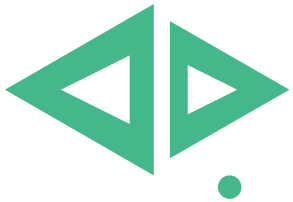
#!/usr/bin/Python

#HappyHacking💻

byNestorCode

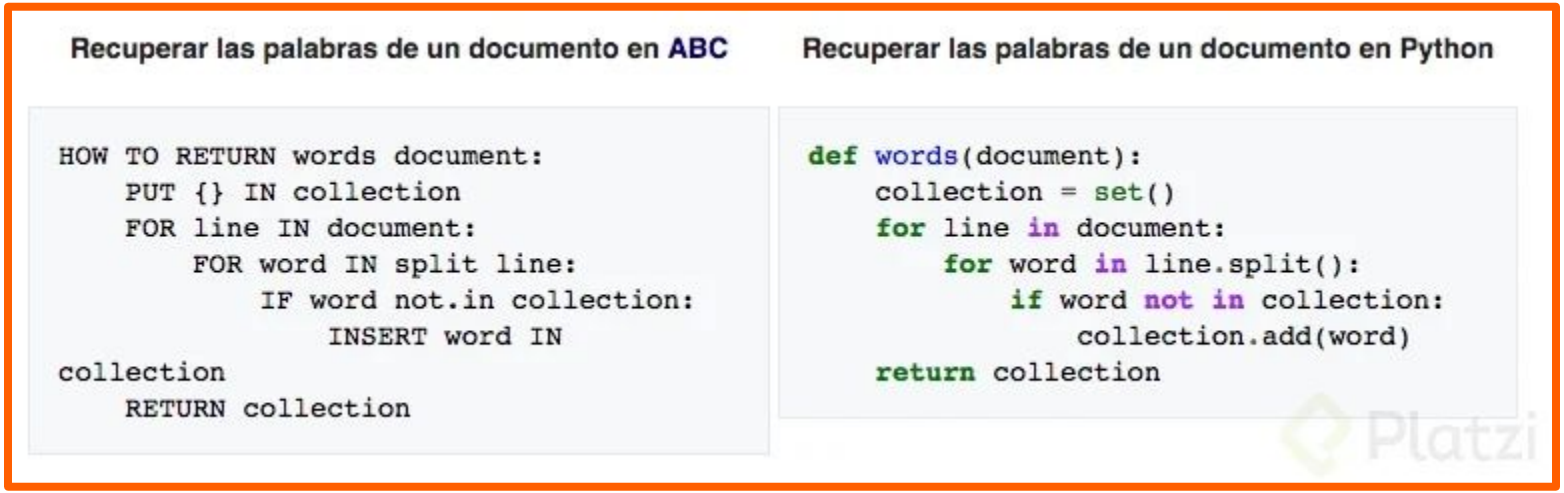
Introducción



Un poco de historia

$ Guido van Rossum

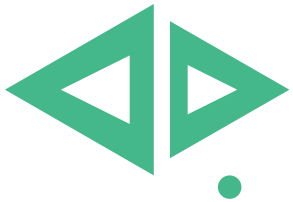
En una navidad de 1989, Guido Van Rossum, quien trabajaba en el CWI (un centro de investigación holandés), decidió empezar un proyecto como pasatiempo dándole continuidad a ABC, un lenguaje de programación que se desarrolló en el CWI.



ABC fue desarrollado a principios de los 80s como alternativa a BASIC, fue pensado para principiantes por su facilidad de aprendizaje y uso. Su código era compacto pero legible.

El proyecto no trascendió ya que el hardware disponible en la época hacía difícil su uso. Así que Van Rossum le dió una segunda vida creando Python.

A Guido Van Rossum le gustaba mucho el grupo Monty Python, por esta razón escogió el nombre del lenguaje. Actualmente Van Rossum sigue ejerciendo el rol central decidiendo la dirección de Python.



Características

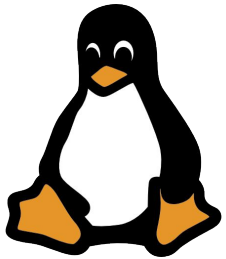
python(“Es un lenguaje de alto nivel”)

Los lenguajes de programación de alto nivel son lenguajes de programación en los cuales las instrucciones que se envían para que el ordenador ejecute algunas órdenes son parecidas al lenguaje humano. Debido a que el ordenador no puede ser capaz de reconocer estas órdenes, se necesita el uso de un guía que traduzca dicho lenguaje de alto nivel a un lenguaje de bajo nivel el cual el sistema pueda entender.

El lenguaje de programación de alto nivel usa palabras similares al inglés, así como símbolos, signos de puntuación y aritméticos de manera que permite el desarrollo de programas.

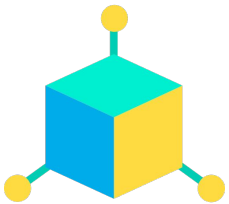
python.multiPlataforma(linux, windows, mac)

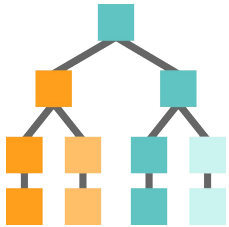
Python es multiplataforma, lo cual es ventajoso para hacer ejecutable su código fuente entre varios sistema operativos:



def multiParadigma(idea):

Es un lenguaje de programación multiparadigma, el cual soporta varios paradigma de programación como orientación a objetos, estructurada, programación imperativa y, en menor medida, programación funcional.





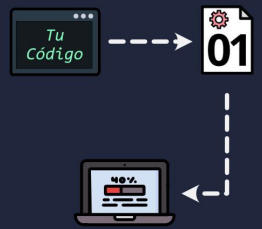
`class execute(interpreter):`

Este lenguaje ¡ya no cuenta con un compilador! El código va directo a la máquina quien ahora tiene un intérprete, que traduce el código y lo convierte a su lenguaje, entonces ¿Un compilador es lo mismo que un intérprete?, bueno, digamos que tienen la misma funcionalidad (traducir), pero su diferencia radica en que el intérprete lo realiza al momento de ejecución (cuando lo solicitas) y al ser en tiempo real puede alentar el proceso.

TIPOS DE LENGUAJE

COMPILADO

Convierte el código a binarios que lee el sistema operativo.

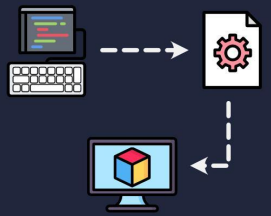


INTERPRETADO

Requieren de un programa que lea la instrucción del código en tiempo real, y la ejecute.

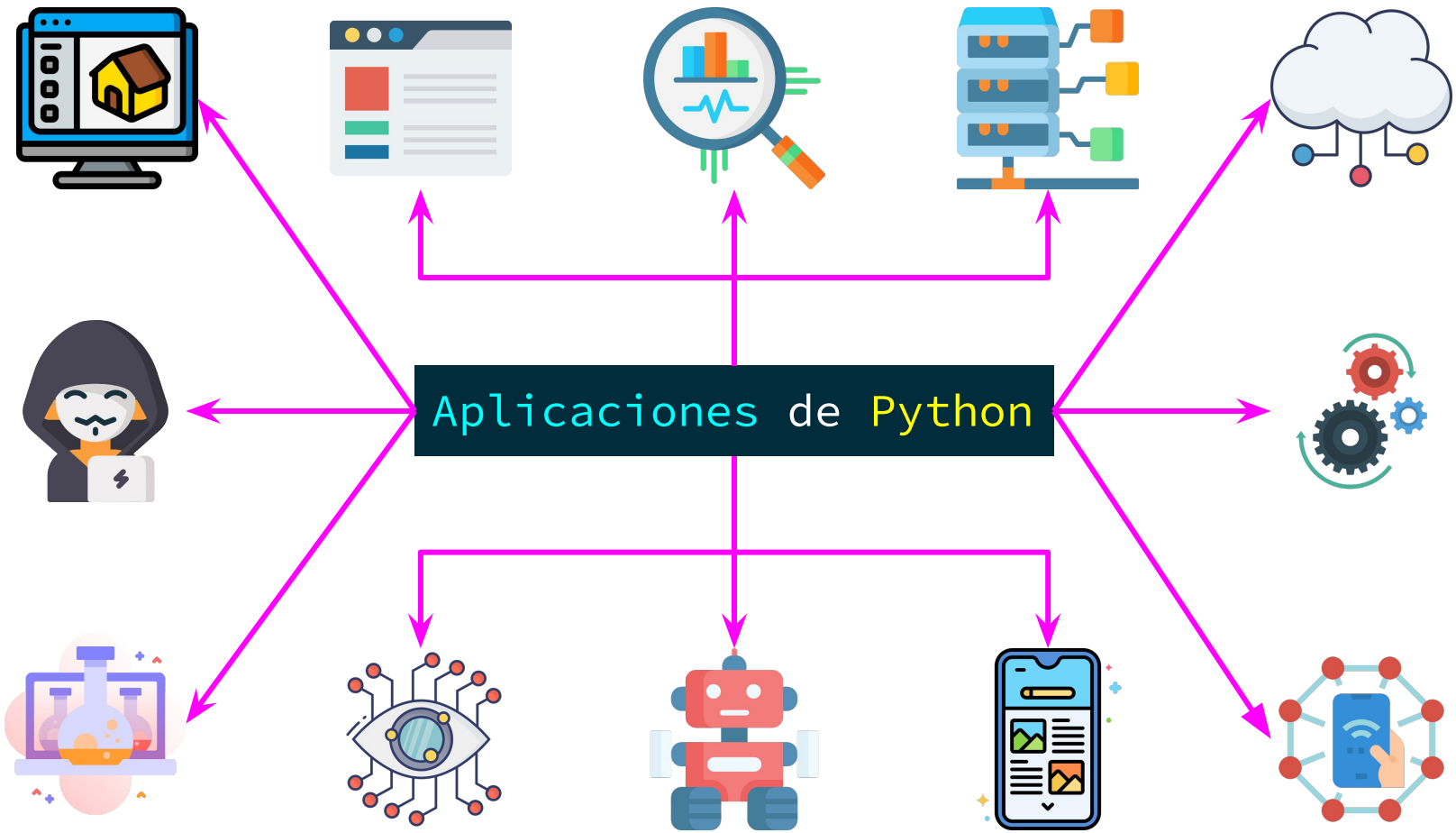
INTERMEDIO

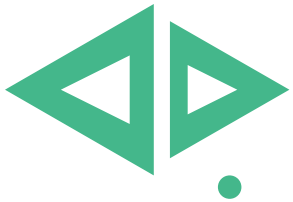
Se compila el código fuente a un lenguaje intermedio y este último se ejecuta en una máquina virtual



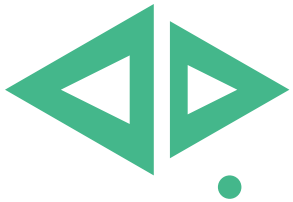
@class Caracteristicas

- Sintaxis muy clara y legible.
- Fuerte capacidad de introspección.
- Orientación a objetos intuitiva.
- Expresión del código procedimental.
- Altamente modular, soporta paquetes jerárquicos.
- Enfocado en el uso de excepciones para el manejo de errores.
- Tipos de datos dinámicos de muy alto nivel.
- Extensa biblioteca estándar (STL) y módulos de terceros para prácticamente todas las tareas.
- Extensiones y módulos fácilmente escritos en C, C + + (o Java para Jython, o. NET para IronPython).
- Integrable dentro de las aplicaciones como una interfaz de scripting.

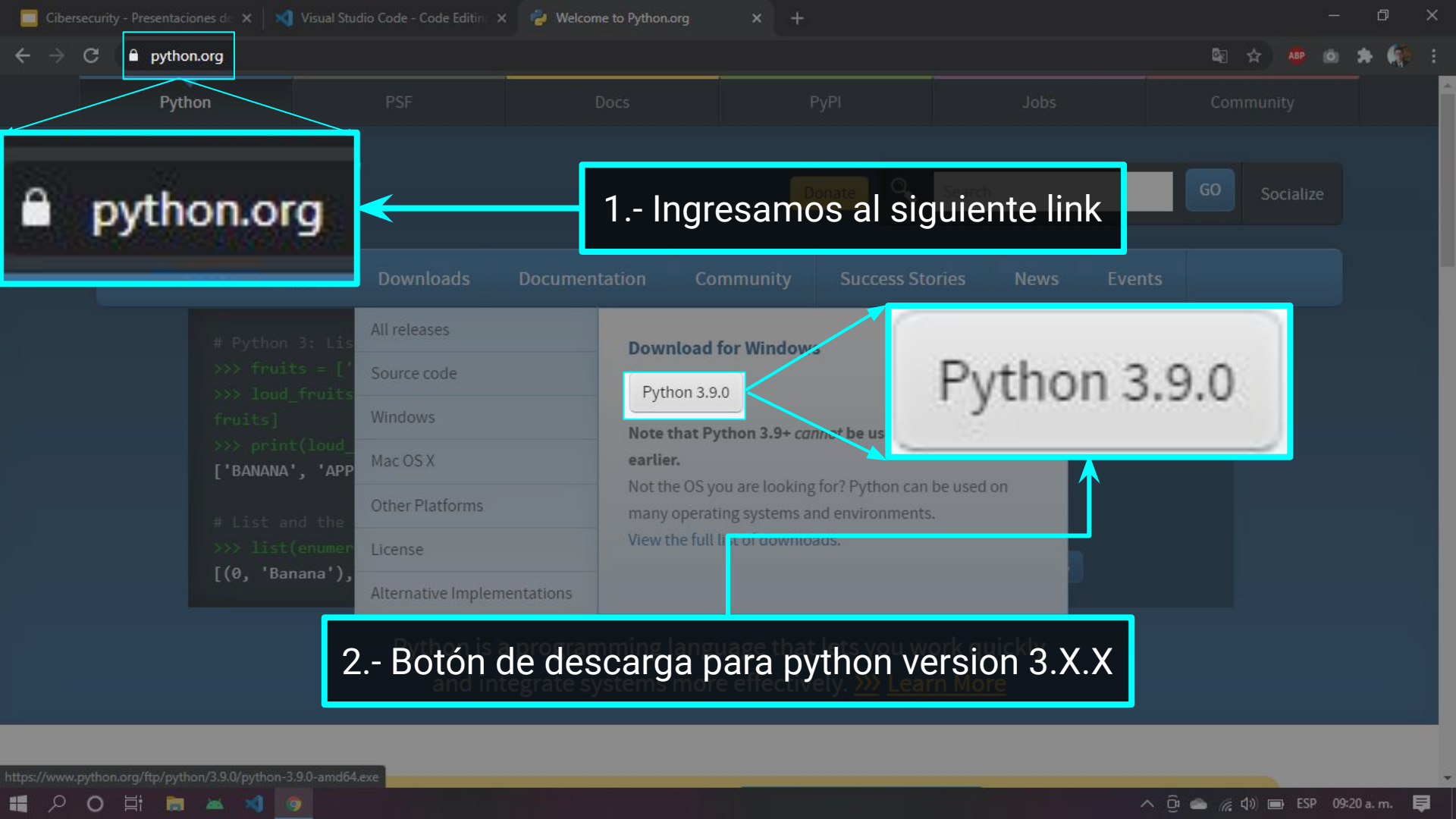

Aplicaciones de Python

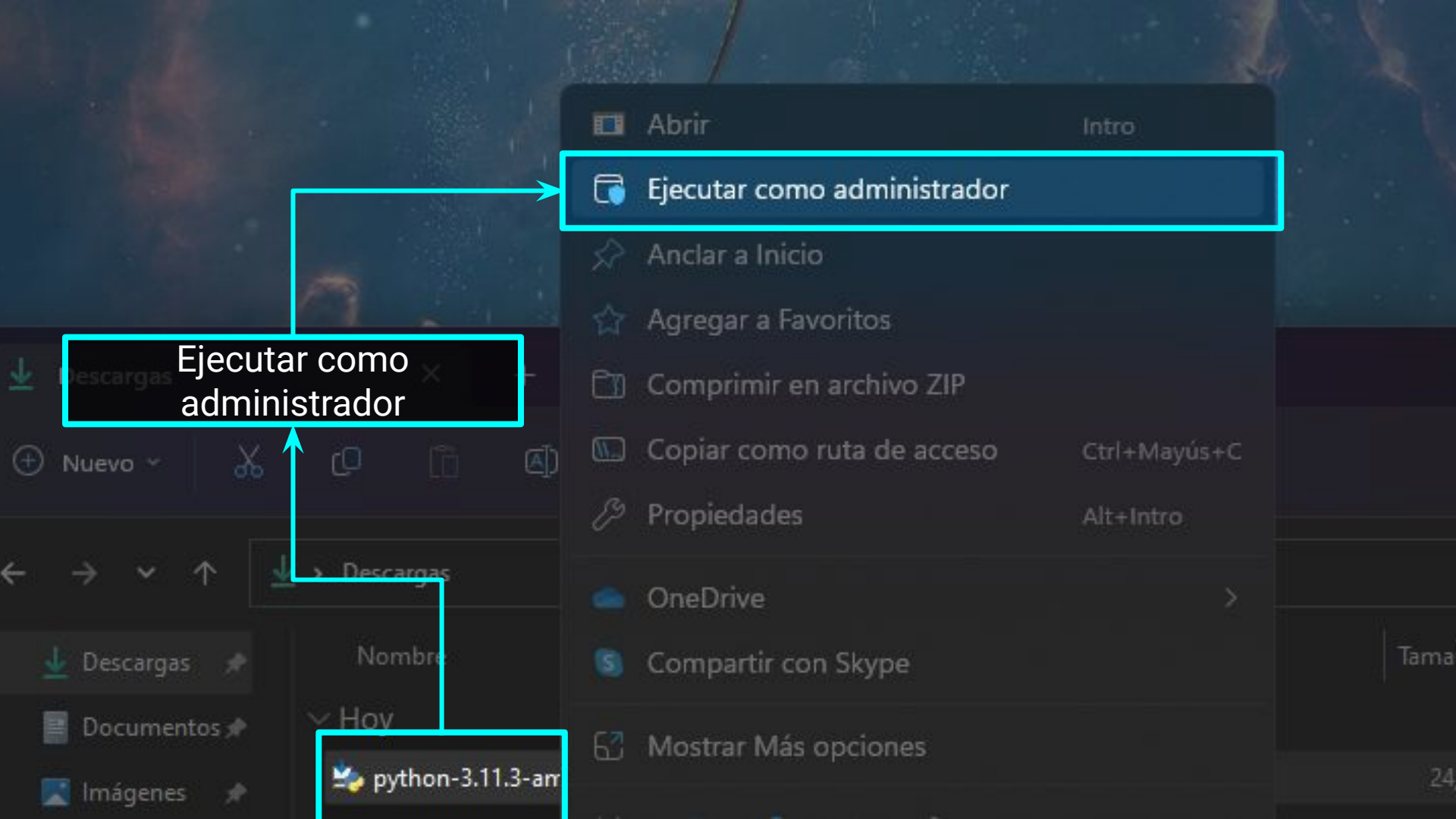


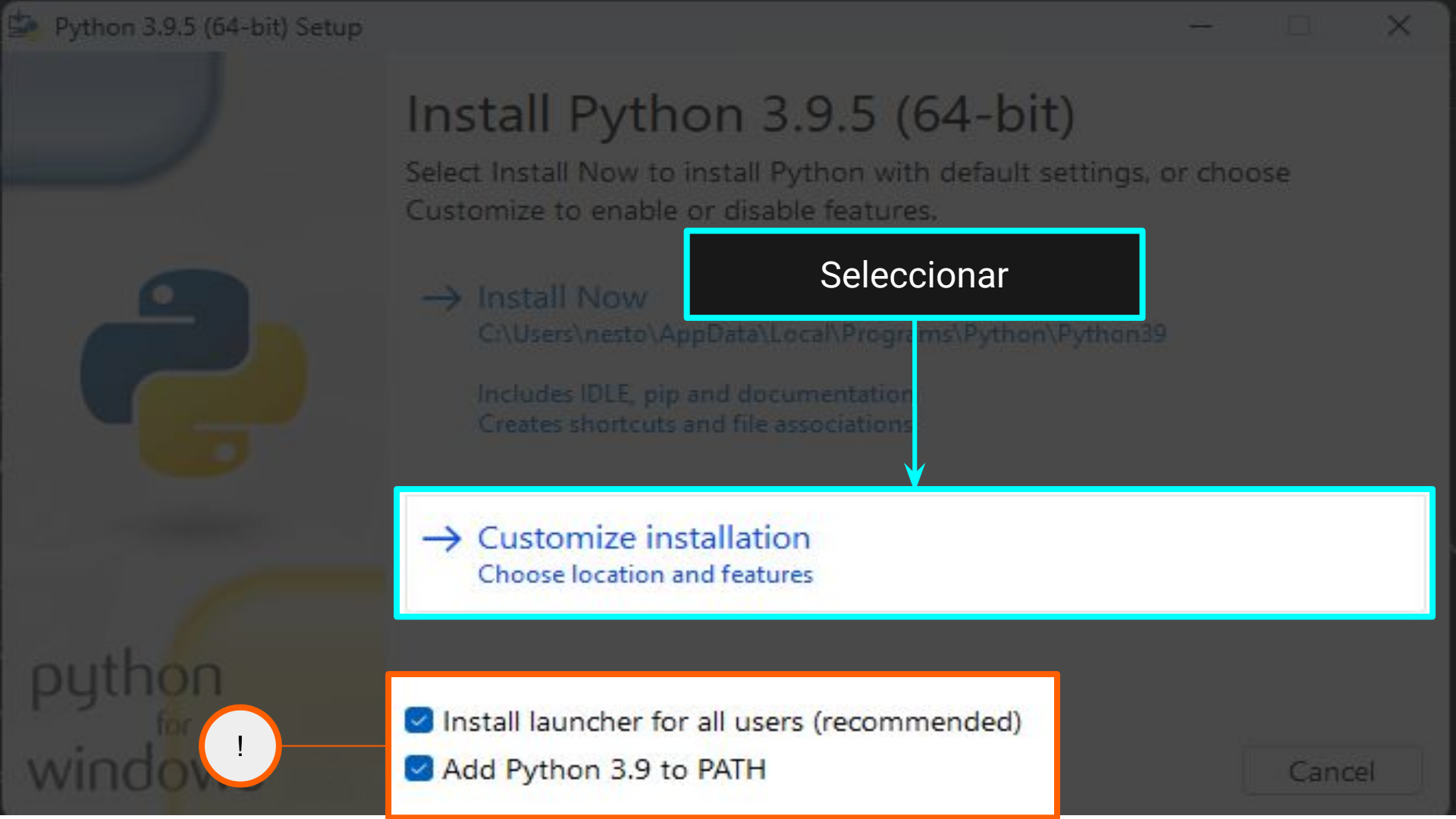
Laboratorio

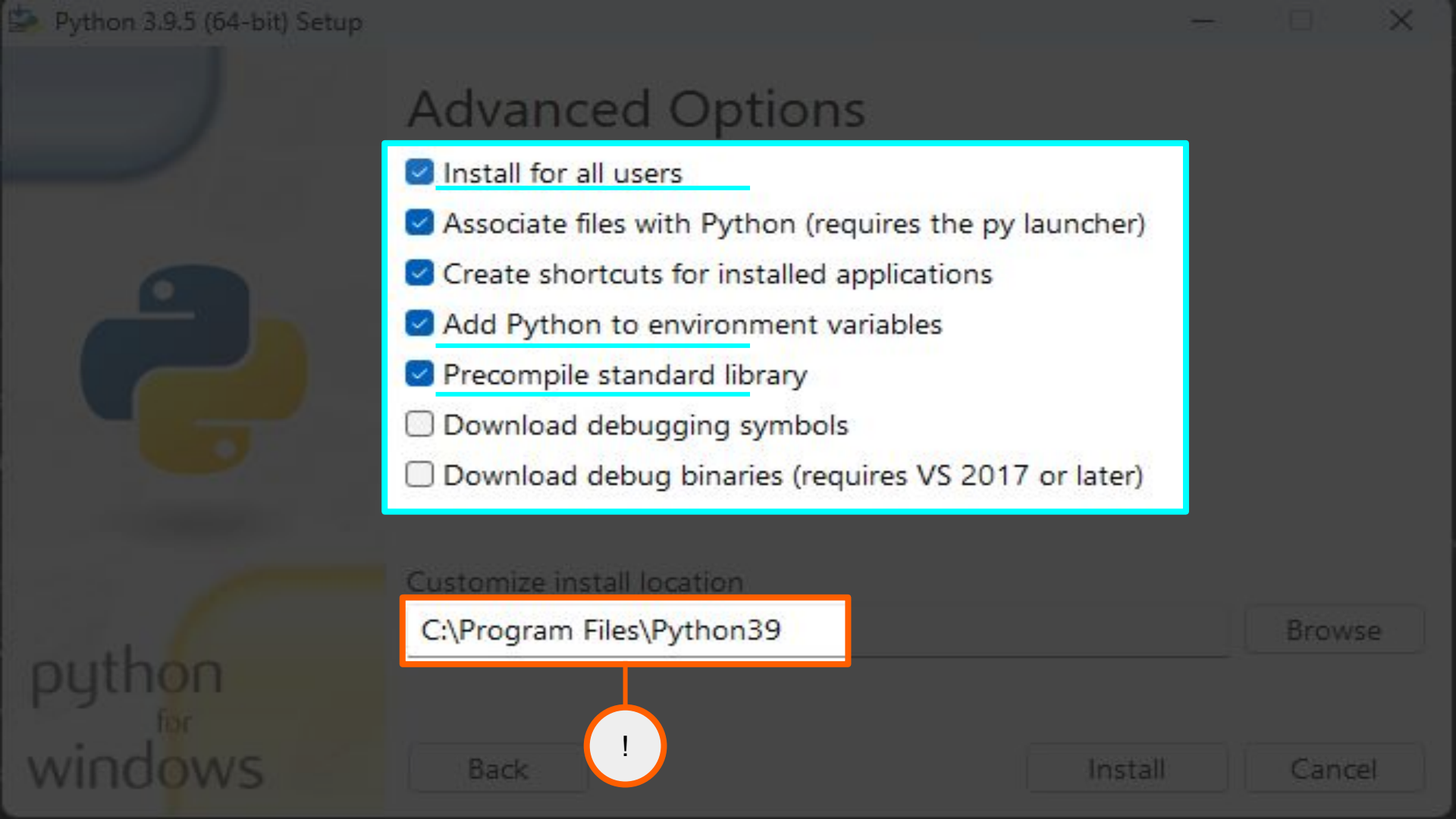


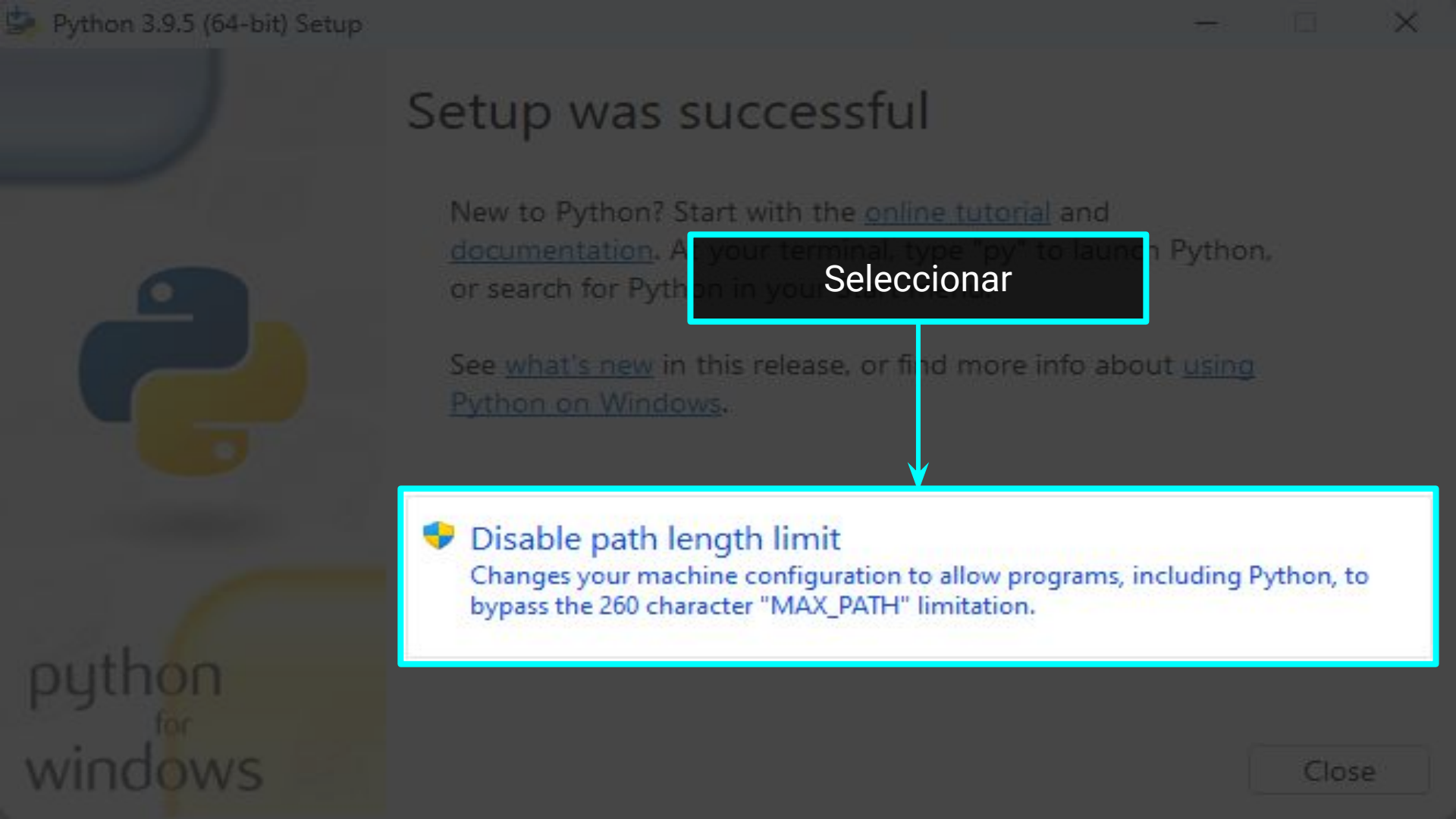
Instalar Python


python.org
1.- Ingresamos al siguiente link
Python 3.9.0
2.- Botón de descarga para python version 3.X.X

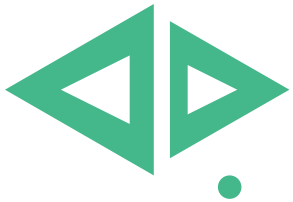

Abrir
Intro
Ejecutar como administrador
Anclar a Inicio
Agregar a Favoritos
Comprimir en archivo ZIP
Copiar como ruta de acceso
Ctrl+Mayús+C
Propiedades
Alt+Intro
OneDrive
Compartir con Skype
Mostrar Más opciones
Ejecutar como administrador
Nuevo
Descargas
Descargas
Documentos
Imágenes
Nombre
Hoy
python-3.11.3-am


Python 3.9.5 (64-bit) Setup
Install Python 3.9.5 (64-bit)
Select Install Now to install Python with default settings, or choose
Customize to enable or disable features.
Seleccionar
Install Now
C:\Users\nesto\AppData\Local\Programs\Python\Python39
Includes IDLE, pip and documentation
Creates shortcuts and file associations
Customize installation
Choose location and features
python
for
windows
!
Install launcher for all users (recommended)
Add Python 3.9 to PATH
Cancel


Python 3.9.5 (64-bit) Setup
Advanced Options
Install for all users
Associate files with Python (requires the py launcher)
Create shortcuts for installed applications
Add Python to environment variables
Precompile standard library
Download debugging symbols
Download debug binaries (requires VS 2017 or later)
Customize install location
C:\Program Files\Python39
Browse
!
python
for
windows
Back
Install
Cancel


Python 3.9.5 (64-bit) Setup
Setup was successful
New to Python? Start with the online tutorial and
documentation. A your terminal, type "py" to launch Python,
or search for Pyth n in you
See what's new in this release, or find more info about using
Python on Windows.
Seleccionar
Disable path length limit
Changes your machine configuration to allow programs, including Python, to
bypass the 260 character "MAX_PATH" limitation.
python
for
windows
Close

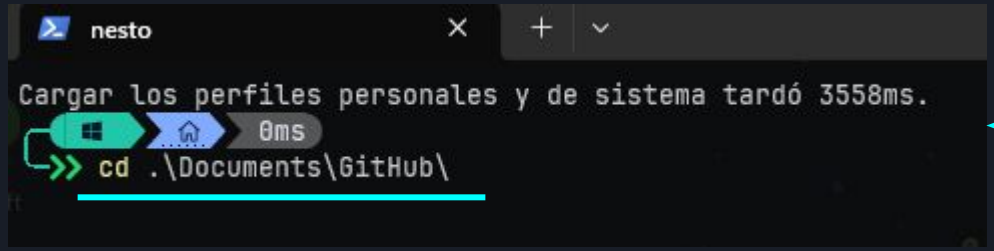

nesto
> python --version
Python 3.11.0
> python --version
Python 3.11.0
pwsh
61%
21:24:08
Comprobamos que tenemos python instalado
python --version
!
3.X.XX
Al día de hoy en la 3.9.2



Configurar entorno virtual

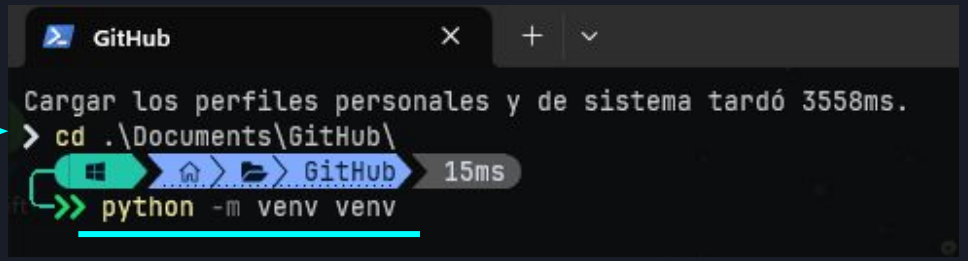
1

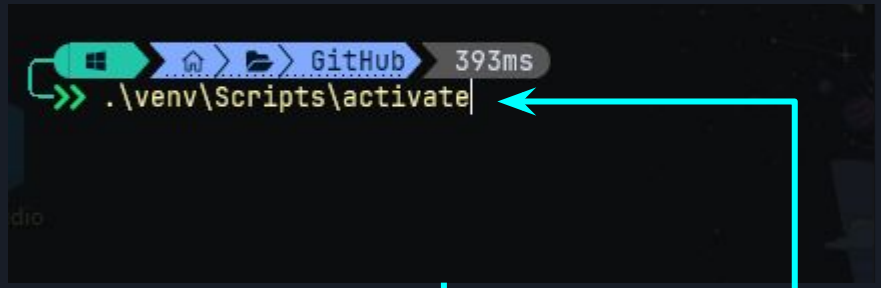
Nos dirigimos a la carpeta donde estará nuestro proyecto



2

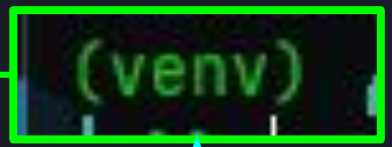
Creamos nuestro entorno virtual

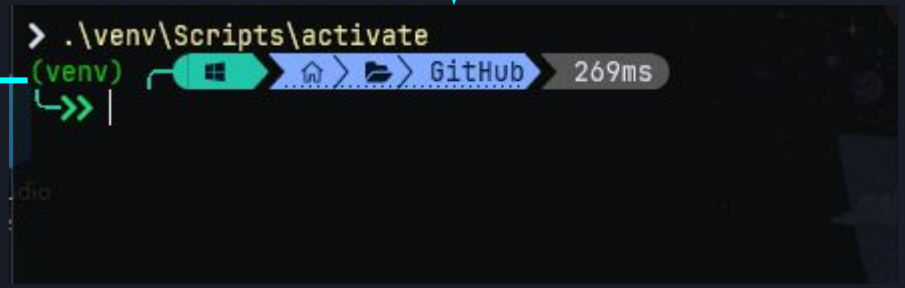


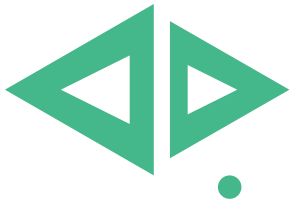


3

Activamos nuestro entorno virtual de la siguiente manera en Powershell o CMD

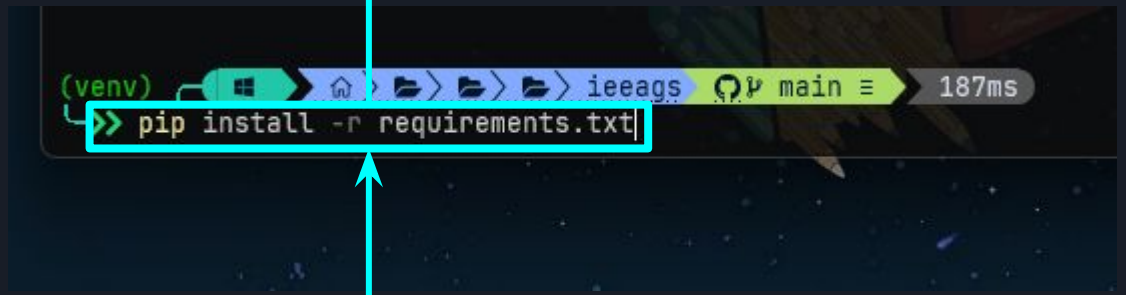






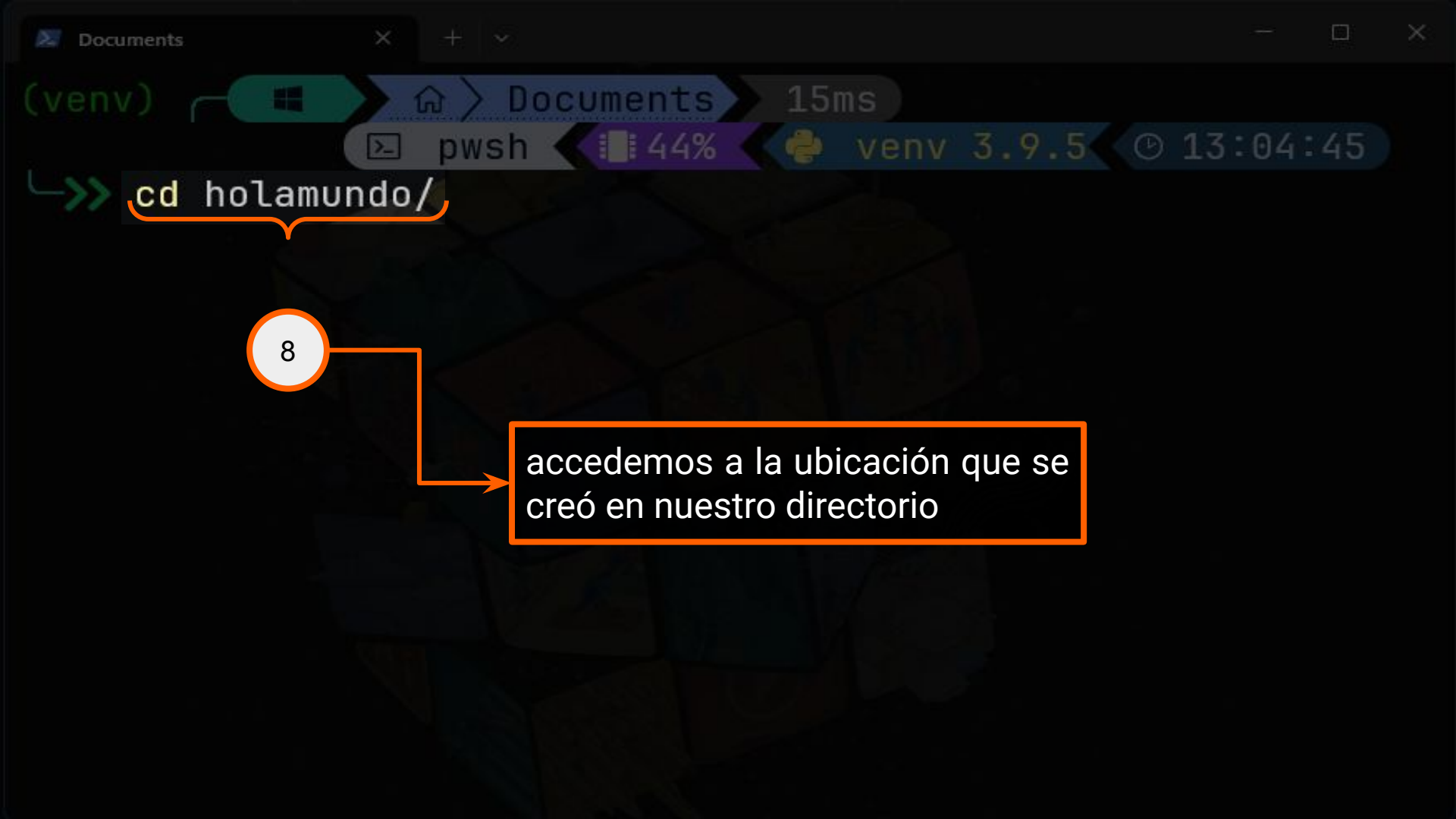
Crear espacio de trabajo

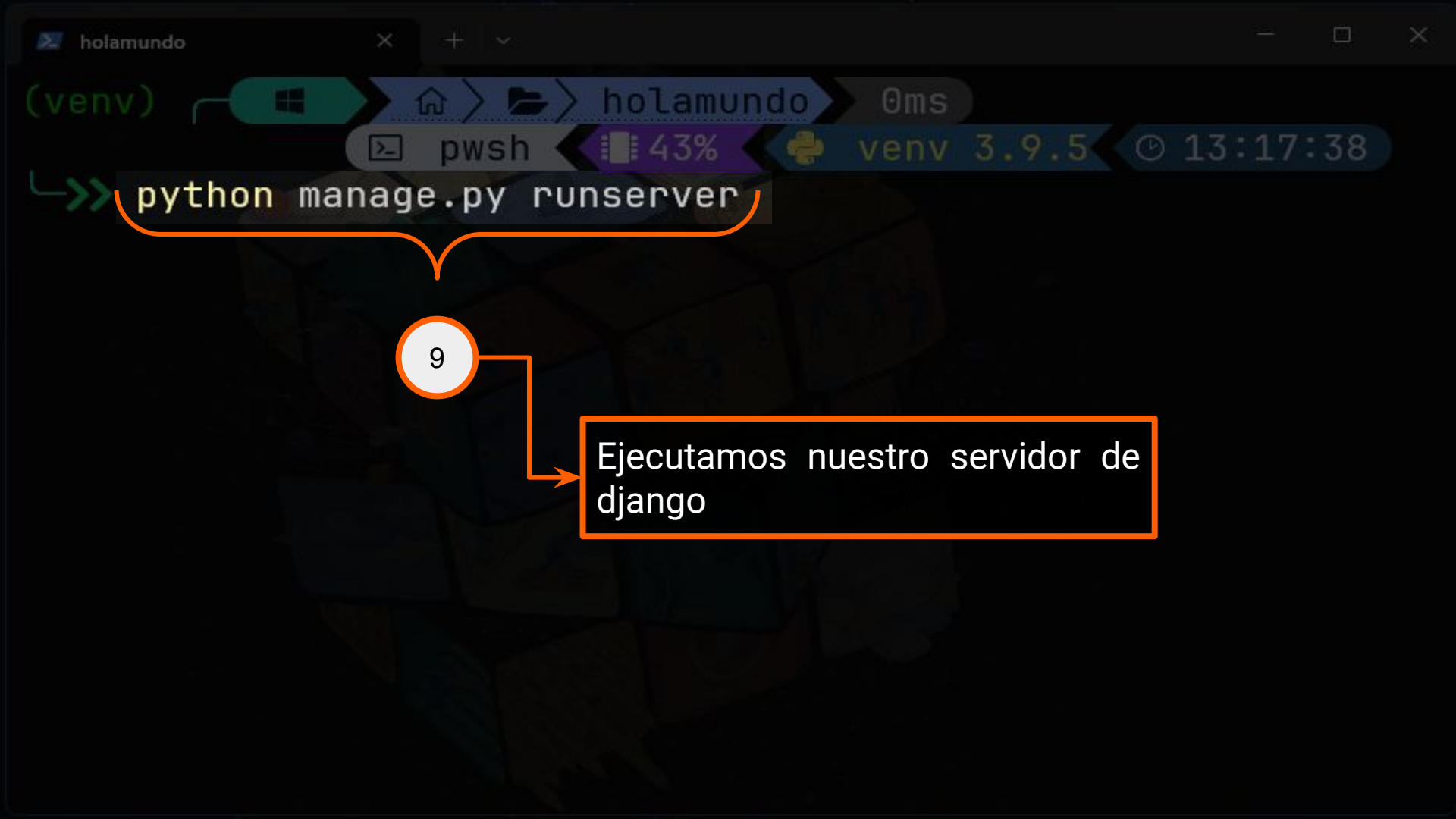
```
pip install -r requirements.txt
```

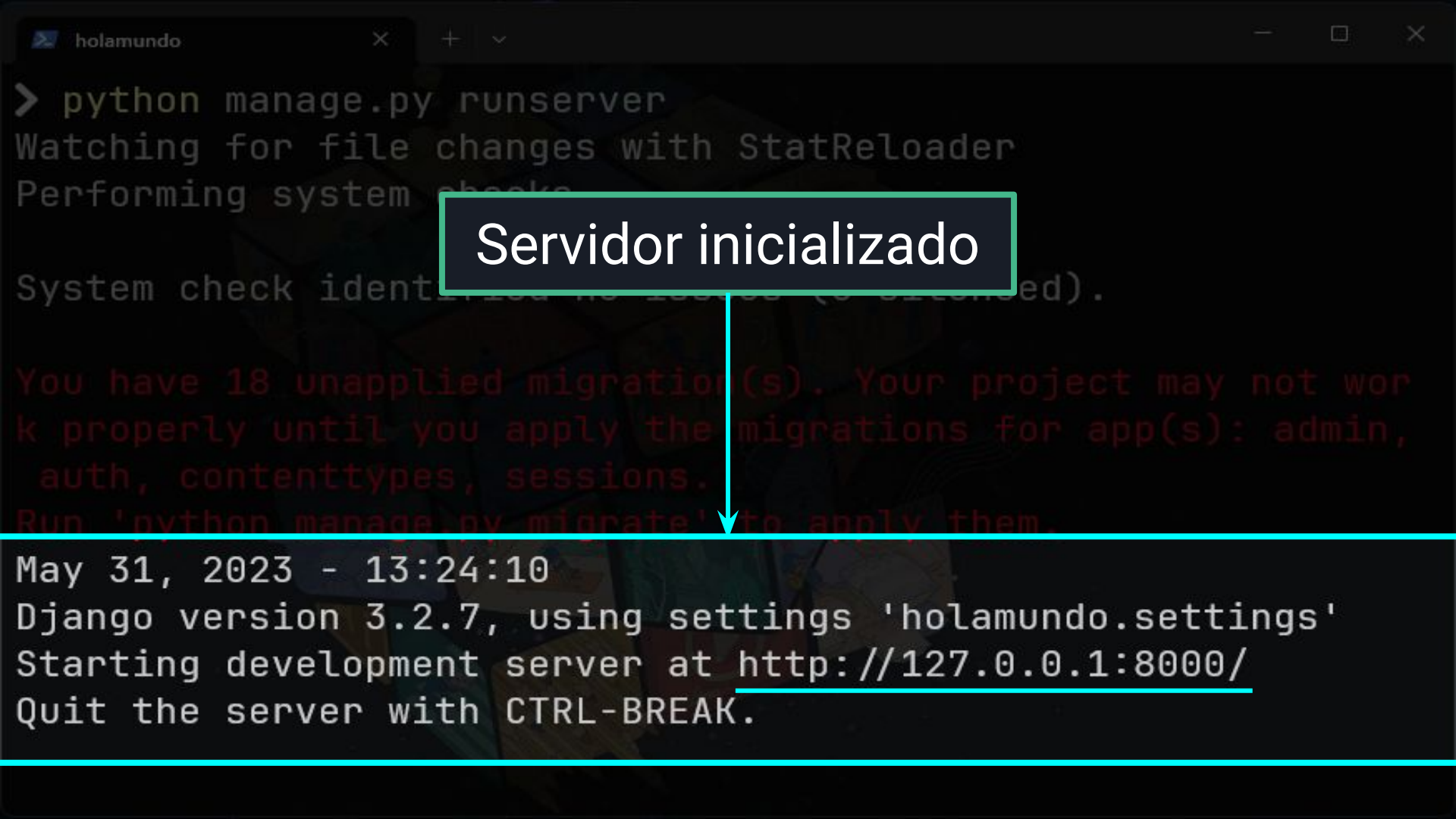


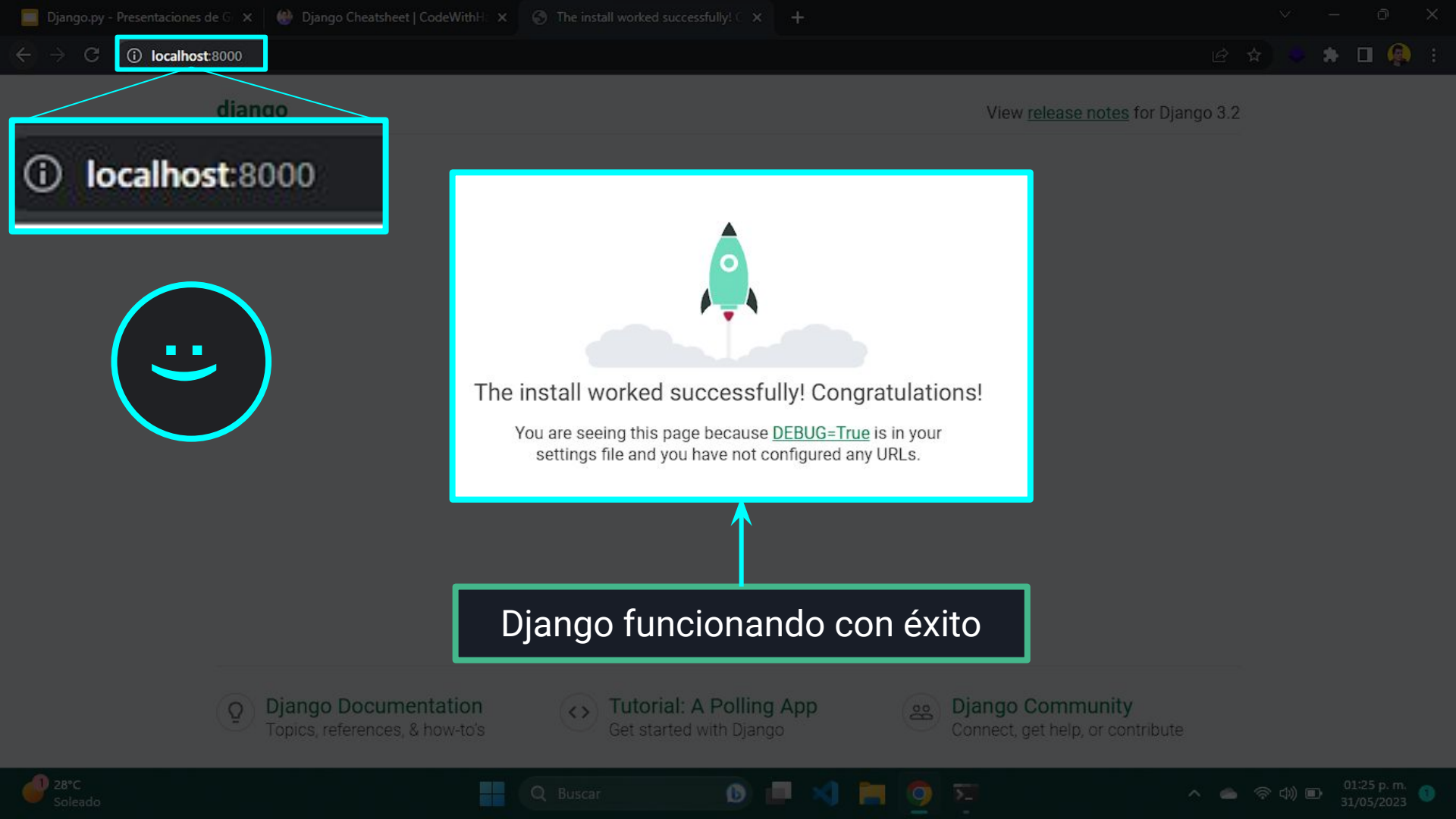
Instalamos el archivo de requerimientos para python

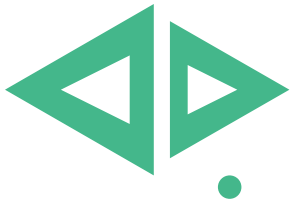

Documents
> python -- version
C:\Program Fi e: can't open file 'C:\Us
ers\nesto\ver ch file or directory
> python --version
Python 3.9.5
> cd .\Documents\
(venv)
Documents
62ms
pwsh
43%
venv 3.9.5
12:18:45
>> django-admin startproject holamundo
Le indicamos que queremos comenzar un proyecto
6
5
7
invocamos a django
Le damos un nombre a nuestro proyecto


Documents
(venv)
Documents
15ms
pwsh
44%
venv 3.9.5
13:04:45
cd holamundo/
8
accedemos a la ubicación que se creó en nuestro directorio


holamundo
(venv)
holamundo
0ms
pwsh
43%
venv 3.9.5
13:17:38
python manage.py runserver
9
Ejecutamos nuestro servidor de django


holamundo
> python manage.py runserver
Watching for file changes with StatReloader
Performing system
System check ident
ed).
You have 18 unapplied migration(s). Your project may not wor
k properly until you apply the migrations for app(s): admin,
auth, contenttypes, sessions.
Run 'python manage.py migrate' to apply them.
Servidor inicializado
May 31, 2023 - 13:24:10
Django version 3.2.7, using settings 'holamundo.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.


localhost:8000
localhost:8000
The install worked successfully! Congratulations!
You are seeing this page because DEBUG=True is in your settings file and you have not configured any URLs.
Django funcionando con éxito
View release notes for Django 3.2
Django Documentation
Topics, references, & how-to's
Tutorial: A Polling App
Get started with Django
Django Community
Connect, get help, or contribute

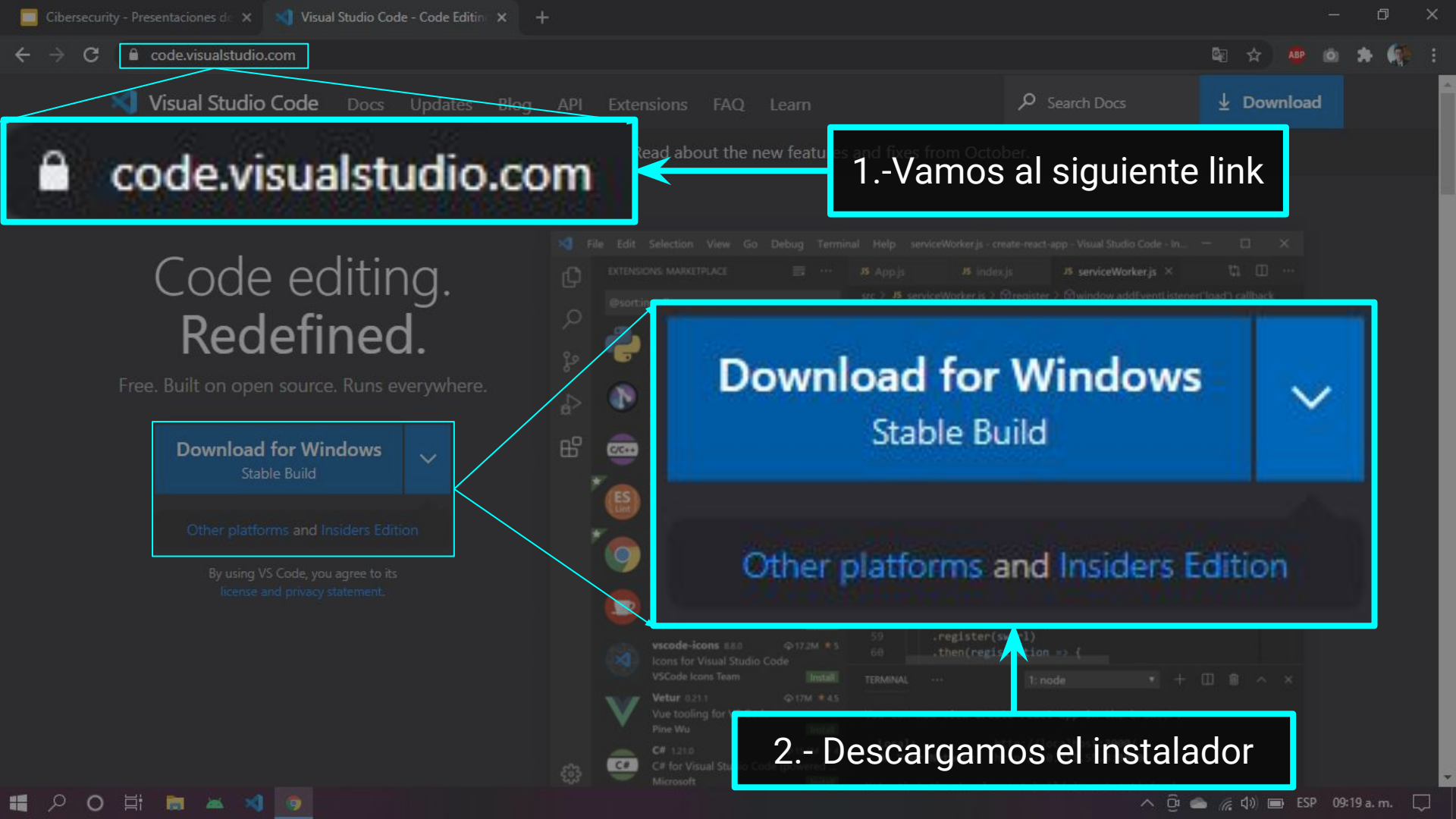


¡Felicidades!

¡Creaste tu primer Hello World!
con django

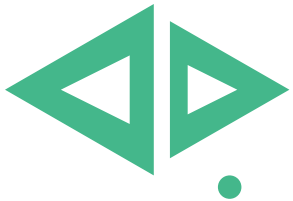
Continuará…

byNestorCode

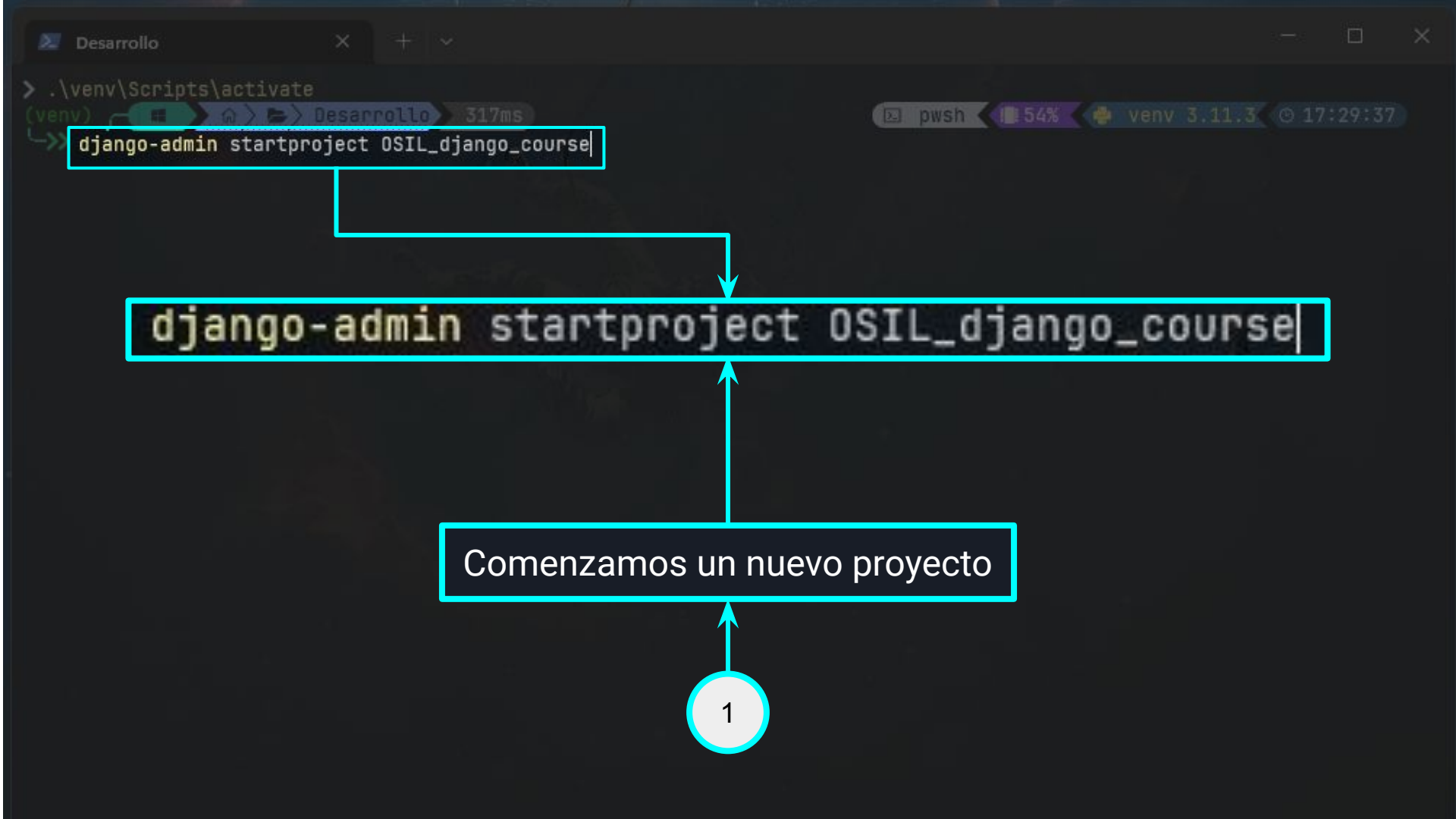

code.visualstudio.com
1.-Vamos al siguiente link
Code editing.
Redefined.
Free. Built on open source. Runs everywhere.
Download for Windows
Stable Build
Other platforms and Insiders Edition
By using VS Code, you agree to its
license and privacy statement.
Download for Windows
Stable Build
Other platforms and Insiders Edition
2.- Descargamos el instalador

django

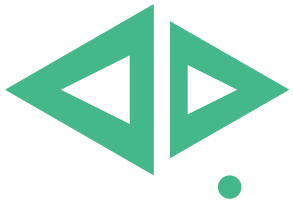
#HappyHacking💻



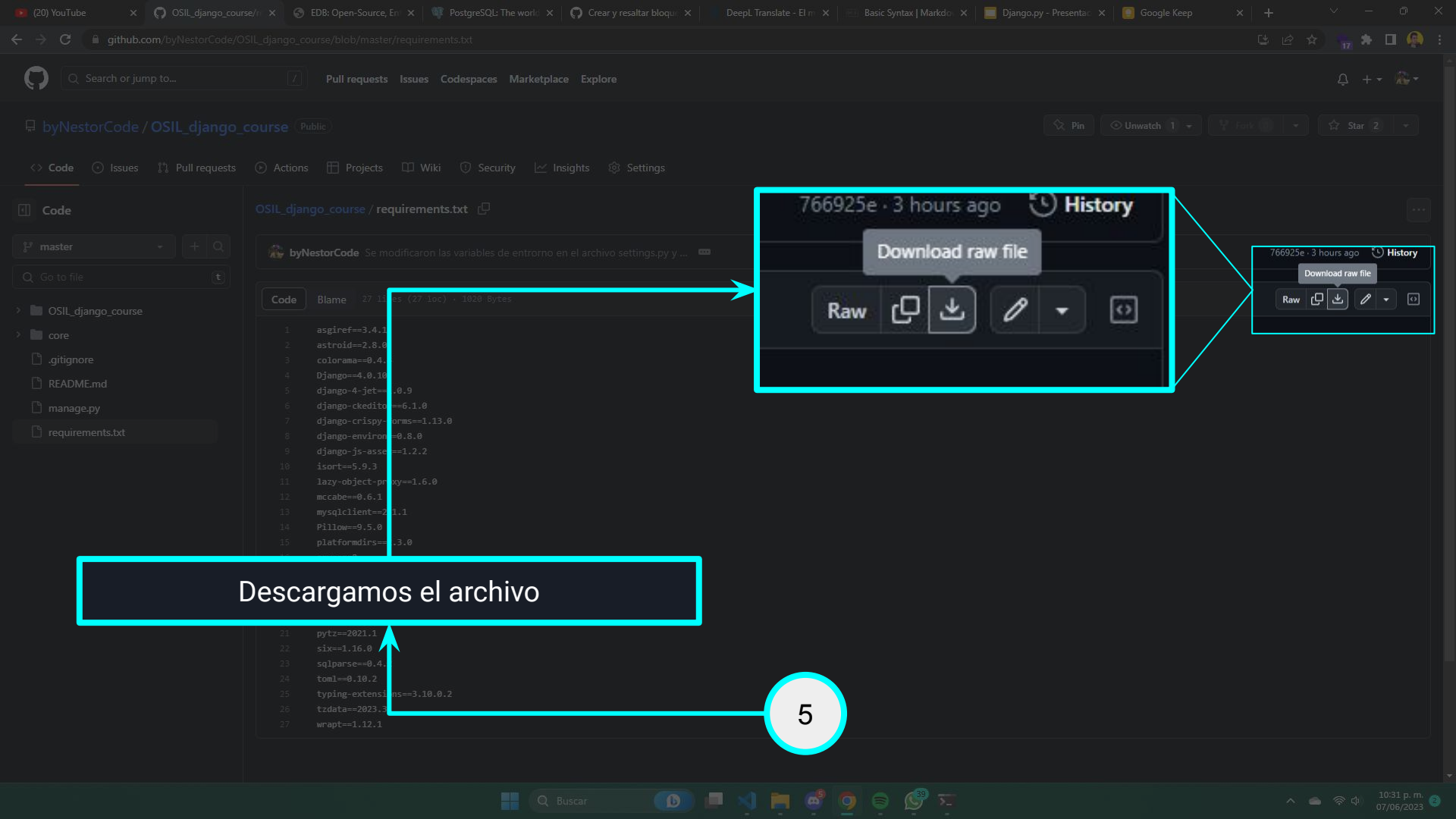
Comenzamos nuestro Proyecto

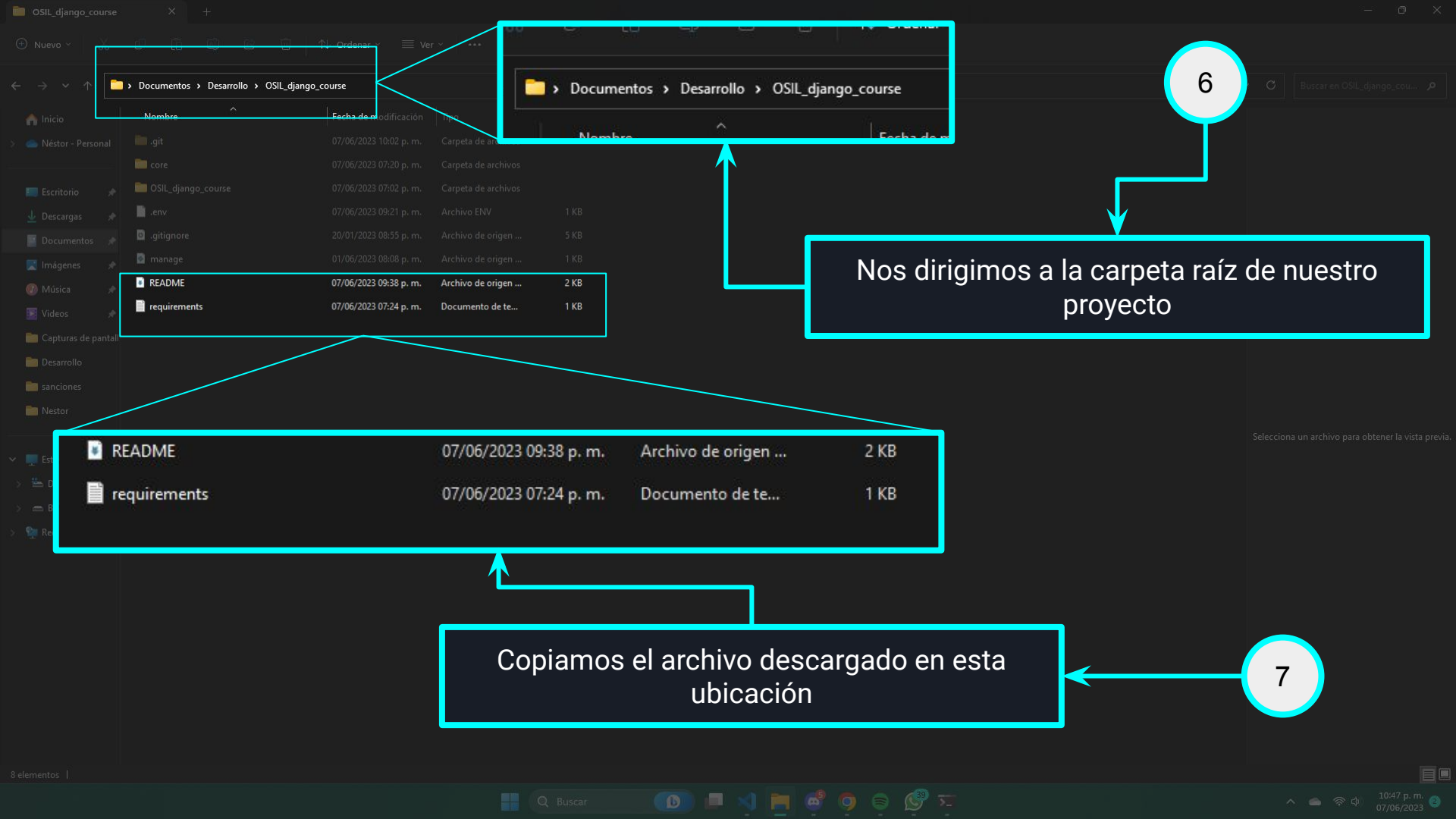

Desarrollo
> .\venv\Scripts\activate
(venv) Desarrollo 317ms
pwsh 54% venv 3.11.3 17:29:37
django-admin startproject OSIL_django_course
django-admin startproject OSIL_django_course
Comenzamos un nuevo proyecto
1

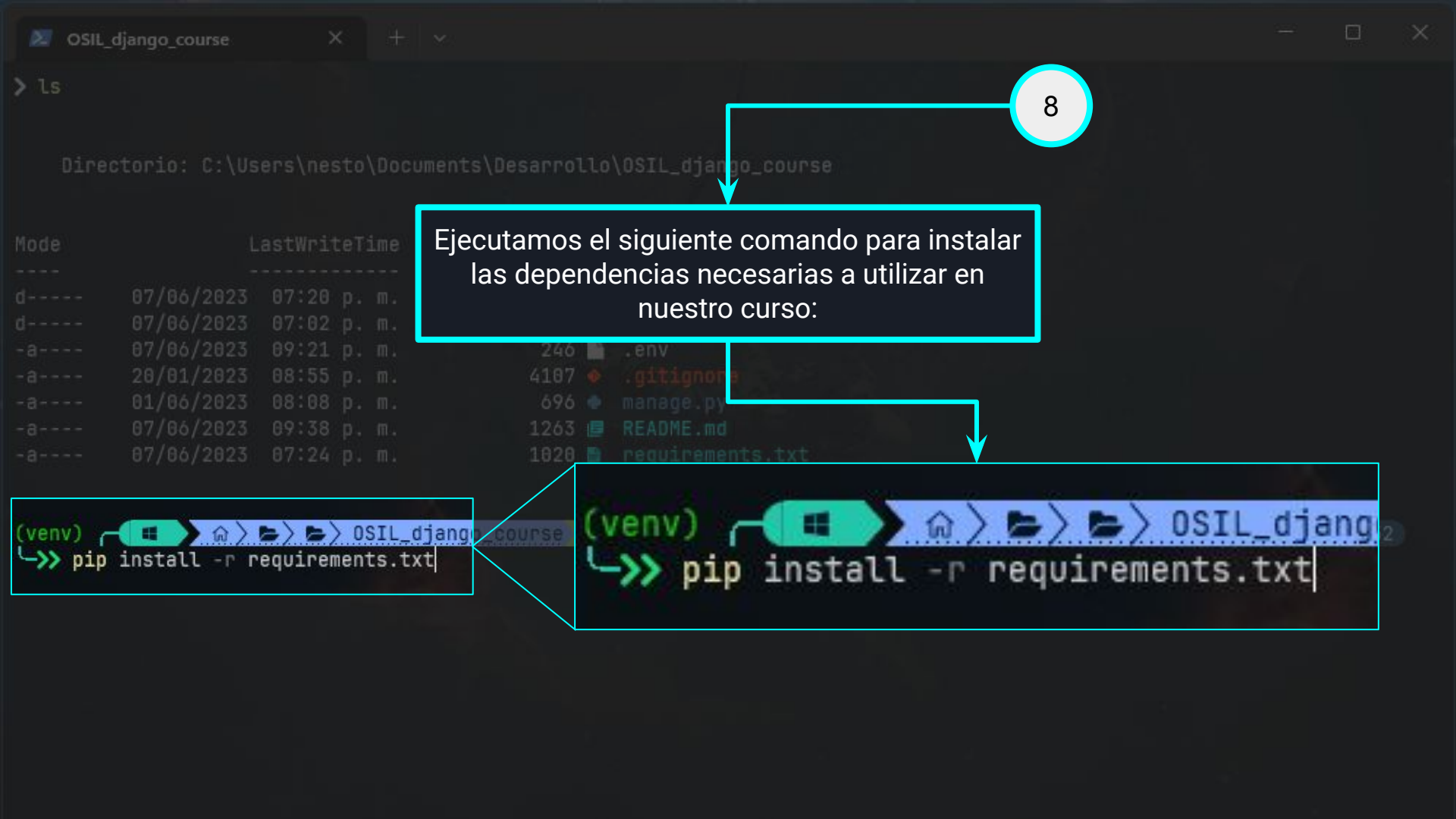

Desarrollo
> ls
(venv) Desarrollo 15ms
cd OSIL_django_course/
Mode
d----- 07/06/2023 07:02 p. m. OSIL_django_course
d----- 30/01/2023 06:49 p. m. sanciones
d----- 01/06/2023 07:49 p. m. venv
-a---- 01/06/2023 05:13 p. m. 2221677 Django.py.pdf
-a---- 01/06/2023 07:51 p. m. 452 requirements.txt
-a---- 01/06/2023 07:52 p. m. 1000 requirements_.txt
(venv) Desarrollo 15ms
cd OSIL_django_course/
Accedemos a la carpeta que se acaba de crear
2



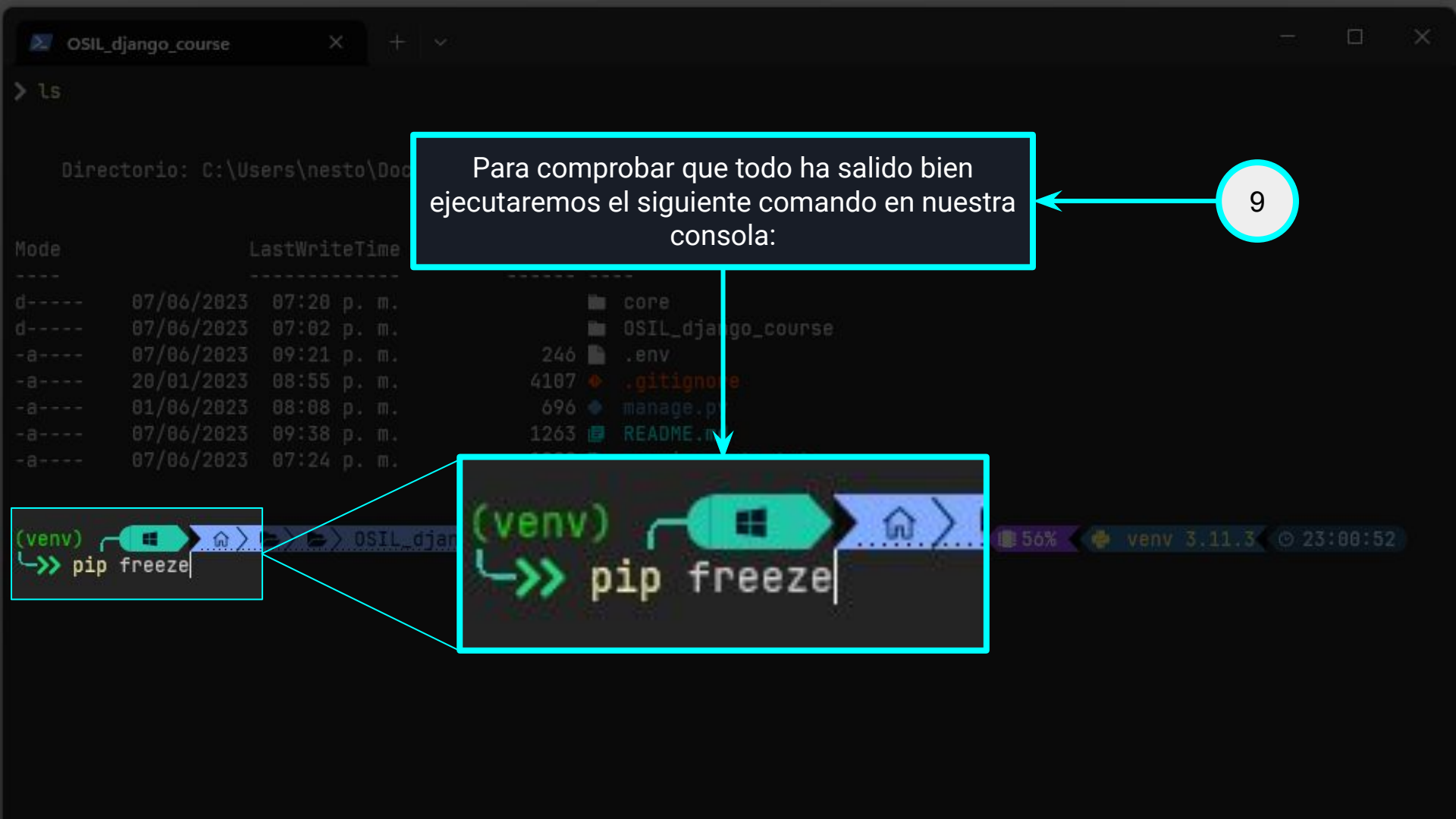
Instalamos el archivo requirements.txt


766925e · 3 hours ago
History
Download raw file
Raw
Descargamos el archivo
5


Documentos › Desarrollo › OSIL_django_course
6
Nos dirigimos a la carpeta raíz de nuestro proyecto
README 07/06/2023 09:38 p. m. Archivo de origen ... 2 KB
requirements 07/06/2023 07:24 p. m. Documento de te... 1 KB
Copiamos el archivo descargado en esta ubicación
7


OSIL_django_course
> ls
Directorio: C:\Users\nesto\Documents\Desarrollo\OSIL_django_course
Mode LastWriteTime
d----- 07/06/2023 07:20 p. m.
d----- 07/06/2023 07:02 p. m.
-a---- 07/06/2023 09:21 p. m. 246 .env
-a---- 20/01/2023 08:55 p. m. 4107 .gitignore
-a---- 01/06/2023 08:08 p. m. 696 manage.py
-a---- 07/06/2023 09:38 p. m. 1263 README.md
-a---- 07/06/2023 07:24 p. m. 1020 requirements.txt
8
Ejecutamos el siguiente comando para instalar las dependencias necesarias a utilizar en nuestro curso:
(venv)
pip install -r requirements.txt

De pronto veremos como Python comienza a instalar cada una de las dependencias solicitadas, solo nos queda esperar un poco…

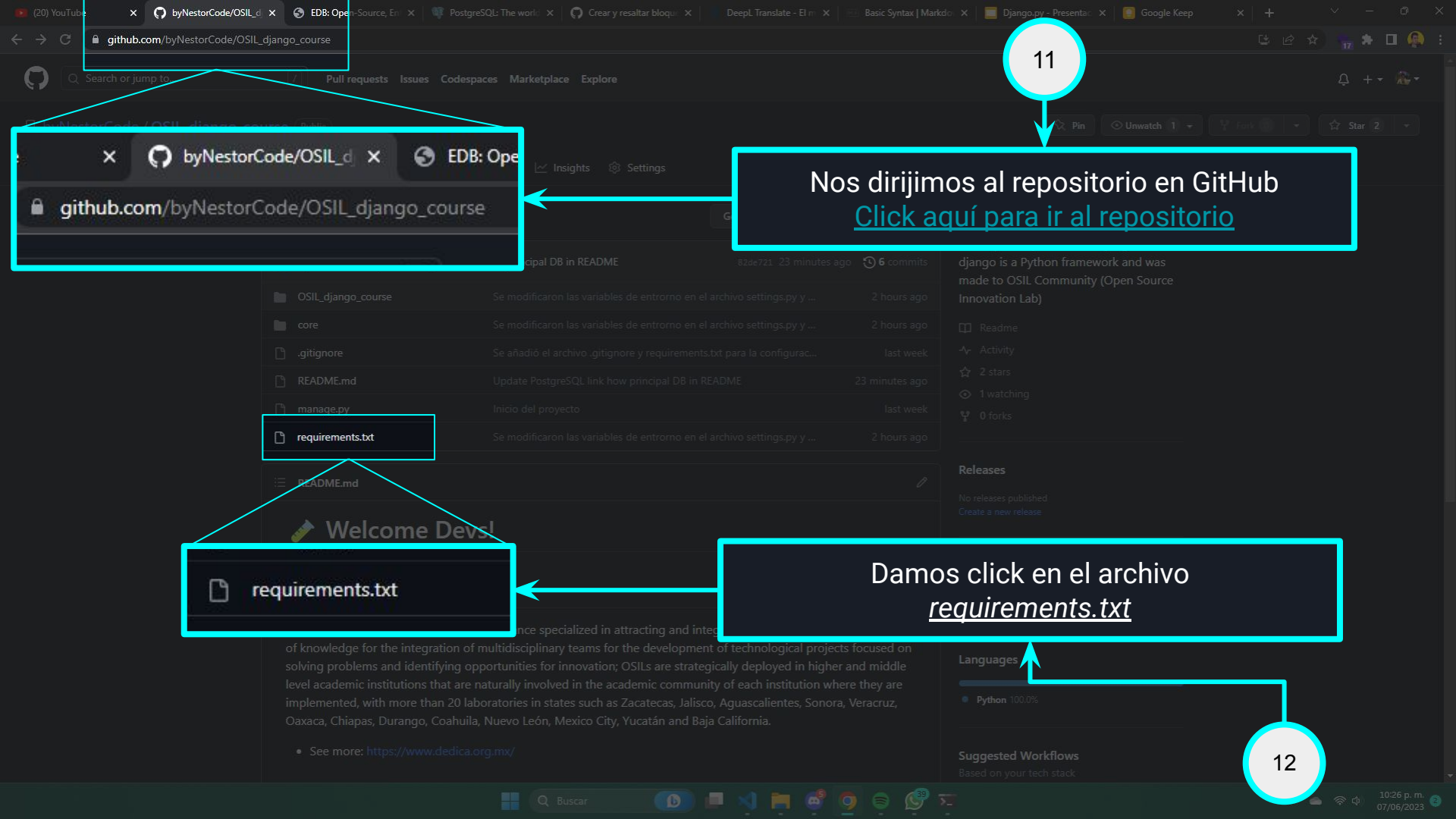

OSIL_django_course
> ls
Directorio: C:\Users\nesto\Doc
Mode LastWriteTime
d----- 07/06/2023 07:20 p. m. core
d----- 07/06/2023 07:02 p. m. OSIL_django_course
-a---- 07/06/2023 09:21 p. m. 246 .env
-a---- 20/01/2023 08:55 p. m. 4107 .gitignore
-a---- 01/06/2023 08:08 p. m. 696 manage.py
-a---- 07/06/2023 09:38 p. m. 1263 README.md
-a---- 07/06/2023 07:24 p. m.
Para comprobar que todo ha salido bien ejecutaremos el siguiente comando en nuestra consola:
9
(venv)
pip freeze
56%
venv 3.11.3
23:00:52

OSIL_django_course

```
> pip freeze
asgiref==3.4.1
astroid==2.8.0
colorama==0.4.4
Django==4.0.10
django-4-jet==1.0.9
django-ckeditor==6.1.0
django-crispy-forms==1.13.0
django-environ==0.8.0
django-js-asset==1.2.2
isort==5.9.3
lazy-object-proxy==1.6.0
mccabe==0.6.1
mysqlclient==2.1.1
Pillow==9.5.0
platformdirs==2.3.0
psycopg2==2.9.6
pylint==2.11.1
pylint-celery==0.3
pylint-django==2.4.4
pylint-plugin-utils==0.6
pytz==2021.1
six==1.16.0
sqlparse==0.4.2
toml==0.10.2
typing-extensions==3.10.0.2
tzdata==2023.3
wrapt==1.12.1
```

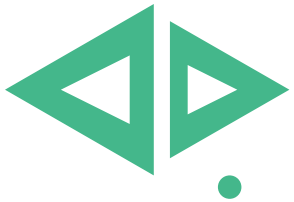
10

Sí todo ha salido, la consola nos devolverá un listado con las dependencias instaladas

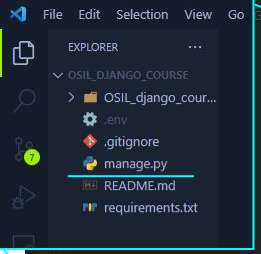

11
Nos dirijimos al repositorio en GitHub
Click aquí para ir al repositorio
byNestorCode/OSIL_d
EDB: Ope
github.com/byNestorCode/OSIL_django_course
requirements.txt
Damos click en el archivo
requirements.txt
12

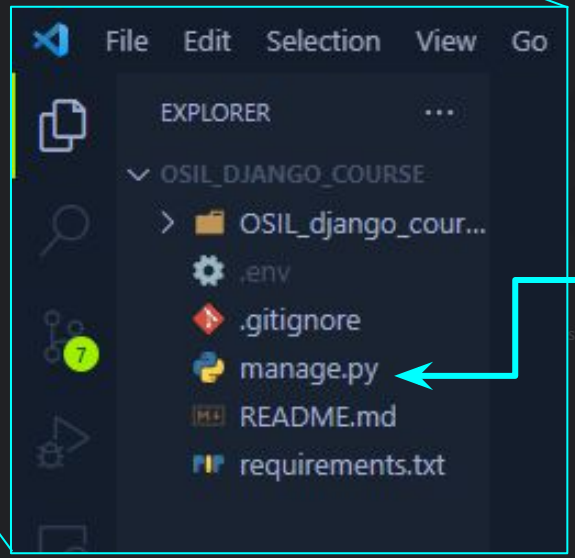
¿Sabías qué?

Un archivo requirements.txt en python es un tipo de archivo que normalmente almacena información sobre todas las librerías, módulos y paquetes específicos del proyecto utilizados mientras se desarrolla un proyecto en particular.

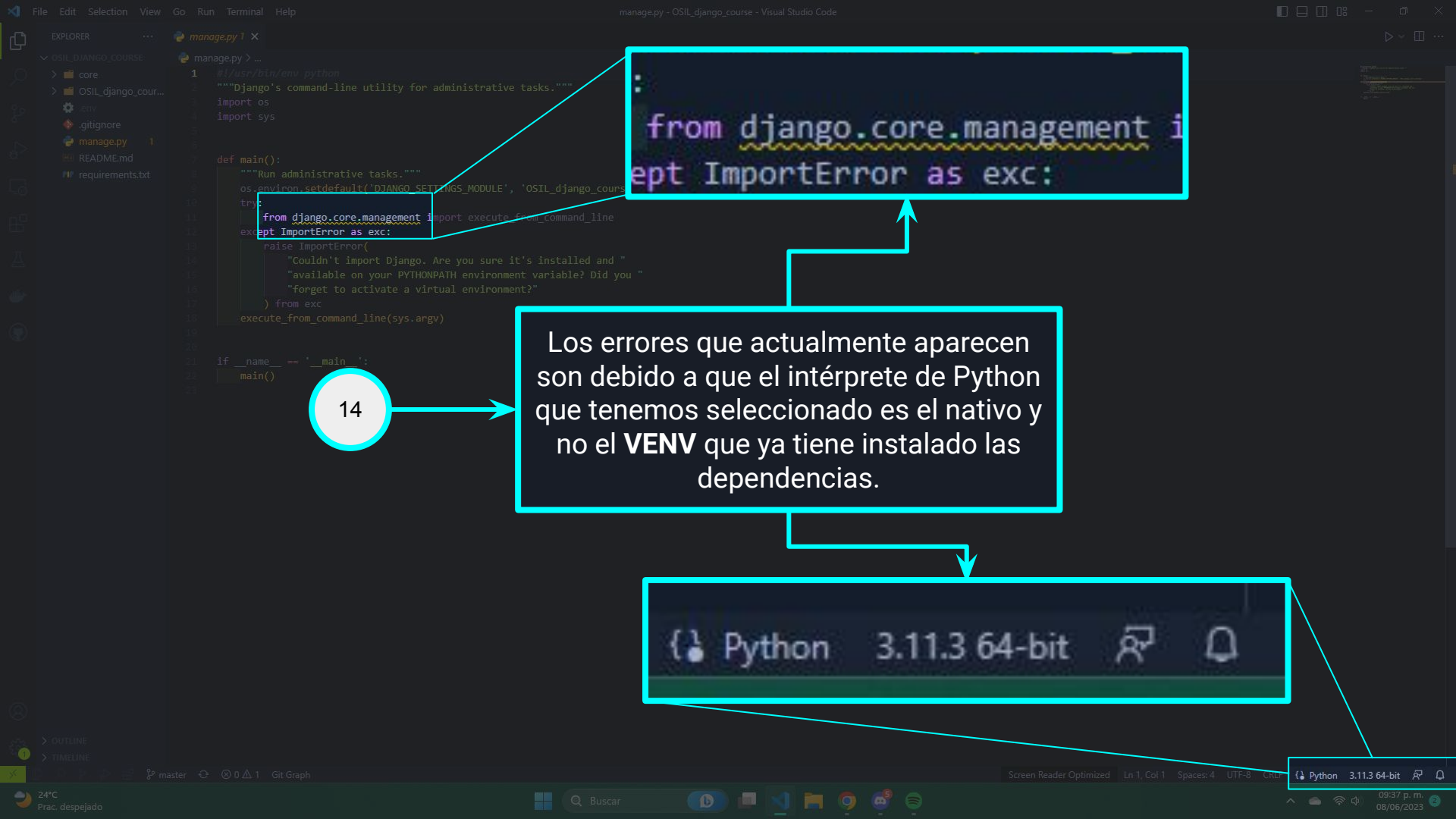


Configuración VS Code



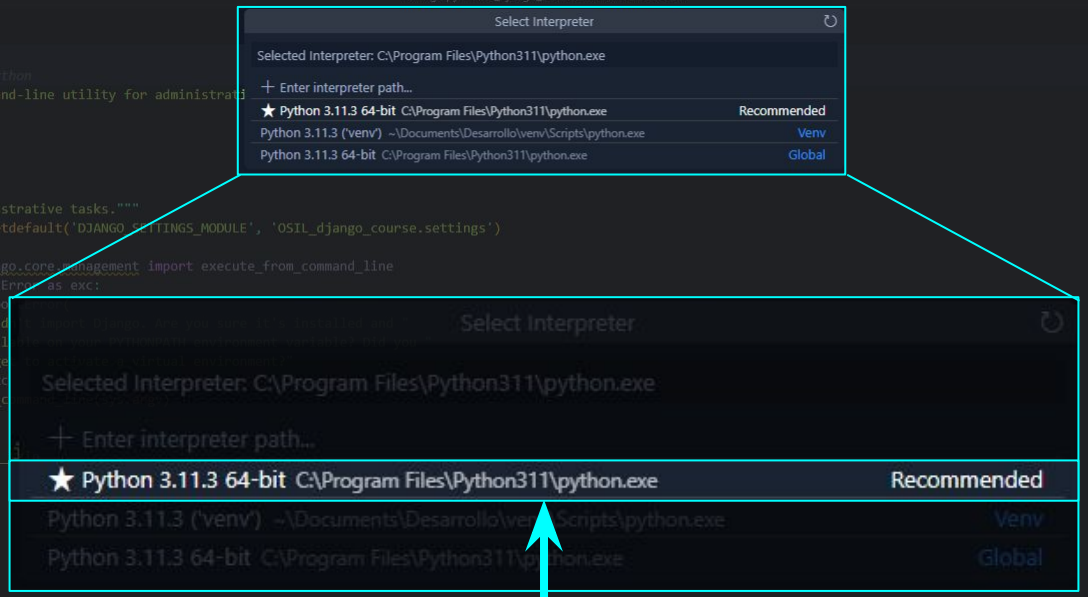


Dentro de la carpeta de nuestro proyecto, abrimos cualquier archivo de python

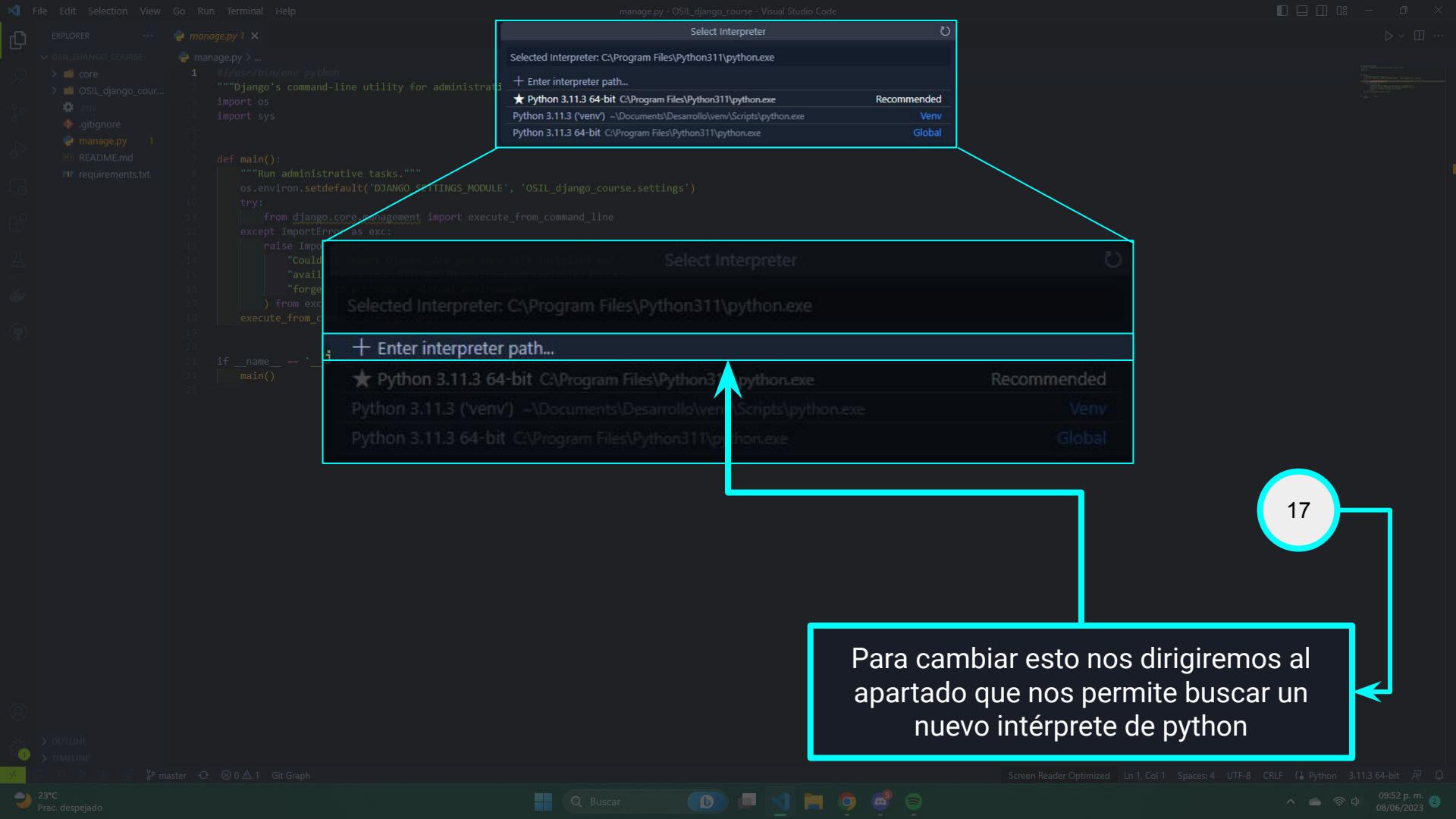

manage.py - OSIL_django_course - Visual Studio Code
manage.py 1
#!/usr/bin/env python
"""Django's command-line utility for administrative tasks."""
import os
import sys
def main():
"""Run administrative tasks."""
os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'OSIL_django_cours
try:
from django.core.management import execute_from_command_line
except ImportError as exc:
raise ImportError(
"Couldn't import Django. Are you sure it's installed and "
"available on your PYTHONPATH environment variable? Did you "
"forget to activate a virtual environment?"
) from exc
execute_from_command_line(sys.argv)
if __name__ == '__main__':
main()
from django.core.management i
ept ImportError as exc:
14
Los errores que actualmente aparecen son debido a que el intérprete de Python que tenemos seleccionado es el nativo y no el VENV que ya tiene instalado las dependencias.
Python 3.11.3 64-bit
Python 3.11.3 64-bit

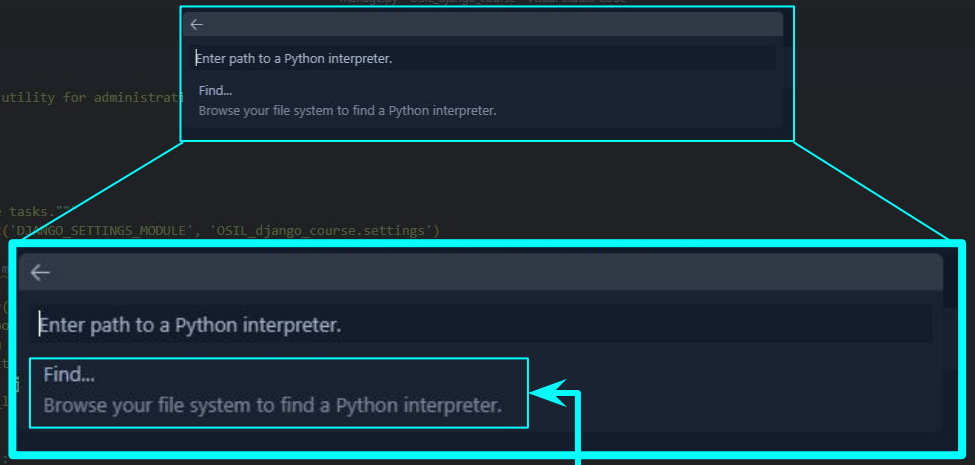
15

Para cambiar esto daremos **click aquí** para seleccionar el intérprete de nuestro **VENV**

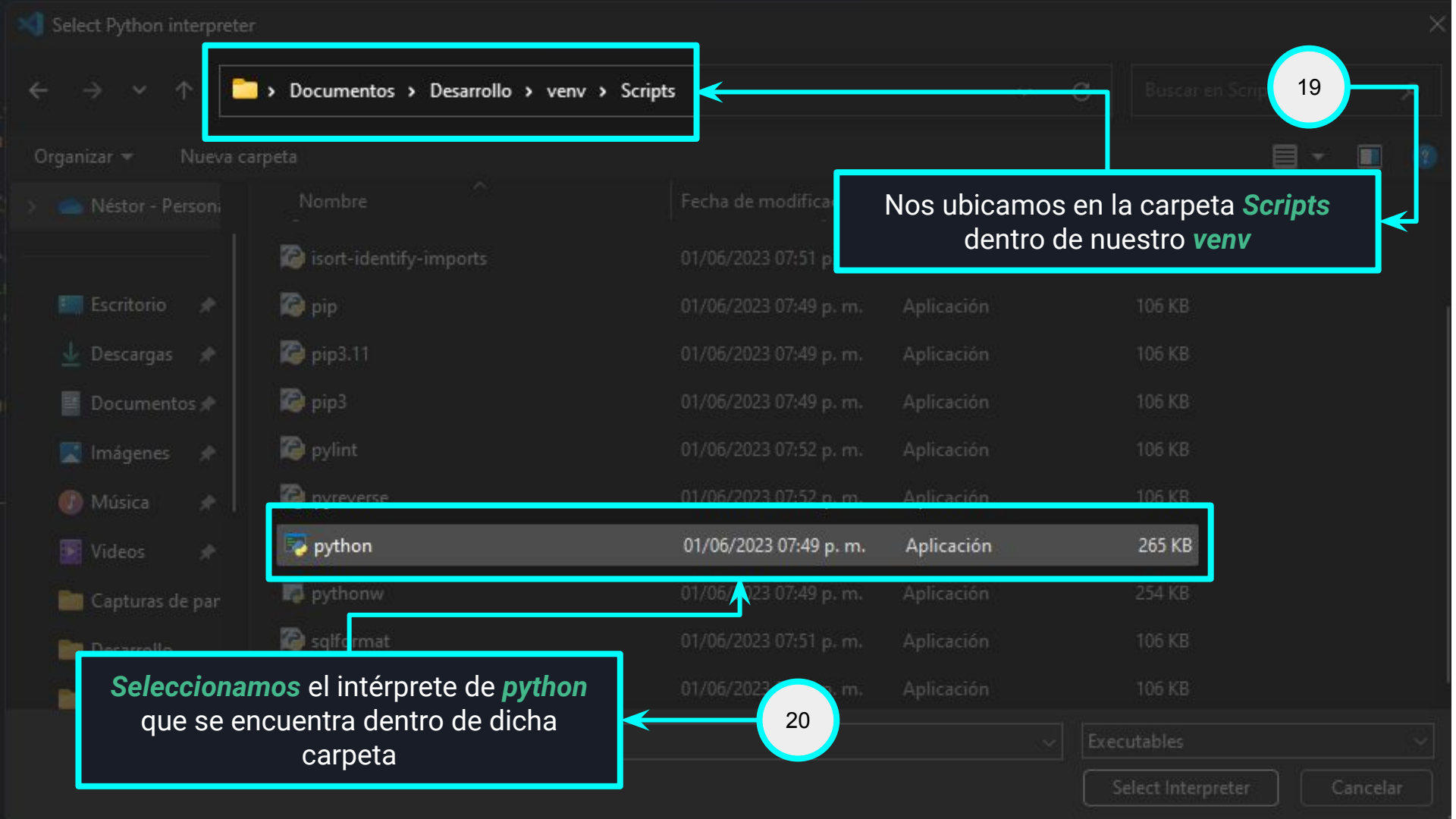


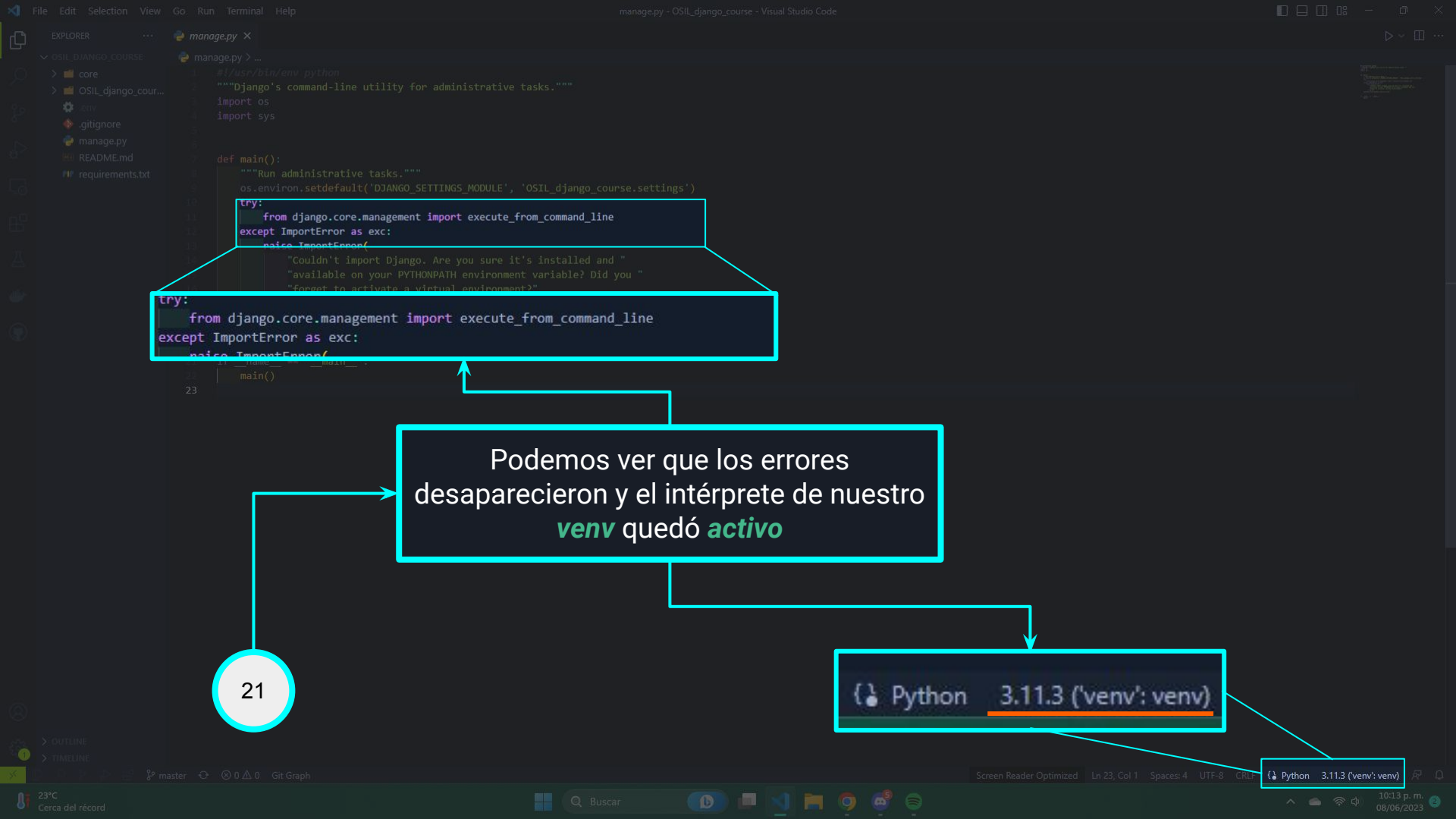
Aquí podemos observar que el intérprete por defecto es la instalación principal de python, pero recuerda que nosotros necesitamos utilizar nuestro **VENV**

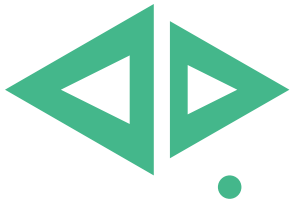

Select Interpreter
Selected Interpreter: C:\Program Files\Python311\python.exe
Enter interpreter path...
Python 3.11.3 64-bit C:\Program Files\Python311\python.exe Recommended
Python 3.11.3 ('venv') ~\Documents\Desarrollo\venv\Scripts\python.exe Venv
Python 3.11.3 64-bit C:\Program Files\Python311\python.exe Global
17
Para cambiar esto nos dirigiremos al apartado que nos permite buscar un nuevo intérprete de python



Damos click en “Find…” para comenzar la búsqueda


Select Python interpreter
Documentos › Desarrollo › venv › Scripts
19
Nos ubicamos en la carpeta Scripts dentro de nuestro venv
Organizar
Nueva carpeta
Nombre
Fecha de modifica
Néstor - Personi
Escritorio
Descargas
Documentos
Imágenes
Música
Videos
Capturas de par
isort-identify-imports 01/06/2023 07:51 p
pip 01/06/2023 07:49 p. m. Aplicación 106 KB
pip3.11 01/06/2023 07:49 p. m. Aplicación 106 KB
pip3 01/06/2023 07:49 p. m. Aplicación 106 KB
pylint 01/06/2023 07:52 p. m. Aplicación 106 KB
pyreverse 01/06/2023 07:52 p. m. Aplicación 106 KB
python 01/06/2023 07:49 p. m. Aplicación 265 KB
pythonw 01/06/2023 07:49 p. m. Aplicación 254 KB
sqlformat 01/06/2023 07:51 p. m. Aplicación 106 KB
01/06/2023 m. Aplicación 106 KB
Seleccionamos el intérprete de python que se encuentra dentro de dicha carpeta
20
Executables
Select Interpreter
Cancelar


manage.py - OSIL_django_course - Visual Studio Code
EXPLORER
manage.py
OSIL_DJANGO_COURSE
core
OSIL_django_cour...
.env
.gitignore
manage.py
README.md
requirements.txt
#!/usr/bin/env python
"""Django's command-line utility for administrative tasks."""
import os
import sys
def main():
"""Run administrative tasks."""
os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'OSIL_django_course.settings')
try:
from django.core.management import execute_from_command_line
except ImportError as exc:
raise ImportError(
"Couldn't import Django. Are you sure it's installed and "
"available on your PYTHONPATH environment variable? Did you "
"forget to activate a virtual environment?"
)
if __name__ == '__main__':
main()
Podemos ver que los errores desaparecieron y el intérprete de nuestro venv quedó activo
21
Python 3.11.3 ('venv': venv)
master
Git Graph
Screen Reader Optimized
Ln 23, Col 1
Spaces: 4
UTF-8
CRLF
Python 3.11.3 ('venv': venv)
23°C
Cerca del récord
Buscar
10:13 p. m.
08/06/2023



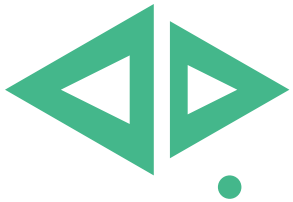
Si todo ha salido correctamente...

¡FELICIDADES!

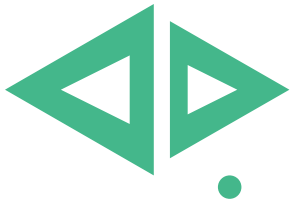
ya puedes comenzar a

HACKEAR al mundo

byNestorCode



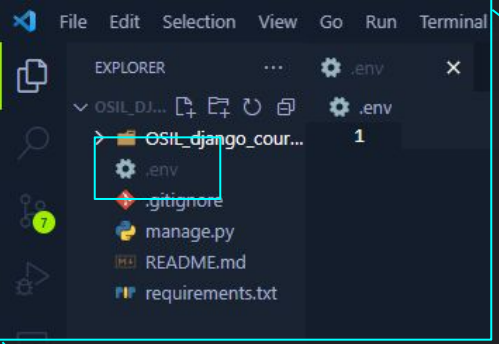
byNestorCode

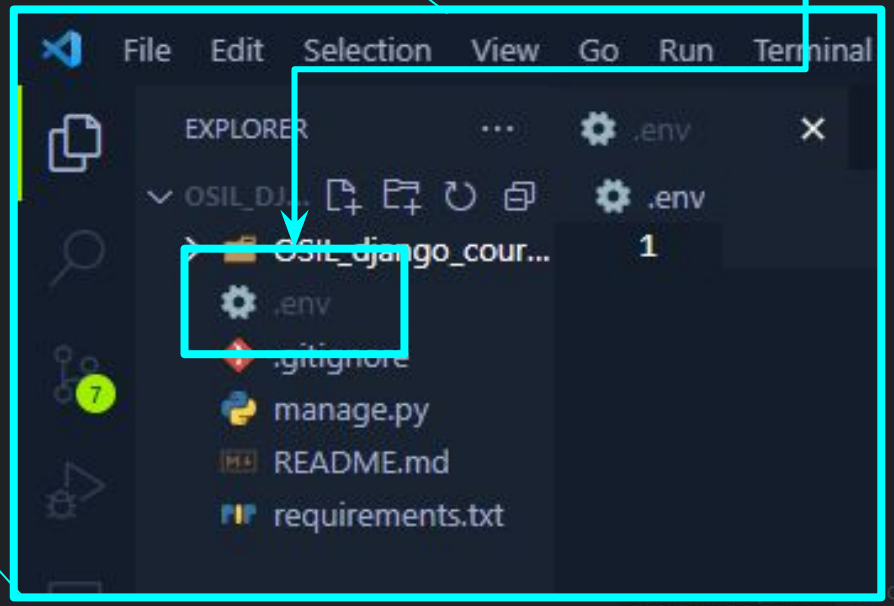


Vamos a darle seguridad a nuestro código

byNestorCode

Crearemos un archivo ***.env*** en nuestro directorio raíz para manejar las variables de entorno de nuestro servidor





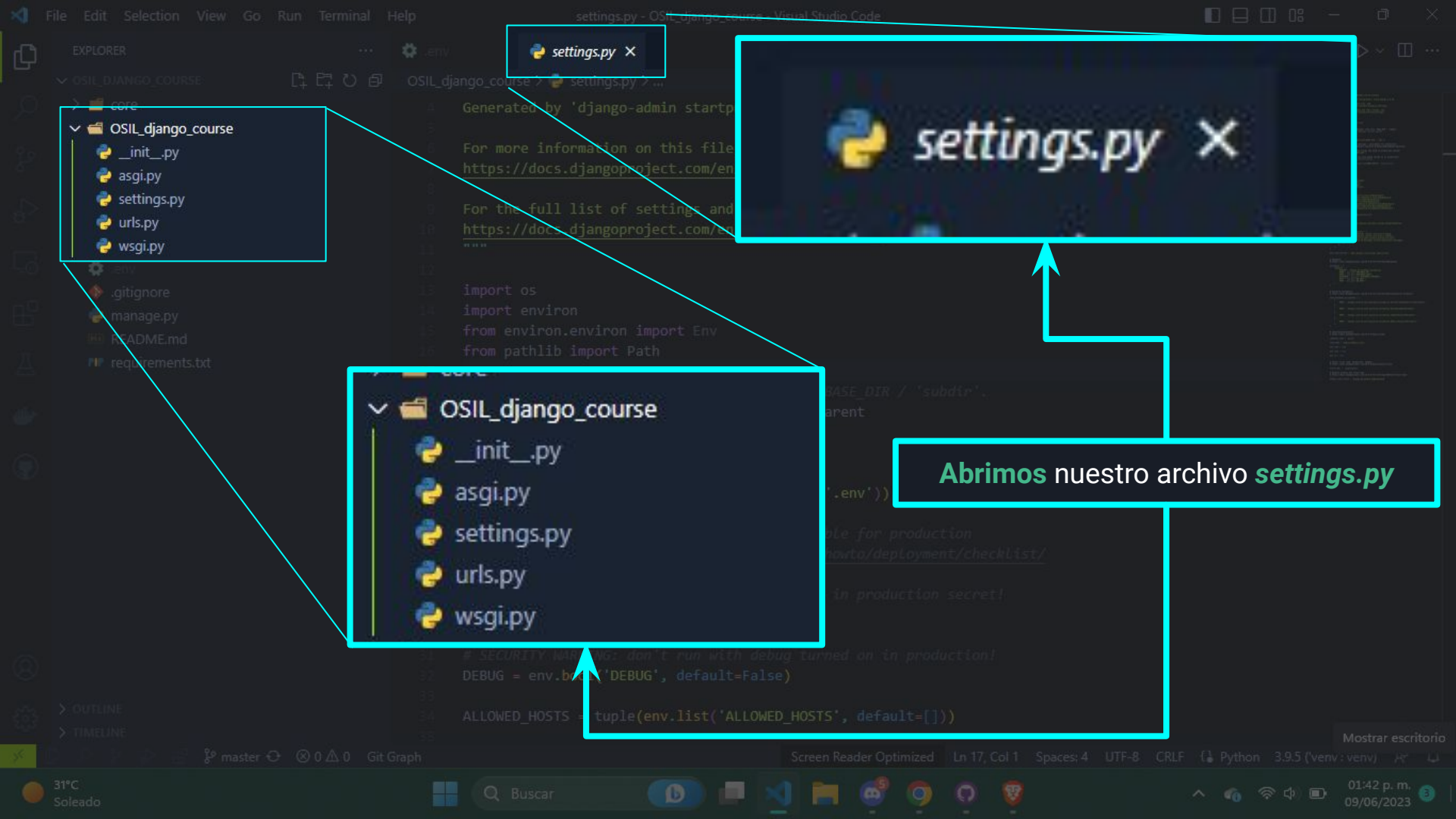
¿Qué es un archivo ***.env***?

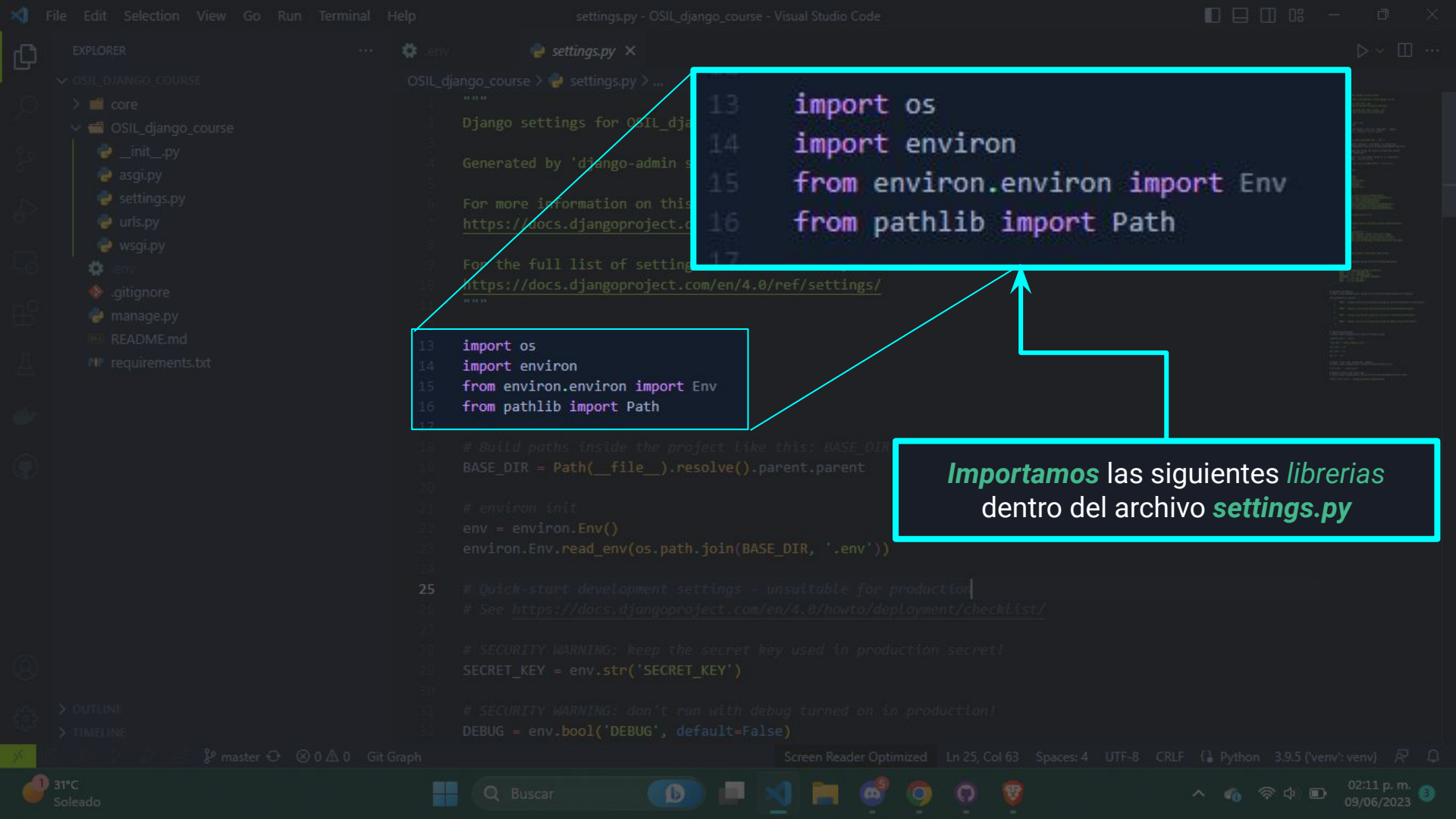
Un archivo ***.env*** no es más que un archivo de texto donde se definen una serie de variables de entorno a las cuales les asignamos un valor y que se agrega en el directorio raíz de un proyecto.

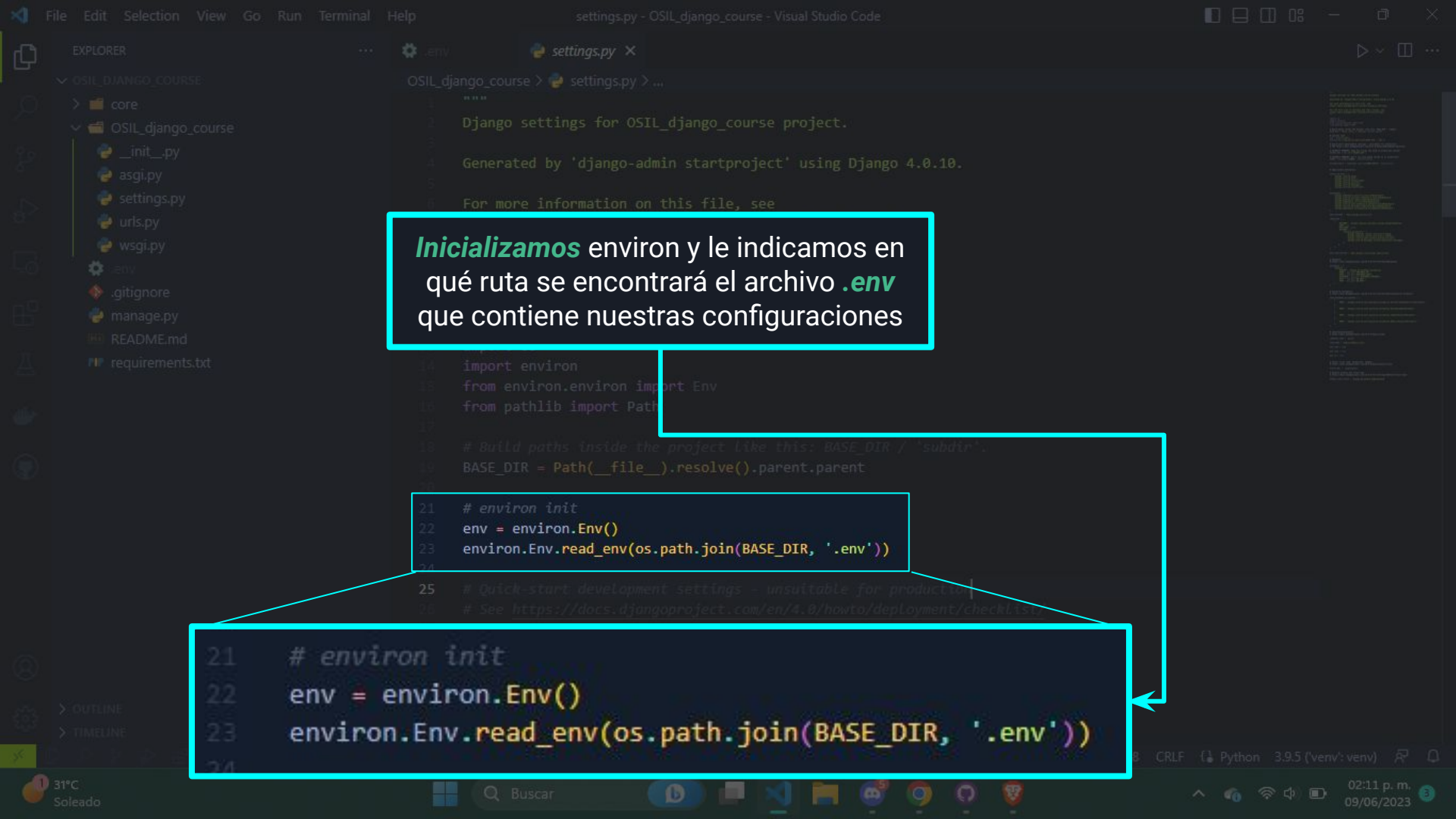
¿Por qué es importante el archivo ***.env***?

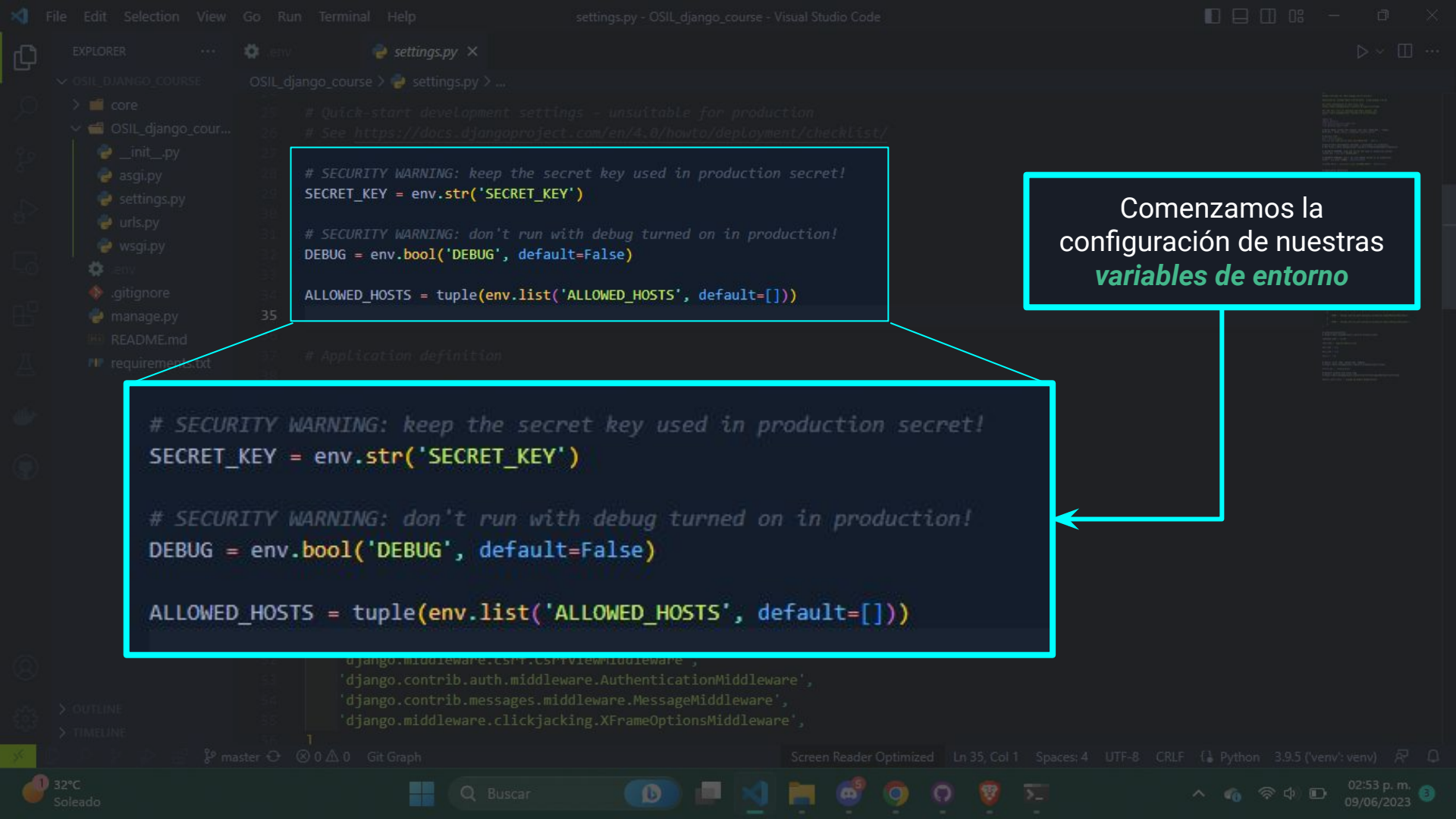
Para aplicar buenas prácticas a nuestro código

Recordemos que no es una buena idea guardar este tipo de datos en scripts que se suban a un repositorio. Por esto debemos recordar que al usar archivos .env debemos agregar al archivo .gitignore el archivo .env para evitar subirlo al repositorio y revelar información sensible.


settings.py
OSIL_django_course
__init__.py
asgi.py
settings.py
urls.py
wsgi.py
Abrimos nuestro archivo settings.py


import os
import environ
from environ.environ import Env
from pathlib import Path
Importamos las siguientes librerias
dentro del archivo settings.py

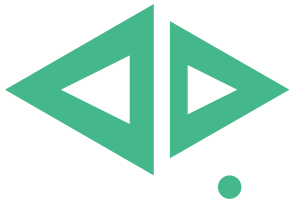

Inicializamos environ y le indicamos en qué ruta se encontrará el archivo ***.env*** que contiene nuestras configuraciones
# environ init
env = environ.Env()
environ.Env.read_env(os.path.join(BASE_DIR, '.env'))


Comenzamos la configuración de nuestras *variables de entorno*
# SECURITY WARNING: keep the secret key used in production secret!
SECRET_KEY = env.str('SECRET_KEY')
# SECURITY WARNING: don't run with debug turned on in production!
DEBUG = env.bool('DEBUG', default=False)
ALLOWED_HOSTS = tuple(env.list('ALLOWED_HOSTS', default=[]))

Así quedaría la configuración de nuestra ***Base de Datos***

```python
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.postgresql',
        'NAME': env.str('POSTGRES_DB'),
        'USER': env.str('POSTGRES_USER'),
        'PASSWORD': env.str('POSTGRES_PASSWORD'),
        'HOST': env.str('DB_HOST'),
        'PORT': env.int('DB_PORT')
    }
}
```


```python
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.postgresql',
        'NAME': env.str('POSTGRES_DB'),
        'USER': env.str('POSTGRES_USER'),
        'PASSWORD': env.str('POSTGRES_PASSWORD'),
        'HOST': env.str('DB_HOST'),
        'PORT': env.int('DB_PORT')
    }
}
```

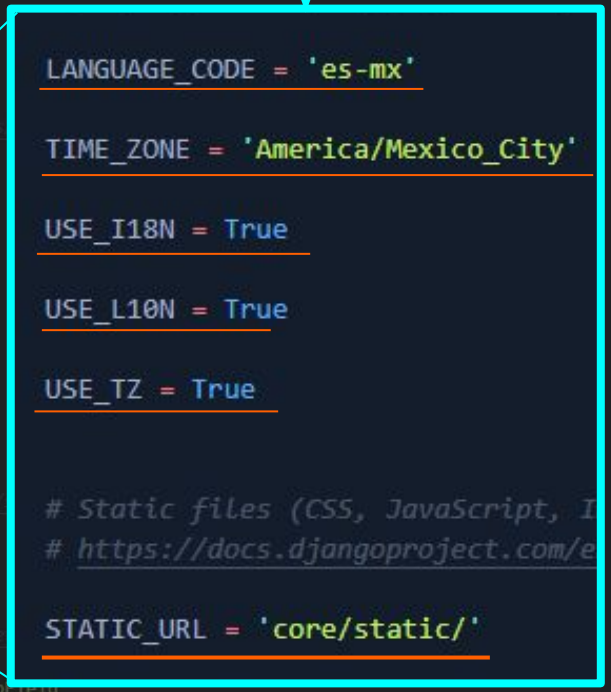


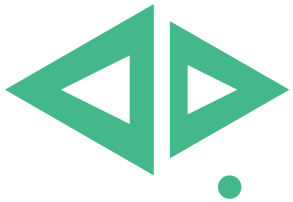
Un ***tip*** que nos ayudará a futuro

byNestorCode

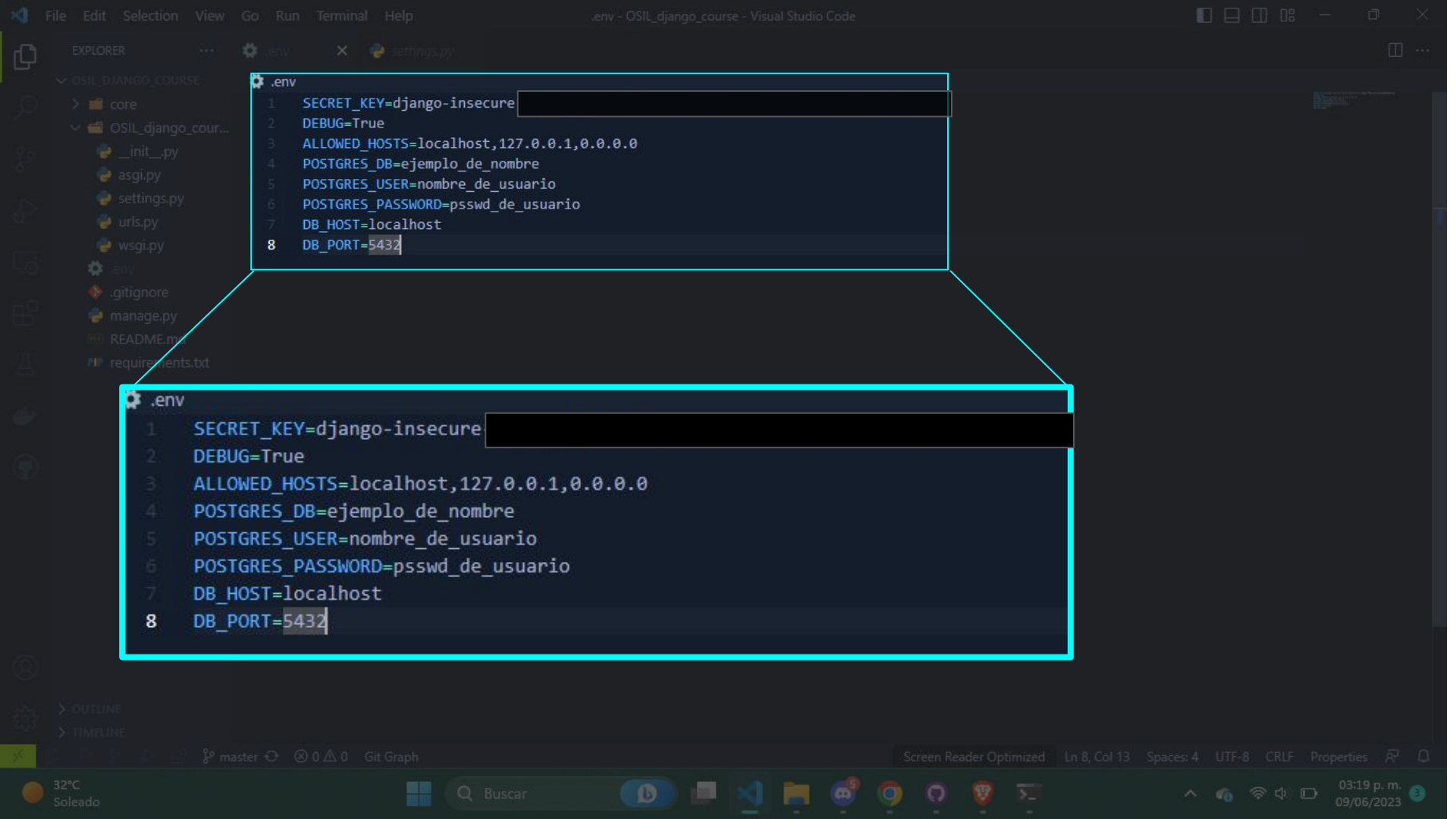
Añadimos estas líneas de código que configuran la ***zona horaria*** de nuestro ***servidor*** y la ***ruta de archivos estáticos***

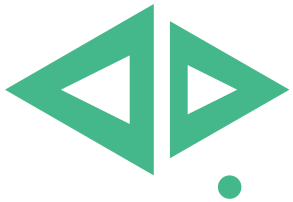
```python
LANGUAGE_CODE = 'es-mx'

TIME_ZONE = 'America/Mexico_City'

USE_I18N = True

USE_L10N = True

USE_TZ = True


# Static files (CSS, JavaScript, Images)
# https://docs.djangoproject.com/en/4.0/howto/static-files/

STATIC_URL = 'core/static/'
```





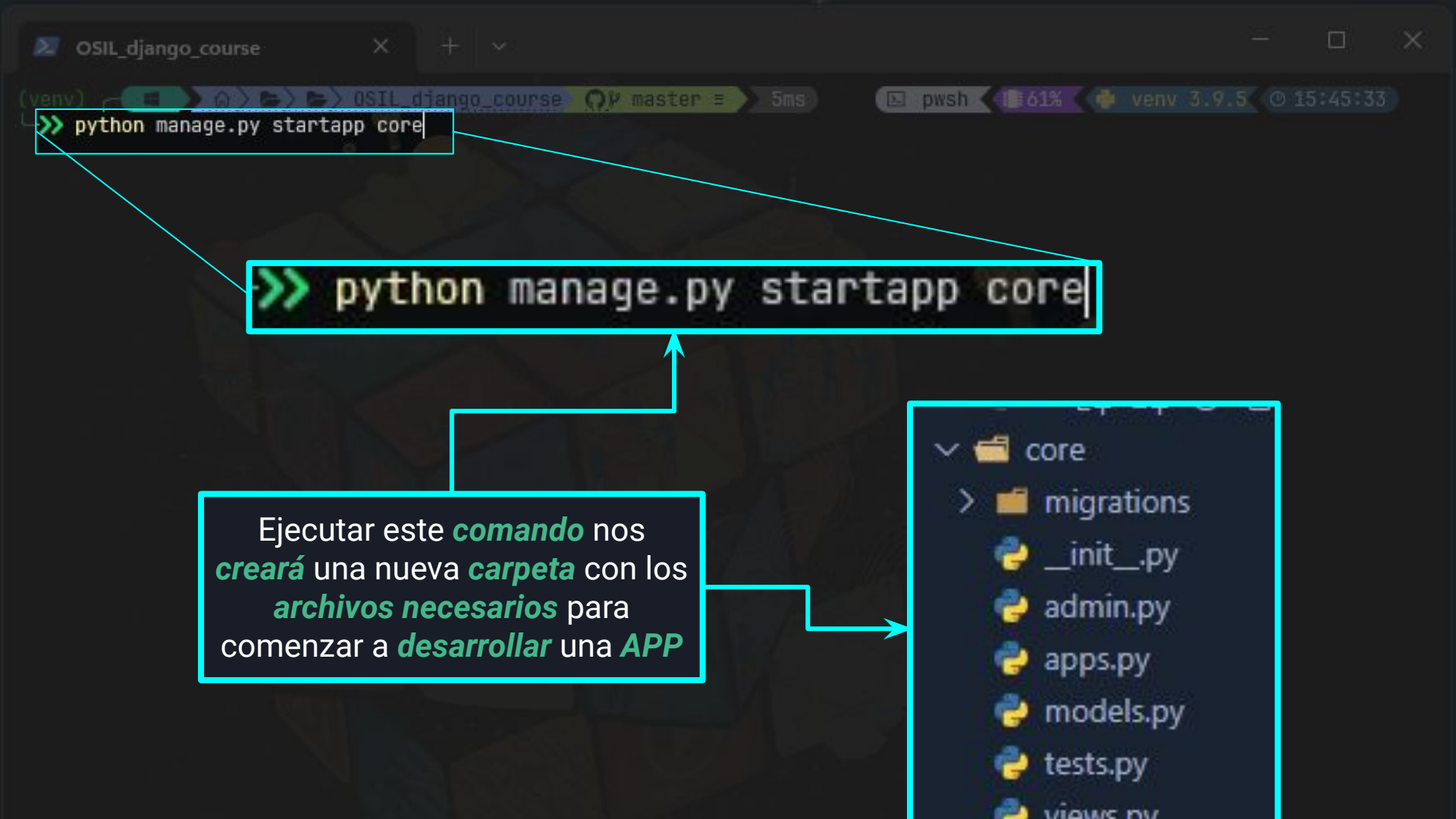
Nuestro archivo .***env*** quedaría así:

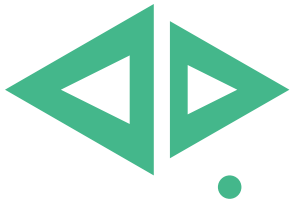

.env
SECRET_KEY=django-insecure
DEBUG=True
ALLOWED_HOSTS=localhost,127.0.0.1,0.0.0.0
POSTGRES_DB=ejemplo_de_nombre
POSTGRES_USER=nombre_de_usuario
POSTGRES_PASSWORD=psswd_de_usuario
DB_HOST=localhost
DB_PORT=5432
.env
SECRET_KEY=django-insecure
DEBUG=True
ALLOWED_HOSTS=localhost,127.0.0.1,0.0.0.0
POSTGRES_DB=ejemplo_de_nombre
POSTGRES_USER=nombre_de_usuario
POSTGRES_PASSWORD=psswd_de_usuario
DB_HOST=localhost
DB_PORT=5432



Añadimos nuestra primer APP:

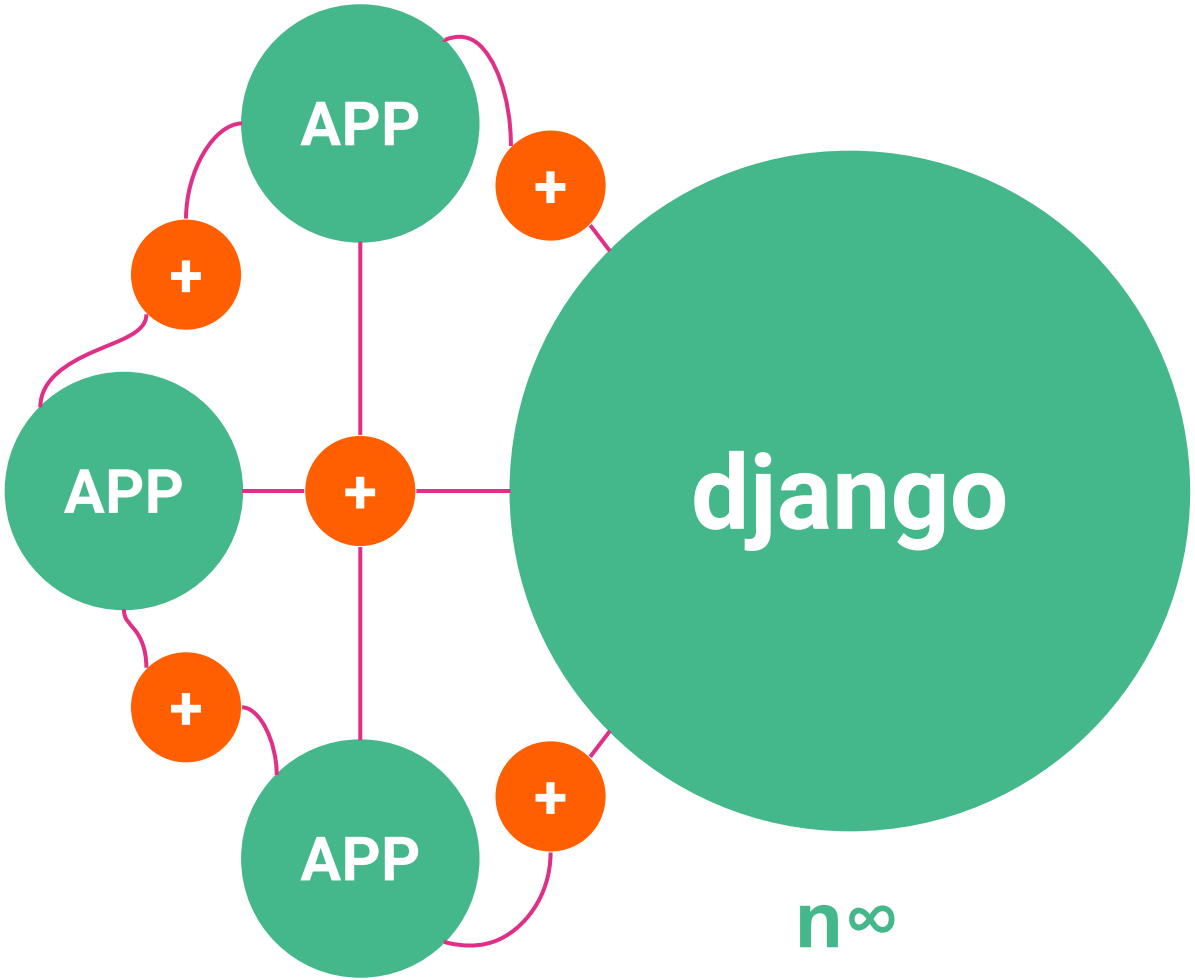
byNestorCode


OSIL_django_course
(venv) OSIL_django_course master 5ms pwsh 61% venv 3.9.5 15:45:33
>> python manage.py startapp core
>> python manage.py startapp core
Ejecutar este comando nos creará una nueva carpeta con los archivos necesarios para comenzar a desarrollar una APP
core
migrations
__init__.py
admin.py
apps.py
models.py
tests.py
views.py



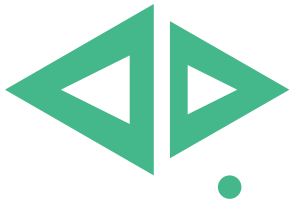
¿Qué es una app en django?

byNestorCode



Cada *app* podría realizar una *función específica* de nuestro servidor principal y además entre estas *apps compartirse información*

app = módulo



Continuará...

byNestorCode

https://ed.team/blog/tipos-de-lenguajes-de-programacion
https://www.pythonista.io/cursos/py101/
https://www.pythonista.io/cursos/py101/expresiones-con-operadores-en-python
https://www.flaticon.com/
https://platzi.com/blog/historia-python/
https://www.codigofuente.org/operadores-en-python/

https://www.python.org/
https://code.visualstudio.com/