**Requirements (要件)**

* Python version 🡪 3.11
* Pip version 🡪23.3.1
* Tkinter 🡪 pip install tkinter
* Ttkbootstrap 🡪 pip install ttkbootstrap
* Pandas 🡪 pip install pandas
* Requests 🡪 pip install requests
* Openpyxl 🡪pip install openpyxl
* Xlswriter 🡪pip install xlswriter
* Datetime 🡪 pip install DateTime
* Calendar 🡪pip install calendar

**Code descriptions(コードの説明)**

1. **Import modules**

from tkinter import \*

import ttkbootstrap as tb

import os

import pandas as pd

import requests

from requests.auth import HTTPDigestAuth

import sys

from tkinter import messagebox

from openpyxl import load\_workbook

from datetime import datetime, timedelta,date

import calendar

import openpyxl

from openpyxl.chart import BarChart, LineChart, Reference

from openpyxl.styles import Border, Side

1. **GUI**

root = tb.Window(themename="superhero")

root.title("ログインフォーム")

root.geometry('400x400')

style = tb.Style()

style.configure('TEntry', font=('Helvetica', 14))

style.configure('TButton', font=('Helvetica', 14))

style.configure('TLabel', font=('Helvetica', 14))

# Create a frame to enclose the login elements

login\_container = tb.Frame(root, padding=20, borderwidth=4, relief="solid", width=300, height=250)

login\_container.place(relx=0.5, rely=0.5, anchor=CENTER)

username\_label = tb.Label(login\_container, text="ユーザー名")

username\_label.grid(row=0, column=0, padx=10, pady=10, columnspan=2)

username\_entry = tb.Entry(login\_container, width=20)

username\_entry.grid(row=1, column=0, padx=10, pady=10, columnspan=2)

password\_label = tb.Label(login\_container, text="パスワード")

password\_label.grid(row=2, column=0, padx=10, pady=10, columnspan=2)

password\_entry = tb.Entry(login\_container, show="\*", width=20)

password\_entry.grid(row=3, column=0, padx=10, pady=10, columnspan=2)

login\_button = tb.Button(login\_container, text="ログイン", command=login)

login\_button.grid(row=4, column=0, padx=10, pady=10, columnspan=2)

root.mainloop()

1. **Log In Box**

def login():

#default username and password

default\_username = "sems2024"

default\_password = "Sm@rtsol@r"

username = username\_entry.get()

password = password\_entry.get()

if username == default\_username and password == default\_password:

root.withdraw()

open\_date\_picker()

else:

messagebox.showerror("エラー", "無効なユーザー名/パスワード")

**Change default\_username and password as you wish!**

必要に応じてdefault\_usernameとpasswordを変更します

**Here messagebox. showerror show error box if username/password are wrongs**

ここで messagebox.showerror ユーザー名/パスワードが間違っている場合にエラー ボックスを表示します

1. **Date picker calendar**

def open\_date\_picker():

top = tb.Toplevel(root)

top.title("日付を選ぶ")

top.geometry('400x400')

# Create a frame to enclose the login elements

container = tb.Frame(top, padding=20, borderwidth=4, relief="solid", width=300, height=250)

container.place(relx=0.5, rely=0.5, anchor=CENTER)

global mydate

mydate = tb.DateEntry(container, bootstyle="danger", width=20, dateformat = r"%Y-%m-%d") # Increase box size

mydate.pack(pady=50)

download\_button = tb.Button(container, text="ダウンロード", command=download)

download\_button.pack()

top.protocol("WM\_DELETE\_WINDOW", sys.exit)

🡪”*download\_button=tb.Button(container,text="ダウンロード",*

*command=download)*” This will call download function

🡪これによりダウンロード関数が呼び出されます

def download():

# selected\_year = year\_var.get()

# selected\_month = month\_var.get()

# print("{}:{}".format(selected\_year,selected\_month))

# locale.setlocale(locale.LC\_TIME, '')

# # Get the date format

# date\_format = locale.nl\_langinfo(locale.D\_FMT)

# print('Date format: ', date\_format)

selected\_date = mydate.entry.get()

selected\_date = selected\_date

start\_date = datetime.strptime(selected\_date, "%Y-%m-%d")

# start\_date = datetime.strptime(selected\_date, "%m/%d/%Y")

formatted\_date = start\_date.strftime("%Y-%m-%d")

print(formatted\_date)

print("Download initiated")

year,month,date= formatted\_date.split('-')

time = year + month

fileyear = time[-4:]

def get\_downloads\_folder():

current\_working\_directory = os.getcwd()

print(current\_working\_directory)

# home\_dir = os.path.expanduser("~")

# downloads\_folder = os.path.join(home\_dir, "Downloads")

# os.makedirs(downloads\_folder, exist\_ok=True)

return current\_working\_directory

def get\_workbook\_folder():

current\_working\_directory = os.getcwd()

# home\_dir = os.path.expanduser("~")

# workbooks\_folder = os.path.join(home\_dir, "Documents\\excel")

path = os.path.join( current\_working\_directory,'保守点検報告書\_ベース.xlsx')

isFile = os.path.isfile(path)

if isFile == False:

print("there is no such files")

messagebox.showerror("エラー", "エクセルファイルが見つかりません。{}パスのベースエクセルシートを確認してください。".format(current\_working\_directory))

else:

print("file found on path {}".format(path))

# os.makedirs(workbooks\_folder, exist\_ok=True)

return current\_working\_directory

downloads\_folder = get\_downloads\_folder()

workbooks\_folder = get\_workbook\_folder()

counter1 = 0

# counter2 = 0

# users = [

# {"device\_id": "CDA3614231168779955966051050305660734614600","filename":"KIC日高\_電力量実績\_"},

# {"device\_id": "CDA1234823544587112915614506550158738364790","filename":"KIC厚木\_電力量実績\_"},

# {"device\_id": "CDA3975154446986497297068957620540553993780","filename":"KIC越谷\_電力量実績\_"},

# {"device\_id": "CDA4995386155929238379015114546939378371328","filename":"岩根中学校\_電力量実績\_"},

# {"device\_id": "CDA1836290764883644058616167299722898321065","filename":"太田中学校\_電力量実績\_"},

# {"device\_id": "CDA8626825421151183593093310386921157478810","filename":"清見台小学校\_電力量実績\_"},

# {"device\_id": "CDA1229397212628199228294670587600616668420","filename":"富来田中学校\_電力量実績\_"},

# {"device\_id": "CDA1066501134050441480013516012951809329362","filename":"畑沢小学校\_電力量実績\_"},

# {"device\_id": "CDA1524381332789749178434543690444562842710","filename":"第二中学校\_電力量実績\_"},

# {"device\_id": "CDA1433063410857518239315949293158629193154","filename":"請西小学校\_電力量実績\_"}

# ]

params = {

"groupid": "1",

"time": time,

"data": "measuringdata",

"format": "csv",

"type": "pcs"

}

users1= [

{"username": "XGM0182319", "password": "MUcjx4kMvCxeMo9","filename":"広野ソーラーパーク","address":"福島県双葉郡広野町折木字東下4-18、外7筆","cover\_name":"合同会社SS福島広野","report\_name":"保守点検報告書","FIT":'=40',"PV":'=0.29\*9372',"opening\_date":"12-12-2019"},

{"username": "CBU0173382", "password": "aiqh9gMgrfLvUgz","filename":"田方郡函南町太陽光発電所","address":"静岡県田方郡函南町軽井沢字浜井場295-8　他10筆","cover\_name":"リニューアブル・ジャパン株式会社","report\_name":"運転監視報告書","FIT":'=40',"PV":'=312\*0.275+7392\*0.245',"opening\_date":"9-26-2018"},

# {"username": "FOO0197257", "password": "rdR9YWWNeumoiEL","filename":"千歳新川低圧\_運転監視報告書\_"},

# {"username": "MUI020F591", "password": "zr9qjgpwdtW4tRe","filename":"大郷町小学校\_電力量実績\_"},

# {"username": "XVK020F592", "password": "kWiEAKcpWc9kfoJ","filename":"大郷町中学校\_電力量実績\_"},

# {"username": "SXR020F590", "password": "zzxf4ge3LmKKRuz","filename":"大郷町文化会館\_電力量実績\_"},

# {"username": "IQH020C788", "password": "zmeEuMcL4zuYjt4","filename":"大郷町役場\_電力量実績\_"},

{"username": "UAP0170823", "password": "AhKC9WrRETcfnTt","filename":"岩手一関ソーラーパーク","address":"岩手県一関市花泉町花泉字大又南沢３－５他地内","cover\_name":"合同会社SS岩手一関","report\_name":"保守点検報告書","FIT":'=36',"PV":'=0.275\*8904',"opening\_date":"12-25-2020"},

{"username": "LLH0162908", "password": "3XC7NhwTsxiRydz","filename":"御田神辺池ソーラー発電所","address":"香川県さぬき市寒川町石田東字御田神辺甲1539","cover\_name":"合同会社香川水上ソーラー第二","report\_name":"運転監視報告書","FIT":'=32',"PV":'=0.295\*5175',"opening\_date":"9-8-2017"}

]

for user in users1:

username = user["username"]

password = user["password"]

filename = user["filename"]

address=user["address"]

address = str(address)

cover\_name=user["cover\_name"]

cover\_name=str(cover\_name)

FIT = user['FIT']

PV = user['PV']

opening\_date = user['opening\_date']

opening\_date = datetime.strptime(opening\_date, "%m-%d-%Y")

# date\_object = datetime.strptime(selected\_date, "%m/%d/%Y")

OPD = opening\_date.strftime("%Y-%m-%d")

oyear,omonth,odate= OPD.split('-')

date\_object = datetime.strptime('{}-{}-{}'.format(omonth, odate, oyear),'%m-%d-%Y')

formatdate = date\_object.strftime('%Y年%m月%d日')

laplace(username, password, filename, time, month, fileyear, downloads\_folder, workbooks\_folder,params,year,address,cover\_name,FIT,PV,formatdate)

counter1 += 1

if counter1 == 4:

messagebox.showinfo("ダウンロードステータス", "ダウンロード完了")

Internal part explanation (内部部品説明)

**get\_downloads\_folder():**

**-->**defining the folder where we need to keep downloaded file

**-->** In this part I am using same folder where .exe file is present

-->ダウンロードしたファイルを保存するフォルダーを定義します

--> この部分では、.exe ファイルが存在するのと同じフォルダーを使用しています。

**get\_workbook\_folder():**

-->Define the folder where base formatted excel file is present

--> 基本形式の Excel ファイルが存在するフォルダーを定義します。

counter1 🡪 total numbers of file to be downloaded.

if counter1 == 4:

messagebox.showinfo("ダウンロードステータス", "ダウンロード完了")

params = {

"groupid": "1",

"time": time,

"data": "measuringdata",

"format": "csv",

"type": "pcs"

}

🡪parameters need to pass for API

**users1=[{…},{…},{…}]**

🡪input data structure that are used later.

🡪If you want to add more user that use same api, please append the list inside this.

**for user in users1:**

…

…

…

….

-🡪 inside this section here we call

-🡪 このセクション内では、ここで呼び出します

laplace(username, password, filename, time, month, fileyear, downloads\_folder, workbooks\_folder,params,year,address,cover\_name,FIT,PV,formatdate)

-🡪 this is the main part of code, creating excel sheets and data manipulation are done inside this section

-🡪これはコードの主要部分であり、Excel シートの作成とデータ操作はこのセクション内で行われます。

**laplace(username, password, filename, time, month, fileyear, downloads\_folder, workbooks\_folder,params,year,address,cover\_name,FIT,PV,formatdate):**

**--->** write\_date\_on\_column(current\_date, end\_date, current\_cell,initial\_row,final\_row):

🡪This is used for writing date at first column of all sheets.

🡪これは、すべてのシートの最初の列に日付を書き込むために使用されます。

if username == "UAP0170823":

df\_destination[destination\_column\_name1] = df[source\_column\_name1]

df\_destination[destination\_column\_name2] = df[source\_column\_name3]

# Save the modified dataframe back to the Excel file

with pd.ExcelWriter(output\_path\_excel, engine='xlsxwriter') as writer:

df.to\_excel(writer, sheet\_name='Sheet1', index=False)

df\_destination.to\_excel(writer, sheet\_name=destination\_sheet\_name, index=False)

else:

df\_destination[destination\_column\_name1] = df[source\_column\_name1]

df\_destination[destination\_column\_name2] = df[source\_column\_name2]

# Save the modified dataframe back to the Excel file

with pd.ExcelWriter(output\_path\_excel, engine='xlsxwriter') as writer:

df.to\_excel(writer, sheet\_name='Sheet1', index=False)

df\_destination.to\_excel(writer, sheet\_name=destination\_sheet\_name, index=False)

* **This section is for copy Kwh/m2 and Kwh data to another sheet. Username “UAP0170823” use kwh/m2 data from '日射量2(kWh/m2)' so need to use if..else{} condtion**
* このセクションは、Kwh/m2 と Kwh データを別のシートにコピーするためのセクションです。 ユーザー名「UAP0170823」は「日射量2(kWh/m2)」のkwh/m2データを使用するため、if..else{}条件を使用する必要があります

for sheet\_name in sheet\_names:

# Check if the sheet exists in the workbook

if sheet\_name in workbook.sheetnames:

sheet = workbook[sheet\_name]

if sheet\_name == '表紙 ':

…

…

…

* This section checks every sheet name of excel file and do perform as required as to do inside those sheets.
* このセクションでは、Excel ファイルのすべてのシート名をチェックし、それらのシート内で行うのと同様に必要に応じて実行します。

final\_row = 49

day\_difference = 31 - numbers\_of\_rows

initial\_row = final\_row - day\_difference

cfinalvalue = initial\_row-1

# Specify the cell in which to start writing dates

start\_cell = 'A18'

current\_date = start\_date

current\_cell = sheet[start\_cell]

write\_date\_on\_column(current\_date, end\_date, current\_cell,initial\_row,final\_row)

TS = workbook.active

TS = workbook[sheet\_name]

# Calculate the number of rows to delete

num\_rows\_to\_delete = final\_row - initial\_row

if num\_rows\_to\_delete !=0:

TS.delete\_rows(initial\_row,num\_rows\_to\_delete)

🡪This section is used for maintaining row on sheets.

🡪このセクションは、シート上の行を維持するために使用されます。

Inside **finally:** section

🡪This section draw chart diagram inside sheet “発電実績”

最後に：セクション

🡪このセクションは、シート "発電実績 "の中にチャート図を描画します。