import pandas as pd

import tkinter as tk

from tkinter import filedialog, simpledialog, messagebox

from sqlalchemy import create\_engine, text

import urllib.parse

import os

def get\_db\_connection(jdbc\_url, username, password):

try:

# Convert JDBC to SQLAlchemy format

# Example JDBC: jdbc:oracle:thin:@host:port/service

jdbc\_url = jdbc\_url.replace("jdbc:oracle:thin:@", "")

# Example formatted\_url: oracle+cx\_oracle://user:pass@host:port/?service\_name=your\_service

parts = jdbc\_url.split("/")

host\_port = parts[0]

service\_name = parts[1]

host, port = host\_port.split(":")

quoted\_password = urllib.parse.quote(password)

# Create SQLAlchemy Engine string

conn\_str = f"oracle+cx\_oracle://{username}:{quoted\_password}@{host}:{port}/?service\_name={service\_name}"

engine = create\_engine(conn\_str)

# Test connection

engine.execute(text("SELECT 1 FROM DUAL"))

return engine

except Exception as e:

messagebox.showerror("Connection Error", f"Database connection failed:\n{e}")

return None

def get\_customer\_ref(engine, user\_id):

query = text("SELECT customer\_ref FROM v\_nonvzw\_customer WHERE userid = :userid")

result = engine.execute(query, {'userid': user\_id}).fetchone()

return result[0] if result else None

def get\_latest\_subscription(engine, customer\_ref):

query = text("""

SELECT start\_date, end\_date

FROM scm\_subscription

WHERE customer\_number = :customer\_number

ORDER BY

CASE

WHEN end\_date IS NULL THEN 1

ELSE 0

END,

end\_date DESC

""")

result = engine.execute(query, {'customer\_number': customer\_ref}).fetchall()

return result[0] if result else (None, None)

def process\_file(file\_path, user\_id\_column, engine):

is\_excel = file\_path.endswith('.xlsx')

df = pd.read\_excel(file\_path) if is\_excel else pd.read\_csv(file\_path)

subscription\_status = []

subscription\_start = []

subscription\_end = []

for user\_id in df[user\_id\_column]:

customer\_ref = get\_customer\_ref(engine, user\_id)

if customer\_ref:

result = get\_latest\_subscription(engine, customer\_ref)

if result:

start\_date, end\_date = result

subscription\_start.append(start\_date)

subscription\_end.append(end\_date)

if end\_date is None:

subscription\_status.append("Subscription Active")

else:

subscription\_status.append("Subscription Canceled")

else:

subscription\_status.append("No Subscription Found")

subscription\_start.append(None)

subscription\_end.append(None)

else:

subscription\_status.append("Customer Ref Not Found")

subscription\_start.append(None)

subscription\_end.append(None)

# Add new columns

df["Subscription\_Status"] = subscription\_status

df["Subscription\_Start\_Date"] = subscription\_start

df["Subscription\_End\_Date"] = subscription\_end

# Save output file

output\_file = os.path.splitext(file\_path)[0] + "\_updated." + ("xlsx" if is\_excel else "csv")

if is\_excel:

df.to\_excel(output\_file, index=False)

else:

df.to\_csv(output\_file, index=False)

messagebox.showinfo("Success", f"Processed file saved as:\n{output\_file}")

def gui\_app():

root = tk.Tk()

root.withdraw()

# Step 1: Get DB Credentials

jdbc\_url = simpledialog.askstring("Database Connection", "Enter Oracle JDBC URL (e.g., jdbc:oracle:thin:@host:port/service):")

username = simpledialog.askstring("Database Username", "Enter Oracle username:")

password = simpledialog.askstring("Database Password", "Enter Oracle password:", show='\*')

if not jdbc\_url or not username or not password:

messagebox.showerror("Error", "All database fields are required.")

return

engine = get\_db\_connection(jdbc\_url, username, password)

if not engine:

return # error already shown

# Step 2: File Selection

file\_path = filedialog.askopenfilename(

title="Select CSV or XLSX file",

filetypes=[("Excel files", "\*.xlsx"), ("CSV files", "\*.csv")]

)

if not file\_path:

messagebox.showerror("Error", "No file selected.")

return

# Step 3: Ask for user\_id column

is\_excel = file\_path.endswith('.xlsx')

df = pd.read\_excel(file\_path) if is\_excel else pd.read\_csv(file\_path)

column\_list = df.columns.tolist()

user\_id\_column = simpledialog.askstring("Column Name", f"Enter the user\_id column from:\n{column\_list}")

if user\_id\_column not in column\_list:

messagebox.showerror("Error", f"Column '{user\_id\_column}' not found in file.")

return

# Step 4: Process

process\_file(file\_path, user\_id\_column, engine)

if \_\_name\_\_ == "\_\_main\_\_":

gui\_app()