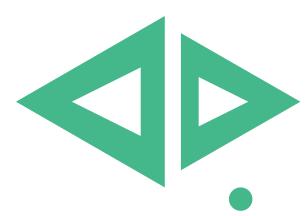
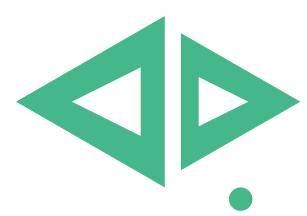


`#!/usr/bin/Python`

`#HappyHacking` 



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.

Recuperar las palabras de un documento en ABC

```
HOW TO RETURN words document:  
    PUT {} IN collection  
    FOR line IN document:  
        FOR word IN split line:  
            IF word not.in collection:  
                INSERT word IN  
collection  
    RETURN collection
```

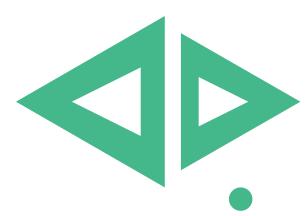
Recuperar las palabras de un documento en Python

```
def words(document):  
    collection = set()  
    for line in document:  
        for word in line.split():  
            if word not in collection:  
                collection.add(word)  
    return collection
```

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

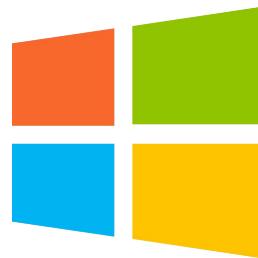
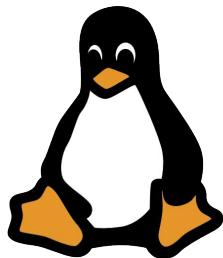
`python("Es un lenguaje de alto nivel")`

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.

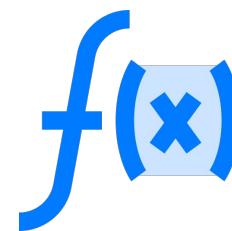
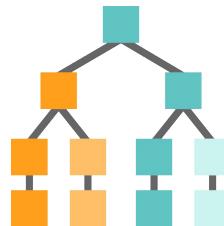
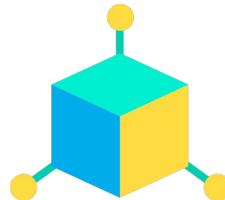
```
python.multiPlataforma(linux, windows, mac)
```

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



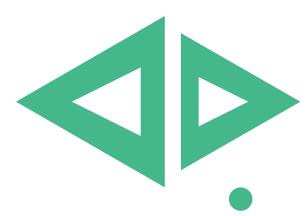
Aprende a programar en cualquier lenguaje (primer curso gratis) en:

👉 ed.team/programacion

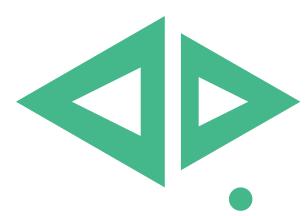
@class Caracteristicas

- 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.





Laboratorio



Instalar Python

Ciberseguridad - Presentaciones de ... X Visual Studio Code - Code Editin... X Welcome to Python.org X +

← → C python.org 🔒

Python PSF Docs PyPI Jobs Community

python.org

Downloads Documentation Community Success Stories News Events

Python 3: Lists
...
List and the
...
All releases
Source code
Windows
Mac OS X
Other Platforms
License
Alternative Implementations

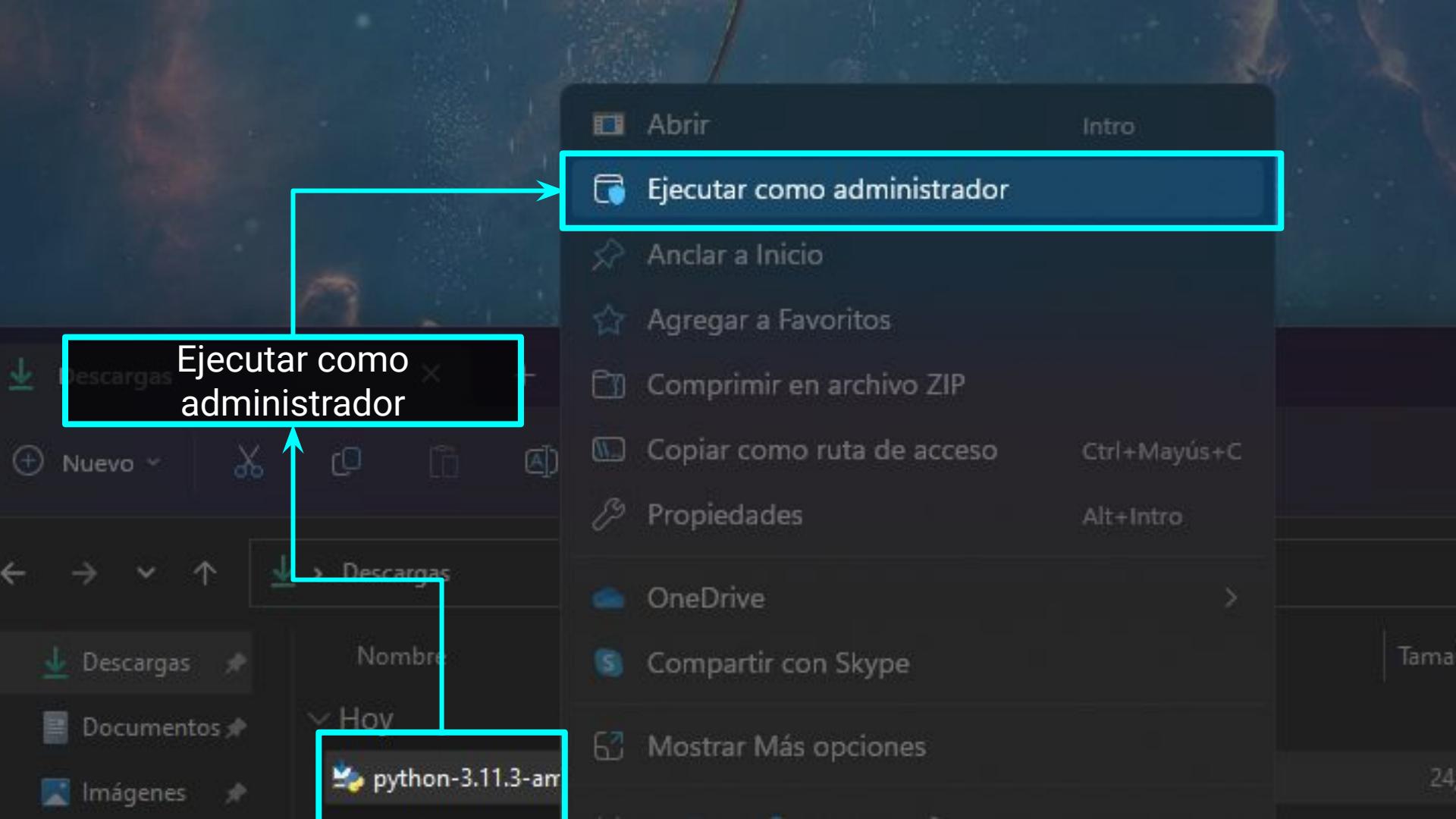
1.- Ingresamos al siguiente link

Download for Windows
Python 3.9.0
Note that Python 3.9+ cannot be used earlier.
Not the OS you are looking for? Python can be used on many operating systems and environments.
View the full list of downloads.

Python 3.9.0

2.- Botón de descarga para python version 3.X.X

https://www.python.org/ftp/python/3.9.0/python-3.9.0-amd64.exe



Install Python 3.9.5 (64-bit)

Select Install Now to install Python with default settings, or choose Customize to enable or disable features.

→ **Install Now**

C:\Users\nesto\AppData\Local\Programs\Python\Python39

Includes IDLE, pip and documentation
Creates shortcuts and file associations

Seleccionar

→ **Customize installation**

Choose location and features



- Install launcher for all users (recommended)
- Add Python 3.9 to PATH

Cancel

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



Back

Install

Cancel

Setup was successful

New to Python? Start with the [online tutorial](#) and [documentation](#). At your terminal, type "py" to launch Python, or search for Python in your Start menu.

Seleccionar

See what's new in this release, or find more info about using [Python on Windows](#).



Disable path length limit

Changes your machine configuration to allow programs, including Python, to bypass the 260 character "MAX_PATH" limitation.

Close

```
> python --version  
Python 3.11.0
```

Kendrick Lamar m Hu

nesto

pwsh 61% 21:24:08

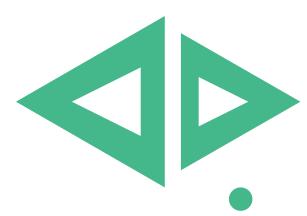
Comprobamos que tenemos python instalado

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

```
nesto          X + ▾  
Cargar los perfiles personales y de sistema tardó 3558ms.  
cd .\Documents\GitHub\ 0ms  
» cd .\Documents\GitHub\
```

2

Creamos nuestro entorno virtual

```
GitHub          X + ▾  
Cargar los perfiles personales y de sistema tardó 3558ms.  
cd .\Documents\GitHub\ 15ms  
» python -m venv venv
```

3

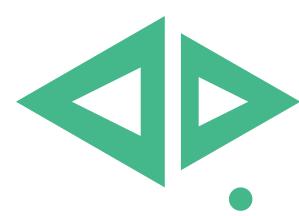
Activamos nuestro entorno virtual de la siguiente manera en Powershell o CMD

(venv)

dio

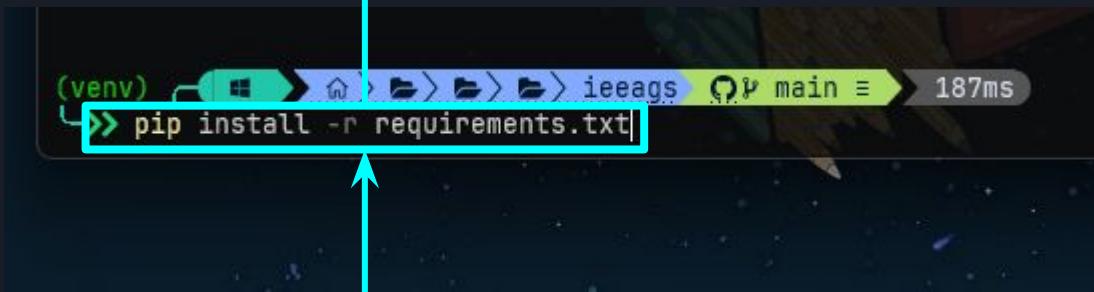
> .\venv\Scripts\activate
(venv) > GitHub > 269ms

dio



Crear espacio de
trabajo

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



A screenshot of a terminal window. The path in the top bar is '(venv) ~ ➜ ieeags ➜ main ➜ 187ms'. The command 'pip install -r requirements.txt' is typed in the bottom input field. The terminal has a dark background with light-colored text and icons.

Instalamos el archivo de
requerimientos para python

```
> python --version  
C:\Program Fi...e: can't open file 'C:\\Us...  
ers\\nesto\\ver... No such file or directory  
> python --version  
Python 3.9.5  
> cd .\\Documents\\  
(venv) 6 62ms  
->> django-admin startproject holamundo  
5  
invocamos a django  
Le indicamos que queremos comenzar un proyecto  
7  
Le damos un nombre a nuestro proyecto
```



A screenshot of a terminal window titled "Documents". The path bar shows "Documents > Documents" with a duration of "15ms". Below the path bar, the command "cd holamundo/" is entered, with the entire command underlined by an orange bracket. The terminal interface includes icons for Windows, Home, File, PowerShell, Taskbar, and Python environment.

8

accedemos a la ubicación que se creó en nuestro directorio

```
(venv) holamundo > python manage.py runserver
```

The screenshot shows a terminal window with a green title bar containing '(venv)' and 'holamundo'. Below the title bar, there's a navigation bar with icons for file, home, folder, and search, followed by '0ms'. The main terminal area shows the command 'python manage.py runserver' being typed. A red bracket and arrow point from a circled number '9' at the bottom left to this command.

9

Ejecutamos nuestro servidor de django

holamundo

x + ×

— □ ×

> python manage.py runserver

Watching for file changes with StatReloader

Performing system checks...

Servidor inicializado

System check identified no issues (0 silenced).

You have 18 unapplied migrations. Your project may not work properly until you apply the migrations for app(s): admin, auth, contenttypes, sessions.

Run 'python manage.py migrate' to apply them.

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

django

View release notes for Django 3.2

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



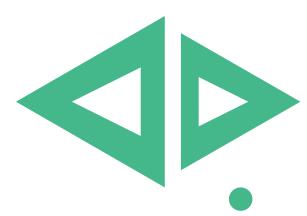
Django Documentation
Topics, references, & how-to's



Tutorial: A Polling App
Get started with Django



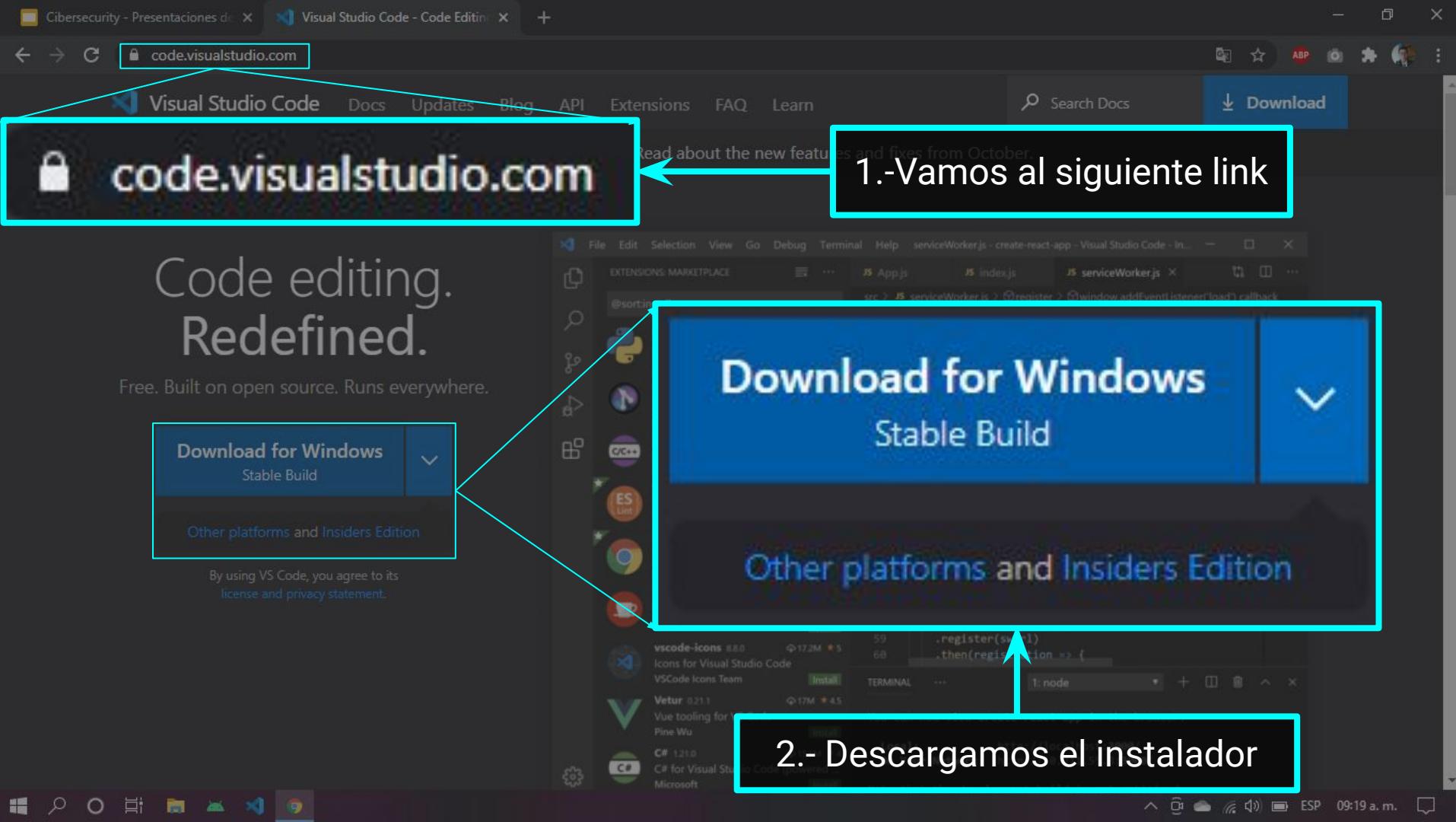
Django Community
Connect, get help, or contribute

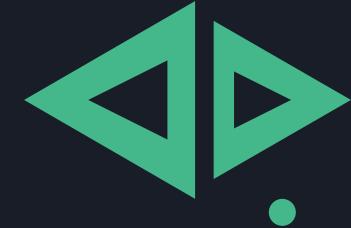


¡Felicitaciones!

¡Create tu primer Hello World!
con django

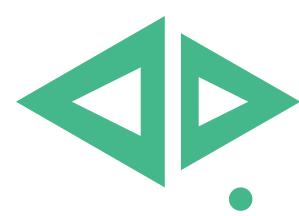
Continuará...





django

#HappyHacking 



Comenzamos
nuestro Proyecto

Desarrollo

X + ▾

- □ X

> .\venv\Scripts\activate

(venv) > @ > Desarrollo > 317ms

↳> django-admin startproject OSIL_django_course|

pwsh

54%

venv 3.11.3

17:29:37

django-admin startproject OSIL_django_course|

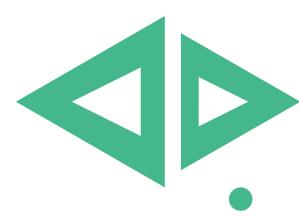
Comenzamos un nuevo proyecto

1

```
> ls
Direc... (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



Instalamos el
archivo
requirements.txt

byNestorCode / OSIL_django_course · Public

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

Code master Go to file

OSIL_django_course / requirements.txt

byNestorCode Se modificaron las variables de entorno en el archivo settings.py y ...

Code Blame 27 lines (27 loc) · 1020 Bytes

```
1 asgiref==3.4.1
2 astroid==2.8.0
3 colorama==0.4.1
4 Django==4.0.18
5 django-4-jet==1.0.9
6 django-ckeditor==6.1.0
7 django-crispy-forms==1.13.0
8 django-environ==0.8.0
9 django-js-asset==1.2.2
10 isort==5.9.3
11 lazy-object-proxy==1.6.0
12 mccabe==0.6.3
13 mysqlclient==2.1.1
14 Pillow==9.5.0
15 platformdirs==3.0.0
16 pytz==2021.1
17 six==1.16.0
18 sqlparse==0.4.0
19 tomli==0.10.2
20 typing-extensions==3.10.0.2
21 tzdata==2023.3
22 wrapt==1.12.1
```

766925e · 3 hours ago History Download raw file

Raw    

766925e · 3 hours ago History Download raw file

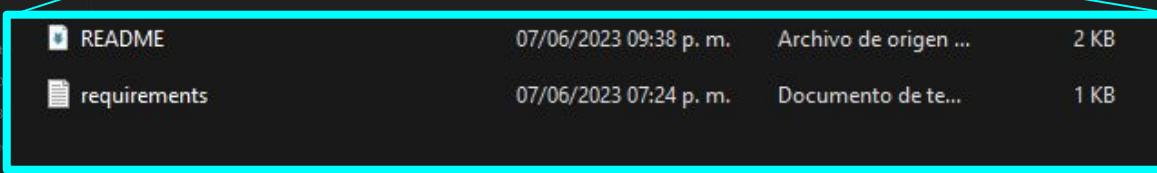
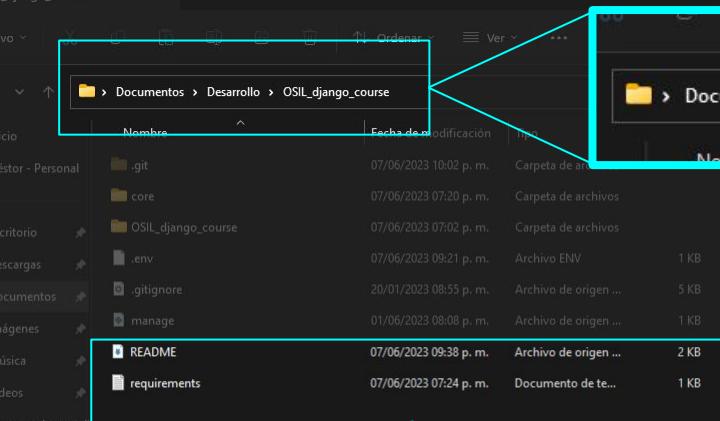
Raw    

Descargamos el archivo

5

6

Nos dirigimos a la carpeta raíz de nuestro proyecto



7

Copiamos el archivo descargado en esta ubicación

8

OSIL_django_course

x + v

- □ ×

> ls

Directorio: C:\Users\nesto\Documents\Desarrollo\OSIL_django_course

Ejecutamos el siguiente comando para instalar las dependencias necesarias a utilizar en nuestro curso:

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.
-a----	20/01/2023 08:55 p. m.
-a----	01/06/2023 08:08 p. m.
-a----	07/06/2023 09:38 p. m.
-a----	07/06/2023 07:24 p. m.

240	.env
4187	gitignore
696	manage.py
1263	README.md
1020	requirements.txt

(venv) ➜ pip install -r requirements.txt

(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 < X + < - < X  
> ls  
  
Directorio: C:\Users\nesto\Doc  
  
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.  
-a---- 20/01/2023 08:55 p. m.  
-a---- 01/06/2023 08:08 p. m.  
-a---- 07/06/2023 09:38 p. m.  
-a---- 07/06/2023 07:24 p. m.  
  
          core  
          OSIL_django_course  
          .env  
          246 .gitignore  
          4107 manage.py  
          696 README.rst  
          1263  
  
(venv) pip freeze  
  
(venv) >>> pip freeze
```

Para comprobar que todo ha salido bien ejecutaremos el siguiente comando en nuestra consola:

9

```
> 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

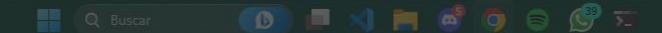
11

Nos dirigimos al repositorio en GitHub
[Click aquí para ir al repositorio](#)

The screenshot shows a GitHub repository page. At the top, there's a navigation bar with links like 'Pull requests', 'Issues', 'Codespaces', 'Marketplace', and 'Explore'. Below the navigation bar, the repository title 'byNestorCode/OSIL_django_course' is displayed, with a callout box labeled '11' pointing to it. The main content area shows a list of files and their commit history. One file, 'requirements.txt', is highlighted with a callout box labeled '12' pointing to it. The commit history includes entries like 'Se modificaron las variables de entorno en el archivo settings.py y ...', 'Update PostgreSQL link how principal DB in README', and 'Inicio del proyecto'. On the right side of the repository page, there are sections for 'Insights', 'Settings', 'django is a Python framework and was made to OSIL Community (Open Source Innovation Lab)', 'Activity', '2 stars', '1 watching', '0 forks', and 'Releases'.

12

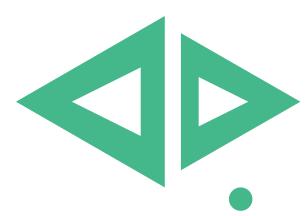
Damos click en el archivo
requirements.txt



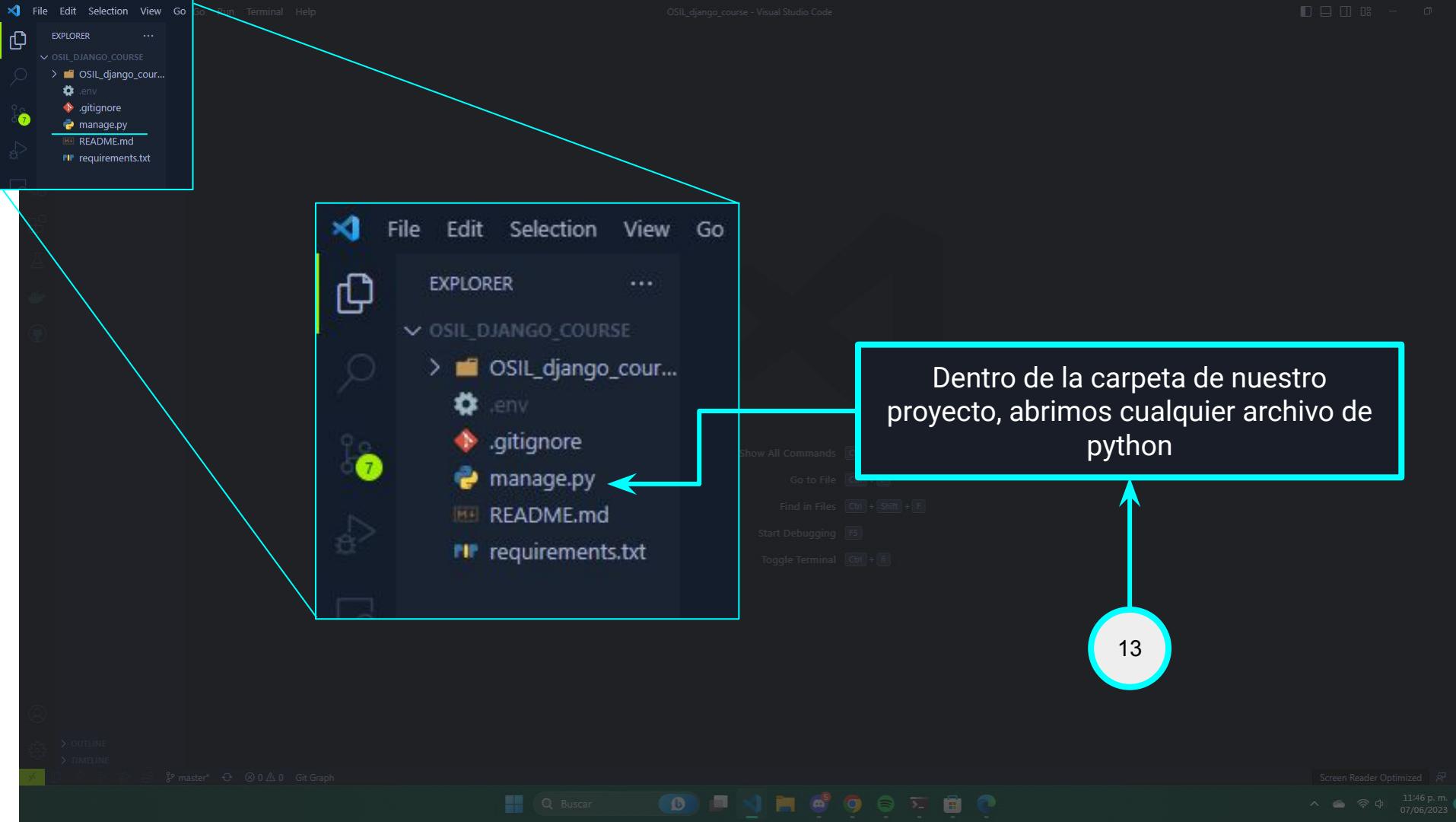


¿Sabías qué?

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



EXPLORER ...

OSIL_DJANGO_COURSE

> OSIL_django_cour...
.env
.gitignore
manage.py
README.md
requirements.txt

File Edit Selection View Go

EXPLORER ...

OSIL_DJANGO_COURSE

> OSIL_django_cour...
.env
.gitignore
manage.py
README.md
requirements.txt

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

13

> OUTLINE
> TIMELINE

master* Git Graph

Screen Reader Optimized

11:46 p. m.
07/06/2023

File Edit Selection View Go Run Terminal Help

manage.py - OSIL_django_course - Visual Studio Code

EXPLORER ... manage.py 1 × manage.py > ...

OSIL_DJANGO_COURSE: core, OSIL_django_course, .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?"
        ) from exc
    execute_from_command_line(sys.argv)

if __name__ == '__main__':
    main()
```

14

from django.core.management import execute_from_command_line
except ImportError as exc:

from django.core.management import execute_from_command_line
except ImportError as exc:

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

Screen Reader Optimized Lin 1, Col 1 Spaces: 4 UTF-8 Ctrl

24°C Práct. despejado

Buscar

09:37 p. m.
08/06/2023

File Edit Selection View Go Run Terminal Help

manage.py - OSIL_django_course - Visual Studio Code

EXPLORER ... manage.py 1 ×

OSIL_DJANGO_COURSE ... manage.py > ...

> core

> OSIL_django

gitignore

manage.py

README.md

requirements.txt

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.settings')

try:

from django.core.management import execute_from_command_line

except ImportError:

raise ImportError(

"Couldn't import Django.

Make sure that Django is installed and

available on your PYTHONPATH.

You may need to activate a virtual environment.")

except Exception as exc:

from .util import log_error

log_error("Error running management command: %s" % exc)

sys.exit(1)

execute_from_command_line(sys.argv)

21 if __name__ == '__main__':

22 main()

15

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

3.11.3 64-bit

Screen Reader Optimized Lin 1, Col 1 Spaces: 4 UTF-8 Chk Python 3.11.3 64-bit

24°C Práct. despejado

master 0 1 Git Graph

Buscar

9:37 p. m. 08/06/2023

```
#!/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.settings')

        try:
            from django.core.management import execute_from_command_line
        except ImportError:
            raise ImportError(
                "Couldn't import Django.\n"
                "Make sure that Django is installed and\n"
                "available on your PYTHONPATH.\n"
                "You may need to activate a virtual environment.")
        except Exception as exc:
            from .util import log_error
            log_error("Error running management command: %s" % exc)
            sys.exit(1)

        execute_from_command_line(sys.argv)

if __name__ == '__main__':
    main()
```

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows a project structure for "OSIL_django_course" containing files like "core", "OSIL_django_course", ".gitignore", "manage.py", "README.md", and "requirements.txt".
- Code Editor:** Displays the content of "manage.py". The code includes imports for os and sys, a main function handling administrative tasks, and a conditional block for running the script.
- Select Interpreter Dialog:** A modal window titled "Select Interpreter" is open, listing three interpreters:
 - Selected Interpreter: C:\Program Files\Python311\python.exe
 - 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)
- Status Bar:** Shows the file is "master", has 0 changes, and the Python version is 3.11.3 64-bit. It also indicates "Screen Reader Optimized".
- Bottom Bar:** Includes icons for file operations, search, and navigation.

A large cyan box highlights the "Select Interpreter" dialog, and a cyan arrow points from the text in the bottom box to the "Venv" interpreter entry in the dialog. A cyan circle with the number "16" is located in the bottom right corner.

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

16

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows a project named "OSIL_django_course" with files like "manage.py", "core", "OSIL_django_course", ".gitignore", "manage.py", "README.md", and "requirements.txt".
- Code Editor:** Displays Python code for "manage.py". The code includes imports for os and sys, a main function that runs administrative tasks, and a check for the __name__ variable.
- Select Interpreter Dialog:** A modal window titled "Select Interpreter" is open, listing three interpreters:
 - Selected Interpreter: C:\Program Files\Python311\python.exe
 - ★ 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
- Bottom Status Bar:** Shows "Screen Reader Optimized", "Ln 1, Col 1", "Spaces: 4", "UTF-8", "CRLF", "Python", "3.11.3 64-bit", and the date "08/06/2023".

A large blue callout box points from the bottom right towards the "Enter interpreter path..." input field in the "Select Interpreter" dialog. Another blue callout box points from the bottom right towards the explanatory text at the bottom right of the slide.

17

Para cambiar esto nos dirigiremos al apartado que nos permite buscar un nuevo intérprete de python

File Edit Selection View Go Run Terminal Help

OSIL_django_course

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_course.settings')
    try:
        from django.core.management import execute_from_command_line
    except ImportError as exc:
        raise ImportError(
            "Couldn't import module '%s'. Are you sure it's available on your PYTHONPATH? Did you forget to activate your environment?") from exc
    execute_from_command_line(sys.argv)

if __name__ == '__main__':
    main()
```

manage.py - OSIL_django_course - Visual Studio Code

Enter path to a Python interpreter.

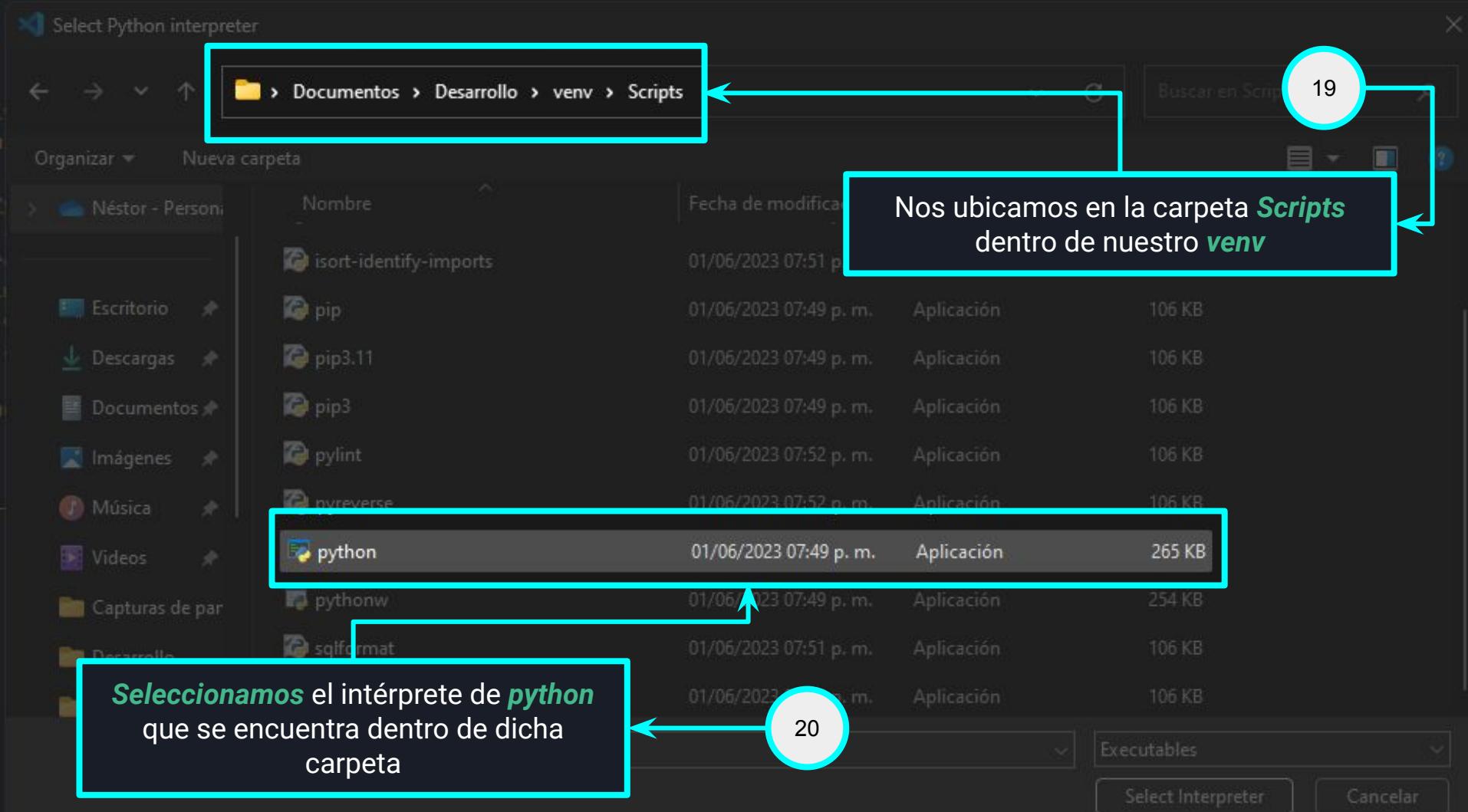
Find...

Browse your file system to find a Python interpreter.

Damos click en "Find..." para comenzar la búsqueda

18

Screen Reader Optimized Lin 1, Col 1 Spaces: 4 UTF-8 CRLF ↵ Python 3.11.3 64-bit ⚡ 23°C 09:59 p. m. 08/06/2023



File Edit Selection View Go Run Terminal Help

manage.py - OSIL_django_course - Visual Studio Code

EXPLORER ... manage.py x

OSIL_DJANGO_COURSE manage.py > ...

> core
> OSIL_django_cour...
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?"
        )
    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?"
        )

    main()

23
```

Podemos ver que los errores desaparecieron y el intérprete de nuestro **venv** quedó **activo**

21

{ Python 3.11.3 ('venv': venv)

Screen Reader Optimized | Lin 23, Col 1 | Spaces:4 | UTF-8 | CTRL + F | { Python 3.11.3 ('venv': venv)

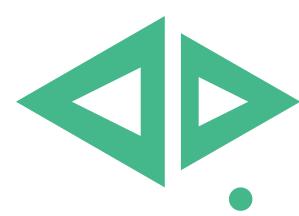
23°C
Cerca del récord

OUTLINE
TIMELINE

master 0 0 △ 0 Git Graph

Buscar

10:13 p. m.
08/06/2023

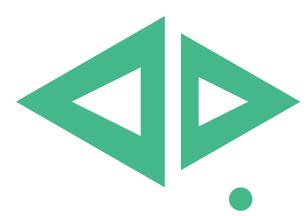


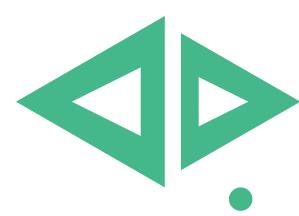
Si todo ha salido correctamente...

¡FELICIDADES!

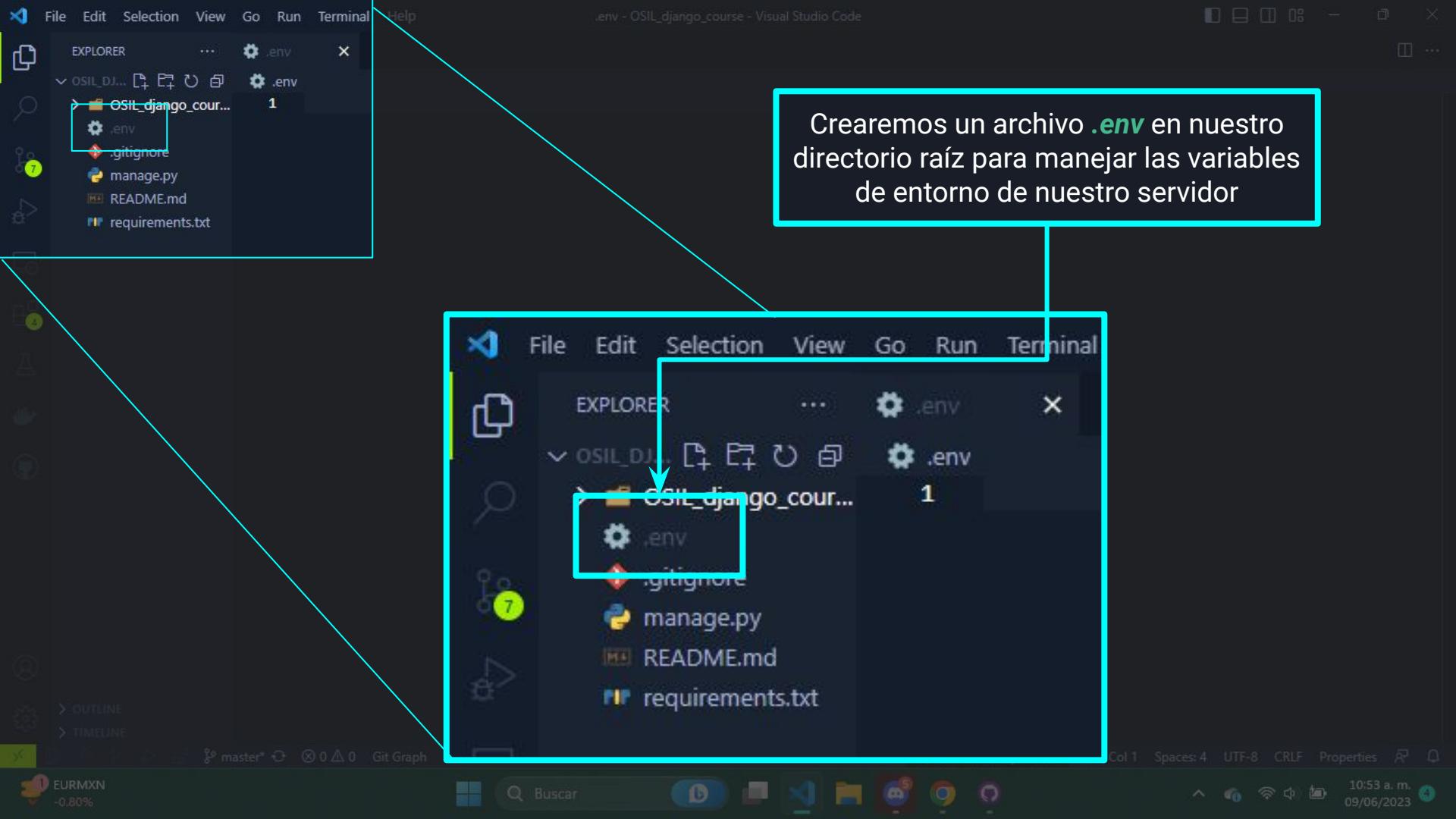
ya puedes comenzar a

HACKEAR al mundo





Vamos a darle
seguridad a
nuestro código



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.

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** On the left, it displays the project structure under "OSIL_django_course". The "core" folder contains files: __init__.py, asgi.py, settings.py, urls.py, and wsgi.py. Other files in the root include .env, .gitignore, manage.py, README.md, and requirements.txt.
- Code Editor:** The main area shows the content of the "settings.py" file. The code is generated by "django-admin startproject". It imports os, environ, and Env from environ, and Path from pathlib. It defines BASE_DIR and parent, and creates a .env variable. It includes sections for production deployment and production secrets. A security warning at the bottom advises against running with DEBUG=True in production.
- Annotations:** Three cyan-colored boxes highlight specific areas:
 - A large box surrounds the entire "core" folder in the File Explorer.
 - A medium box surrounds the "settings.py" file in the code editor.
 - A small box points to the first line of the "settings.py" code in the editor.
- Text Overlay:** A cyan-bordered box in the bottom right corner contains the text "Abrimos nuestro archivo **settings.py**".
- Bottom Status Bar:** Shows the current branch (master), commit count (0), file graph status, screen reader optimization, line and column numbers (Ln 17, Col 1), spaces (4), encoding (UTF-8), CRLF, Python 3.9.5 (venv: venv), and a system status bar with battery level (31°C Soledad), signal strength, and date/time (01:42 p.m., 09/06/2023).



EXPLORER

OSIL_DJANGO COURSE
core
OSIL_django_course
__init__.py
asgi.py
settings.py
urls.py
wsgi.py
env
.gitignore
manage.py
README.md
requirements.txt

... settings.py x

```
13 import os
14 import environ
15 from environ.environ import Env
16 from pathlib import Path
```

For more information on this
<https://docs.djangoproject.com/en/4.0/ref/settings/>

```
13 import os
14 import environ
15 from environ.environ import Env
16 from pathlib import Path
```

```
17 # Build paths inside the project like this: BASE_DIR = Path(__file__).resolve().parent.parent
18
19 # environ init
20 env = environ.Env()
21 environ.Env.read_env(os.path.join(BASE_DIR, '.env'))
22
23 # Quick-start development settings - unsuitable for production!
24 # See https://docs.djangoproject.com/en/4.0/howto/deployment/checklist/
25
26 # SECURITY WARNING: keep the secret key used in production secret!
27 SECRET_KEY = env.str('SECRET_KEY')
28
29 # SECURITY WARNING: don't run with debug turned on in production!
30 DEBUG = env.bool('DEBUG', default=False)
```

Importamos las siguientes *librerías*
dentro del archivo **settings.py**



EXPLORER

OSIL_DJANGO COURSE

- > core
- > OSIL_django_course
 - __init__.py
 - asgi.py
 - settings.py
 - urls.py
 - wsgi.py
- .env
- .gitignore
- manage.py
- README.md
- requirements.txt

.env settings.py x

OSIL_django_course > settings.py > ...

```
"""
Django settings for OSIL_django_course project.

Generated by 'django-admin startproject' using Django 4.0.10.

For more information on this file, see

```

Inicializamos environ y le indicamos en qué ruta se encontrará el archivo .env que contiene nuestras configuraciones

```
14 import environ
15 from environ import Env
16 from pathlib import Path
17
18 # Build paths inside the project like this: BASE_DIR / "subdir".
19 BASE_DIR = Path(__file__).resolve().parent.parent
20
21 # environ init
22 env = environ.Env()
23 environ.Env.read_env(os.path.join(BASE_DIR, '.env'))
24
25 # Quick-start development settings - unsuitable for production
26 # See https://docs.djangoproject.com/en/4.0/howto/deployment/checklist/
27
```

```
21 # environ init
22 env = environ.Env()
23 environ.Env.read_env(os.path.join(BASE_DIR, '.env'))
```





EXPLORER



settings.py



OSIL_DJANGO_COURSE

> core
< OSIL_django_cour..._init_.py
asgi.py
settings.py
urls.py
wsgi.py

env

.gitignore

manage.py

README.md

requirements.txt

OSIL_django_course > settings.py > ...

Quick-start development settings - unsuitable for production
See https://docs.djangoproject.com/en/4.0/howto/deployment/checklist/# 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=[]))

35 # Application definition

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=[]))

django.middleware.csrf.CsrfViewMiddleware',
'django.contrib.auth.middleware.AuthenticationMiddleware',
'django.contrib.messages.middleware.MessageMiddleware',
'django.middleware.clickjacking.XFrameOptionsMiddleware',

g master □ 0 △ 0 Git Graph

1

Screen Reader Optimized Ln 35, Col 1 Spaces: 4 UTF-8 CRLF Python 3.9.5 ('venv': venv) R Q

32°C
Soledad

Buscar



B



F



S



G



D



A

02:53 p.m.
09/06/2023Comenzamos la configuración de nuestras **variables de entorno**

EXPLORER

Así quedaría la
configuración de nuestra
Base de Datos

WSGI_APPLICATION = 'OSIL_django_course.wsgi.application'

Database

<https://docs.djangoproject.com/en/4.0/ref/settings/#>

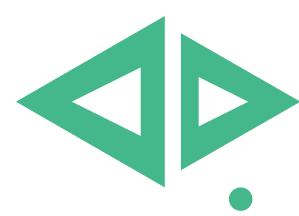
```
 DATABASES = {  
     'default': {  
         'ENGINE': 'django.db.backends.postgresql',  
         'NAME': env.str('POSTGRES_DB'),  
         'USER': env.str('POSTGRES_USER'),  
         'PASSWORD': env.str('POSTGRES_PASSWORD'),  
         'HOST': env.str('DB_HOST'),  
         'PORT': env.int('DB_PORT')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

Password validation

<https://docs.djangoproject.com/en/4.0/ref/settings/#auth-password-validation>

```
AUTH_PASSWORD_VALIDATORS = [  
    {  
        'NAME': 'django.contrib.auth.password_validation.UserAttributeSimilarityValidator',  
    },  
    {  
        'NAME': 'django.contrib.auth.password_validation.MinimumLengthValidator',  
    },  
]
```

```
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

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

```
    .Contrib.auth.password_validation.NumericPasswordValidator',
```

```
LANGUAGE_CODE = 'es-mx'
```

```
TIME_ZONE = 'America/Mexico_City'
```

```
USE_I18N = True
```

```
USE_L10N = True
```

```
USE_TZ = True
```

```
LANGUAGE_CODE = 'es-mx'
```

```
TIME_ZONE = 'America/Mexico_City'
```

```
USE_I18N = True
```

```
USE_L10N = True
```

```
USE_TZ = True
```

```
# Static files (CSS, JavaScript, Images)
```

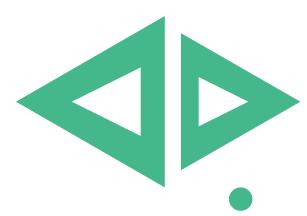
```
# https://docs.djangoproject.com/en/4.0/howto/static-files/
```

```
STATIC_URL = 'core/static/'
```

```
# Default primary key field type
```

```
# https://docs.djangoproject.com/en/4.0/ref/settings/#default-field-type
```

```
DEFAULT_AUTO_FIELD = 'django.db.models.BigAutoField'
```



Nuestro archivo
.env quedaría así:



EXPLORER



OSIL_DJANGO COURSE

- > core
- > OSIL_django_cour...
- __init__.py
- asgi.py
- settings.py
- urls.py
- wsgi.py

ENV

.env

.gitignore

manage.py

README.md

requirements.txt

.env

```
1 SECRET_KEY=django-insecure
2 DEBUG=True
3 ALLOWED_HOSTS=localhost,127.0.0.1,0.0.0.0
4 POSTGRES_DB=ejemplo_de_nombre
5 POSTGRES_USER=nombre_de_usuario
6 POSTGRES_PASSWORD=psswd_de_usuario
7 DB_HOST=localhost
8 DB_PORT=5432
```

.env

```
1 SECRET_KEY=django-insecure
2 DEBUG=True
3 ALLOWED_HOSTS=localhost,127.0.0.1,0.0.0.0
4 POSTGRES_DB=ejemplo_de_nombre
5 POSTGRES_USER=nombre_de_usuario
6 POSTGRES_PASSWORD=psswd_de_usuario
7 DB_HOST=localhost
8 DB_PORT=5432
```

> OUTLINE

> TIMELINE

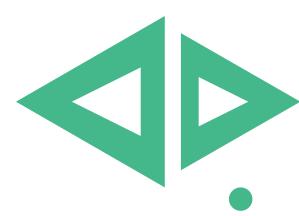
g° master ⌂ ⑧ 0 △ 0 Git Graph

Screen Reader Optimized Ln 8, Col 13 Spaces: 4 UTF-8 CRLF Properties

32°C
Soleado

Buscar

03:19 p.m.
09/06/2023

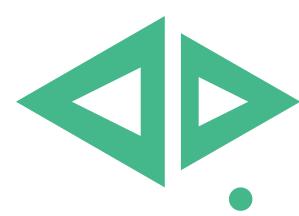


Añadimos
nuestra primer
APP:

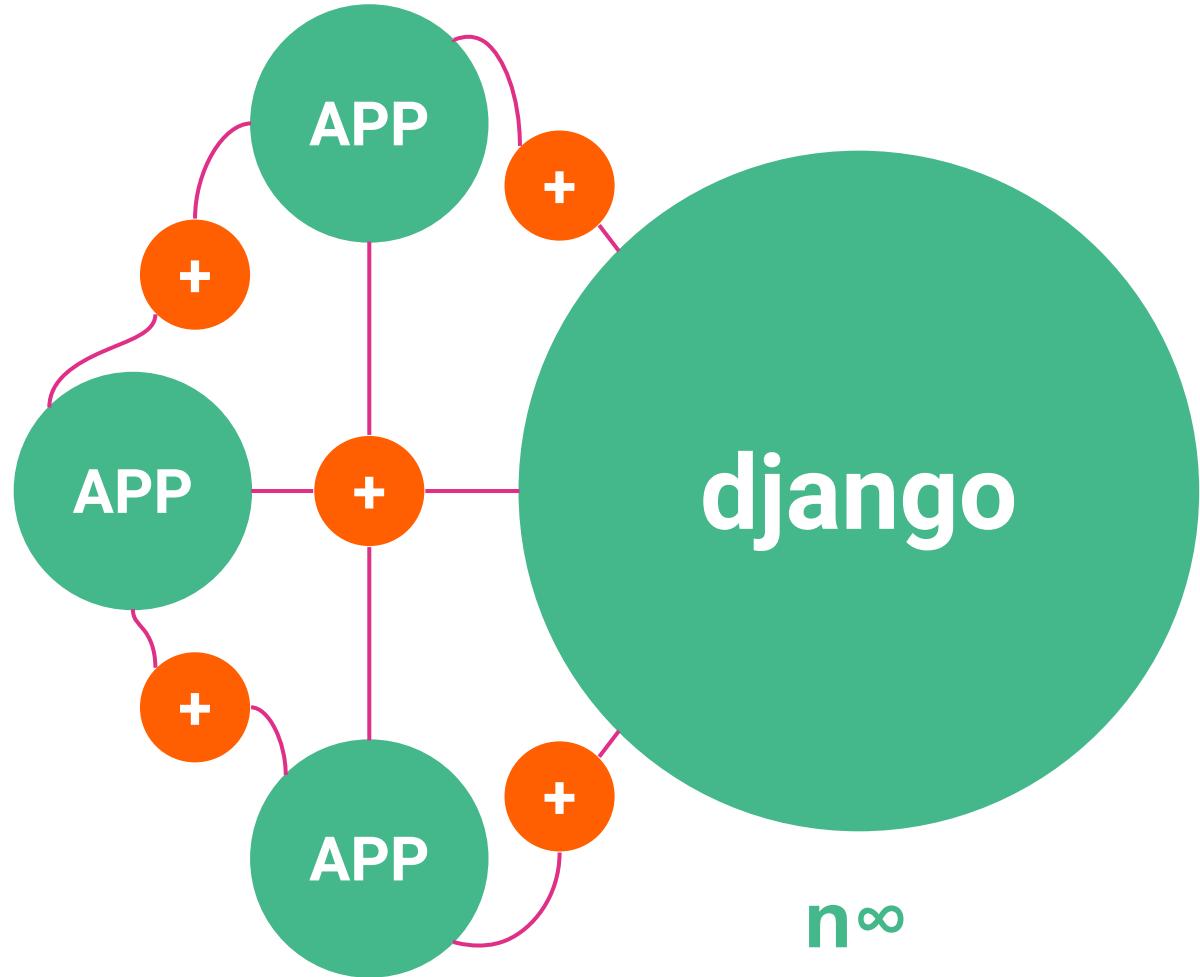
Ejecutar este **comando** nos **creará** una nueva **carpeta** con los **archivos necesarios** para comenzar a **desarrollar** una **APP**

```
>>> python manage.py startapp core
```

```
core
migrations
__init__.py
admin.py
apps.py
models.py
tests.py
views.py
```

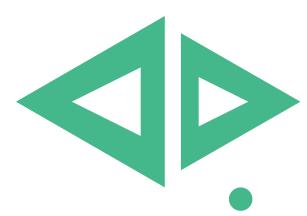


¿Qué es una app en django?



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

app = módulo



Continuará...

<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/>