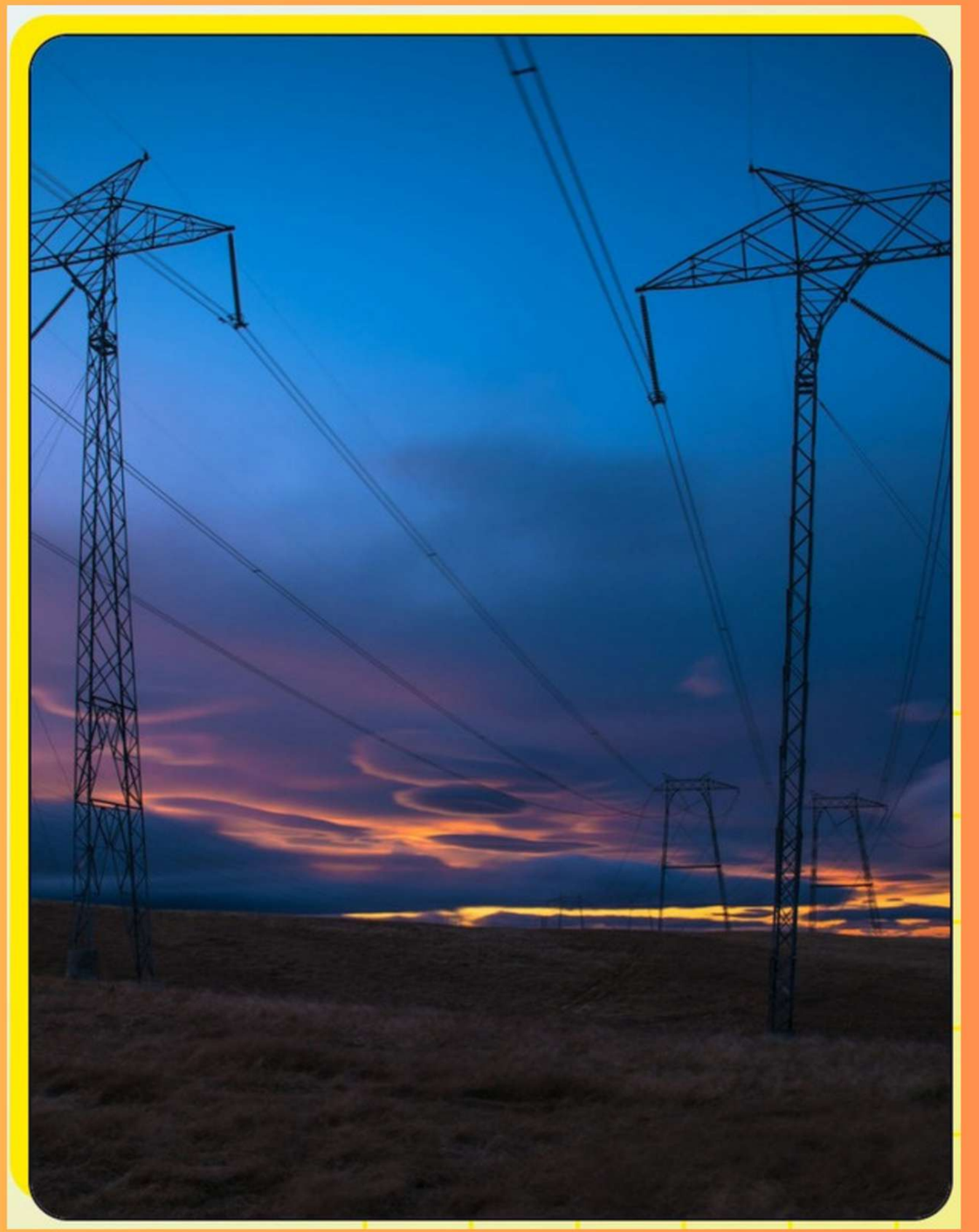
Haz click

Y que se haga la luz

Ana Pineda
Juan Durán
Silvia Alonso

Recomendador de Tarifas de Luz

Un sistema innovador para recomendar la tarifa más conveniente y sugerir soluciones de energía solar.



Motivación del proyecto

La elección del tema se debió a varios aspectos principales:

- Utilidad del proyecto
- Practicidad
- Reto para aplicar lo aprendido

Estructura del proyecto ››››

1 2 3 4 5

Obtención de datos

Obtención de tarifas con webscrapping a través de las páginas de las distintas compañías.

Limpieza de datos

Limpieza de los datos obtenidos y formateo de los mismos.

Creación de formulario

Desarrollo de un sistema que sugiere tarifas de luz basado en entradas del usuario.

Placas solares

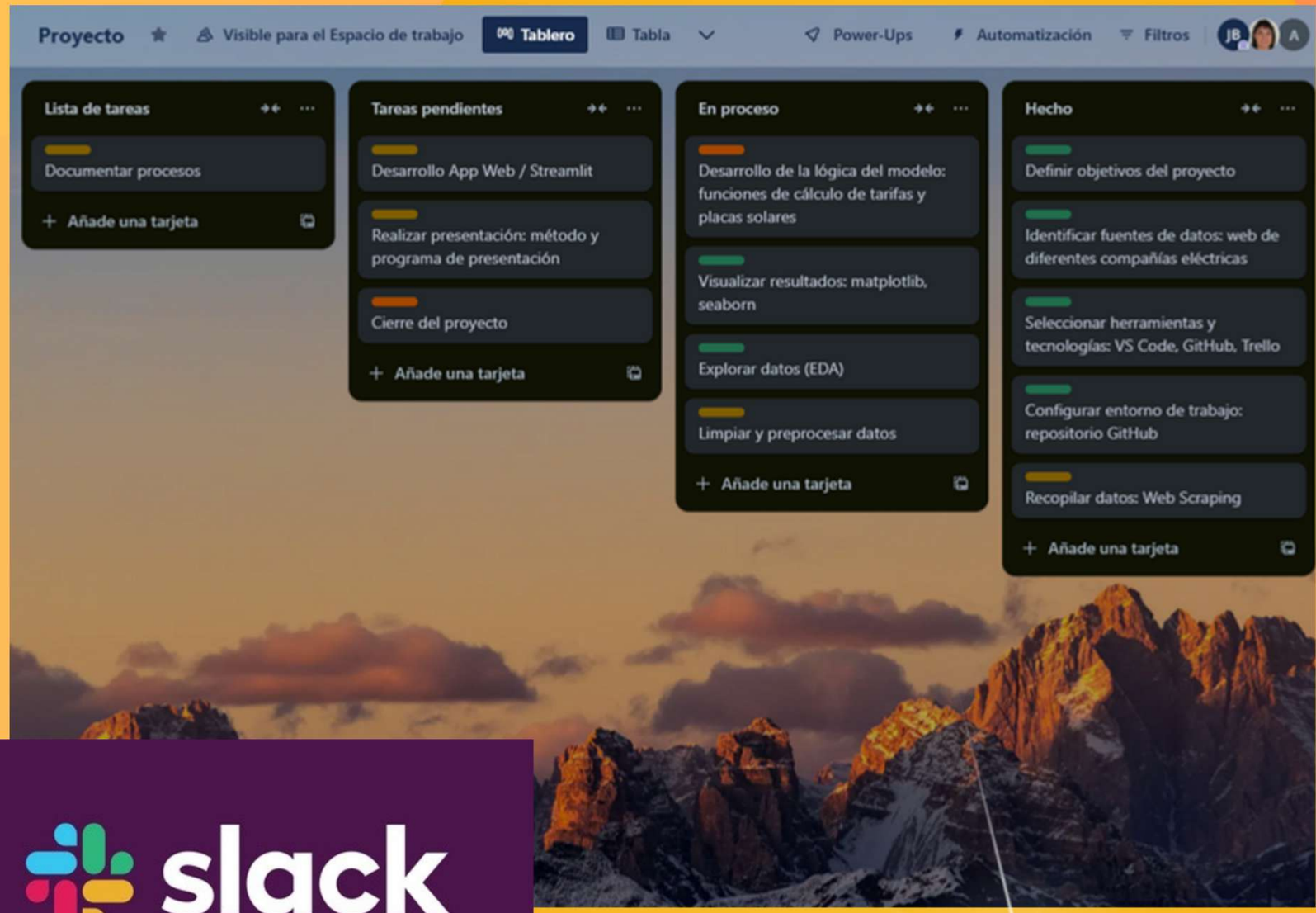
Calcula si la instalación de placas solares sería economicamente beneficioso.

Aplicación web

Queremos que todos puedan utilizar la aplicación, aunque no tengan de sus conocimientos de Python.

Organización

- Uso de **Trello** para la organización y seguimiento de tareas.
- Uso de **Slack** para mantener la comunicación del equipo.



slack

1.- Obtención de datos

Uso de web scraping

Realizamos la extracción de datos directamente desde los sitios web de varias compañías energéticas en España.

Creación de dataframes

Obteniendo datasets útiles para el almacenamiento y posterior uso de los datos requeridos en el programa.

```
energy.es/precios"

Solicitud HTTP
equests.get(url)
autifulSoup(response.content, 'html.parser')

ntrar las tarjetas de tarifas
etas = soup.find_all('div', {'data-testid': 'tariff-card'}

xtraer los datos de cada tarjeta
s = []
tarjeta in tarjetas:
titulo = tarjeta.find('h3').text.strip() if tarjeta.find('
descripcion = tarjeta.find('p', class_='sc-192m2ep-0 jZCOH

# Extraer los precios de energía consumida
energia_punta = "No encontrado"
energia_llano = "No encontrado"
energia_valle = "No encontrado"
energia_labels = tarjeta.find_all('p', class_='sc-192m2ep-
energia_values = tarjeta.find_all('h5', class_='sc-192m2ep

for label, value in zip(energia_labels, energia_values):
    if "Punta" in label.text and "€/kWh" in value.text:
```

2.- Limpieza de datos

-Algunas compañías tienen varias tarifas y tuvimos que unificarlas.

-Conseguimos un dataframe por compañía.

-Limpiar y unificar formatos.

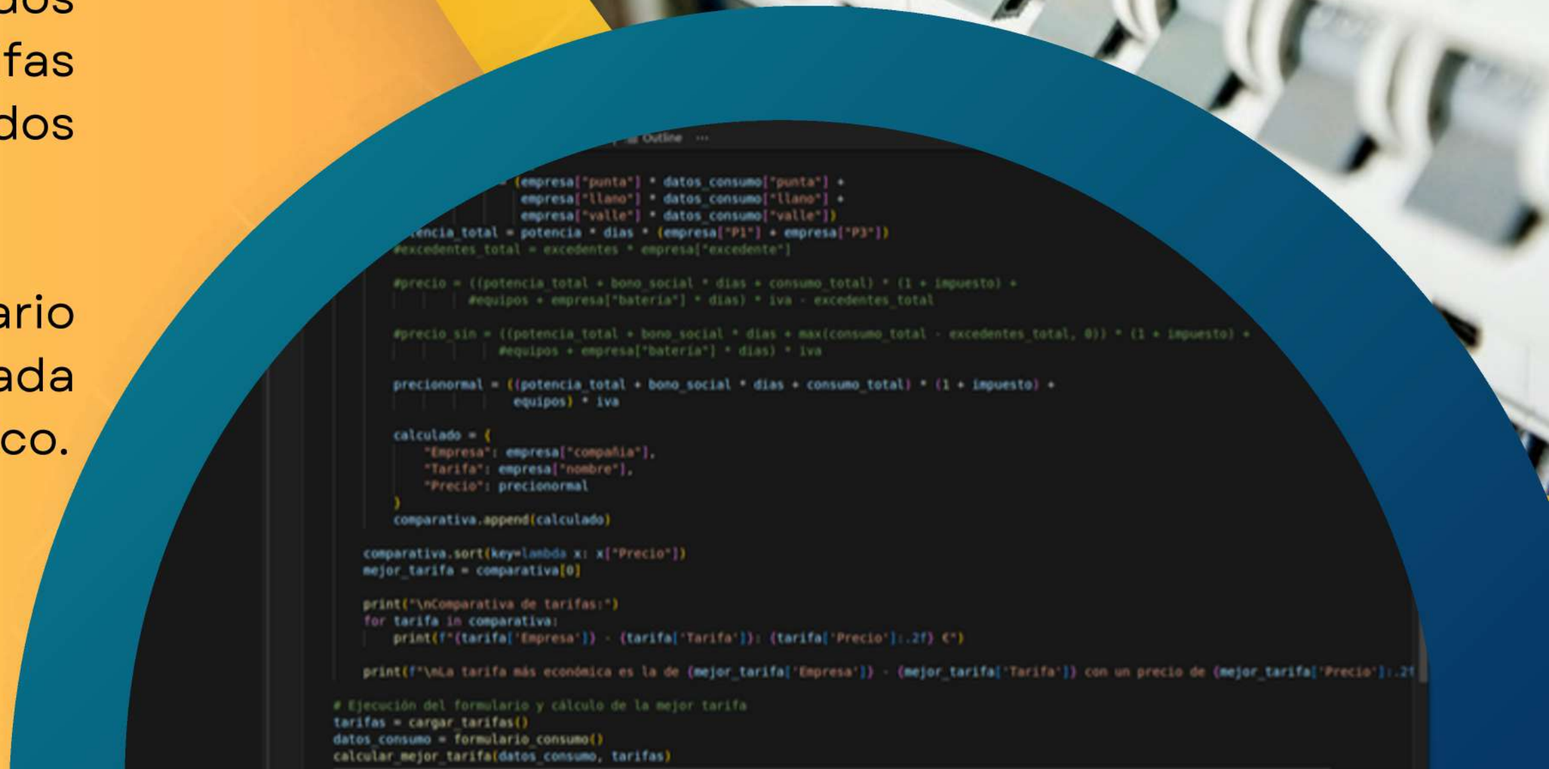
-Corregir datos faltantes.

-Sacar datos horarios de .txt y pasar a dataframe para poder utilizarlos.

```
                                                    ([\d,]+)', df_naturgy.loc[0,
                                                    a las columnas 'Punta', 'Llano' y 'Valle' en la
                                                    'Punta'] = valor_inicial if valor_inicial else None
                                                    [0, 'Llano'] = valor_inicial if valor_inicial else None
                                                    at[0, 'Valle'] = valor_inicial if valor_inicial else None

                                                    gimos la fila 1
                                                    1 = df_naturgy.loc[2, 'Punta']
                                                    or_punta = re.search(r'Punta: ([\d,]+€/kWh)', fila_1)
                                                    lor_llano = re.search(r'Llano: ([\d,]+€/kWh)', fila_1)
                                                    alor_valle = re.search(r'Valle: ([\d,]+€/kWh)', fila_1)
                                                    df_naturgy.at[2, 'Punta'] = valor_punta.group(1) if valor_punta else None
                                                    df_naturgy.at[2, 'Llano'] = valor_llano.group(1) if valor_llano else None
                                                    df_naturgy.at[2, 'Valle'] = valor_valle.group(1) if valor_valle else None

                                                    #Unimos todos los datos en una misma dataframe

                                                    df_companias = pd.concat([df_naturgy,df_natsolar,df_octopus,df_repsol,df_x

                                                    # Añadimos el coste de la bateria virtual de Repsol

                                                    df_companias['Bateria'] = df_companias['Excedentes']
                                                    df_companias.loc[2, 'Bateria'] = 0.0
                                                    df_companias.loc[6, 'Bateria'] = 0.0
                                                    df_companias.loc[10, 'Bateria'] = 1.80

                                                    # Cambiamos los NaN por 0

                                                    f_companias = df_companias.fillna(0)

                                                    ordenamos las columnas
                                                    = ['Empresa', 'Tarifa', 'Descripción', 'Punta', 'Llano',
                                                    'Valle', 'P1', 'P3', 'Excedentes', 'Coste de gestión', 'Bateria']
                                                    las=df_companias[orden]

                                                    unidades(celda):
```

3.- Formulario de recomendación de tarifas

- Diseñamos un formulario interactivo para captar los datos del usuario y ofrecerle una recomendación sobre la tarifa energética más adecuada.
- Los datos ingresados son comparados con nuestro conjunto de datos de tarifas y los coeficientes de consumo ajustados según el perfil de hogar.
- Al completar el formulario, el usuario recibe una recomendación optimizada de tarifas, basada en su perfil energético.



4.- Estudio de placas solares

Con los datos del usuario comprobamos si el uso de placas solares supondría un ahorro considerable en su caso.

Son varios los factores tenidos en cuenta para esta recomendación:

- Ciudad
- Horas de sol por mes
- Irradiación solar
- Cálculo de la distribución del consumo

```
'Nov': 30,
'Dec': 31}

# Crear el diccionario para almacenar la
meses = df_ciudad.columns.difference(['D
horas_sol = {mes: df_ciudad[mes].sum()*f
porcen_sol = {mes: round(df_ciudad[mes].
# Aproximadamente entre el 40% y el 50%
# Calcular el total de todos los meses

# Para un mes dado
mes = 'Jul'
c_punta = 188.39
c_llano = 233.13
c_valle = 355.02

#mes = 'Feb'
#c_punta = 188.39
#c_llano = 233.13
#c_valle = 355.02

total = c_punta + c_llano + c_valle

consumo anual de energía por parte
0 %  de la energía anual generada

l >= e_generada *0.8

= int(c_total / (pot_placa * 0.

ada = n_placas * pot_placa * 0.7
ada #912.0481066666669

= c_total * 0.4
```

5.- Aplicación web

-Interfaz de usuario amigable

-Posibilidad de introducir datos de consumo

-Interfaz basada en web usando Streamlit

-Cualquier persona con conexión a internet puede usarla.

Entra y prueba en:

https://electric-and-solar-tariff-recommender-project-aklabfg6vkkxekht.streamlit.app/

Menú de Navegación

- Introducción
- Objetivos
- Metodología
- Visualizaciones
- Recomendador

Formulario de Consumo

¿Para qué mes deseas hacer la comparativa?

enero

¿En qué provincia te encuentras?

¿Cuántas personas viven en tu domicilio?

1

¿Conoces tu consumo en kWh para los periodos de Valle, Llano y Punta?

- Sí
- No

Vamos a estimar tu consumo. Por favor, indica si tienes los siguiente

¿Tienes Frigorífico?

- Sí
- No

¿Tienes Lavadora?

- Sí
- No

¿Tienes Lavavajillas?

- Sí
- No

¿Tienes Televisor?

- Sí
- No

¿Tienes Aire Acondicionado?

Dificultades enfrentadas

Variabilidad de las páginas web para el webscraping

- La diversidad en la estructura de las páginas web complicó la recolección de datos de manera uniforme.

Gran cantidad de datos

- Trabajamos con muchos archivos.

Lógica del programa

- La lógica del programa se complicaba cada vez más.

Nuevas herramientas y librerías

- Para lograr los resultados deseados, tuvimos que investigar y utilizar nuevas librerías y herramientas.

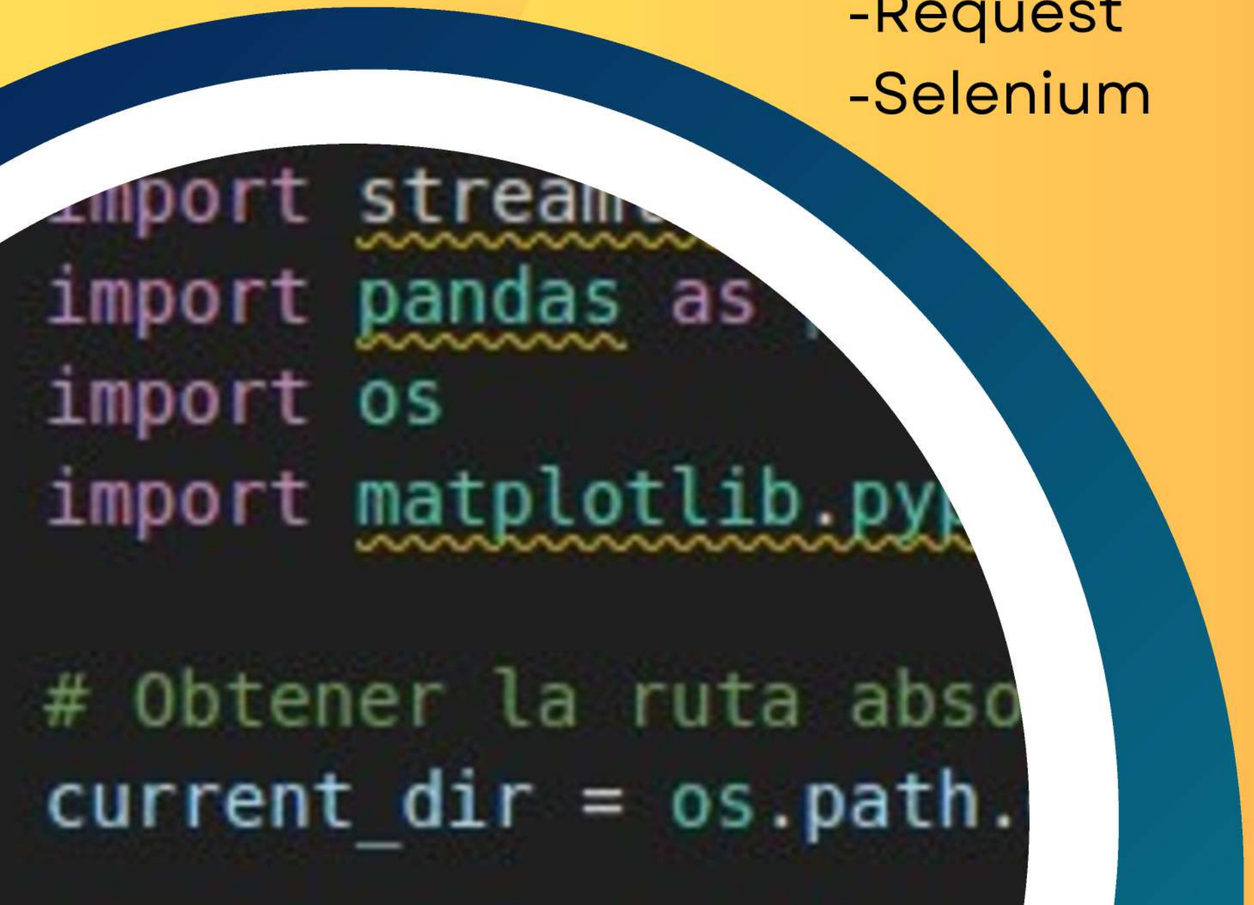
Librerías utilizadas

Para **webscrapping**:
-BeautifulSoup
-Request
-Selenium

Para la **limpieza**:
-pandas

Para la **aplicación**:
-streamlit

Para los **gráficos**:
-seaborn
-matplotlib



Gracias por vuestra atención

https://electric-and-solar-tariff-recommender-project-aklabfg6vkkxekht.streamlit.app/