Nice choice 🙌 Let’s build your **“Excel Report Generator”** project step by step. We’ll use **pandas** for data wrangling, **matplotlib** for charts, **openpyxl** for Excel export + styling, and **Tkinter** for the GUI.

**Project Structure**

excel-report-generator/

app.py # Tkinter GUI app

requirements.txt # Dependencies

sample.csv # Example input CSV

output\_report.xlsx # Generated report

screenshots/ # UI screenshots

README.md # Usage guide

**requirements.txt**

pandas>=2.0

openpyxl>=3.1

matplotlib>=3.8

tkinter # built-in

**app.py**

import pandas as pd

import matplotlib.pyplot as plt

import tkinter as tk

from tkinter import filedialog, messagebox

from openpyxl import load\_workbook

from openpyxl.styles import Font, PatternFill

from openpyxl.utils.dataframe import dataframe\_to\_rows

import os

# -------------------------

# Helpers

# -------------------------

def load\_csv(file\_path):

return pd.read\_csv(file\_path)

def create\_pivot(df):

# Example pivot: sum of numeric cols grouped by first column

numeric\_cols = df.select\_dtypes(include="number").columns

if len(numeric\_cols) > 0:

pivot = df.pivot\_table(index=df.columns[0], values=numeric\_cols, aggfunc="sum")

return pivot

return pd.DataFrame()

def create\_summary(df):

desc = df.describe(include="all")

return desc

def create\_chart(df, filename="chart.png"):

plt.figure(figsize=(6, 4))

df[df.columns[1]].head(10).plot(kind="bar")

plt.title("Sample Chart")

plt.tight\_layout()

plt.savefig(filename)

plt.close()

def export\_to\_excel(df, pivot, summary, save\_path):

with pd.ExcelWriter(save\_path, engine="openpyxl") as writer:

df.to\_excel(writer, sheet\_name="Raw Data", index=False)

pivot.to\_excel(writer, sheet\_name="Pivot Table")

summary.to\_excel(writer, sheet\_name="Summary Stats")

# Add basic styling

wb = load\_workbook(save\_path)

for sheet in wb.sheetnames:

ws = wb[sheet]

for cell in ws[1]:

cell.font = Font(bold=True, color="FFFFFF")

cell.fill = PatternFill("solid", fgColor="4F81BD")

wb.save(save\_path)

# -------------------------

# GUI Functions

# -------------------------

def open\_file():

file\_path = filedialog.askopenfilename(filetypes=[("CSV Files", "\*.csv")])

if file\_path:

entry\_csv.delete(0, tk.END)

entry\_csv.insert(0, file\_path)

def generate\_report():

file\_path = entry\_csv.get()

if not os.path.exists(file\_path):

messagebox.showerror("Error", "Select a valid CSV file.")

return

try:

df = load\_csv(file\_path)

pivot = create\_pivot(df)

summary = create\_summary(df)

create\_chart(df, "chart.png")

save\_path = filedialog.asksaveasfilename(defaultextension=".xlsx",

filetypes=[("Excel Files", "\*.xlsx")],

title="Save Report As")

if save\_path:

export\_to\_excel(df, pivot, summary, save\_path)

messagebox.showinfo("Success", f"Report saved as {save\_path}")

except Exception as e:

messagebox.showerror("Error", str(e))

# -------------------------

# Tkinter UI

# -------------------------

root = tk.Tk()

root.title("📊 Excel Report Generator")

root.geometry("500x250")

tk.Label(root, text="Select CSV File:").pack(pady=10)

frame = tk.Frame(root)

frame.pack()

entry\_csv = tk.Entry(frame, width=40)

entry\_csv.pack(side=tk.LEFT, padx=5)

tk.Button(frame, text="Browse", command=open\_file).pack(side=tk.LEFT)

tk.Button(root, text="Generate Report", command=generate\_report,

bg="green", fg="white", font=("Arial", 12)).pack(pady=20)

root.mainloop()

**README.md**

# Excel Report Generator

## Objective

Generate Excel summaries from CSV inputs.

## Tools

- pandas – load + transform CSV

- matplotlib – create charts

- openpyxl – styled Excel export

- tkinter – file picker + UI

## Features

1. Load CSV into pandas.

2. Generate pivot table (sum by first column).

3. Create summary statistics.

4. Export styled Excel with \*\*Raw Data\*\*, \*\*Pivot Table\*\*, \*\*Summary Stats\*\* sheets.

5. Generate sample chart.

6. GUI for file upload + save dialog.

## Run

```bash

pip install -r requirements.txt

python app.py

**Deliverables**

* Excel file (output\_report.xlsx)
* Sample CSV (sample.csv)
* Screenshots of UI (/screenshots)

\*\*Deliverables covered\*\*:

- GUI app (`app.py`)

- \*\*Excel file\*\* with raw data, pivot, and summary

- Sample chart inside export

- Sample CSV + screenshots