En este enlace encontrarás la documentación de la API de una tienda de mascotas: https://petstore.swagger.io/

*1. Crea tu usuario mediante petición HTTP y posteriormente recupera sus datos llamando al servicio correspondiente.*

**Creación de usuario**

POST Request URL: <https://petstore.swagger.io/v2/user>

Body:

{

  "id": 555,

  "username": "eabramian",

  "firstName": "Eric",

  "lastName": "Abramian",

  "email": "eyabramian@gmail.com",

  "password": "Pw\_163@",

  "phone": "687053389",

  "userStatus": 1

}

Response:

{

    "code": 200,

    "type": "unknown",

    "message": "555"

}

**Recuperación de datos llamando al servicio correspondiente**

GET Request URL: <https://petstore.swagger.io/v2/user/eabramian>

Response:

{

  "id": 555,

  "username": "eabramian",

  "firstName": "Eric",

  "lastName": "Abramian",

  "email": "eabramian@gmail.com",

  "password": "Pw\_163@",

  "phone": "687053389",

  "userStatus": 1

}

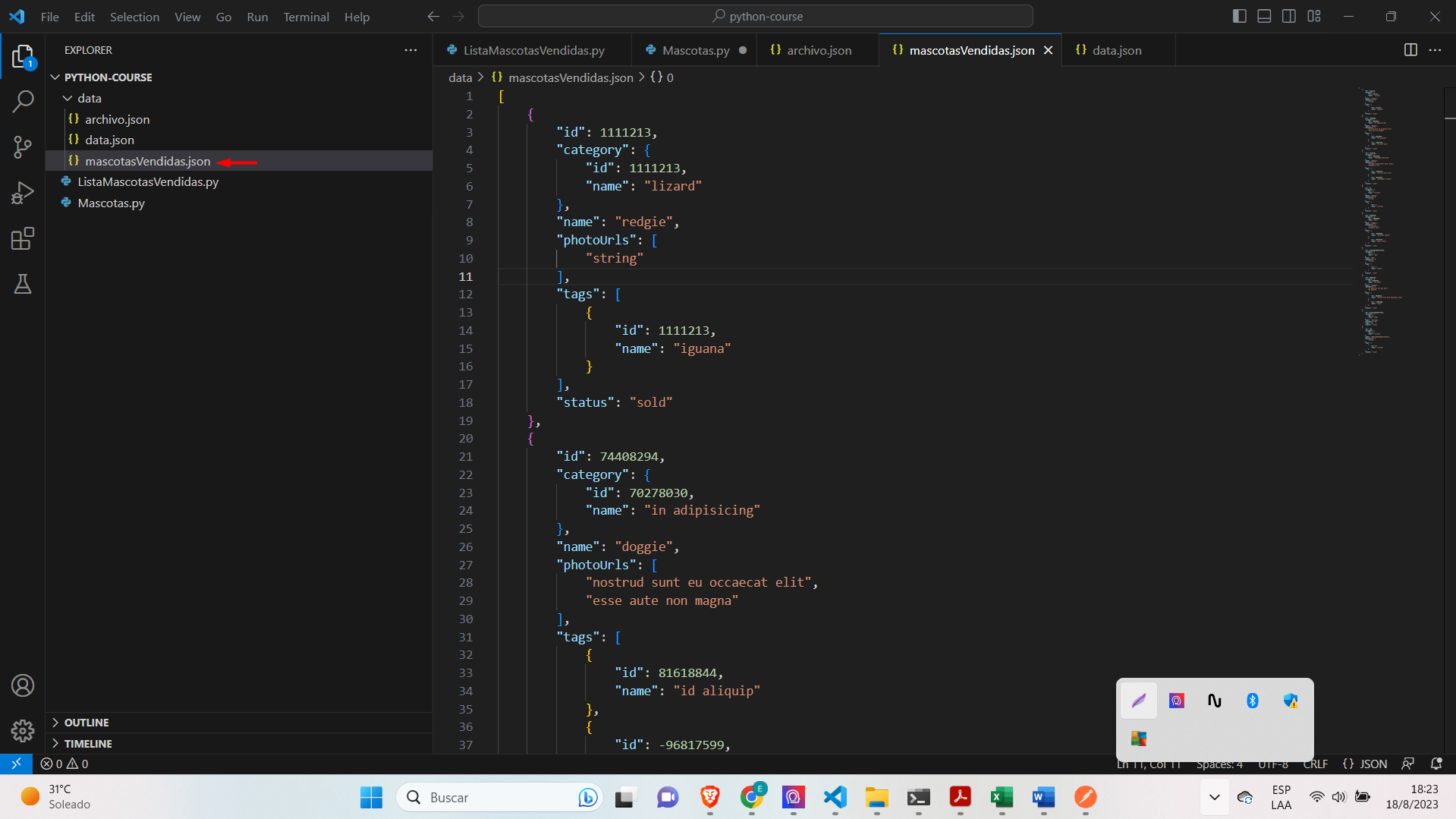
*2. Recoge mediante petición HTTP, el JSON que retorna el endpoint /pet/findByStatus y lista mediante una función los nombres de las mascotas que se hayan vendido.*

*- El formato de salida deberá estar formado por la tupla {id, name}.*

*- Puedes utilizar la estructura de datos que prefieras.*

GET Request URL: <https://petstore.swagger.io/v2/pet/findByStatus?status=sold>

-Se genera un archivo .json con el response del request ejecutado previamente y es subido al editor de código:



-Se genera en lenguaje Python la función la cual lista los IDs y nombres de las mascotas que se hayan vendido:

import json

def cargar\_datos(ruta):

    with open(ruta) as contenido:

        mascotasVendidas = json.load(contenido)

        for mascotaVendida in mascotasVendidas:

            print('ID: '+str((mascotaVendida).get('id')), 'Name: '+str(mascotaVendida.get('name')))

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

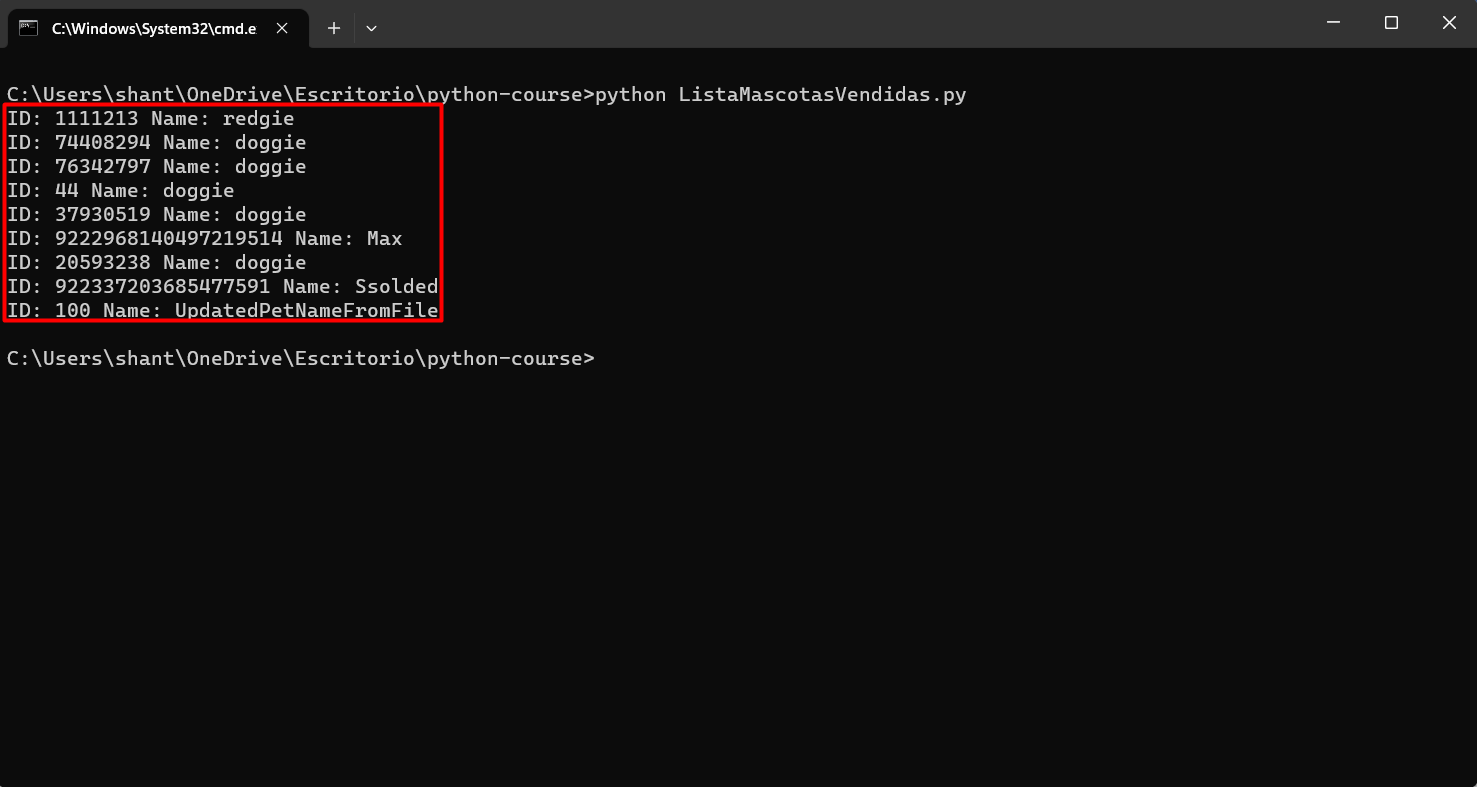
    ruta = 'data/mascotasVendidas.json'

    cargar\_datos(ruta)

Archivo de la clase:



-Se ejecuta el test para validar que figuren correctamente los IDs y nombres de las mascotas que fueron vendidas:



*3. Crea una clase cuyo constructor requiera de la estructura de datos anterior y realiza un método que pueda recorrerla para poder identificar cuantas mascotas se llaman igual.*

*- Ejemplo de salida: {“William”: 11, “ Floyd”: 2} Como output, te pediremos el código (puedes separarlo en archivos como quieras) y los resultados de salida de los puntos anteriores.*

*- Recuerda que puedes utilizar el lenguaje que prefieras y cualquier mejora adicional será bien considerada*

-Se crea la clase en lenguaje Python

from collections import Counter

import json

#Se crea la clase y constructor

class Mascota:

    def \_\_init\_\_(self, id, name):

        self.id = id

        self.name = name

def cargaDeDatos(ruta):

    with open(ruta) as contenido:

        mascotasVendidas = json.load(contenido)

    mascotas = []

    for item in mascotasVendidas:

        id = item['id']

        name = item['name']

        mascota = Mascota(id, name)

        mascotas.append(mascota)

    return mascotas

#Funcion para encontrar nombres repetidos

def encuentraNombresRepetidos(mascotas):

    contadorNombres = Counter([mascota.name for mascota in mascotas])

    nombresRepetidos = {name: count for name, count in contadorNombres.items() if count > 1}

    return nombresRepetidos

#Define la ruta y nombre del archivo json de donde obtendra los datos

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

    ruta = 'data/MascotasVendidas.json'

    mascotas = cargaDeDatos(ruta)

#Imprime nombre y cantidad de las mascotas que se llaman igual

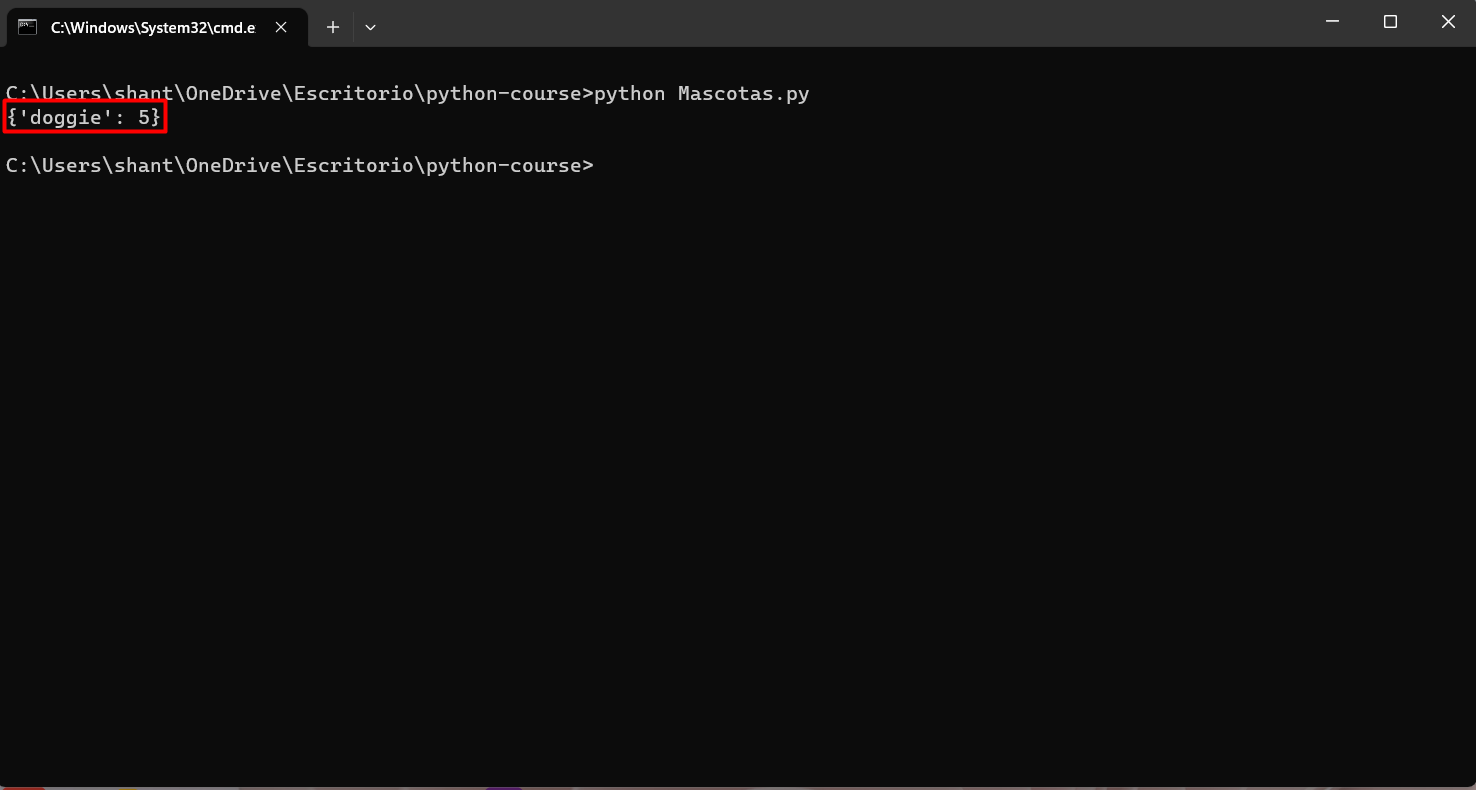
counts = encuentraNombresRepetidos(mascotas)

print(counts)

Archivo de la clase:



-Se ejecuta el test para validar la identificación de cuantas mascotas se llaman igual:



Archivo .json usado para los datos:

