

## Act. 2.2 Crear Microservicio Parte 01

Repositorio:

```
https://github.com/Gilberto-Guzman/Act.-2.2-Crear-Microservicio-Parte-01
```

Comandos utilizados:

```
python3 -m venv proyectoweb  
http://10.33.26.185:8080/crear%20registro
```

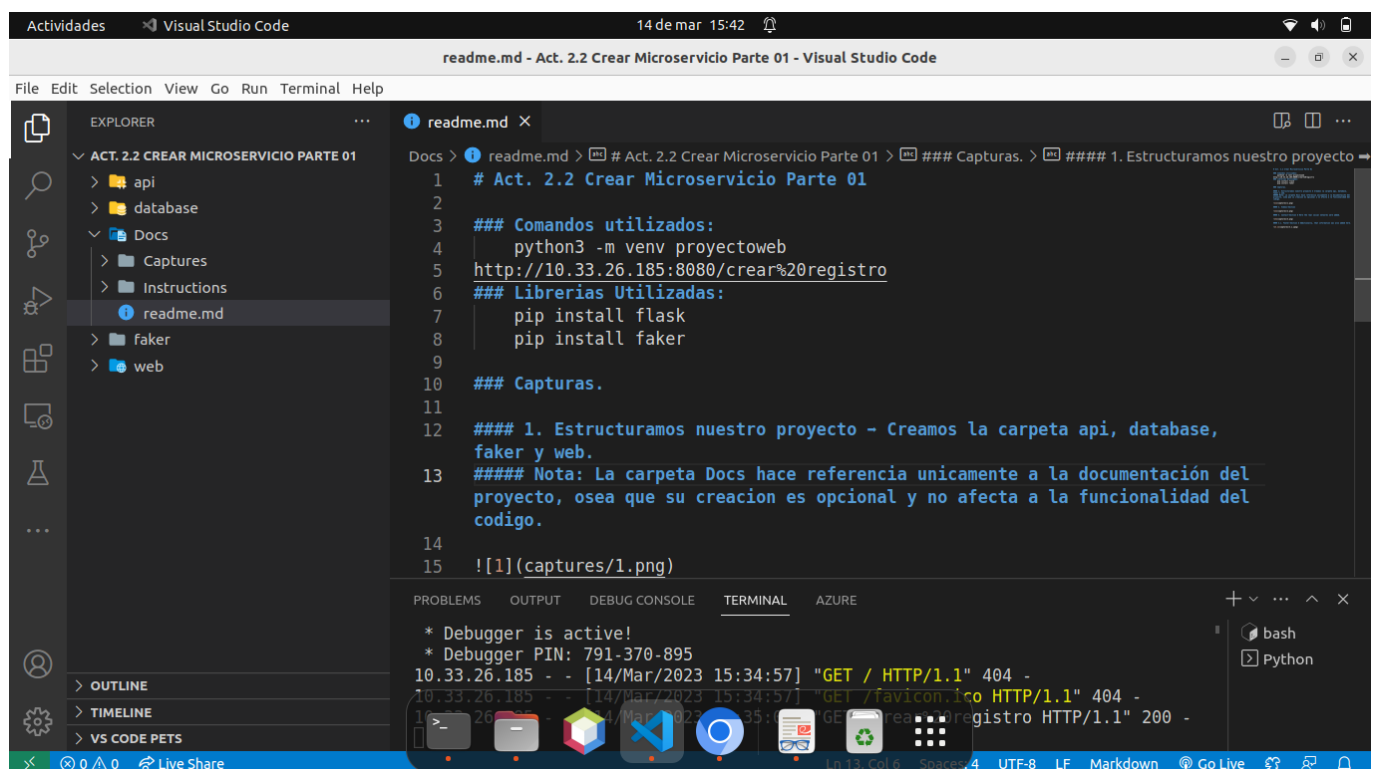
Librerías Utilizadas:

```
pip install flask  
pip install faker
```

Capturas.

**1. Estructuramos nuestro proyecto → Creamos la carpeta api, database, faker y web.**

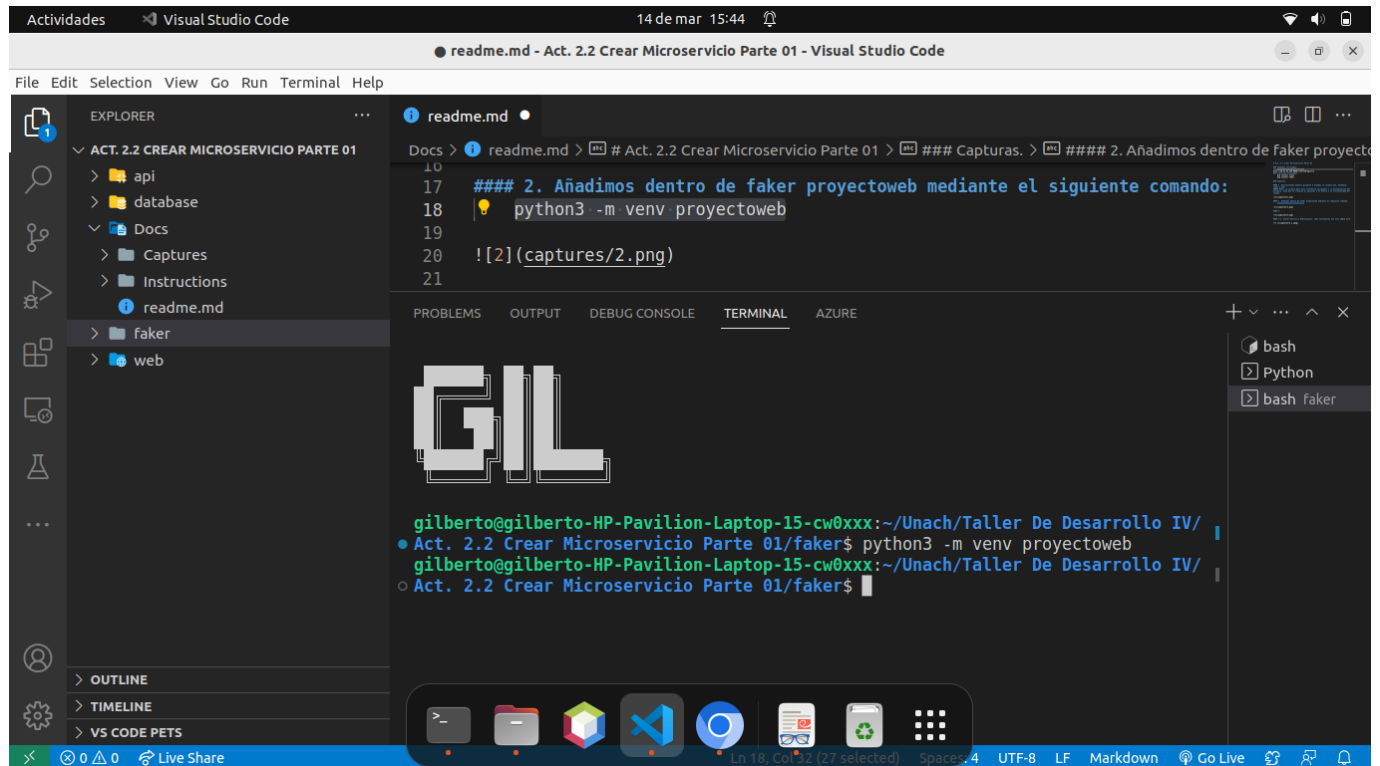
**Nota: La carpeta Docs hace referencia unicamente a la documentación del proyecto, osea que su creacion es opcional y no afecta a la funcionalidad del codigo.**



## 2. Creación de proyectoweb → Añadimos dentro de faker proyectoweb mediante el siguiente comando:

**Nota:** Este nos proovera de una plantilla la cual nos servira proximamente.

```
python3 -m venv proyectoweb
```



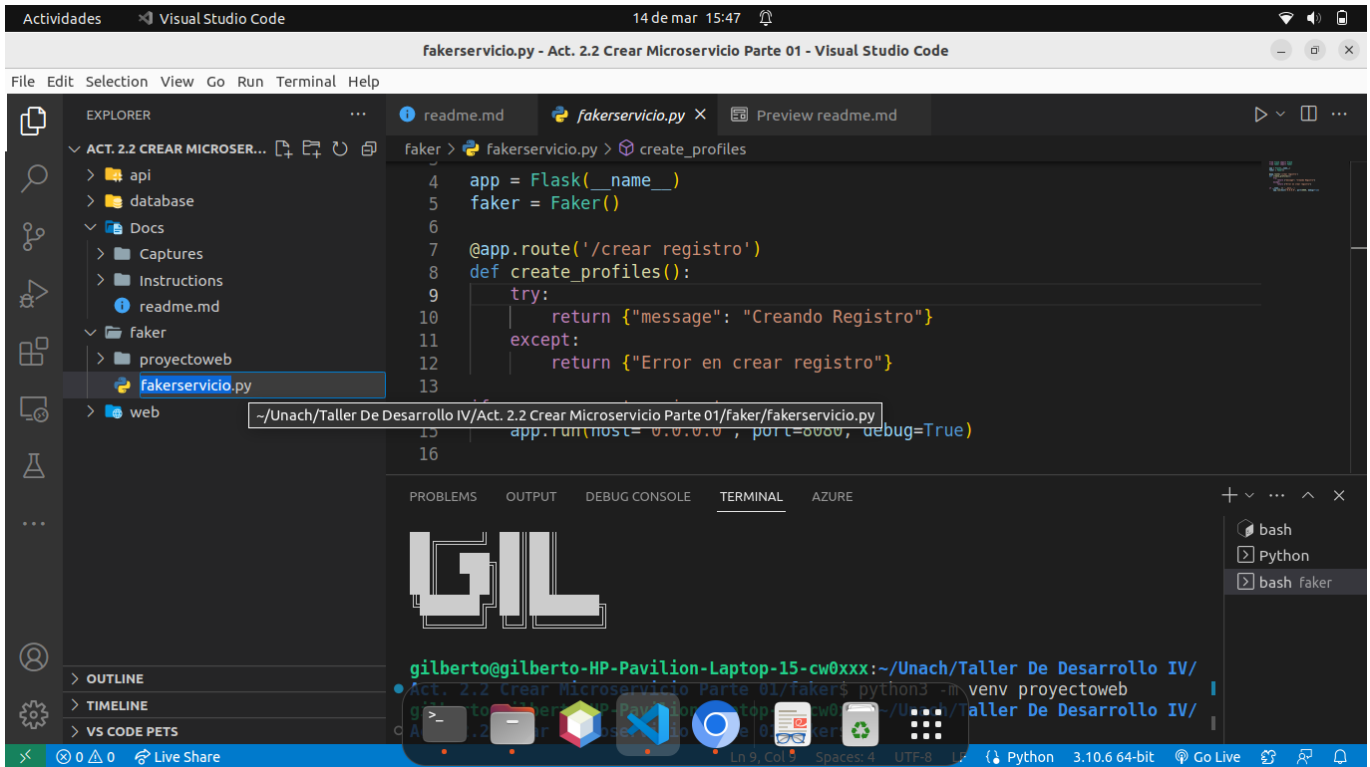
## 3. Creación de fakerservicio → Creamos un archivo de python llamado fakerservicio.py , el cual contendra el siguiente codigo:

```
from flask import Flask
from faker import Faker

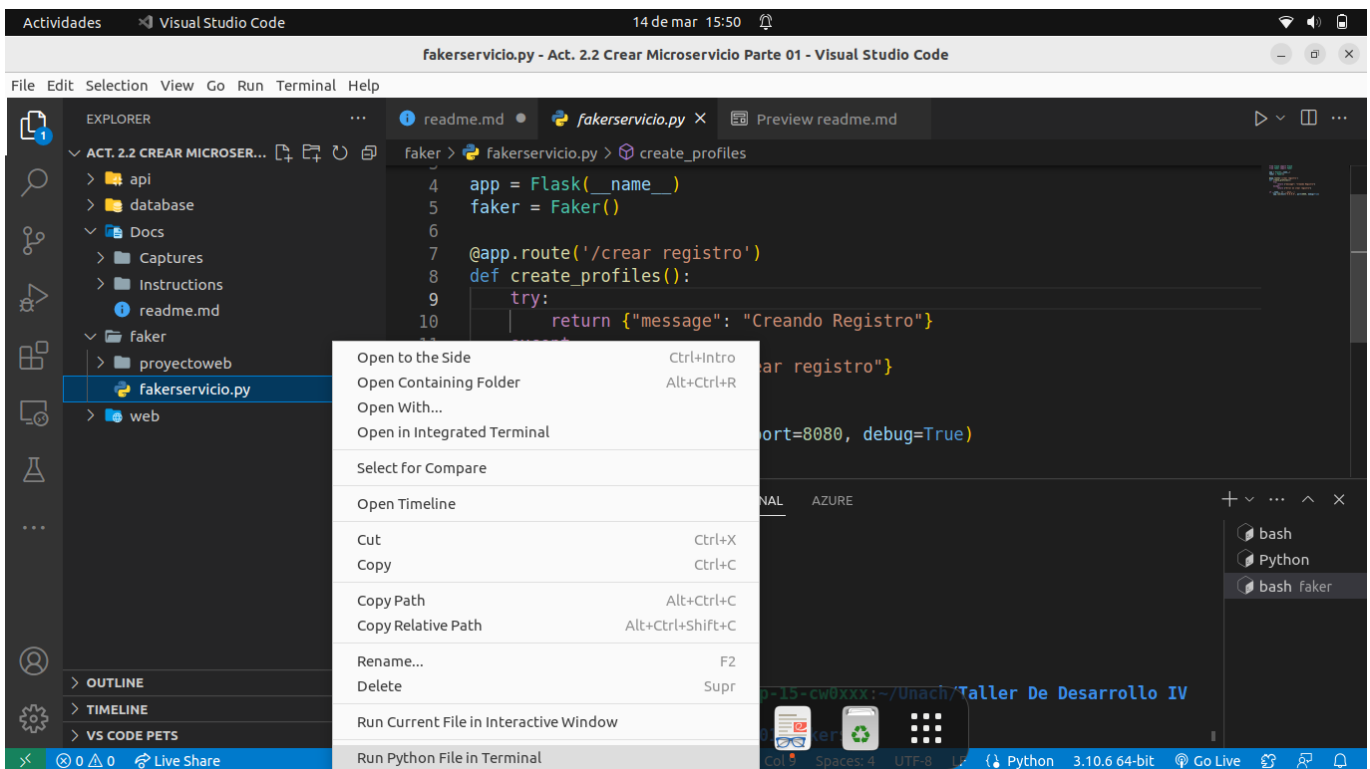
app = Flask(__name__)
faker = Faker()

@app.route('/crear registro')
def create_profiles():
    try:
        return {"message": "Creando Registro"}
    except:
        return {"Error en crear registro"}

if __name__ == '__main__':
    app.run(host='0.0.0.0', port=8080, debug=True)
```

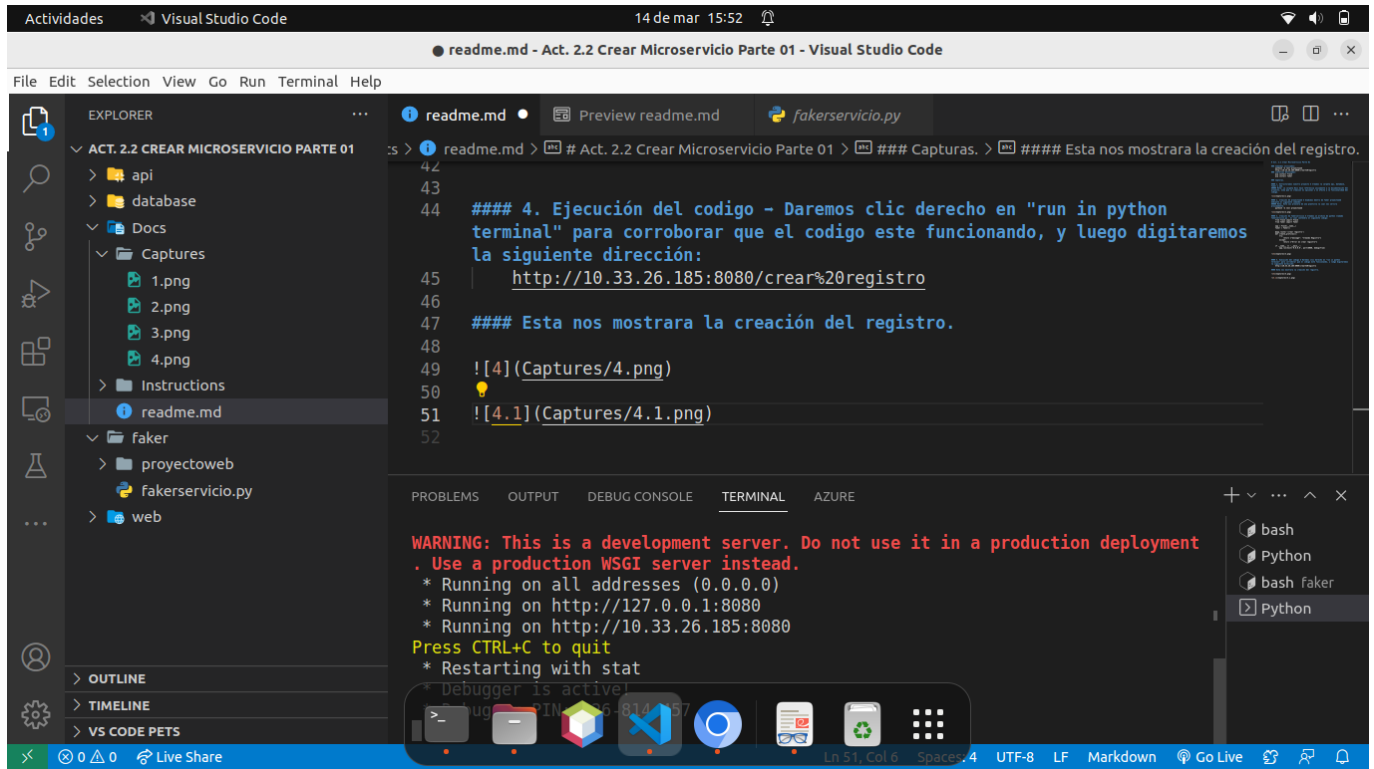


**4. Ejecución del código → Daremos clic derecho en "run in python terminal" para corroborar que el código este funcionando.**



**Y luego en el navegador, digitaremos la siguiente dirección:**

`http://10.33.26.185:8080/crear%20registro`



The screenshot shows the Visual Studio Code interface. The Explorer panel on the left displays the project structure for 'ACT. 2.2 CREAR MICROSERVICIO PARTE 01', including folders like 'api', 'database', 'Docs', 'Captures', 'Instructions', 'faker', 'proyectoweb', 'fakerservicio.py', and 'web'. The main editor area shows the 'readme.md' file with the following content:

```

42
43
44 #### 4. Ejecución del código - Daremos clic derecho en "run in python
45 terminal" para corroborar que el código este funcionando, y luego digitaremos
46 la siguiente dirección:
47 http://10.33.26.185:8080/crear%20registro
48
49 #### Esta nos mostrara la creación del registro.
50
51 ![4](Captures/4.png)
52
53

```

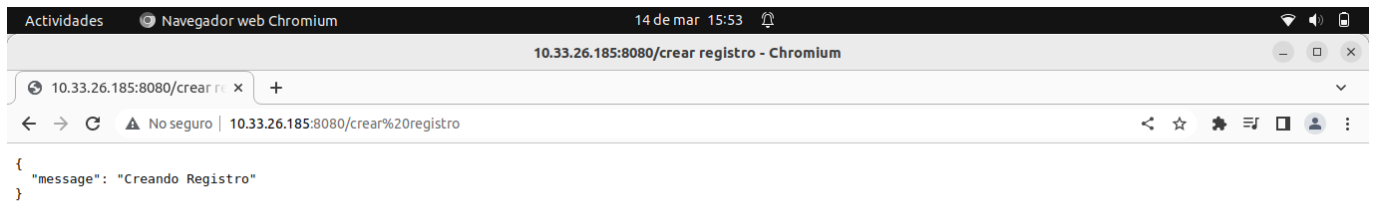
The Terminal panel at the bottom shows the output of a Python script running on a development server. The output includes a warning about using a development server in production, followed by the server's status and the URL it is running on. The output also shows the server restarting with 'stat' and a message about the debugger being active.

```

WARNING: This is a development server. Do not use it in a production deployment
. Use a production WSGI server instead.
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:8080
* Running on http://10.33.26.185:8080
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!

```

Esta nos mostrara la creación del registro.



The screenshot shows a web browser window with the address bar displaying '10.33.26.185:8080/crear registro - Chromium'. The browser's address bar shows the URL '10.33.26.185:8080/crear%20registro'. The page content displays a JSON response from the API:

```

{
  "message": "Creando Registro"
}

```