Trabajo: Entrega Final

Alumna: Verónica Soledad Rocha

Objetivo:

Hacer que el modelo muestre una estadistica de la cantidad de tareas que tenemos previstas para el sprint, clasificadas por su estado

1. Instalamos las librerias

In [5]:
```
#!pip install -U google-generativeai
#!pip install openai==0.28

import google.generativeai as genai
import openai
import requests
import os
from io import BytesIO
```

2. Defino mi apikey

In [6]:
```
genai.configure(api_key="AIzaSyBeYfAwyUSfOf4BsO86rd0RXC242OboWJ8")

openai.api_key = "AIzaSyBeYfAwyUSfOf4BsO86rd0RXC242OboWJ8"
```

3. Generamos la respuesta para el cliente en base al prompt proporcionado

In [7]:
```
def generate_text_with_gemini(prompt):
    model = genai.GenerativeModel('gemini-1.5-flash')
    response = model.generate_content(f"{prompt} indicar que se revisará lo que

    #get the response´s text
    generated_text = response.candidates[0].content.parts[0].text

    return generated_text
```

4. Importamos más librerias

In [8]:
```
import sys
print ("Python que ejecuta el notebook:",sys.executable)
```

```
Python que ejecuta el notebook: C:\Users\VS\miniconda3\envs\generacion-de-promt\p
ython.exe
```

In [9]:
```
import numpy
print ("numpy:",numpy.__version__, "en", numpy.__file__)
```

```
numpy: 1.23.5 en C:\Users\VS\miniconda3\envs\generacion-de-promt\lib\site-package
s\numpy\__init__.py
```

In [10]:
```python
import pandas as pd
print ("pandas:",pd.__version__)
```

```
pandas: 1.5.0
```

In [7]:
```python
import pandas as pd
import tkinter as tk
from tkinter import filedialog, messagebox
```

5. Interactuamos con el cliente para preguntarle ¿qué quiere revisar?

In [11]:
```python
initial_prompt = input("Hola, qué es lo que quieres revisar, clasificar?")

print("Revisaremos el documento y te indicaremos los resultados. Por favor, prop
```

```
Revisaremos el documento y te indicaremos los resultados. Por favor, proporciónam
e el archivo que deseas revisar
```

6. Le pedimos que importe el archivo que desea revisar

In [12]:
```python
from ipywidgets import FileUpload
from IPython.display import display

# Creamos y mostramos el widget de subida
uploader = FileUpload(accept='.xlsx,.xls', multiple=False)
display(uploader)
```

```
FileUpload(value=(), accept='.xlsx,.xls', description='Upload')
```

7. Leemos el archivo suministrado y lo mostramos por pantalla

In [13]:
```python
import pandas as pd
from io import BytesIO
from IPython.display import display

# Comprobamos que el usuario ya subió algo
if uploader.value:
    val = uploader.value
    if isinstance(val, dict):
        Content = next(iter(val.values()))['contente']
    else:
        first = val [0]
        if isinstance(first,dict):
            content = first['content']
        else:
            content = first.content

    df = pd.read_excel(BytesIO(content))
    display(df)
else:
    print("No se ha subido ningun archivo")
```

```
C:\Users\VS\miniconda3\envs\generacion-de-promt\lib\site-packages\openpyxl\worksheet\_read_only.py:85: UserWarning: Cell L19 is marked as a date but the serial value 6689546 is outside the limits for dates. The cell will be treated as an error.
  for idx, row in parser.parse():
```

	Objetivo	Historia de usuario (HdU)	Código tarea	Descripción tarea	Sprint	Sprint goal	Prioridad	R
0	A. Modelo de prevención de fraude	A.1. Existen políticas, procedimientos y roles...	A1.T1	"Relevar y documentar todas las normas, políti...	NaN	NaN	Must	
1	NaN	A.2. Verificar mediante una muestra de casos d...	A2.T1	Identificar si este punto fue cubierto en algu...	Sprint 1	NaN	Must	
2	B. Contratación de productos de inversión en o...	B.1.Existe un proceso con definición de respon...	B1.T1	Verificar la existencia y vigencia de normativ...	NaN	NaN	Could	
3	NaN	B.2 Los empleados de sucursal que asisten a lo...	B2.T1	"Relevar la normativa vigente del regulador qu...	NaN	NaN	Could	
4	NaN	NaN	B2.T2	Relevar el circuito de asignación de la funció...	NaN	NaN	Could	
5	NaN	NaN	B2.T3	"Listar a los colaboradores sujetos al requisi...	NaN	NaN	Should	
6	NaN	B.3 Las funciones para operar en mercado, está...	B3.T1	"Detectar a los colaboradores que poseen permi...	NaN	NaN	Must	
7	C. Controles de efectivo en sucursales	C.1.Para el mes de marzo de 2025, se realizaro...	C1.T1	Elaborar un resumen ejecutivo de la normativa ...	NaN	NaN	Should	
8	NaN	NaN	C1.T2	Obtener los registros de ejecución de los sigu...	Sprint 1	Si	Must	
9	NaN	NaN	C1.T3	Definir el universo de sucursales que sí indic...	Sprint 1	Si	Must	
10	NaN	NaN	C1.T4	Seleccionar una muestra	Sprint 1	Si	Must	

	Objetivo	Historia de usuario (HdU)	Código tarea	Descripción tarea	Sprint	Sprint goal	Prioridad	R
				de sucursales y justif...				
11	NaN	NaN	C1.T5	A partir de la muestra de sucursales seleccion...	Sprint 1	Si	Must	
12	D. Alta de clientes y productos en oficinas	D.1.El proceso de alta de clientes/productos s...	D1.T1	Identificar la norma aplicable para la solicit...	NaN	NaN	Could	
13	NaN	D.2. Seleccionar una muestra de clientes/produ...	D2.T2	Identificar el universo de clientes de banca i...	Sprint 1	NaN	Must	
14	NaN	NaN	D2.T3	Mediante un tratamiento masivo de datos: evalu...	NaN	NaN	Should	
15	NaN	D.3.Las visitas a clientes empresas, a los que...	D3.T1	"Identificar el universo de clientes de Banca ...	Sprint 1	NaN	Must	
16	NaN	NaN	D3.T2	"Realizar un análisis masivo de datos para ver...	Sprint 1	Si	Must	
17	NaN	NaN	D3.T3	"Seleccionar una muestra de clientes del unive...	Sprint 1	NaN	Must	
18	NaN	NaN	D3.T4	"Identificar la normativa interna que establec...	Sprint 1	NaN	Must	
19	NaN	NaN	D3.T5	"Sobre la muestra seleccionada, evaluar el gra...	Sprint 1	NaN	Must	
20	E. Funcionamiento de los controles	E.1. Evaluación del diseño del mapa de riesgos...	E1.T1	Obtener del RCA la matriz de riesgos y control...	NaN	NaN	Must	

	Objetivo	Historia de usuario (HdU)	Código tarea	Descripción tarea	Sprint	Sprint goal	Prioridad	R
21	NaN	NaN	E1.T2	Realizar un mapeo entre el resto de HdU (cread...	NaN	NaN	Must	
22	NaN	NaN	E1.T3	Realizar una revisión preliminar del mapa de r...	NaN	NaN	Must	
23	NaN	NaN	NaN	NaN	NaN	NaN	NaN	

24 rows × 21 columns

8. Realizamos el recuento de tareas y clasificación solicitada

In [14]:
```
#1 volvemos a leer el archivo
df = pd.read_excel(BytesIO(content))

# 2. Filtra solo las tareas de Sprint 1
sprint1 = df[df["Sprint"] == 'Sprint 1']

# 3. Total de tareas definidas en Sprint 1
total_tareas = sprint1.shape[0]
print(f"Total de tareas definidas en Sprint 1: {total_tareas}\n")

# 4. Clasifica esas tareas por estado (columna "Status")
conteo_por_estado = (
    sprint1["Status"]
    .value_counts()
    .rename_axis("Status")
    .reset_index(name="Cantidad")
)
```

```
Total de tareas definidas en Sprint 1: 11
```

```
C:\Users\VS\miniconda3\envs\generacion-de-promt\lib\site-packages\openpyxl\worksheet\_read_only.py:85: UserWarning: Cell L19 is marked as a date but the serial value 6689546 is outside the limits for dates. The cell will be treated as an error.
  for idx, row in parser.parse():
```

9. Realizamos el conteo y cálculo del porcentaje de representatividad de cada estado y lo exponemos

In [15]:
```
# 1) Recalculamos el conteo por estado
vc = sprint1 ['Status'].value_counts()
pct = sprint1 ['Status'].value_counts(normalize=True)

# 2) Montamos el DataFrame con cantidad y porcentaje
conteo_por_estado = (
    pd.DataFrame({
        'Status': vc.index,
        'Cantidad': vc.values,
```

```
        'Porcentaje': (pct.values * 100).round(2)
    })
)

# 3) Opcional: añadir el símbolo "%"
conteo_por_estado['Porcentaje'] = conteo_por_estado['Porcentaje'].astype(str) +

# 4) Mostramos el resultado
print(f"Total de tareas en Sprint 1: {total_tareas}\n")
display(conteo_por_estado)
```

```
Total de tareas en Sprint 1: 11
```

	Status	Cantidad	Porcentaje
0	To review	4	36.36%
1	Bloqueada	3	27.27%
2	Doing	3	27.27%
3	Con comentarios Vero	1	9.09%

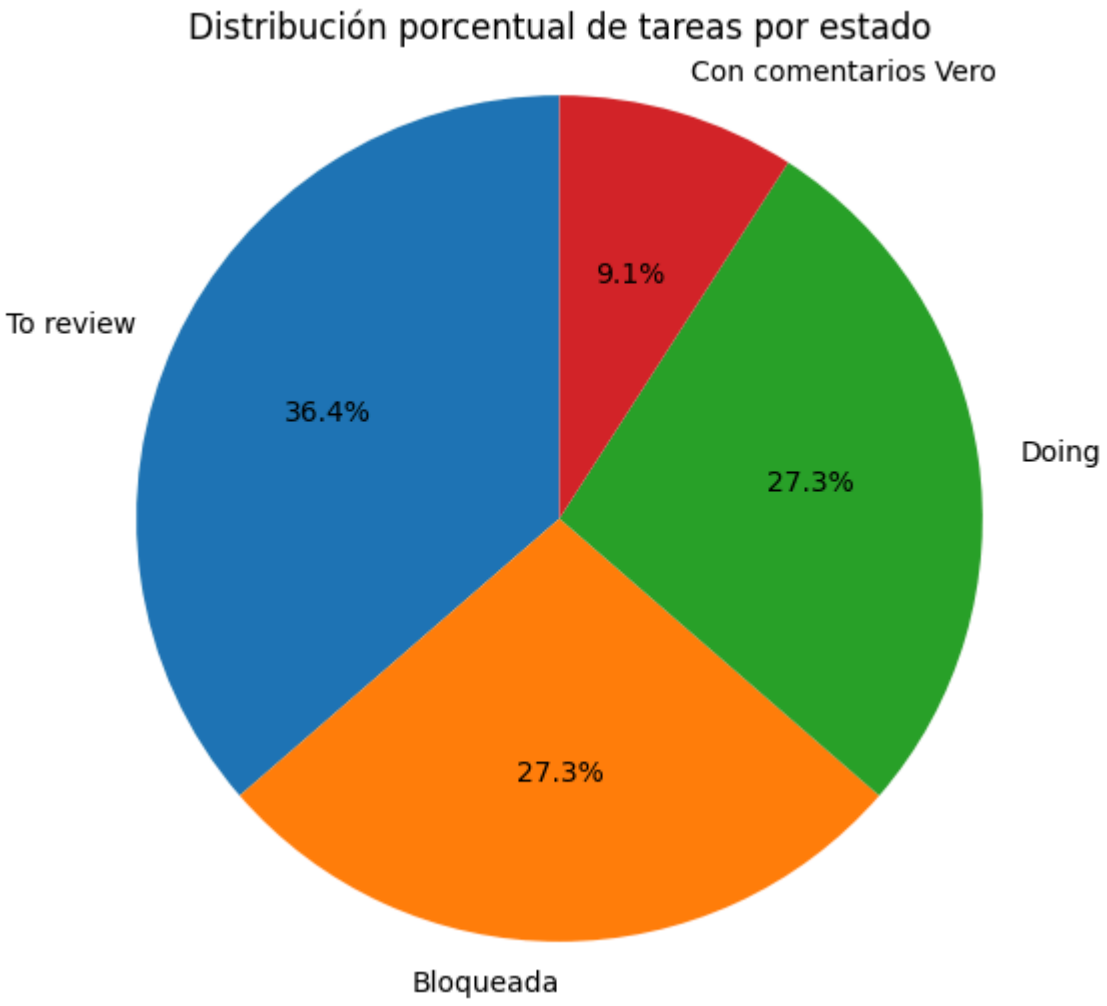
11. Importamos matplotlib

In [16]:
```
import matplotlib.pyplot as plt
```

In [17]:
```
# 1) Definir los datos
# ——————————————————————
conteo_por_estado = pd.DataFrame({
    'Status': ['To review', 'Bloqueada', 'Doing', 'Con comentarios Vero'],
    'Porcentaje': ['36.36%', '27.27%', '27.27%', '9.09%']
    })
```

In [18]:
```
# 2) Convertir la columna de porcentaje a valores numéricos
# ——————————————————————
conteo_por_estado['Porcentaje_num'] = (
    conteo_por_estado['Porcentaje']
    .str.rstrip('%')      # quita el '%'
    .astype(float)        # convierte a float
)
```

In [19]:
```
# 3) Dibujar el gráfico de torta
# ——————————————————————
plt.figure(figsize=(6,6))  # tamaño de la figura
plt.pie(
    conteo_por_estado['Porcentaje_num'],
    labels=conteo_por_estado['Status'],
    autopct='%1.1f%%',     # formato de las etiquetas internas
    startangle=90          # rota el inicio para mejor visualización
)
plt.title('Distribución porcentual de tareas por estado')
plt.axis('equal')          # para asegurar proporciones circulares
plt.show()
```



10. Imprimo la notebook en pdf

In [1]:
```
pip install pyppeteer nest_asyncio
```

```
Requirement already satisfied: pyppeteer in c:\users\vs\miniconda3\envs\generacio
n-de-promt\lib\site-packages (2.0.0)
Requirement already satisfied: nest_asyncio in c:\users\vs\miniconda3\envs\genera
cion-de-promt\lib\site-packages (1.6.0)
Requirement already satisfied: appdirs<2.0.0,>=1.4.3 in c:\users\vs\miniconda3\en
vs\generacion-de-promt\lib\site-packages (from pyppeteer) (1.4.4)
Requirement already satisfied: certifi>=2023 in c:\users\vs\miniconda3\envs\gener
acion-de-promt\lib\site-packages (from pyppeteer) (2025.4.26)
Requirement already satisfied: importlib-metadata>=1.4 in c:\users\vs\miniconda3
\envs\generacion-de-promt\lib\site-packages (from pyppeteer) (8.7.0)
Requirement already satisfied: pyee<12.0.0,>=11.0.0 in c:\users\vs\miniconda3\env
s\generacion-de-promt\lib\site-packages (from pyppeteer) (11.1.1)
Requirement already satisfied: tqdm<5.0.0,>=4.42.1 in c:\users\vs\miniconda3\envs
\generacion-de-promt\lib\site-packages (from pyppeteer) (4.67.1)
Requirement already satisfied: urllib3<2.0.0,>=1.25.8 in c:\users\vs\miniconda3\e
nvs\generacion-de-promt\lib\site-packages (from pyppeteer) (1.26.20)
Requirement already satisfied: websockets<11.0,>=10.0 in c:\users\vs\miniconda3\e
nvs\generacion-de-promt\lib\site-packages (from pyppeteer) (10.4)
Requirement already satisfied: typing-extensions in c:\users\vs\miniconda3\envs\g
eneracion-de-promt\lib\site-packages (from pyee<12.0.0,>=11.0.0->pyppeteer) (4.1
3.2)
Requirement already satisfied: colorama in c:\users\vs\miniconda3\envs\generacion
-de-promt\lib\site-packages (from tqdm<5.0.0,>=4.42.1->pyppeteer) (0.4.6)
Requirement already satisfied: zipp>=3.20 in c:\users\vs\miniconda3\envs\generaci
on-de-promt\lib\site-packages (from importlib-metadata>=1.4->pyppeteer) (3.21.0)
Note: you may need to restart the kernel to use updated packages.
```

In [3]:

```python
import os
import nbformat
import nest_asyncio
import asyncio
from pyppeteer import launch

# Aplicamos el parche para permitir bucles anidados en Jupyter
nest_asyncio.apply()

# Nombre de notebook y destino
NOTEBOOK   = "Entrega Final.ipynb"
HTML_FILE  = "Entrega Final.html"
DOWNLOADS  = os.path.join(os.path.expanduser("~"), "Downloads")
OUTPUT_PDF = os.path.join(DOWNLOADS, "Entrega1.pdf")

# 1) Convertimos .ipynb → .html
os.makedirs(DOWNLOADS, exist_ok=True)
os.system(f'jupyter nbconvert --to html "{NOTEBOOK}" --output "{HTML_FILE}"')

# 2) Función asíncrona para abrir el HTML en Edge/Chromium y volcarlo a PDF
async def html_to_pdf(input_html, output_pdf):
    browser = await launch(
        executablePath=r"C:\Program Files (x86)\Microsoft\Edge\Application\msedg
        headless=True,
        args=['--no-sandbox']
    )
    page = await browser.newPage()
    await page.goto(f'file:///{os.path.abspath(input_html)}', waitUntil='network
    await page.pdf({
        'path': output_pdf,
        'format': 'A4',
        'printBackground': True,
        'margin': {'top':'0.5in','bottom':'0.5in','left':'0.5in','right':'0.5in'
```

```
    })
    await browser.close()

# 3) Ejecutamos la tarea asíncrona en el mismo bucle de Jupyter
asyncio.get_event_loop().run_until_complete(
    html_to_pdf(HTML_FILE, OUTPUT_PDF)
)

print(f"✅ PDF creado correctamente en:\n   {OUTPUT_PDF}")
```

```
✅ PDF creado correctamente en:
   C:\Users\VS\Downloads\Entrega1.pdf
```

In [ ]:

In [ ]: