMOUNT)

        piDP\_AMOUNT = int(piDP\_AMOUNT)

        piTENOR = int(piTENOR)

        piEST\_CICILAN = int(piEST\_CICILAN)

        piINCOME = int(piINCOME)

        # for image in images:

        #     file\_name = str(image)

        #     path = default\_storage.save("attachment1\%s"%(file\_name), ContentFile(image.read()))

        #     tmp\_file = os.path.join(settings.MEDIA\_ROOT, path)

        # source\_test = "D:\\"

        images\_dict = {

            'attachment1': img\_ATTACHMENT1,

            'attachment2': img\_ATTACHMENT2,

            'attachment3': img\_ATTACHMENT3,

            'attachment4': img\_ATTACHMENT4,

            'attachment5': img\_ATTACHMENT5

            }

        sql\_dir = """SELECT NAME\_MARKETPLACE, PATH\_FILE FROM M\_MKT\_MARKETPLACE\_MASTER WHERE ID\_MARKETPLACE= %s"""%(piID\_PARTNER)

        cursor.execute(sql\_dir)

        row = cursor.fetchall()

        path\_attachment = row[0][1]

        print (settings.MEDIA\_ROOT)

        di = settings.MEDIA\_ROOT

        print ('\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_--')

        di\_hardcode = "D:\\django"

        print ('AAAAAAAAAAAAA')

        print (default\_storage)

        print ('AAAAAAAAAAAAA')

        for image in images\_dict:

            file\_name = piORDER\_ID + '\_' + str(images\_dict[image])

            folder = str('marketplace')

            # path = default\_storage.save("%s"%(image)+"\\"+"%s"%(file\_name), ContentFile(images\_dict[image].read()))

            path = default\_storage.save("%s"%(folder)+"\\"+"%s"%(file\_name), ContentFile(images\_dict[image].read()))

            # file\_dest = default\_storage.save("%s"%(file\_name), ContentFile(images\_dict[image].read()))

            print (path)

            tmp\_file = os.path.join(settings.MEDIA\_ROOT, path)

            dst\_file = os.path.join(path\_attachment,file\_name)

            a = shutil.move(tmp\_file, "%s"%(dst\_file))

            print (a)

            print ('aaaaaa')

        # os.rename("path", "D:/to/new/destination/for/file.foo")

        # shutil.move("path", "D")

        # os.replace("path", "path/to/new/destination/for/file.foo")

        ###########

        ### end upload image

        ###########

        params = (

                piID\_PARTNER,

                piORDER\_ID,

                piORDER\_DT,

                piTRANSACTION\_ID,

                piPRODUCT\_TYPE,

                piFINANCE\_AMOUNT,

                piDP\_AMOUNT,

                piTENOR,

                piEST\_CICILAN,

                piFULL\_NAME,

                piID\_NO,

                piBIRTH\_PLACE,

                piBIRTH\_DT,

                piGENDER,

                piMOTHER\_MAIDEN\_NAME,

                piPHONE\_NUMBER,

                piCITY,

                piADDR,

                piPROFESSION,

                piINDUSTRY\_TYPE,

                piINCOME,

                piNPWP\_NO,

                piJOB\_ADDR,

                piJOB\_CITY,

                piEMERGENCY\_CNTCT\_NAME,

                piPHONE\_NUMBER\_EMER\_CNTCT,

                piMANUFACTURING\_YEAR,

                piID\_JAMINAN,

                piNAMA\_JAMINAN,

                piMERK,

                piMODEL,

                piTYPE,

                piATTACHMENT1,

                piATTACHMENT2,

                piATTACHMENT3,

                piATTACHMENT4,

                piATTACHMENT5,

            )

        try:

            sql = """

                        DECLARE @TRANSACTION\_ID\_OUT varchar (10);

                        EXECUTE sp\_WOM\_MKT\_MARKETPLACE\_INS\_ACQUISITION

                        @piID\_PARTNER = %s,

                        @piORDER\_ID = %s ,

                        @piORDER\_DT = %s , -- GATEDATE()

                        @piTRANSACTION\_ID = %s ,

                        @piPRODUCT\_TYPE = %s ,

                        @piFINANCE\_AMOUNT = %s,

                        @piDP\_AMOUNT = %s,

                        @piTENOR = %s,

                        @piEST\_CICILAN = %s,

                        @piFULL\_NAME = %s ,

                        @piID\_NO = %s ,

                        @piBIRTH\_PLACE = %s ,

                        @piBIRTH\_DT = %s ,

                        @piGENDER = %s ,

                        @piMOTHER\_MAIDEN\_NAME = %s ,

                        @piPHONE\_NUMBER = %s ,

                        @piCITY = %s ,

                        @piADDR = %s ,

                        @piPROFESSION = %s ,

                        @piINDUSTRY\_TYPE = %s ,

                        @piINCOME = %s,

                        @piNPWP\_NO = %s ,

                        @piJOB\_ADDR = %s ,

                        @piJOB\_CITY = %s ,

                        @piEMERGENCY\_CNTCT\_NAME = %s ,

                        @piPHONE\_NUMBER\_EMER\_CNTCT = %s ,

                        @piMANUFACTURING\_YEAR = %s ,

                        @piID\_JAMINAN = %s ,

                        @piNAMA\_JAMINAN = %s ,

                        @piMERK = %s ,

                        @piMODEL = %s ,

                        @piTYPE = %s ,

                        @piATTACHMENT1 = %s,

                        @piATTACHMENT2 = %s,

                        @piATTACHMENT3 = %s,

                        @piATTACHMENT4 = %s,

                        @piATTACHMENT5 = %s,

                        @poRESP\_TRANSACTION\_ID = @TRANSACTION\_ID\_OUT OUTPUT

                        ;

                        SELECT @TRANSACTION\_ID\_OUT AS code\_output;

                        """

            # params = (piID\_PARTNER, piORDER\_ID, piORDER\_DT, piPRODUCT\_TYPE)

            cursor.execute(sql, params)

            try:

                row = cursor.fetchall()

                content = {'status\_code': {'responseCode': row[0]}}

                response\_api =  Response(content, status = status.HTTP\_200\_OK)

                print (row)

            except:

                print('errror')

            cursor.commit()

            return response\_api

        except Exception as e:

            print (e)

            content = {'AppData': {'responseCode': '25', 'responseMessage': 'System Error'}}

            return Response(content, status=status.HTTP\_204\_NO\_CONTENT)

    def post\_hardcode\_work(self,request):

        print (request.POST)

        content = {'test': 'test'}

        piID\_PARTNER = request.POST.get('ID\_PARTNER')

        piORDER\_ID = request.POST.get('ORDER\_ID')

        piORDER\_DT = request.POST.get('ORDER\_DT')

        piTRANSACTION\_ID = request.POST.get('TRANSACTION\_ID')

        piPRODUCT\_TYPE = request.POST.get('PRODUCT\_TYPE')

        piFINANCE\_AMOUNT = request.POST.get('FINANCE\_AMOUNT')

        piDP\_AMOUNT = request.POST.get('DP\_AMOUNT')

        piTENOR = request.POST.get('TENOR')

        piEST\_CICILAN = request.POST.get('EST\_CICILAN')

        piFULL\_NAME = request.POST.get('FULL\_NAME')

        piID\_NO = request.POST.get('ID\_NO')

        piBIRTH\_PLACE = request.POST.get('BIRTH\_PLACE')

        piBIRTH\_DT = request.POST.get('BIRTH\_DT')

        piGENDER = request.POST.get('GENDER')

        piMOTHER\_MAIDEN\_NAME = request.POST.get('MOTHER\_MAIDEN\_NAME')

        piPHONE\_NUMBER = request.POST.get('PHONE\_NUMBER')

        piCITY = request.POST.get('CITY')

        piADDR = request.POST.get('ADDR')

        piPROFESSION = request.POST.get('PROFESSION')

        piINDUSTRY\_TYPE = request.POST.get('INDUSTRY\_TYPE')

        piINCOME = request.POST.get('INCOME')

        piNPWP\_NO = request.POST.get('NPWP\_NO')

        piJOB\_ADDR = request.POST.get('JOB\_ADDR')

        piJOB\_CITY = request.POST.get('JOB\_CITY')

        piEMERGENCY\_CNTCT\_NAME = request.POST.get('EMERGENCY\_CNTCT\_NAME')

        piPHONE\_NUMBER\_EMER\_CNTCT = request.POST.get('PHONE\_NUMBER\_EMER\_CNTCT')

        piMANUFACTURING\_YEAR = request.POST.get('MANUFACTURING\_YEAR')

        piID\_JAMINAN = request.POST.get('ID\_JAMINAN')

        piNAMA\_JAMINAN = request.POST.get('NAMA\_JAMINAN')

        piMERK = request.POST.get('MERK')

        piMODEL = request.POST.get('MODEL')

        piTYPE = request.POST.get('TYPE')

        piATTACHMENT1 = request.POST.get('ATTACHMENT1')

        piATTACHMENT2 = request.POST.get('ATTACHMENT2')

        piATTACHMENT3 = request.POST.get('ATTACHMENT3')

        piATTACHMENT4 = request.POST.get('ATTACHMENT4')

        piATTACHMENT5 = request.POST.get('ATTACHMENT5')

        # piSOURCE\_DATA = request.POST.get('SOURCE\_DATA')

        # piDTM\_CRT = request.POST.get('DTM\_CRT')

        # piUSR\_CRT = request.POST.get('USR\_CRT')

        # piDTM\_UPD = request.POST.get('DTM\_UPD')

        # piUSR\_UPD = request.POST.get('USR\_UPD')

        print (type(int(piORDER\_ID)))

        print ('------------')

        params = (

                piID\_PARTNER ,

                piORDER\_ID  ,

                piORDER\_DT  ,

                piTRANSACTION\_ID  ,

                piPRODUCT\_TYPE  ,

                int(piFINANCE\_AMOUNT)  ,

                int(piDP\_AMOUNT)  ,

                int(piTENOR)  ,

                int(piEST\_CICILAN)  ,

                piFULL\_NAME  ,

                piID\_NO  ,

                piBIRTH\_PLACE  ,

                piBIRTH\_DT  ,

                piGENDER  ,

                piMOTHER\_MAIDEN\_NAME  ,

                piPHONE\_NUMBER  ,

                piCITY  ,

                piADDR  ,

                piPROFESSION  ,

                piINDUSTRY\_TYPE  ,

                int(piINCOME)  ,

                piNPWP\_NO  ,

                piJOB\_ADDR  ,

                piJOB\_CITY  ,

                piEMERGENCY\_CNTCT\_NAME  ,

                piPHONE\_NUMBER\_EMER\_CNTCT  ,

                piMANUFACTURING\_YEAR  ,

                piID\_JAMINAN  ,

                piNAMA\_JAMINAN  ,

                piMERK  ,

                piMODEL  ,

                piTYPE  ,

                piATTACHMENT1  ,

                piATTACHMENT2  ,

                piATTACHMENT3  ,

                piATTACHMENT4  ,

                piATTACHMENT5

            )

        cursor = connection.cursor()

        try:

            print (123333333)

            sql = """

                        DECLARE @TRANSACTION\_ID\_OUT varchar (10);

                        EXECUTE sp\_WOM\_MKT\_MARKETPLACE\_INS\_ACQUISITION

                        @piID\_PARTNER = 'test',

                        @piORDER\_ID = 1 ,

                        @piORDER\_DT = NULL ,

                        @piTRANSACTION\_ID = 1 ,

                        @piPRODUCT\_TYPE = 'test' ,

                        @piFINANCE\_AMOUNT = 1001 ,

                        @piDP\_AMOUNT = 1001 ,

                        @piTENOR = 20 ,

                        @piEST\_CICILAN = 100 ,

                        @piFULL\_NAME = 'test' ,

                        @piID\_NO = 'test' ,

                        @piBIRTH\_PLACE = 'test' ,

                        @piBIRTH\_DT = NULL ,

                        @piGENDER = 'test' ,

                        @piMOTHER\_MAIDEN\_NAME = 'test' ,

                        @piPHONE\_NUMBER = 'test' ,

                        @piCITY = 'test' ,

                        @piADDR = 'test' ,

                        @piPROFESSION = 'test' ,

                        @piINDUSTRY\_TYPE = 'test' ,

                        @piINCOME = 10 ,

                        @piNPWP\_NO = 'test' ,

                        @piJOB\_ADDR = 'test' ,

                        @piJOB\_CITY = 'test' ,

                        @piEMERGENCY\_CNTCT\_NAME = 'test' ,

                        @piPHONE\_NUMBER\_EMER\_CNTCT = 'test' ,

                        @piMANUFACTURING\_YEAR = 'test' ,

                        @piID\_JAMINAN = 'test' ,

                        @piNAMA\_JAMINAN = 'test' ,

                        @piMERK = 'test' ,

                        @piMODEL = 'test' ,

                        @piTYPE = 'test' ,

                        @piATTACHMENT1 = 'Test',

                        @piATTACHMENT2 = 'Test',

                        @piATTACHMENT3 = 'Test',

                        @piATTACHMENT4 = 'Test',

                        @piATTACHMENT5 = 'Test',

                        @poRESP\_TRANSACTION\_ID = @TRANSACTION\_ID\_OUT OUTPUT

                        ;

                        SELECT @TRANSACTION\_ID\_OUT AS test;

                        """

            # params = (piID\_PARTNER, piORDER\_ID, piORDER\_DT, piPRODUCT\_TYPE)

            cursor.execute(sql)

            try:

                row = cursor.fetchall()

                print (row)

            except:

                print('errror')

            cursor.commit()

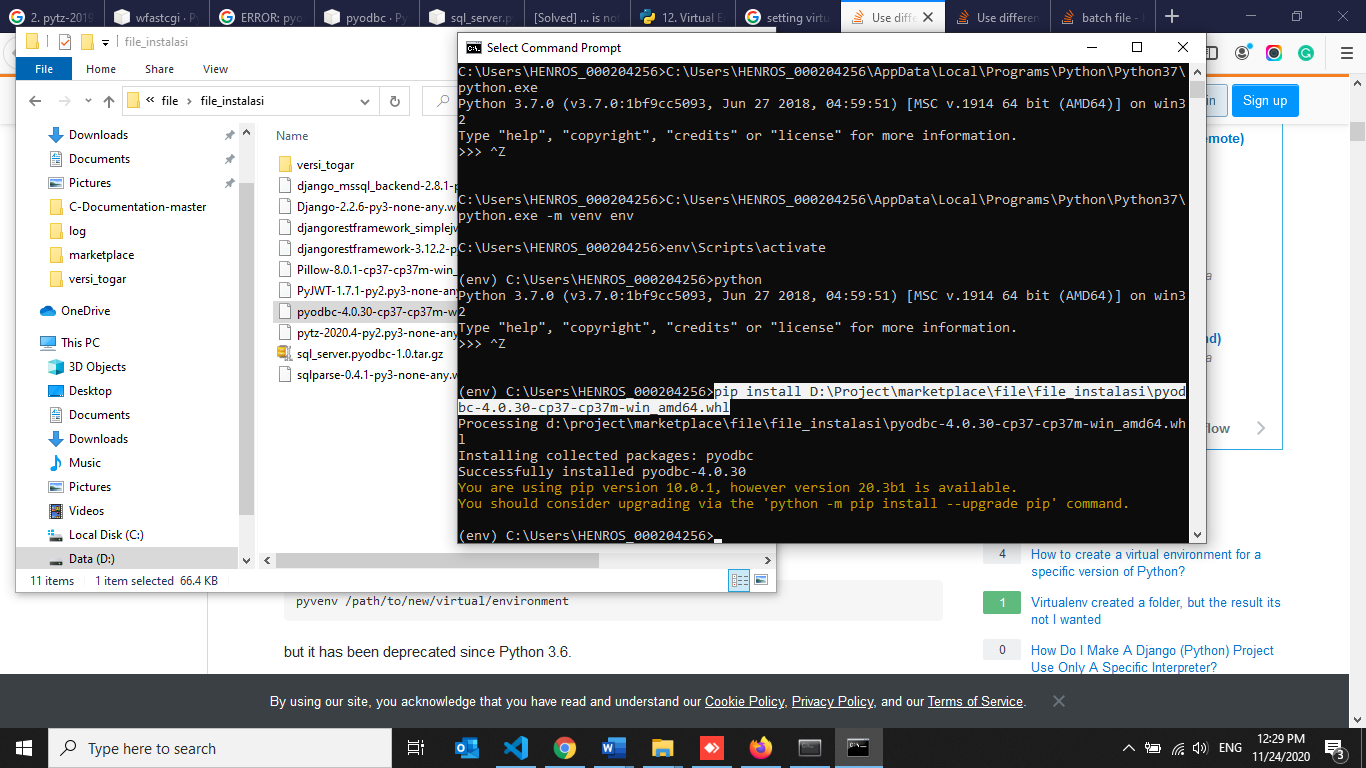
            return Response(content)

        except:

            content = {'inquiryData': {'responseCode': '25', 'responseMessage': 'System Error'}}

            return Response(content)

carabuat environment denganversi python tertentu, contoh 3.7



Cara setting timezone Django menjadisettingan Jakarta:

Masukke settings.py, kemudianuntuk setting timezonebuatsepertiberikut:

TIME\_ZONE = 'Asia/Jakarta'

caramemindahkan file pakai python.

Berikut 3 cara, yaitu:

1. os.rename("path", "D:/to/new/destination/for/file.foo")
2. shutil.move("path", "D")
3. os.replace("path", "path/to/new/destination/for/file.foo")

fungsiketigacara di atasmemilikitujuan yang sama.

Cara melihatpenyebab error di python:

Contoh:

try:

statement

except Exception as e:

  print (e)

daraicontohitu, e merupakan variable ygmenampungpenyebab error

untukmembuat custom model di Django, pastikanjanganpernahada migrate sebelumnya. Jadi migrate pertamaitusetelahkitabuat custom user modelnya.

Don’t forget to point [**AUTH\_USER\_MODEL**](https://docs.djangoproject.com/en/3.0/ref/settings/#std:setting-AUTH_USER_MODEL) to it. Do this before creating any migrations or running **manage.py migrate** for the first time.

Cara menkonvert base64 ke file gambar (decode)

f = open('D:\\add\_script.jpeg', 'rb')

        img = f.read()

        b64 = base64.b64encode(img)

        # print (b64)

        img = base64.b64decode(b64)

        filename = 'D:\\some\_image.jpg'

        # with open(filename, 'wb') as f:

        #     f.write(img)

        # print (img)

        print ('checkkkkkkkkkkkkkkkkkkk')

        # return True

Error request apike advance ai

requests.exceptions.ConnectionError: HTTPSConnectionPool(host='api.advance.ai', port=443): Max retries exceeded with url: /openapi/verification/v1/tele-status-query (Caused by NewConnectionError('<urllib3.connection.HTTPSConnection object at 0x0000018A71B2F8E0>: Failed to establish a new connection: [WinError 10060] A connection attempt failed because the connected party did not properly respond after a period of time, or established connection failed because connected host has failed to respond'))

(<class 'pyodbc.ProgrammingError'>, 'telestatus.py', 327) : disebabkan query yang error

Pyodbc - Previous SQL was not a query (initerjadikitapanggil procedure dengan parameter output dari python odbc)

Solusi:

Sebelum query ygkitabuatuntuk statement insert tambahkan

SET NOCOUNT ON

Contoh:

USE [DBNAME]

GO

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

ALTERprocedure [dbo].[GET\_PROCESS\_ID](

@PROVIDER\_ID INT,

@PROCESS\_ID INTOUTPUT

)

AS

BEGIN

SET NOCOUNT ON

INSERTINTO processes(provider\_id) values(@PROVIDER\_ID)

SET @PROCESS\_ID= SCOPE\_IDENTITY()

SELECT @PROCESS\_ID AS PROCESS\_ID

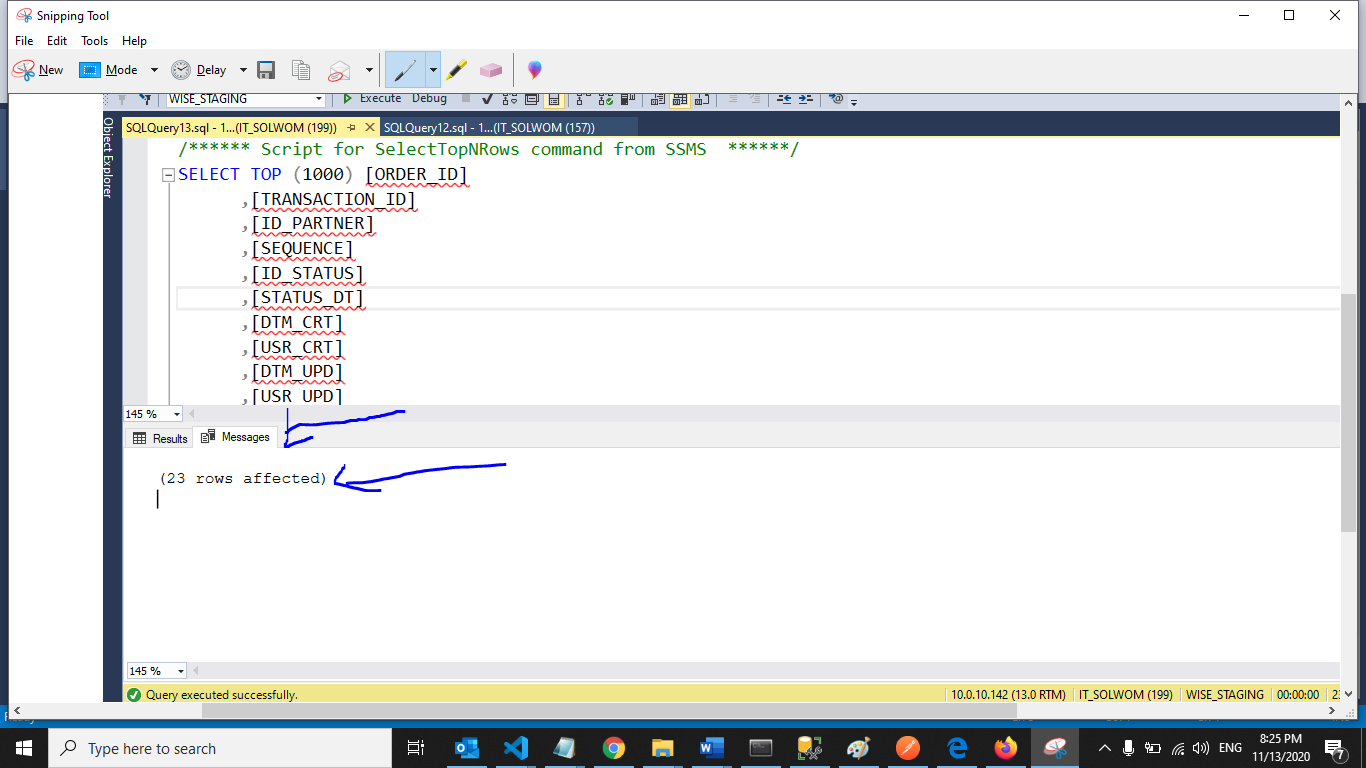
END

Sumber:

<https://stackoverflow.com/questions/7753830/mssql2008-pyodbc-previous-sql-was-not-a-query>

jikadari python tidakberhasilpanggil SP, pastikan Ketika di jalansp di sql server tidakadapesan warning, maupung rows lebihdari 1.

Contohberikuadabenar



Maksudnyajangan 1 ada 2 baris tulisan pesannya.

Contoh:

1 rows affected

1 rows affected

Ataumungkinadapesan warning

Untuk mem fix pesan warning tidakmuncul:

Buat code berikutdidalam begin paling

SETANSI\_WARNINGSOFF

Untukmemfixpesanrows lebihdarisatu:

SETNOCOUNTON

Mengenaichoice\_sest di tutorial Django:

You created a foreign key on Choice which relates each one to a Question.

So, each Choice explicitly has a question field, which you declared in the model.

Django's ORM follows the relationship backwards from Question too, automatically generating a field on each instance called foo\_set where Foo is the model with a ForeignKey field to that model.

choice\_set is a RelatedManager which can create querysets of Choice objects which relate to the Question instance, e.g. q.choice\_set.all()

If you don't like the foo\_set naming which Django chooses automatically, or if you have more than one foreign key to the same model and need to distinguish them, you can choose your own overriding name using the [related\_name](http://docs.djangoproject.com/en/dev/ref/models/fields/#django.db.models.ForeignKey.related_name) argument to ForeignKey.

**forloop.counter: merupakanberapabanyak loop dijalankan. Inibisakitapakaijadipenomoran**

Test driven

*Nb: inti dari test iniadalahkitamembuat scenario yang nantinyaakandibandingkandengan return dari function yang kitabuat*

Jenis-jenis test:

1. test unit
2. test aplikaction
3. test project
4. test compliance

Django sudahmefasilitasisebuah yang Namanya Testcase. Dimana fungsiinikitabisapakaiuntukotomatis test aplikasikita. Nama file untuk test di Django adalah test.py. file inikitaletakkandisetiap app (aplikasi) di django project kita. Juga, namaawalandari function iniadalahtest\_nama\_functionkita.

Contoh:

def test\_was\_published\_recently\_with\_future\_question(self):

        time = timezone.now() + datetime.timedelta(days=30)

        future\_question = Question(pub\_date=time)

        self.assertIs(future\_question.was\_published\_recently(), False)

    def test\_was\_published\_recently\_with\_old\_question(self):

        """

        was\_published\_recently() returns False for questions whose pub\_date

        is older than 1 day.

        """

        time = timezone.now() - datetime.timedelta(days=1, seconds=1)

        old\_question = Question(pub\_date=time)

        self.assertIs(old\_question.was\_published\_recently(), False)

    def test\_was\_published\_recently\_with\_recent\_question(self):

        """

        was\_published\_recently() returns True for questions whose pub\_date

        is within the last day.

        """

        time = timezone.now() - datetime.timedelta(hours=23, minutes=59, seconds=59)

        recent\_question = Question(pub\_date=time)

        self.assertIs(recent\_question.was\_published\_recently(), True)

jikakitaperhatikan di class :QuestionModelTests di atas, itumeng-inherit class TestCasemiliksiDjango.test

kemudian di dalamnyaada 3 function, yang dimanaawalandarisetiapmethodnyaadalah test kemudiandiikutindengannama function yang kitamau.

assertIs (param1, param2)

assertIsbisadiartikansamadengan (=), param1 merupakan output dariygkita test. Baikitufungsi, view, dll. Param2 merupakannilaiygkitaharapkan.

Contoh:

Di class polls berikutkitamembuat 1 function ygnamanya: was\_published\_recently

class Question(models.Model):

    question\_text   = models.CharField(max\_length=200)

    pub\_date        = models.DateTimeField('date published')

    def \_\_str\_\_(self):

        return self.question\_text

    def was\_published\_recently(self):

        return self.pub\_date >= timezone.now() - datetime.timedelta(days=1)

fungsiitukitabuatdengantujuankitainginmengetahuisebuah data di table di create recently (baru-baruini) atautidak. Jika kitalihatcodenya, return dari function ituadalahBoolean : True atau False.  
ygdimana**true** jikaself.pub\_date>= semalam. Kalaukitalihatdarireturnnya, itusedikitada yang salah, dimanasemuayglebihbesarsamadengan( **>=** ) semalam, akangdianggap was published recently. Jadi jikapub\_datenya 3 harikedepan, juga akandianggap True. Nah untukhal-halsepertiini, bisaditangkapataukitakehauibakalada issue melalui proses test.

Contohuntukmengetest function tersebut:

class QuestionModelsTests(TestCase):

    def test\_was\_published\_recently\_with\_future\_question(self):

        time = timezone.now() + datetime.timedelta(days=30)

        future\_question = Question(pub\_date=time)

        self.assertIs(future\_question.was\_published\_recently(), False)

kemudiankitajalankan function dari terminal (cmd) dengancara:

python manage.py test polls

kemudianhasilnyaadalahsepertiberikut:

Creating test database **for** alias 'default'...

System check identified no issues (0 silenced).

F

======================================================================

FAIL: test\_was\_published\_recently\_with\_future\_question (polls.tests.QuestionModelTests)

----------------------------------------------------------------------

Traceback (most recent call last):

File "/path/to/mysite/polls/tests.py", line 16, **in**test\_was\_published\_recently\_with\_future\_question

self.assertIs(future\_question.was\_published\_recently(), **False**)

**AssertionError**: **TrueisnotFalse**

----------------------------------------------------------------------

Ran 1 test **in**0.001s

FAILED (failures=1)

Destroying test database **for** alias 'default'...

Dari sanakitatahu, bahwasannya response dari function was\_published\_recently di model Question kita, ada issue. Oleh karenaitu, sekarangkitasudahtahuharusmenfix yang mana. Makakitabisa fix bugs tersebutdenganmengupdatefunctionnyasepertiberikut:

def was\_published\_recently(self):

        now = timezone.now()

        return now - datetime.timedelta(days=1) <= self.pub\_date <= now

**kesimpulan:**

django test:

kitamembuatskenarion test.

kitalangsungbuat di code.

keuntungannyaadalahkitabisagunakan code tersebutsecaraberulang-ulangtanpaharus manual test lagidikemudianharijikaadaperubahan code.

**Reverse**

Reverse merupakanfungsibawaan Django, dimanatujuanyaadalahuntuksupayakitatidakmengharcodesebuahurl. Ini fungsinya hamper sama dengan pembuatan url di file .html kita.

Contoh di html:

<a href="{% url 'url\_name' %}">link which calls some\_view</a>

Penjelasan:

url\_name adalah name di dalam url\_patterns di file urls.py kita.

Dengan cara seperti itu kita sudah tidak mengharcode url kita. Jadi jika ada perubahan url di kemudianhari, kitacukuphanyamenggantinya di file urls.py tanpaharusmengubahnya di berbagai file yang memakaiurlitu.

Ituadalah di dalam file .html. Namunbagaimanakitainginmenghandleurluntuktidak hardcode seperti yang di urltersebut. Disinilahkitamenggunakanfungsi reverse darisi Django.

Contohpenggunaan:

from django.urls import reverse

return HttpResponseRedirect(reverse('url\_name'))

ygjikatidakpakai reverse, makakitaakanmeredirecnyadengnacara:

return HttpResponseRedirect('/foo/')

foo adalah link ygkitabuat di url\_pattern di file urls.py. jikalangsungpakai link, makaituadalah hardcode. Dimana dikemudianharijikakitainginmerubahurl foo tersebut, kitaharusmerubah di semua file. Namundengancara reverse di atas, kitacukuphanyamerubah di urls.py saja.

PACKAGES

Ketika mengimport packages sebaikanya langsung module-module yg diperlukan saja. Biar tidak memakan waktu. Sebab kalau from packages import \* itu akan mengimport semua file2 yang ada didalam package itu sehingga memakan waktu yang lebih lama. Ada baiknya langsung insert file/module yang diperlukan saja. Contoh: from packages import penjumlahan.

Apakah file \_\_init\_\_.py penting?

Sorce: https://python-forum.io/Thread-Do-packages-really-require-init-py

No,from Python 3.3+ supports Implicit Namespace Packages that allows to create a package without an \_\_init\_\_.py file.  
This however only applies to empty \_\_init\_\_.py files.  
So empty \_\_init\_\_.py files are no longer necessary and can be omitted.  
  
To give a example,i always have one \_\_init\_\_.py that has content to lift sub modules,  
this to avoid long import statement and make it easier for users of package.

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | my\_pack\  |-- \_\_init\_\_.py    color\    |-- base\_color.py      gradient\      |-- gradient.py |

\_\_init\_\_.py

|  |  |
| --- | --- |
| 1  2 | from .color import base\_color  from .color.gradient import gradient |

base\_color.py

|  |  |
| --- | --- |
| 1  2 | def red():      return 'rgb(255, 0, 0)' |

gradient.py

|  |  |
| --- | --- |
| 1  2 | def hsl\_value():      return 'rgba(255, 100, 50, 1.0)' |

As i have lifted with \_\_init\_\_.py there is no need to say color.gradient.hsl\_value() when import.  
Usage:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | λ ptpython  >>> from my\_pack import base\_color, gradient    >>> base\_color.red()  'rgb(255, 0, 0)'      >>> gradient.hsl\_value()  'rgba(255, 100, 50, 1.0)' |

And stop using very old %s string formatting.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14 | >>> for word in 'f-strings are awesome'.split():  ...     print(f'{word.upper():~^20}')  ~~~~~F-STRINGS~~~~~~  ~~~~~~~~ARE~~~~~~~~~  ~~~~~~AWESOME~~~~~~~    >>> # f-strings support any Python expressions inside the curly braces  >>> name = 'f-string'  >>> print(f"My cool string is called {name.upper()}.")  My cool string is called F-STRING.    >>> a, b = 5, 7  >>> f'{a}/{b} = {a/b:.2}'  '5/7 = 0.71' |

Django

CommandError: You must set settings.ALLOWED\_HOSTS if DEBUG is False.

Hal initerjadi Ketika di settings.py Django kitabuat debug=False, tapisettingan

ALLOWED\_HOSTS nyakitatidakatur. Jadi carafixnya: aturALLOWED\_HOSTS.

Contoh:

Untukiptertentu:

ALLOWED\_HOSTS = ['localhost', '127.0.0.1', '111.222.333.444', 'mywebsite.com']

Untuksemuaip:

ALLOWED\_HOSTS = ['\*']

Jika terjadi error sslsaat install library simple jwt (ataumungkin library yg lain), perhatikanapasajaygkita install saat proses berjalan.

Contohkemarin di server 227 install simple jwt, ternyata di ada requirement install yg lain. Di 27 itukita install manual filenya, karenatidakada internet di 27. Oleh karenaitu, semua requirement harus install dulusecara manual juga. (untuk install secara manual, otomatis file libarynyaharus di sediakandulu)

Static File Django

Static file dalamhaliniakanberisicss, javascript, gambar, dan lain-lain yang bersifat static. Untukmenambah static file kitabisalangsungmenambah static file dimasing-masingaplikasi. Dalamcontohmysite, kitaakanmembericsskepadaaplikasi polls. Ikutcara di bawahini:

1. Buat 1 folder di dalamaplikasi polls Namanya **static** kemudian di dalam folder static tambahkansebuah folder dengannama polls. Tujuandibuatlagi folder adalahuntukmenghindariadaconfilic name denganaplikasilain, jikakedepanada file yang sama.
2. Kemudian di dalam folder static/polls/ tambahkansebuah file dengannama style.css

Kemudianisidengan script berikut:

**lia** {

**color**: **green**;

}

1. Di index.html (polls/template/polls/index.html), modify code yang ada di dalamnya. Tambahkanperintah : {% load static %}

Kemudian di header dari html nyatambahkan style.css dengancara:

<**link**rel="stylesheet"type="text/css"href="{%**static**'polls/style.css'%}">

Akan terlihatsepertiberikut:

{% load static %}

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Document</title>

    <link rel="stylesheet" type="text/css" href="{% static 'polls/style.css' %}">

</head>

<body>

Kemudiancobajalankanaplikasikitadari browser:

<http://127.0.0.1:8000/polls/>

tulisan list kitaharusnyasudahberwarnahijausesuai yang kitabuat di style.css

penjelasan:

{%**load**static%}

Akan mengarahkepada folder static yang ada di aplikasikita, sesuai file htmlnyaberada. Jika tidakada folder static di dalam folder aplikasi (1 aplikasidengan file.html nya), maka Django akanmencarike folder utama, jikatidakketemu juga, maka aka nada error.

Untukmenambahgambar, kitacukuptambahsebuah folder di dalam polls dengannama images, kemudianletakkansebuahgambardengannama background.gif

Di dalamcsskitaberiperintahberikut:

body {

background: white url("images/background.gif") no-repeat;

}

Load static pengaturannyaada di file settings.py, di bagianberikut:

STATIC\_URL = '/static/'

**ENCRYPTION**

Rumus :

Encrypt : encrypt Key (str) = str\_encrypted

Decrypt: decrypt Key(str\_encrypted) = str

Catatan: pastikan key nyasama. Waktu generate key pertama kali, kitaharussimpansupayakitabisagunakan pada saatinginimengembalikan (decrypt) encrypted str

b = bytes(mystring, 'utf-8')

b = mystring.encode('utf-8')

b'gAAAAABf02oB5TEGbaoOQD4R3FlHB8c42xoKW2WNfNPKsIu1t6EpgfgqyVpzb3EIwxVdfoD2PMnyVH7LlOsk9DSl3LQd77Y5INUyI-5d6rMvSrkO18pbBr0='

cara:

from cryptography.fernet import Fernet

key = Fernet.generate\_key() #this is your "password"

cipher\_suite = Fernet(key)

encoded\_text = cipher\_suite.encrypt(b"Hellostackoverflow!")

decoded\_text = cipher\_suite.decrypt(encoded\_text)

password : 5.212

gAAAAABf027DbLGRsCYN0dxiBLjqOY7ql6ABCRuJWMAD6BCgBItG1uI6tGMB7Y-JYJyhXPTBYWZX0Vfj13O86dzqS0XFEew0MrAuOOAZa7pp6BcmdfpG420=

iniadalahcontoh key ygsudahkita generate denganerintah:

key = Fernet.generate\_key()

print (key)

output:

b'e0bau9o-eoxHhbSlQFGXNq9FRFvaQ1VPKXZCKkssdno='

kemudian key itukitasimpan, yang natinyakitagunakanuntukmendecrypthasilygSudahkita encrypt

key chiper\_suite=

b'e0bau9o-eoxHhbSlQFGXNq9FRFvaQ1VPKXZCKkssdno='

itsol@2020

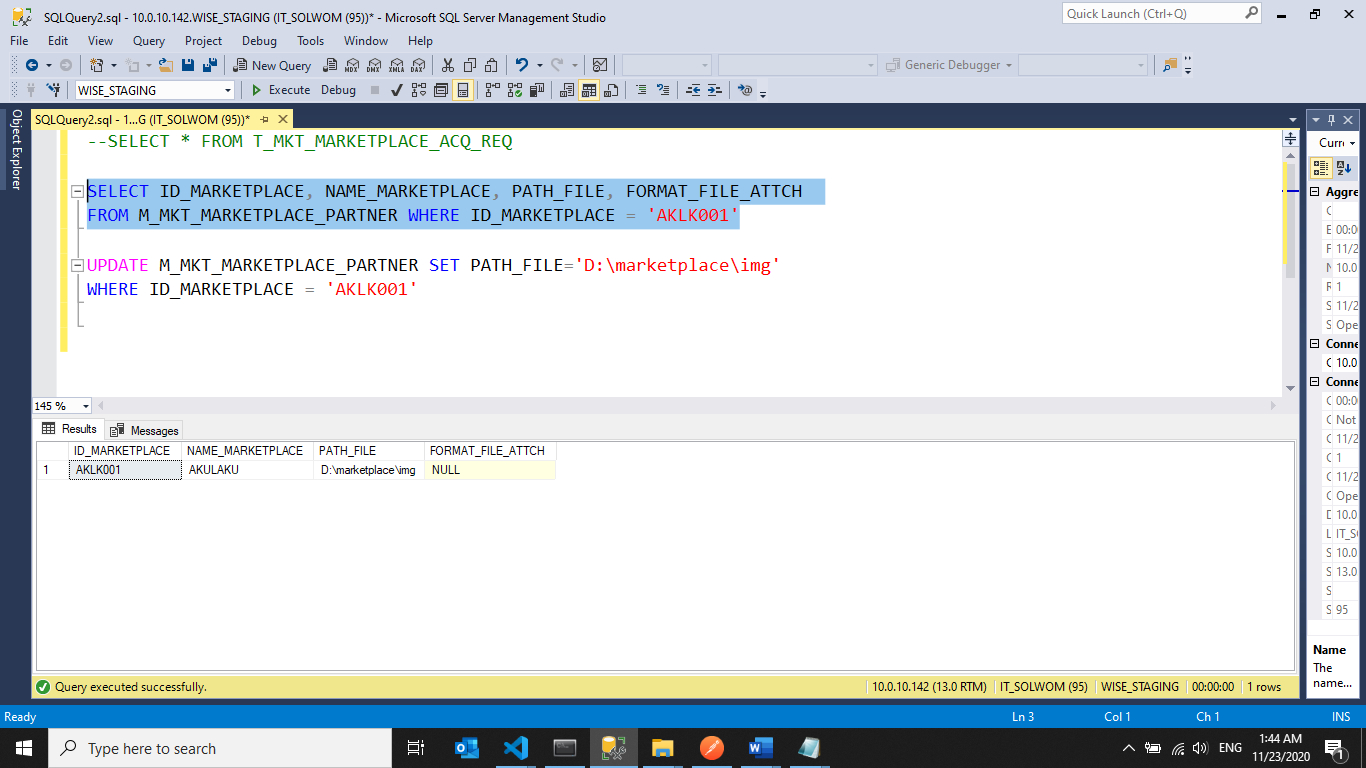
Error Ketika menguploadgambar di Django:

marketplace/10\_KTP\_I033qes.jpeg

namaseharusnyaadalah 10\_KTP.jpg

ituberubahjadi 10\_KTP\_I033qes.jpeg

ituadatandanyaadayg salah. Ituternyatasetelah di check path directorynya di tabel parameter masih none.



Sebelumnyakolompath\_filenyamasih NULL

Cara mengupdatenilaidari field object ygdirelasikanmelalui field relasi di Django:

Contoh:

Object ygakandirelasikan:

class Choice(models.Model):

Question        = models.ForeignKey(Question, on\_delete=models.CASCADE)

Object utama

class Question(models.Model):

    question\_text   = models.CharField(max\_length=200)

Penjelasan:

Object choice memiliki relation ke object Question. Jadi Ketika kitainginmengakses object Question dari object/model Choice, kitabisacukupmemanggilnyadengan:

Nama\_model+ \_set:

Contoh:

choice\_set.all()

choice\_set.get(pk=1)

dll.

Sedangkanuntukmengupdatenyakitabisadengancara:

selected\_choice = question.choice\_set.get(pk=request.POST['choice'])

selected\_choice.votes += 1

selected\_choice.save()

Dan error ygkitatemukan di error\_log.txt adalah:

expected str, bytes or os.PathLike object, not NoneType

itubersumberdari code saatkitainginmemindahkan file tersebut.

tmp\_file = os.path.join(settings.MEDIA\_ROOT, path)

dst\_file = os.path.join(path\_attachment,file\_name)

shutil.move(tmp\_file, "%s"%(dst\_file))

caramendapatkan data dari form ygdikirimlewat forms

request.POST[‘NAME\_FIELD\_FORM’]

contoh:

**request.POST['choice']**

**socket**

ketika menjalankan server, kemudian kita ingin men-stop nya, caranya adalah di cmd server kita tekan ctrl + c sebanyak 2 kali. Setelah kita tekan itu, tidak akan ke close. Supaya ke close adlah, kita perlu melakukan request lagi dari cmd client

# [“No installed app with label 'admin'” running Django migration.](https://stackoverflow.com/questions/29565665/no-installed-app-with-label-admin-running-django-migration-the-app-is-insta)

Error sepretiitukitatemukan Ketika, kitabarumengganti database di settings.py, tapibelum migrate.

Ataumungkinkarenasettingan database, tidakada driver ygcocokterinstall. Contohkita setting databasenyakesql server, sementarauntukpenyambung/extentions Django dengansql server belumada (mssql-backend).

TypeError: \_\_str\_\_ returned non-string (type Question)

Inimungukindisebabkan oleh def \_\_str\_\_ yg di models Django kita, kita return field type many2one ataupun relation field. Jadi pastikan field yg di return bukan field relation, sebabitufieldnyaadalahberupa intention (object)

ERROR KETIKA file di ftp remote tidak ada. Ini bisa karena salah directory di table partner tidak di setting dengan benar. Untuk menampilkan pesan ini, harus debug = True di file settings.py django

File "D:\Project\marketplace\server63\MARKETPLACE\_API\63\_env\lib\site-packages\django\core\servers\basehttp.py", line 171, in handle self.handle\_one\_request() File "D:\Project\marketplace\server63\MARKETPLACE\_API\63\_env\lib\site-packages\django\core\servers\basehttp.py", line 179, in handle\_one\_request self.raw\_requestline = self.rfile.readline(65537) File "C:\Users\HENROSU\_000205123\Anaconda3\envs\nama\_enviroment\lib\socket.py", line 589, in readinto return self.\_sock.recv\_into(b) ConnectionResetError: [WinError 10054] An existing connection was forcibly closed by the remote host

Module

Untuk menjalankan/memanggil sebuah module, sebaiknya di luar module itu sendiri.

Contoh:

Main.py

Modules/

moduleA.py

module.py

yang dimana isinya seperti berikut:

moduleA.py

#!/usr/bin/env python3

# Exported function

def as\_int(a):

return int(a)

# Test function for module

def \_test():

assert as\_int('1') == 1

if \_\_name\_\_ == '\_\_main\_\_':

\_test()

moduleB.py

#!/usr/bin/env python3

from .mymodule import as\_int

# Exported function

def add(a, b):

return as\_int(a) + as\_int(b)

# Test function for module

def \_test():

assert add('1', '1') == 2

if \_\_name\_\_ == '\_\_main\_\_':

\_test()

**Perhatikan:**

moduleA.py dan module.py merupakan module yang sama-sama berada di dalam folder/package modules. Seperti yang kita perhatikan di dalam **moduleB** kita mengimport **moduleA :** from .mymodule import as\_int. hal ini akan menyebabkan error di python: **SystemError: Parent module '' not loaded, cannot perform relative import**.

Oleh karena itu memanggil sebuah module, seharusnya dari luar sebuah package itu sendiri. Contoh dari main.py

#!/usr/bin/env python3

from mypackage.myothermodule import add

def main():

print(add('1', '1'))

if \_\_name\_\_ == '\_\_main\_\_':

main()

namun jika kita ingin tetap memanggil dari dalam package yang sama, dalam hal ini adalah dari moduleB.py, caranya supaya errornya tidak muncul adalah dengan menghapus tanda dot nya.

Contoh :

Dari :

from .mymodule import as\_int

menjadi :

from mymodule import as\_int

contoh multiprocess di python

action\_process = Process(target=execute\_send\_customer)

        # We start the process and we block for 5 seconds.

        action\_process.start()

        action\_process.join(timeout=20)

        # We terminate the process.

        action\_process.terminate()

        print("Hey there! I timed out! You can do things after me!")

cara mendapatkan sumber code dari sebuah function, library, atau yg lainnya:

import inspect

inspect.getsource(nama\_source)

contoh:

import inspect

from modules import Math\_module

import requests

print (inspect.getsource(requests.sessions.Request))

**ZIP**

zip digunakan untuk membentuk 2 list menjadi key dan value berdasarkan urutan.

Rumus:

zip(list\_1, list\_2)

contoh:

a = ['nama', 'alamat']

b = ['Andi', 'Siantar']

c = zip(a, b)

jika kita print c :

print (c),

maka hasilnya adalah object:

<zip object at 0x000002C1D46E4B40>

untuk melihat hasil sebenarnya, maka kita harus convert dulu ke mejadi dictionary atau tuple.

contoh:

hasil\_akhir = dict(c)

>>> hasil\_akhir = dict(c)

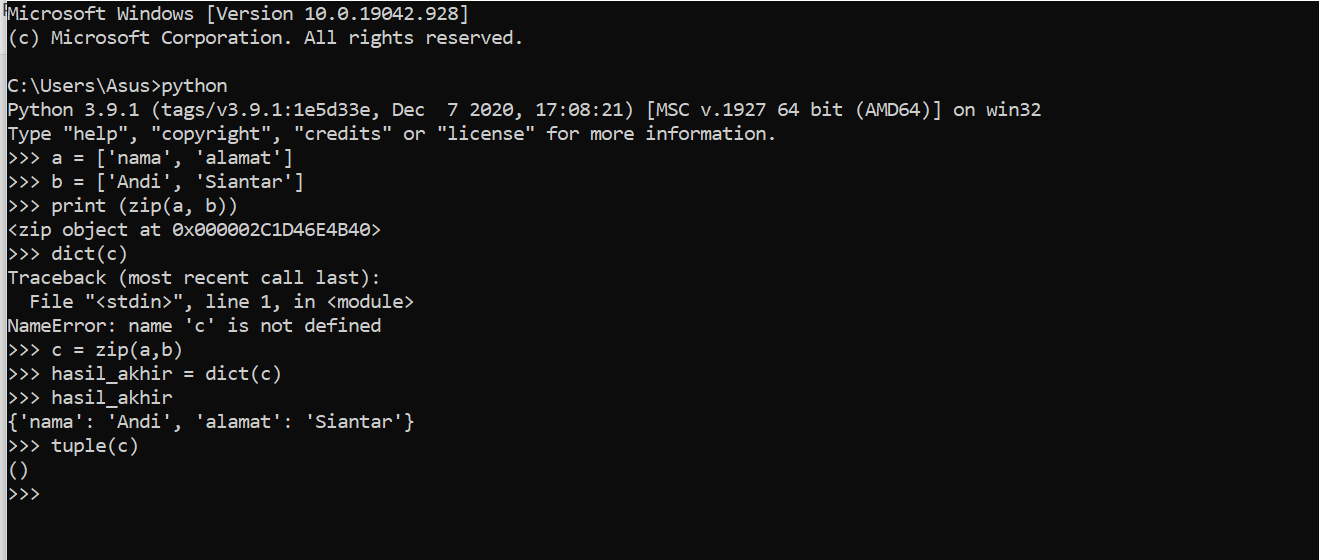
>>> hasil\_akhir

{'nama': 'Andi', 'alamat': 'Siantar'}

Perlu diketahui, setelah kita convert variable C itu k edict, maka variable c itu otomatis menjadi kosong lagi. Sehingga ketika kita ingin menkovert variable C itu ke tuple, maka otomatis hasilnya adalah kosong.

>>> tuple(c)

()



Map function

Ini hasil dari function juga object, sama seperti zip function. Oleh karena itu kita harus mengconvertnya ke list atau dictionary dulu

**Set**

Set merupakan collection atau sejenis type data array tapi tidak ada duplicate data.

Datanya berada di dalam tanda {}, mirip dengan dictionary. Bedanya adalah dictionary terdiri dari key dan value, sementara langsung value.

Cth:

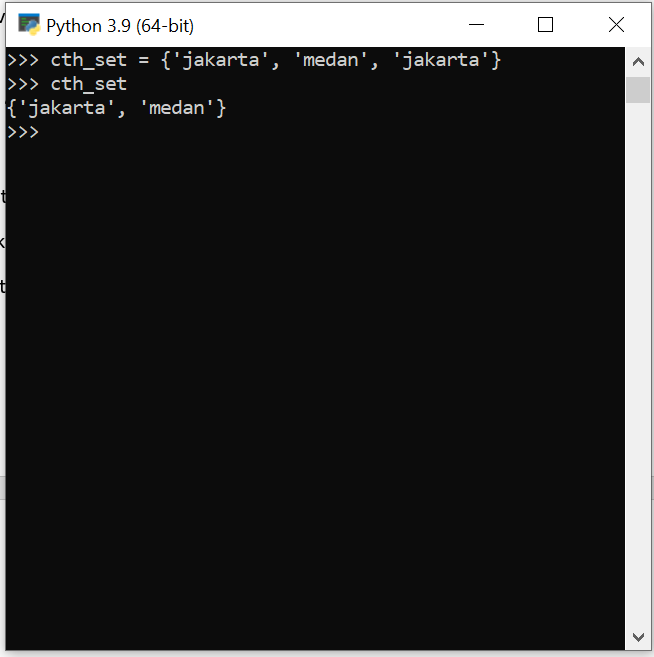
Cth\_Dictionary = {‘nama’: ‘rudi’, ‘alamat’: ‘Jakarta’}

Cth\_Set = {‘rudi’, ‘jakarta’}

Contoh yang tidak boleh duplicate:

cth\_set = {‘jakarta’, ‘medan’, ‘jakarta’}

print (cth\_set) # output: Jakarta, medan. Jakarta yang paling akhir tidak ditampilkan.



Mencari index di list berdasarkan nilainya:

# vowels list

vowels = ['a', 'e', 'i', 'o', 'i', 'u']

# index of 'e' in vowels

index = vowels.index('e')

print('The index of e:', index)

cara mengkoneksikan python ke sql server di local:

import pyodbc

conn = pyodbc.connect('Driver={SQL Server};'

                      'Server=localhost;'

                      'Database=wise\_staging\_local;'

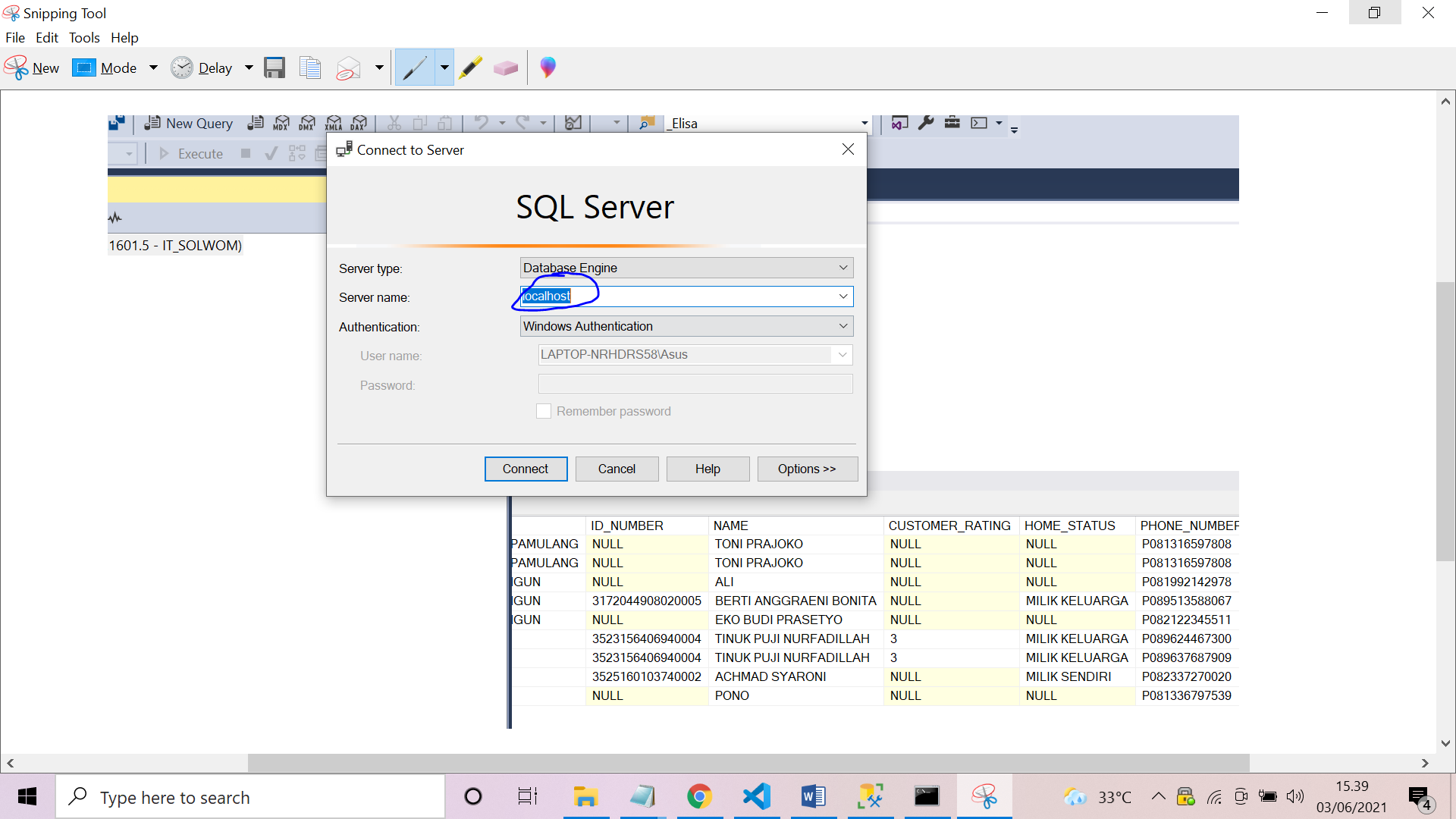
                      'Trusted\_Connection=yes;')

cursor = conn.cursor()

cursor.execute('SELECT \* FROM RPT\_LIST')

for row in cursor:

    print(row)

nama servernya adalah diambil dari gambar berikut:  


send email dengan python:

import smtplib

smtpserver = smtplib.SMTP("smtp.gmail.com", 587)

smtpserver.ehlo()

smtpserver.starttls()

smtpserver.ehlo()

resp = smtpserver.login('henrotanjung@gmail.com', 'K4cih789123')

msg = 'test pesan'

smtpserver.sendmail('henrotanjung@gmail.com', 'h3nro777@gmail.com', msg)

smtpserver.quit()

print (resp)

nb:

pastikan account email untuk login smtp nya , menggallow less secure access dan non-aktifkan two-authentication key

experience error

I am trying to debug python code, I want to pin point the line number in which error occurs. As per the posts found [asked here](https://stackoverflow.com/questions/1278705/python-when-i-catch-an-exception-how-do-i-get-the-type-file-and-line-number) the code gives the line no of the function being called. eg

if \_\_name\_\_ == '\_\_main\_\_':

try:

foo()

except:

<the code to print line no when error occurs>

But it gives me the line no of foo(), Please help to find the exact line no in which error occurs.

Thanks,

Jawaban, fix:

import sys, os, traceback

try:

raise NotImplementedError("No error")

except Exception as e:

exc\_type, exc\_obj, exc\_tb = sys.exc\_info()

tb = traceback.extract\_tb(exc\_tb)[-1]

print(exc\_type, tb[2], tb[1])

error\_experience

pyodbc.DataError: ('22001', '[22001] [Microsoft][ODBC Driver 17 for SQL Server][SQL Server]String or binary data would be truncated. (8152) (SQLExecDirectW); [22001] [Microsoft][ODBC Driver 17 for SQL Server][SQL Server]The statement has been terminated. (3621)')

ini biasanya disebabkan oleh karena maksimal character yang di setting di type data field table , lebih sedikit dari yang akan di inser maupun diupdate dari query.

**ImportError: attempted relative import with no known parent package**

Ini disebabkan karena kita menjalankan sebuah file python di folder yang sama dengan file module yang kita import dari file yang kita jalankan tadi.

Contoh di folder yang sama:

my\_module.py

main.py

di file main.py , kita ada meng-import my module (baik class, maupun attribute yang lain).

Cara importnya pakai tanda titik (**.**)

Contoh:

from .buah import Fruit

Ketika kita execute file main.py : python main.py , maka error seperti di atas akan muncul. Python 3 hanya mengijinkan kita mengimport diluar dari folder itu.

Supaya tidak error, caranya adalah:

Tanda titik (**.**) tidak perlu diikutkan, cukup langsung :

from buah import Fruit

tapi cara yang benar adalah dengan mengimport dari luar beda folder

**TypeError: a bytes-like object is required, n**

Ini disebabkan Karen yang dibutuhkan adalah object, tapi kita provide dengan data lain (str, dict, dll).

Cth:

Ketika kita membuka sebuah file dengan mode rb:

with open(fname, 'rb') as f:

kemudian kita write dengan argument text: fname.write(‘ini isinya’). Error diatas akan muncul. Yang benar adalah kita harus membuat argument textnya dalam bentuk byte juga. Yaitu menambah b didepan string nya (diluar tanda petik 1 atau 2). Cth: fname.write(b’ini isinya’)

cnxn\_ws = pyodbc.connect('DRIVER={ODBC Driver 13 for SQL Server};SERVER='+server+';DATABASE='+database\_ws+';UID='+username+';PWD='+ password) pyodbc.InterfaceError: ('IM002', '[IM002] [Microsoft][ODBC Driver Manager] Data source name not found and no default driver specified (0) (SQLDriverConnect)')

karena driver odbc Driver 13 for SQL tidak ditemukan. Check driver dari button start windows , kettikkan odbc

dipunya saya hanya ada versi 17

