



Tarea Final

Sistemas de Gestión
Empresarial

Andrés Cuevas Rodríguez,
Pablo Rodríguez Sosa



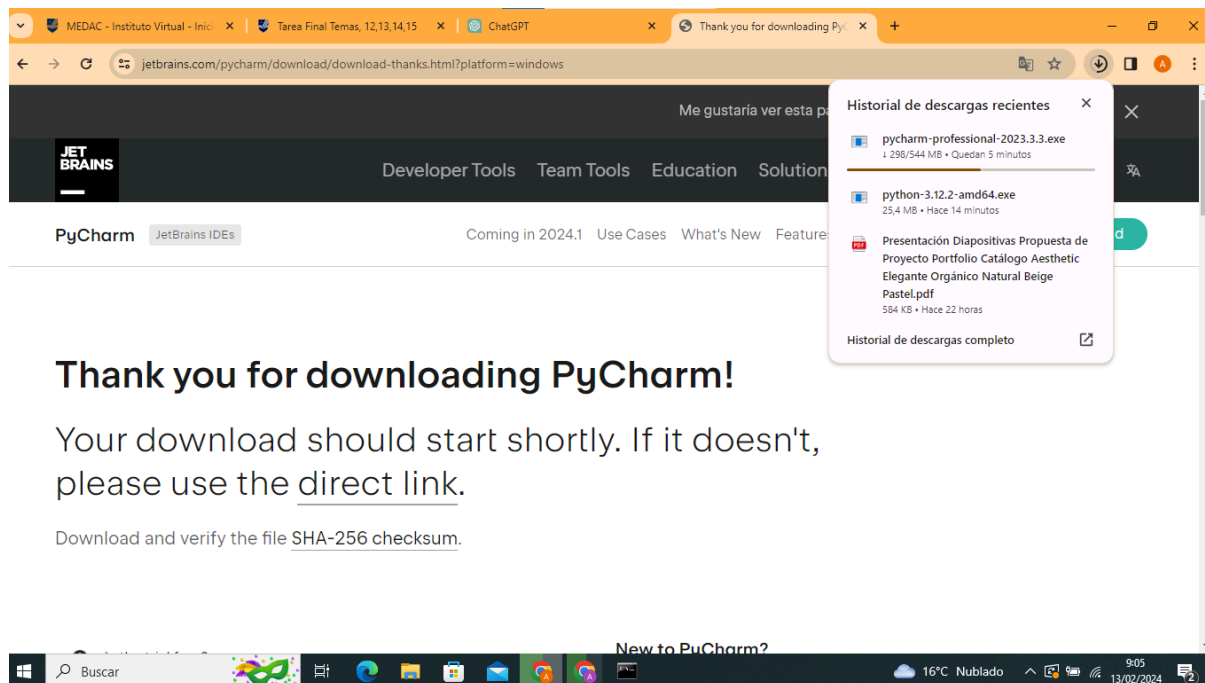
Índice

Tarea Final	1
1. Instalación de Python y PyCharm	3
2. Fase de preparación	5
3. Creación de módulos	6
4. Exportación de informes	8
5. Integración del programa en ERP-CRM	10

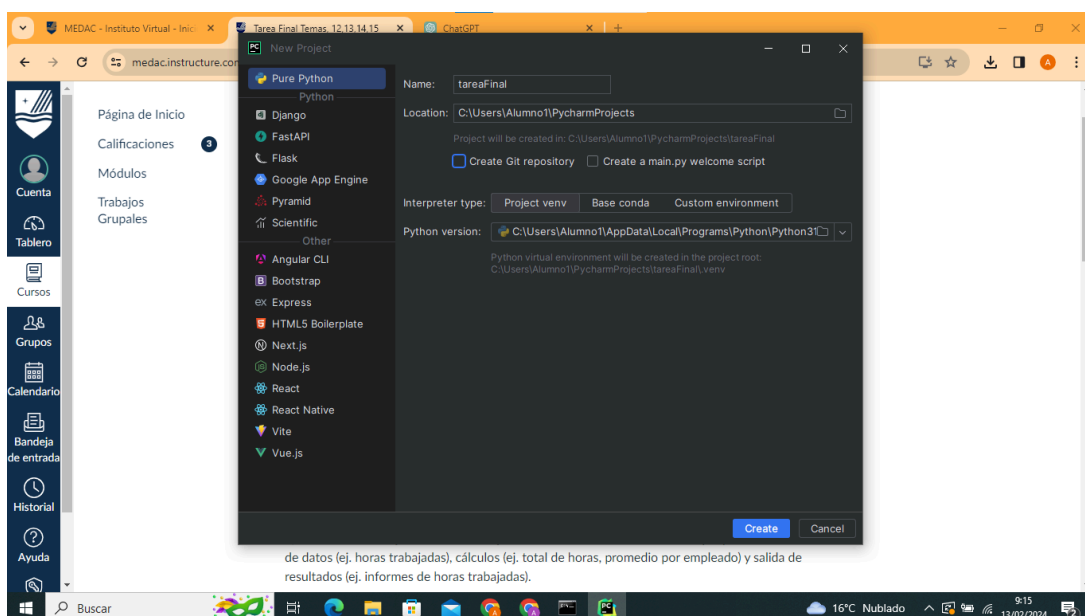


1. Instalación de Python y PyCharm

Instalamos Python y PyCharm (IDE de Python) desde las webs oficiales, para comenzar a trabajar.

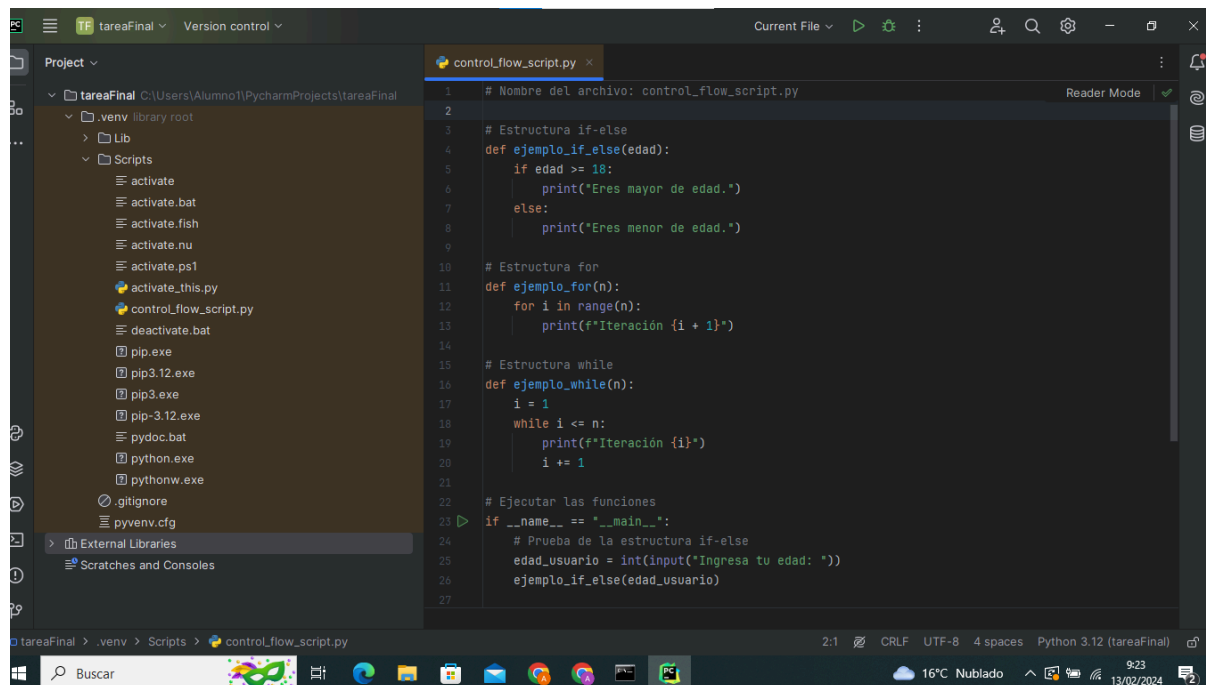


Creamos un proyecto Python en PyCharm.



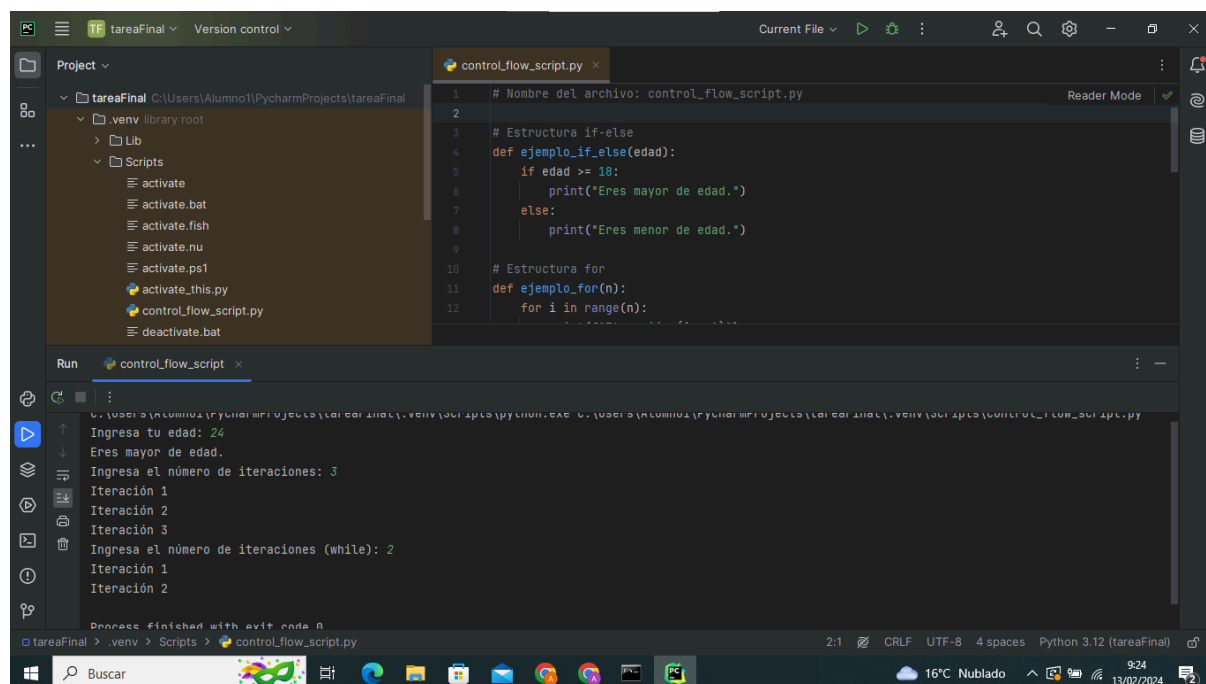


Creamos un script básico de flujo para comprobar la funcionalidad y adaptarnos al nuevo IDE.



```
1 # Nombre del archivo: control_flow_script.py
2
3 # Estructura if-else
4 def ejemplo_if_else(edad):
5     if edad >= 18:
6         print("Eres mayor de edad.")
7     else:
8         print("Eres menor de edad.")
9
10 # Estructura for
11 def ejemplo_for(n):
12     for i in range(n):
13         print(f"Iteración {i + 1}")
14
15 # Estructura while
16 def ejemplo_while(n):
17     i = 1
18     while i <= n:
19         print(f"Iteración {i}")
20         i += 1
21
22 # Ejecutar las funciones
23 if __name__ == "__main__":
24     # Prueba de la estructura if-else
25     edad_usuario = int(input("Ingresa tu edad: "))
26     ejemplo_if_else(edad_usuario)
27
```

Ejecutamos el programa y comprobamos su funcionamiento.



```
Run control_flow_script
C:\Users\Alumno1\PycharmProjects\tareaFinal\.venv\Scripts\python.exe C:\Users\Alumno1\PycharmProjects\tareaFinal\.venv\Scripts\python.exe C:\Users\Alumno1\PycharmProjects\tareaFinal\control_flow_script.py
Ingresa tu edad: 24
Eres mayor de edad.
Ingresa el número de iteraciones: 3
Iteración 1
Iteración 2
Iteración 3
Ingresa el número de iteraciones (while): 2
Iteración 1
Iteración 2
Process finished with exit code 0
```

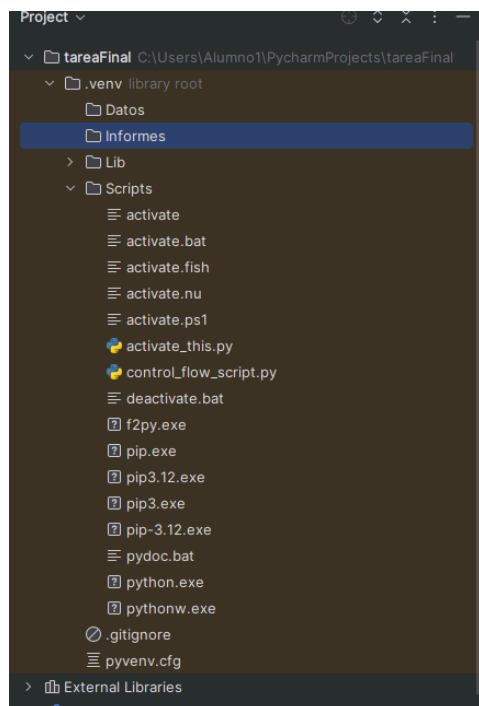


2. Fase de preparación

Primero instalamos las dependencias, para ello, abrimos una nueva terminal e instalamos la biblioteca Pandas.

```
(.env) PS C:\Users\Alumno1\PycharmProjects\tareaFinal> pip install pandas
Collecting pandas
  Obtaining dependency information for pandas from https://files.pythonhosted.org/packages/87/03/fe50521919aa981f0a1c197037da4623a207b0e5f42240d09ba048e80da3/
pandas-2.2.0-cp312-cp312-win_amd64.whl.metadata
  Downloading pandas-2.2.0-cp312-cp312-win_amd64.whl.metadata (19 kB)
Collecting numpy<2,>=1.26.0 (from pandas)
  Obtaining dependency information for numpy<2,>=1.26.0 from https://files.pythonhosted.org/packages/16/2e/86f24451c2d530c88daf997cb8d0ac622c1d40d19f5a031ed68
a4b73a374/numpy-1.26.4-cp312-cp312-win_amd64.whl.metadata
  Downloading numpy-1.26.4-cp312-cp312-win_amd64.whl.metadata (61 kB)
  Downloading numpy-1.26.4-cp312-cp312-win_amd64.whl (11.5 MB)
    61.0/61.0 kB 1.1 MB/s eta 0:00:00
Collecting python-dateutil<2.8.2 (from pandas)
  Downloading python_dateutil-2.8.2-py2.py3-none-any.whl (247 kB)
    247.7/247.7 kB 2.5 MB/s eta 0:00:00
Collecting pytz>=2020.1 (from pandas)
  Obtaining dependency information for pytz>=2020.1 from https://files.pythonhosted.org/packages/9c/3d/a121f284241f08268b21359bd425f7d4825cffe5ac5cd0e1b3d82ff
d2b10/pytz-2024.1-py2.py3-none-any.whl.metadata
  Downloading pytz-2024.1-py2.py3-none-any.whl.metadata (22 kB)
Collecting tzdata>=2022.7 (from pandas)
  Obtaining dependency information for tzdata>=2022.7 from https://files.pythonhosted.org/packages/65/58/f9c9e6be752e9fcb8b0a0ee9fb87e6e7a1f6bcab2cdc73f02bb7b
a91ada0/tzdata-2024.1-py2.py3-none-any.whl.metadata
  Downloading tzdata-2024.1-py2.py3-none-any.whl.metadata (1.4 kB)
Collecting six>=1.5 (from python-dateutil<2.8.2->pandas)
  Downloading six-1.16.0-py2.py3-none-any.whl (11 kB)
  Downloading pandas-2.2.0-cp312-cp312-win_amd64.whl (11.5 MB)
    11.5/11.5 MB 1.2 MB/s eta 0:00:00
```

Organizamos la estructura del proyecto.



3. Creación de módulos

Primero, creamos una función para ingresar las horas trabajadas y calcularlas.

```
control_flow_script.py script_horas.py script_calculos.py script_informes.py
1 # En tu script dentro de la carpeta 'scripts/'
2
3 def ingresar_horas():
4     try:
5         horas_trabajadas = float(input("Ingresa las horas trabajadas: "))
6         if horas_trabajadas < 0:
7             raise ValueError("Las horas trabajadas no pueden ser negativas.")
8         return horas_trabajadas
9     except ValueError as e:
10        print(f"Error: {e}")
11        return None
12
13 def calcular_total_horas(lista_horas):
14     return sum(lista_horas)
15
16 def calcular_promedio_horas(lista_horas):
17     if not lista_horas:
18         return 0
19     return sum(lista_horas) / len(lista_horas)
20
21 def generar_informe(horas_por_empleado):
22     for empleado, horas in horas_por_empleado.items():
23         print(f"Empleado: {empleado} - Horas Trabajadas: {horas}")
24
25 def main():
26     # Llamada a la función ingresar_horas()
27     horas_trabajadas = ingresar_horas()
```

```
28
29 def generar_informe(horas_por_empleado):
30     for empleado, horas in horas_por_empleado.items():
31         print(f"Empleado: {empleado} - Horas Trabajadas: {horas}")
32
33 def main():
34     # Llamada a la función ingresar_horas()
35     horas_trabajadas = ingresar_horas()
36
37     # Validación del resultado de ingresar_horas()
38     if horas_trabajadas is not None:
39         # Si la entrada es válida, proceder con los cálculos
40         lista_horas = [horas_trabajadas] # Puedes almacenar las horas en una lista
41         total_horas = calcular_total_horas(lista_horas)
42         promedio_horas = calcular_promedio_horas(lista_horas)
43
44         # Imprimir resultados
45         print(f"Total de Horas: {total_horas}")
46         print(f"Promedio de Horas: {promedio_horas}")
47
48 if __name__ == "__main__":
49     main()
```



Ejecución de prueba:

```
Run script_horas x
C:\Users\Alumno1\PycharmProjects\tareaFinal\.venv\Scripts\python.exe C:\Users\Alumno1\PycharmProjects\tareaFinal\.venv\Scripts\script_horas.py
Ingresa las horas trabajadas: 8
Total de Horas: 8.0
Promedio de Horas: 8.0
```

Ahora vamos a ampliar el código para que genere un informe para las horas que ha trabajado cada empleado:

```
Project view:
- activate.bat
- activate.fish
- activate.nu
- activate.ps1
- activate_this.py
- control_flow_script.py
- deactivate.bat
- f2py.exe
- pip.exe
- pip3.12.exe
- pip3.exe
- pip-3.12.exe
- pydoc.bat
- python.exe
- pythonw.exe
- script_horas.py
- gitignore
- pyvenv.cfg
- Binary Skeletons
- DLLs
- Extended Definitions
- Lib
- Python312 library root
- site-packages
- Typed stubs
- Scratches and Consoles

script_horas.py:
25 def main():
26     horas_por_empleado = {}
27
28     while True:
29         nombre_empleado = input("Ingresa el nombre del empleado (o 'exit' para salir): ")
30         if nombre_empleado.lower() == 'exit':
31             break
32
33         horas_trabajadas = ingresar_horas()
34
35         if horas_trabajadas is not None:
36             if nombre_empleado in horas_por_empleado:
37                 horas_por_empleado[nombre_empleado]['horas'].append(horas_trabajadas)
38             else:
39                 horas_por_empleado[nombre_empleado] = {'horas': [horas_trabajadas], 'total': 0, 'promedio': 0}
40
41         # Actualizar total y promedio
42         horas_por_empleado[nombre_empleado]['total'] = calcular_total_horas(horas_por_empleado[nombre_empleado])
43         horas_por_empleado[nombre_empleado]['promedio'] = calcular_promedio_horas(horas_por_empleado[nombre_empleado])
44
45         # Imprimir informe al salir
46         generar_informe(horas_por_empleado)
47
48 if __name__ == "__main__":
49     main()
50
Footer:
- calcular_promedio_horas()
- if not lista_horas
- 18:17 CRLF UTF-8 4 spaces Python 3.12 (tareaFinal)
```

Ejecutamos:

```
Run script_horas x
C:\Users\Alumno1\PycharmProjects\tareaFinal\.venv\Scripts\python.exe C:\Users\Alumno1\PycharmProjects\tareaFinal\.venv\Scripts\script_horas.py
Ingresa el nombre del empleado (o 'exit' para salir): Guille
Ingresa las horas trabajadas: 9
Ingresa el nombre del empleado (o 'exit' para salir): Marcos
Ingresa las horas trabajadas: 8
Ingresa el nombre del empleado (o 'exit' para salir): exit
Empleado: Guille - Horas Trabajadas: 9.0 - Promedio: 9.0
Empleado: Marcos - Horas Trabajadas: 8.0 - Promedio: 8.0
Process finished with exit code 0
```

Funciona correctamente.



4. Exportación de informes

Instalamos las dependencias de openpyxl y reportlab también.

```
Terminal Local x
PS C:\Users\Alumno1\PycharmProjects\tareaFinal> pip install reportlab openpyxl
Collecting reportlab
  Obtaining dependency information for reportlab from https://files.pythonhosted.org/packages/d2/70/c44e5fb6099cf28d01255ff1dfc6a4c8f2b981f314707018c802ac179e
e/reportlab-4.1.0-py3-none-any.whl.metadata
  Downloading reportlab-4.1.0-py3-none-any.whl.metadata (1.4 kB)
Collecting openpyxl
aFinal > .venv > Scripts > script_horas.py
```

Actualizamos el código, añadiendo las funciones de exportación para pdf y excel.

```
script_horas.py x
1 # En tu script dentro de la carpeta 'scripts/'
2 import pandas as pd
3 from reportlab.pdfgen import canvas
4
5 def ingresar_horas():
6     try:
7         horas_trabajadas = float(input("Ingresa las horas trabajadas: "))
8         if horas_trabajadas < 0:
9             raise ValueError("Las horas trabajadas no pueden ser negativas.")
10        return horas_trabajadas
11    except ValueError as e:
12        print(f"Error: {e}")
13        return None
14
15 def calcular_total_horas(lista_horas):
16     return sum(lista_horas)
17
18 def calcular_promedio_horas(lista_horas):
19     if not lista_horas:
20         return 0
21     return sum(lista_horas) / len(lista_horas)
22
23 def generar_informe(horas_por_empleado):
24     for empleado, horas in horas_por_empleado.items():
25         print(f"Empleado: {empleado} - Horas Trabajadas: {horas['total']} - Promedio: {horas['promedio']}")
```

```
27 def exportar_a_pdf(horas_por_empleado):
28     pdf_filename = "informe.pdf"
29     c = canvas.Canvas(pdf_filename)
30
31     for empleado, horas in horas_por_empleado.items():
32         c.drawString(x=100, y=750, text=f"Empleado: {empleado} - Horas Trabajadas: {horas['total']} - Promedio: {horas['promedio']}")
33         c.drawString(x=100, y=730, text="-" * 50)
34
35     c.save()
36     print(f"Informe exportado a {pdf_filename}")
37
38 def exportar_a_excel(horas_por_empleado):
39     excel_filename = "informe.xlsx"
40     df = pd.DataFrame(horas_por_empleado)
41     df = df.T # Transponer para tener empleados en filas y atributos en columnas
42     df.to_excel(excel_filename, index_label="Empleado")
43     print(f"Informe exportado a {excel_filename}")
```




```
script_horas.py x
45 def main():
46     horas_por_empleado = {}
47
48     while True:
49         nombre_empleado = input("Ingresa el nombre del empleado (o 'exit' para salir): ")
50         if nombre_empleado.lower() == 'exit':
51             break
52
53         horas_trabajadas = ingresar_horas()
54
55         if horas_trabajadas is not None:
56             if nombre_empleado in horas_por_empleado:
57                 horas_por_empleado[nombre_empleado]['horas'].append(horas_trabajadas)
58             else:
59                 horas_por_empleado[nombre_empleado] = {'horas': [horas_trabajadas], 'total': 0, 'promedio': 0}
60
61             # Actualizar total y promedio
62             horas_por_empleado[nombre_empleado]['total'] = calcular_total_horas(horas_por_empleado[nombre_empleado])
63             horas_por_empleado[nombre_empleado]['promedio'] = calcular_promedio_horas(horas_por_empleado[nombre_empleado])
64
65         # Imprimir informe al salir
66         generar_informe(horas_por_empleado)
67
68         # Exportar informe a PDF y Excel
69         exportar_a_pdf(horas_por_empleado)
70         exportar_a_excel(horas_por_empleado)
```

Probamos el funcionamiento:

```
Run script_horas.py
C:\Users\Alumno1\PycharmProjects\tareaFinal\.venv\Scripts\python.exe C:\Users\Alumno1\PycharmProjects\tareaFinal\.venv\Scripts\script_horas.py
Ingresa el nombre del empleado (o 'exit' para salir): Guille
Ingresa las horas trabajadas: 9
Ingresa el nombre del empleado (o 'exit' para salir): Marcos
Ingresa las horas trabajadas: 8
Ingresa el nombre del empleado (o 'exit' para salir): exit
Empleado: Guille - Horas Trabajadas: 9.0 - Promedio: 9.0
Empleado: Marcos - Horas Trabajadas: 8.0 - Promedio: 8.0
Informe exportado a informe.pdf
Informe exportado a informe.xlsx
Process finished with exit code 0
```

informe.pdf

Archivo | C:\Users\Alumno1\PycharmProjects\tareaFinal\.venv\Scripts\informe.pdf

Empleado: Guille - Horas Trabajadas: 8.0 - Promedio: 8.0

5. Integración del programa en ERP-CRM

Finalmente, tenemos que integrar nuestro programa en nuestro ERP-CRM existente. Para ello, agregamos la conexión en nuestro programa.

```
script_horas.py x
4 def enviar_datos_a_erp_crm(datos):
5     url = "https://medazzz.odoo.com/enviar_datos"
6     headers = {"Authorization": "Bearer 21873781812"}
7     response = requests.post(url, json=datos, headers=headers)
8
9     if response.status_code == 200:
10         print("Datos enviados correctamente al ERP-CRM")
11     else:
12         print(f"Error al enviar datos al ERP-CRM. Código de estado: {response.status_code}")
13
14 def obtener_informacion_erp_crm(id_empleado):
15     url = f"https://medazzz.odoo.com/obtener_informacion/{id_empleado}"
16     headers = {"Authorization": "Bearer 21873781812"}
17     response = requests.get(url, headers=headers)
18
19     if response.status_code == 200:
20         informacion = response.json()
21         print(f"Información del ERP-CRM para el empleado {id_empleado}: {informacion}")
22     else:
23         print(f"Error al obtener información del ERP-CRM. Código de estado: {response.status_code}")
```

Ponemos nuestra url del ERP-CRM correspondiente y el token de authorization obtenido en la base de datos de nuestro ERP-CRM.