E.D.W.A.R.D.

The Electronic Data Writer Analyzer Reporter and Documenter

(a play on the SEC's EDGAR)

(Python Application)

By

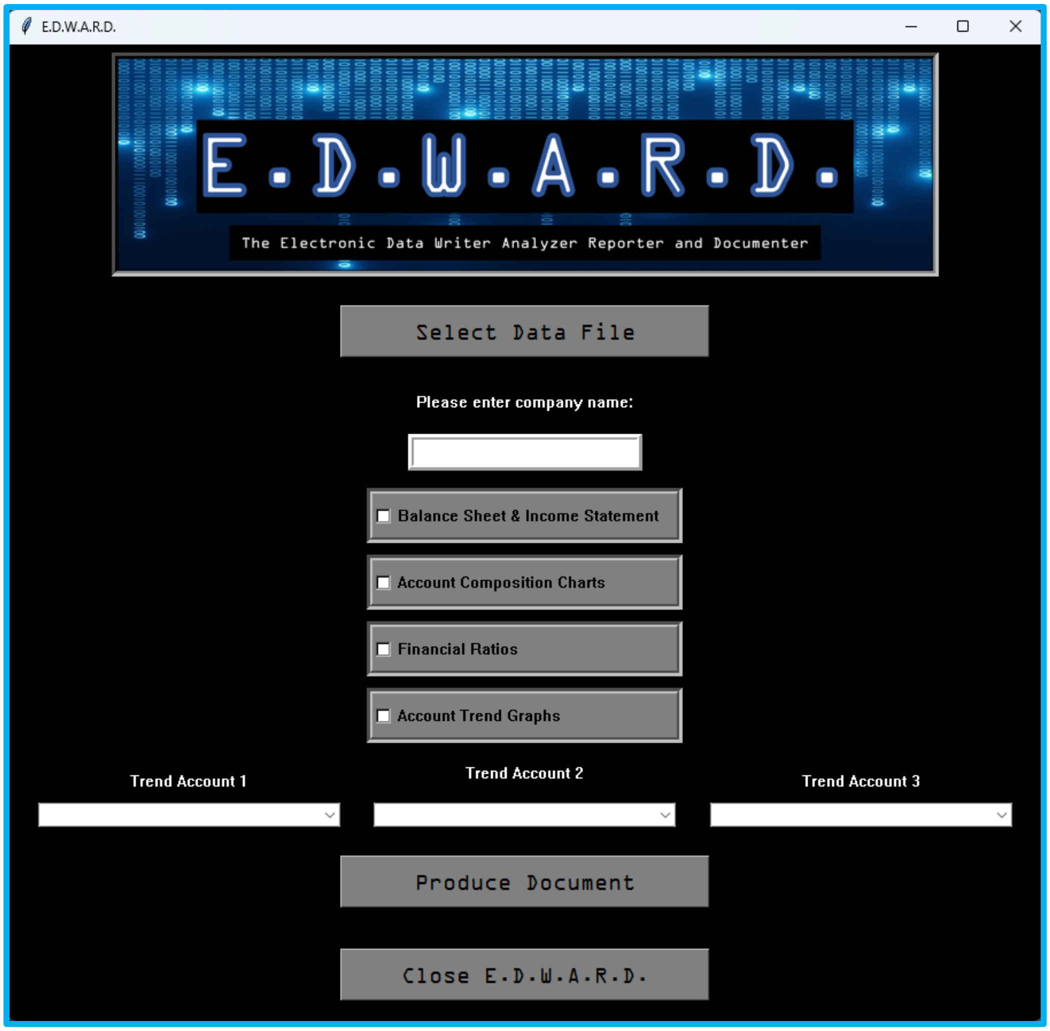
Nate Boyle

1/10/2024

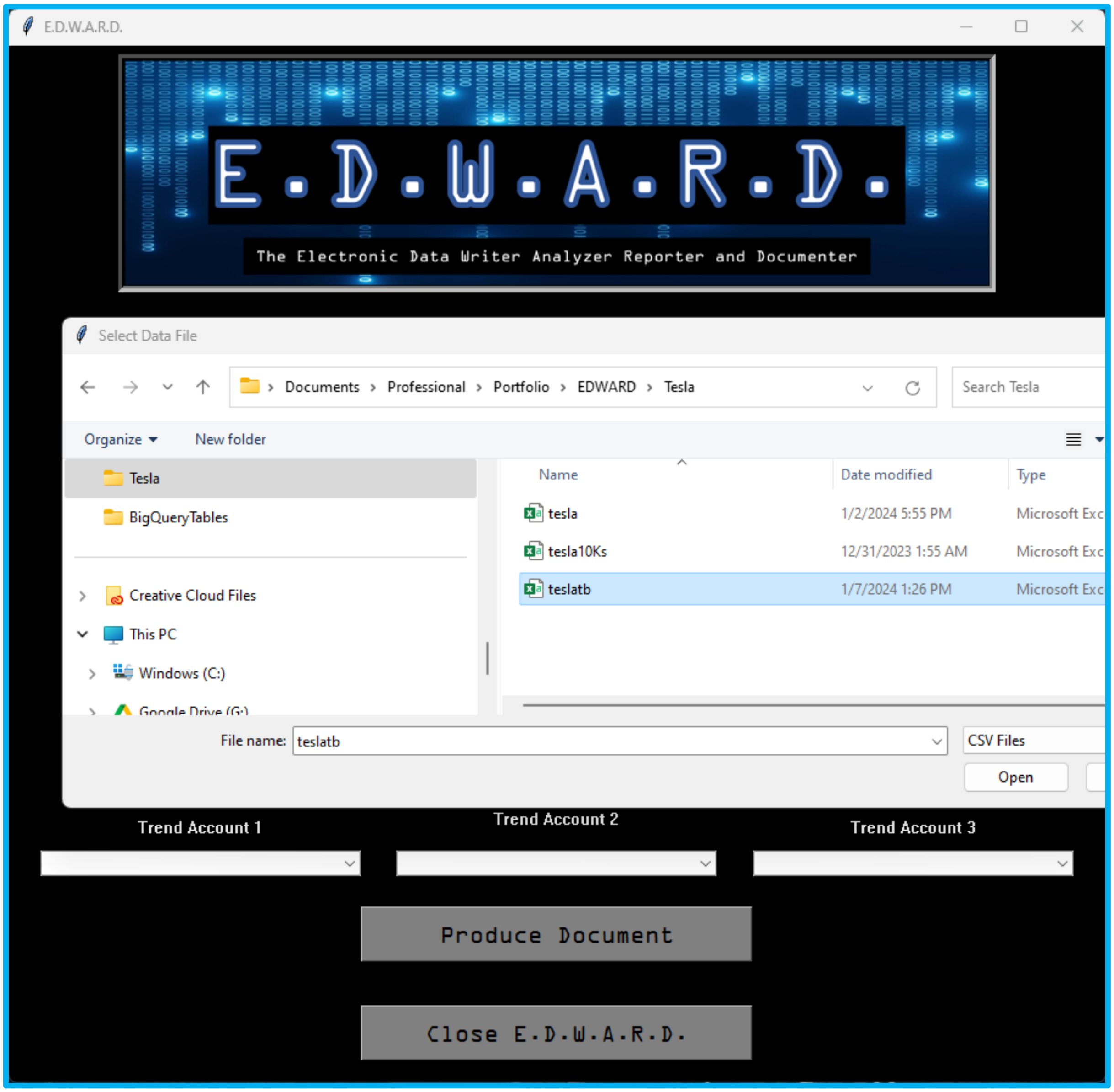
G.U.I.

Users can:

- Select a slightly formatted trial balance CSV file for upload with several options available:
    - Produce a mock balance sheet and income statement for the current and prior fiscal year
    - Include the following financial data visualizations to the produced document:
        - Account Composition Charts
        - Financial Ratios
        - Trend graphs of up to three accounts



File Selection



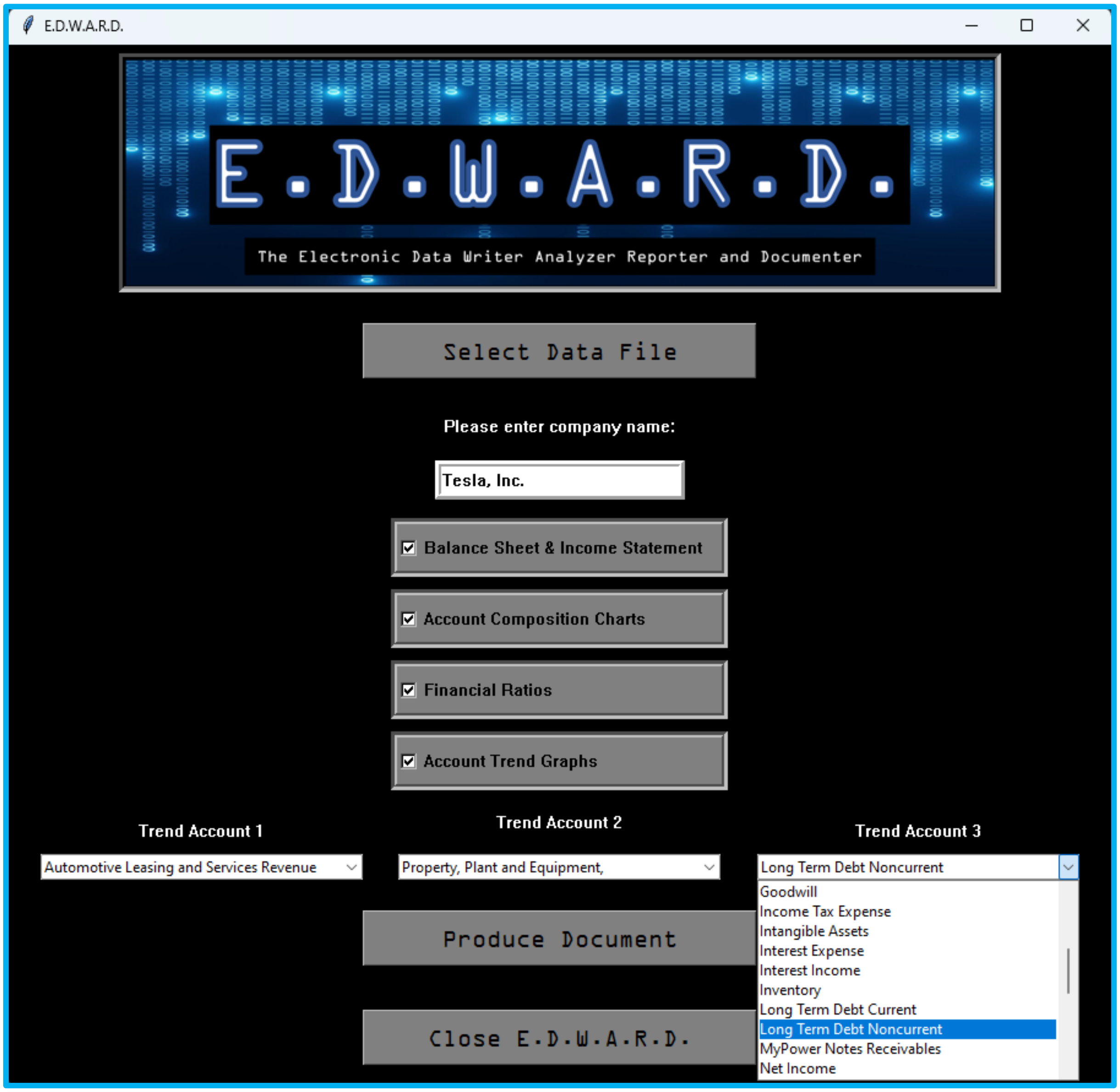
Name	Date modified	Type
tesla	1/2/2024 5:55 PM	Microsoft Exc
tesla10Ks	12/31/2023 1:55 AM	Microsoft Exc
teslatb	1/7/2024 1:26 PM	Microsoft Exc

File name: teslatb

CSV Files

Note: The Tesla trial balance file used in the demonstration was created using BigQuery to query the S.E.C. public dataset for general ledger accounts and cash flow activities.

Feature Selection



Please enter company name:

Tesla, Inc.

☑ Balance Sheet & Income Statement

☑ Account Composition Charts

☑ Financial Ratios

☑ Account Trend Graphs

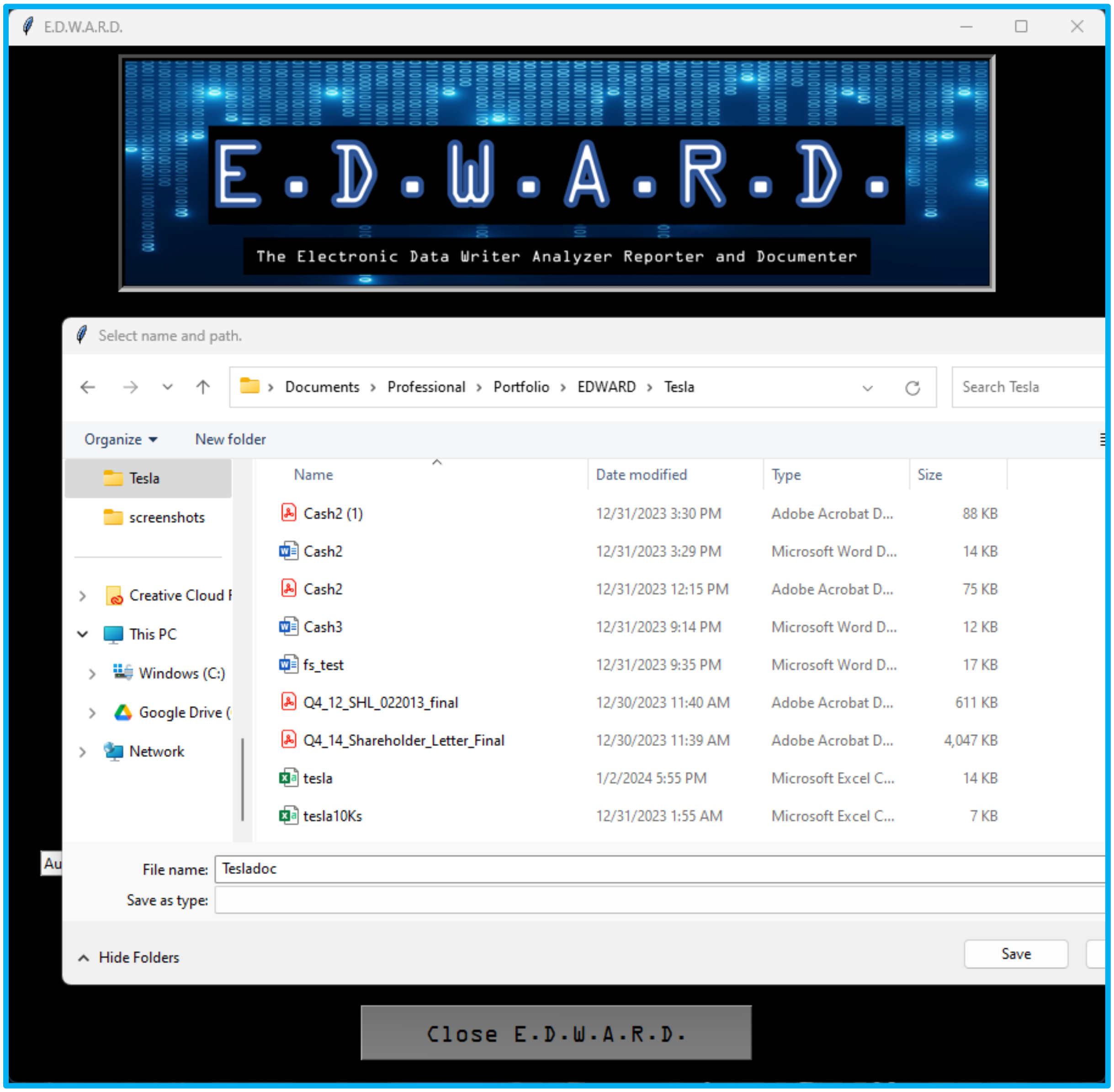
Trend Account 1: Automotive Leasing and Services Revenue

Trend Account 2: Property, Plant and Equipment,

Trend Account 3: Long Term Debt Noncurrent

Note: Users must enter in a name for the company that will be used in the header of the financial statements.

Document Production



Documents › Professional › Portfolio › EDWARD › Tesla

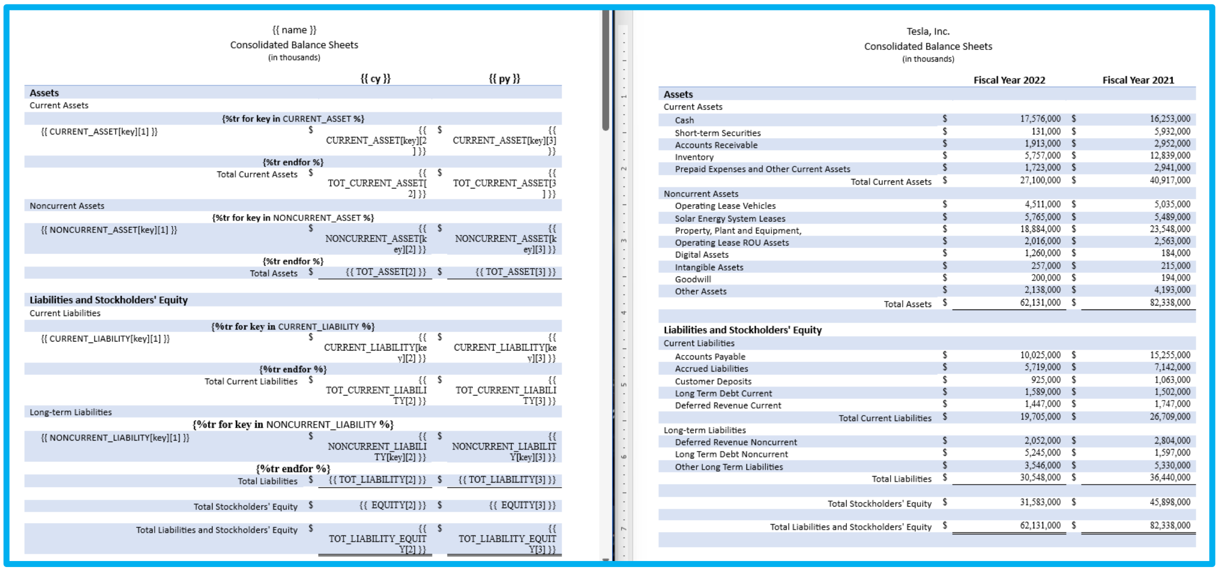
Name	Date modified	Type	Size
Cash2 (1)	12/31/2023 3:30 PM	Adobe Acrobat D...	88 KB
Cash2	12/31/2023 3:29 PM	Microsoft Word D...	14 KB
Cash2	12/31/2023 12:15 PM	Adobe Acrobat D...	75 KB
Cash3	12/31/2023 9:14 PM	Microsoft Word D...	12 KB
fs_test	12/31/2023 9:35 PM	Microsoft Word D...	17 KB
Q4_12_SHL_022013_final	12/30/2023 11:40 AM	Adobe Acrobat D...	611 KB
Q4_14_Shareholder_Letter_Final	12/30/2023 11:39 AM	Adobe Acrobat D...	4,047 KB
tesla	1/2/2024 5:55 PM	Microsoft Excel C...	14 KB
tesla10Ks	12/31/2023 1:55 AM	Microsoft Excel C...	7 KB

File name: Tesladoc

Save as type:

Document Results

Balance Sheet



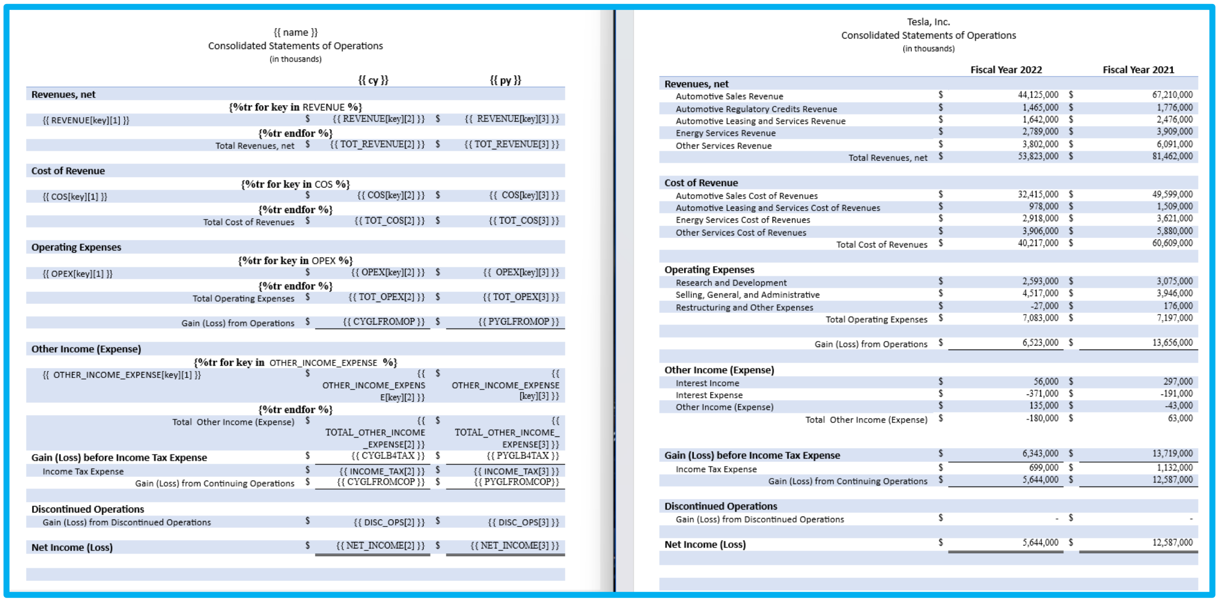
{{ name }}
Consolidated Balance Sheets
(in thousands)

		{{ cy }}		{{ py }}
Assets				
Current Assets				
{%tr for key in CURRENT_ASSET %}				
{{ CURRENT_ASSET[key][1] }}	$	{{ CURRENT_ASSET[key][2] }}	$	{{ CURRENT_ASSET[key][3] }}
{%tr endfor %}				
Total Current Assets	$	{{ TOT_CURRENT_ASSET[2] }}	$	{{ TOT_CURRENT_ASSET[3] }}
Noncurrent Assets				
{%tr for key in NONCURRENT_ASSET %}				
{{ NONCURRENT_ASSET[key][1] }}	$	{{ NONCURRENT_ASSET[key][2] }}	$	{{ NONCURRENT_ASSET[key][3] }}
{%tr endfor %}				
Total Assets	$	{{ TOT_ASSET[2] }}	$	{{ TOT_ASSET[3] }}
Liabilities and Stockholders' Equity				
Current Liabilities				
{%tr for key in CURRENT_LIABILITY %}				
{{ CURRENT_LIABILITY[key][1] }}	$	{{ CURRENT_LIABILITY[key][2] }}	$	{{ CURRENT_LIABILITY[key][3] }}
{%tr endfor %}				
Total Current Liabilities	$	{{ TOT_CURRENT_LIABILITY[2] }}	$	{{ TOT_CURRENT_LIABILITY[3] }}
Long-term Liabilities				
{%tr for key in NONCURRENT_LIABILITY %}				
{{ NONCURRENT_LIABILITY[key][1] }}	$	{{ NONCURRENT_LIABILITY[key][2] }}	$	{{ NONCURRENT_LIABILITY[key][3] }}
{%tr endfor %}				
Total Liabilities	$	{{ TOT_LIABILITY[2] }}	$	{{ TOT_LIABILITY[3] }}
Total Stockholders' Equity	$	{{ EQUITY[2] }}	$	{{ EQUITY[3] }}
Total Liabilities and Stockholders' Equity	$	{{ TOT_LIABILITY_EQUITY[2] }}	$	{{ TOT_LIABILITY_EQUITY[3] }}

Tesla, Inc.
Consolidated Balance Sheets
(in thousands)

		Fiscal Year 2022		Fiscal Year 2021
Assets				
Current Assets				
Cash	$	17,576,000	$	16,253,000
Short-term Securities	$	131,000	$	5,932,000
Accounts Receivable	$	1,913,000	$	2,952,000
Inventory	$	5,757,000	$	12,839,000
Prepaid Expenses and Other Current Assets	$	1,723,000	$	2,941,000
Total Current Assets	$	27,100,000	$	40,917,000
Noncurrent Assets				
Operating Lease Vehicles	$	4,511,000	$	5,035,000
Solar Energy System Leases	$	5,765,000	$	5,489,000
Property, Plant and Equipment,	$	18,884,000	$	23,548,000
Operating Lease ROU Assets	$	2,016,000	$	2,563,000
Digital Assets	$	1,260,000	$	184,000
Intangible Assets	$	257,000	$	215,000
Goodwill	$	200,000	$	194,000
Other Assets	$	2,138,000	$	4,193,000
Total Assets	$	62,131,000	$	82,338,000
Liabilities and Stockholders' Equity				
Current Liabilities				
Accounts Payable	$	10,025,000	$	15,255,000
Accrued Liabilities	$	5,719,000	$	7,142,000
Customer Deposits	$	925,000	$	1,063,000
Long Term Debt Current	$	1,589,000	$	1,502,000
Deferred Revenue Current	$	1,447,000	$	1,747,000
Total Current Liabilities	$	19,705,000	$	26,709,000
Long-term Liabilities				
Deferred Revenue Noncurrent	$	2,052,000	$	2,804,000
Long Term Debt Noncurrent	$	5,245,000	$	1,597,000
Other Long Term Liabilities	$	3,546,000	$	5,330,000
Total Liabilities	$	30,548,000	$	36,440,000
Total Stockholders' Equity	$	31,583,000	$	45,898,000
Total Liabilities and Stockholders' Equity	$	62,131,000	$	82,338,000

Income Statement



{{ name }}
Consolidated Statements of Operations
(in thousands)

		{{ cy }}		{{ py }}
Revenues, net				
{%tr for key in REVENUE %}				
{{ REVENUE[key][1] }}	$	{{ REVENUE[key][2] }}	$	{{ REVENUE[key][3] }}
{%tr endfor %}				
Total Revenues, net	$	{{ TOT_REVENUE[2] }}	$	{{ TOT_REVENUE[3] }}
Cost of Revenue				
{%tr for key in COS %}				
{{ COS[key][1] }}	$	{{ COS[key][2] }}	$	{{ COS[key][3] }}
{%tr endfor %}				
Total Cost of Revenues	$	{{ TOT_COS[2] }}	$	{{ TOT_COS[3] }}
Operating Expenses				
{%tr for key in OPEX %}				
{{ OPEX[key][1] }}	$	{{ OPEX[key][2] }}	$	{{ OPEX[key][3] }}
{%tr endfor %}				
Total Operating Expenses	$	{{ TOT_OPEX[2] }}	$	{{ TOT_OPEX[3] }}
Gain (Loss) from Operations	$	{{ CYGLFROMOP }}	$	{{ PYGLFROMOP }}
Other Income (Expense)				
{%tr for key in OTHER_INCOME_EXPENSE %}				
{{ OTHER_INCOME_EXPENSE[key][1] }}	$	{{ OTHER_INCOME_EXPENSE[key][2] }}	$	{{ OTHER_INCOME_EXPENSE[key][3] }}
{%tr endfor %}				
Total Other Income (Expense)	$	{{ TOTAL_OTHER_INCOME_EXPENSE[2] }}	$	{{ TOTAL_OTHER_INCOME_EXPENSE[3] }}
Gain (Loss) before Income Tax Expense	$	{{ CYGLB4TAX }}	$	{{ PYGLB4TAX }}
Income Tax Expense	$	{{ INCOME_TAX[2] }}	$	{{ INCOME_TAX[3] }}
Gain (Loss) from Continuing Operations	$	{{ CYGLFROMCOP }}	$	{{ PYGLFROMCOP}}
Discontinued Operations				
Gain (Loss) from Discontinued Operations	$	{{ DISC_OPS[2] }}	$	{{ DISC_OPS[3] }}
Net Income (Loss)	$	{{ NET_INCOME[2] }}	$	{{ NET_INCOME[3] }}

Tesla, Inc.
Consolidated Statements of Operations
(in thousands)

		Fiscal Year 2022		Fiscal Year 2021
Revenues, net				
Automotive Sales Revenue	$	44,125,000	$	67,210,000
Automotive Regulatory Credits Revenue	$	1,465,000	$	1,776,000
Automotive Leasing and Services Revenue	$	1,642,000	$	2,476,000
Energy Services Revenue	$	2,789,000	$	3,909,000
Other Services Revenue	$	3,802,000	$	6,091,000
Total Revenues, net	$	53,823,000	$	81,462,000
Cost of Revenue				
Automotive Sales Cost of Revenues	$	32,415,000	$	49,599,000
Automotive Leasing and Services Cost of Revenues	$	978,000	$	1,509,000
Energy Services Cost of Revenues	$	2,918,000	$	3,621,000
Other Services Cost of Revenues	$	3,906,000	$	5,880,000
Total Cost of Revenues	$	40,217,000	$	60,609,000
Operating Expenses				
Research and Development	$	2,593,000	$	3,075,000
Selling, General, and Administrative	$	4,517,000	$	3,946,000
Restructuring and Other Expenses	$	-27,000	$	176,000
Total Operating Expenses	$	7,083,000	$	7,197,000
Gain (Loss) from Operations	$	6,523,000	$	13,656,000
Other Income (Expense)				
Interest Income	$	56,000	$	297,000
Interest Expense	$	-371,000	$	-191,000
Other Income (Expense)	$	135,000	$	-43,000
Total Other Income (Expense)	$	-180,000	$	63,000
Gain (Loss) before Income Tax Expense	$	6,343,000	$	13,719,000
Income Tax Expense	$	699,000	$	1,132,000
Gain (Loss) from Continuing Operations	$	5,644,000	$	12,587,000
Discontinued Operations				
Gain (Loss) from Discontinued Operations	$	-	$	-
Net Income (Loss)	$	5,644,000	$	12,587,000

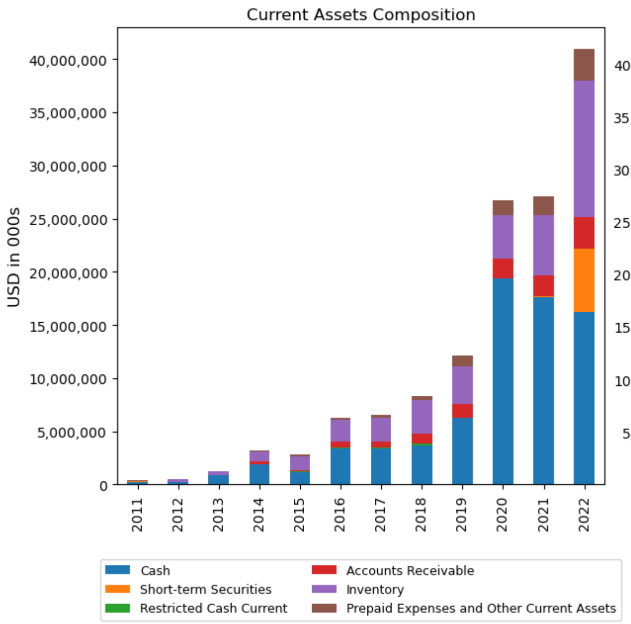
Note: The application uses a Word document template as one of its assets to produce the balance sheet and income statement, however, the number of accounts and their names are not limited or predetermined by this template.

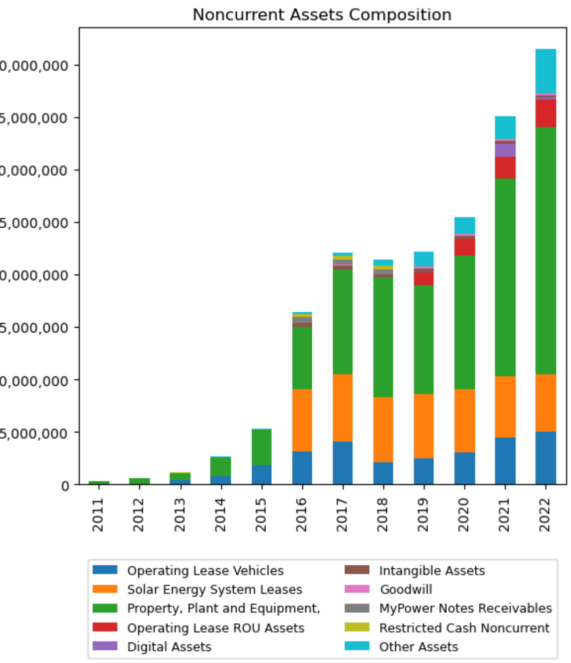
Visualizations

Note: These are the actual results from the demonstration and not just screenshots.

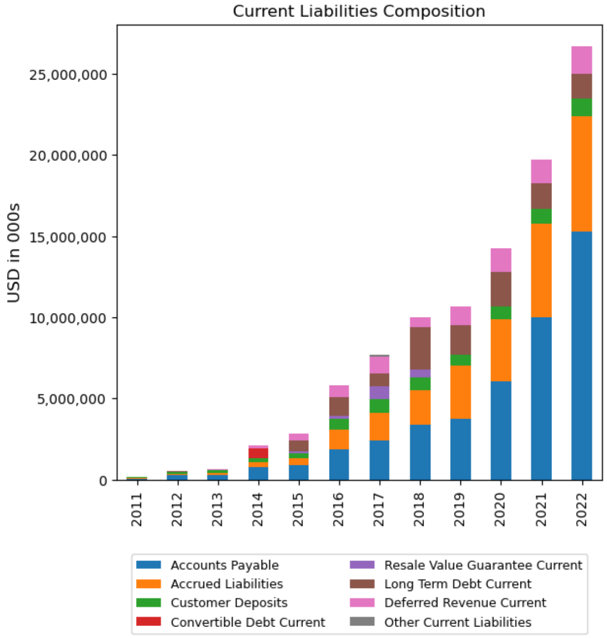
Account Composition Charts

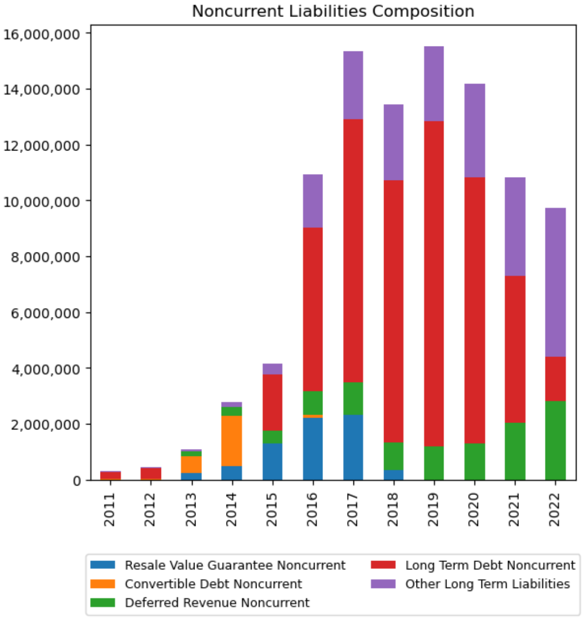
Asset Composition



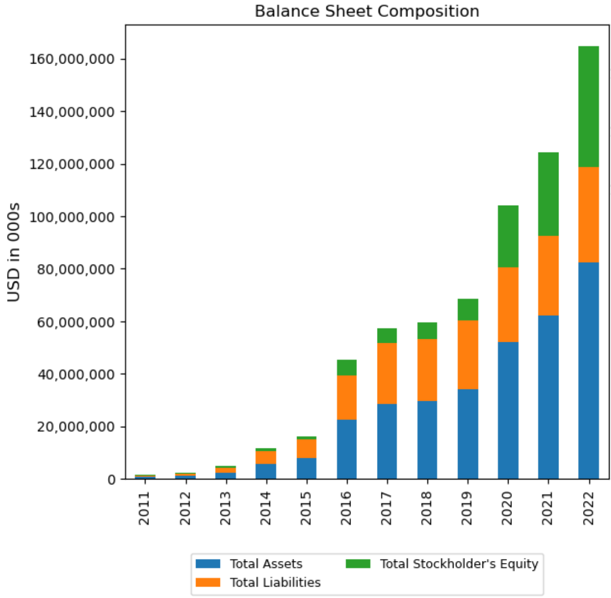


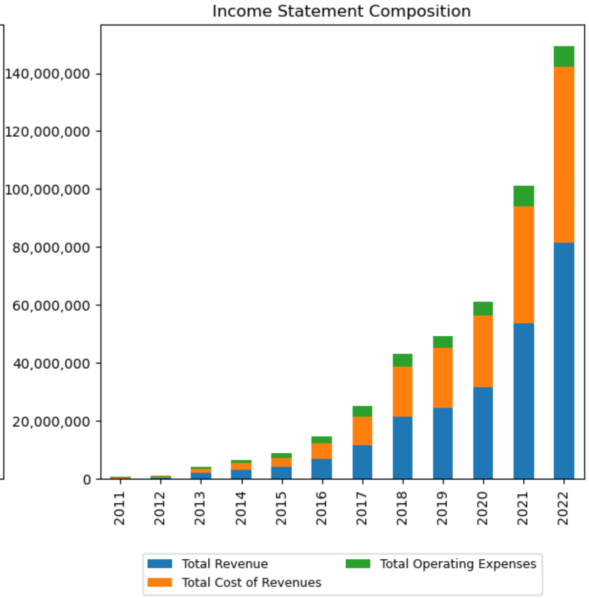
Liabilities Composition



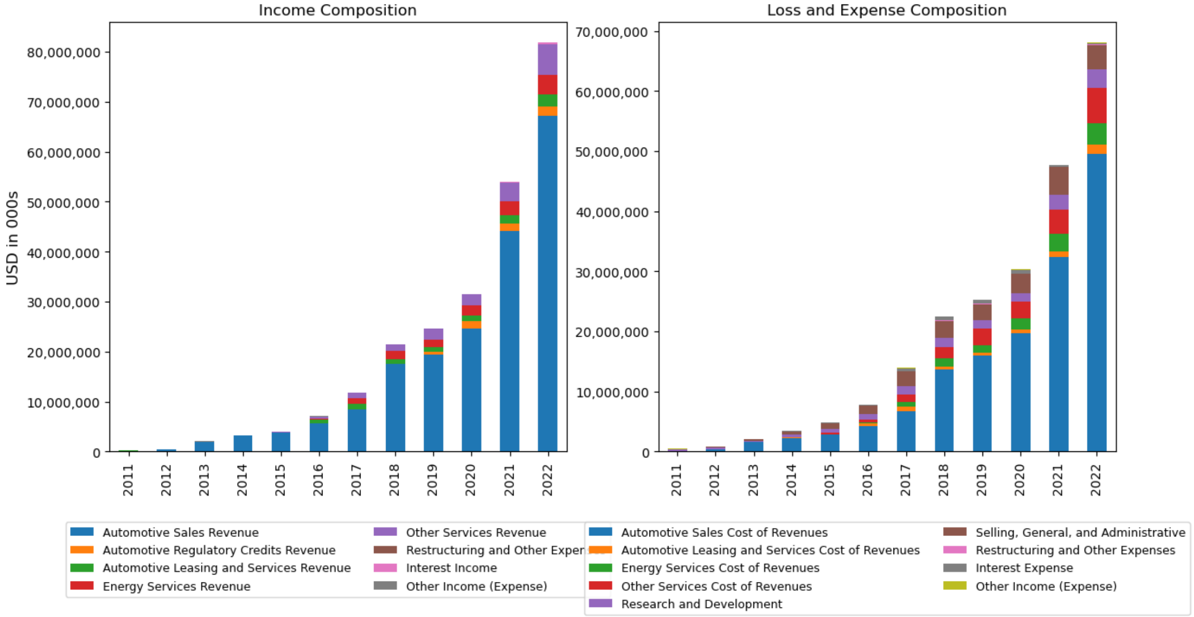


Balance Sheet and Income Statement Composition

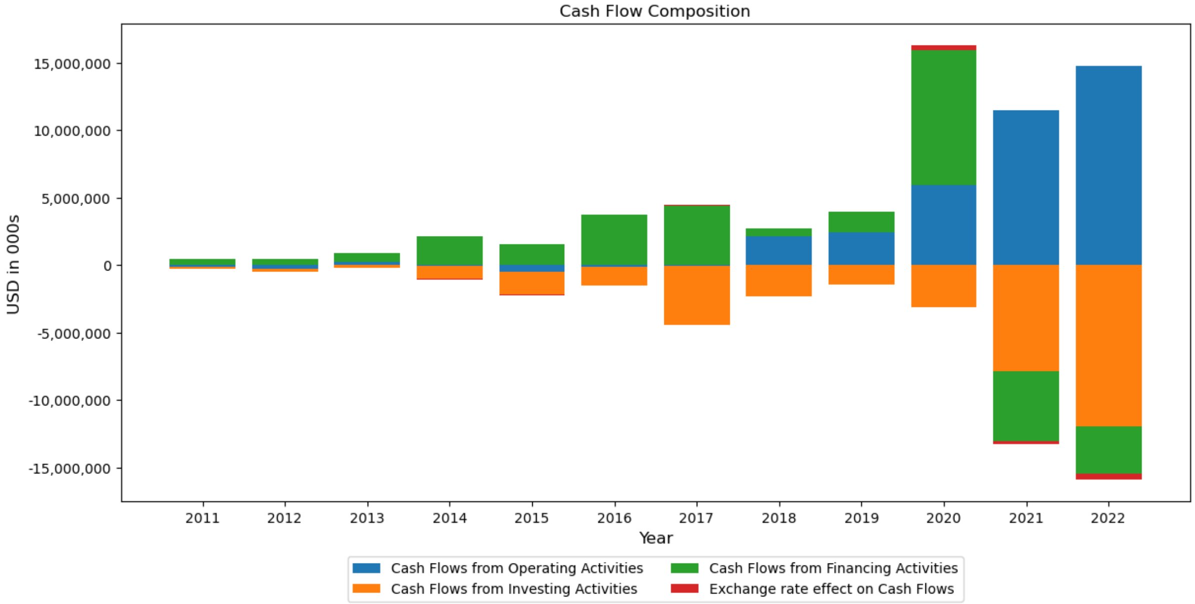




Income and Loss and Expense Composition

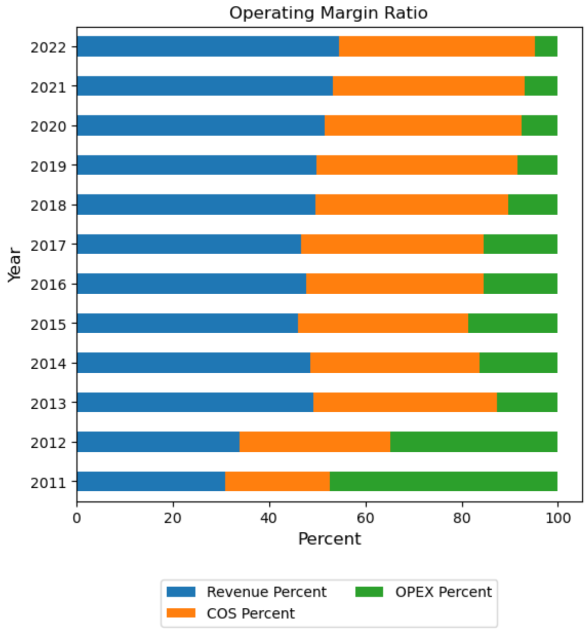


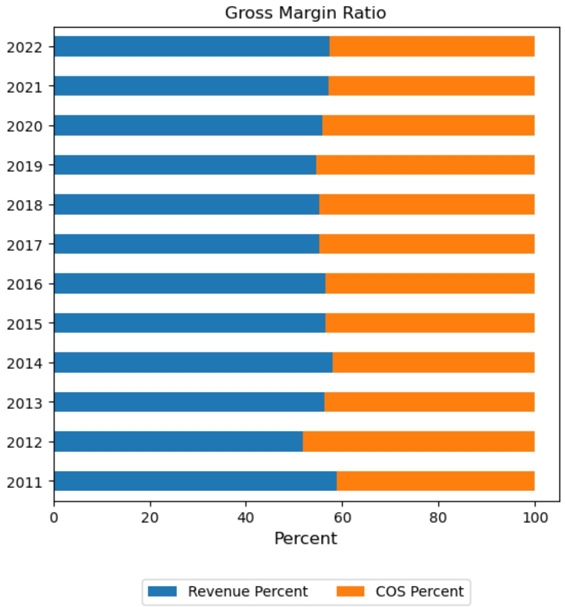
Cash Flow Composition



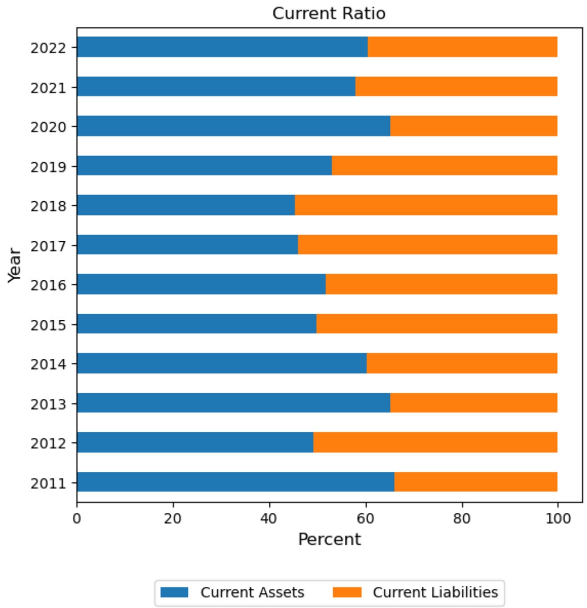
Financial Ratio Visualizations

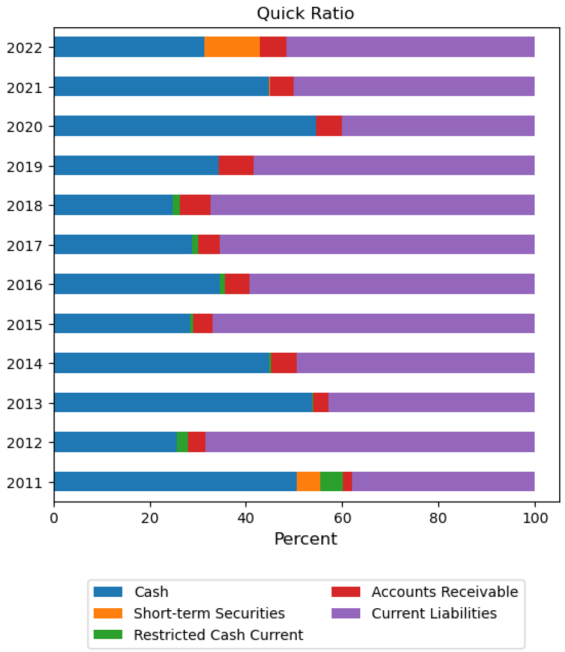
Profit Margin Ratios



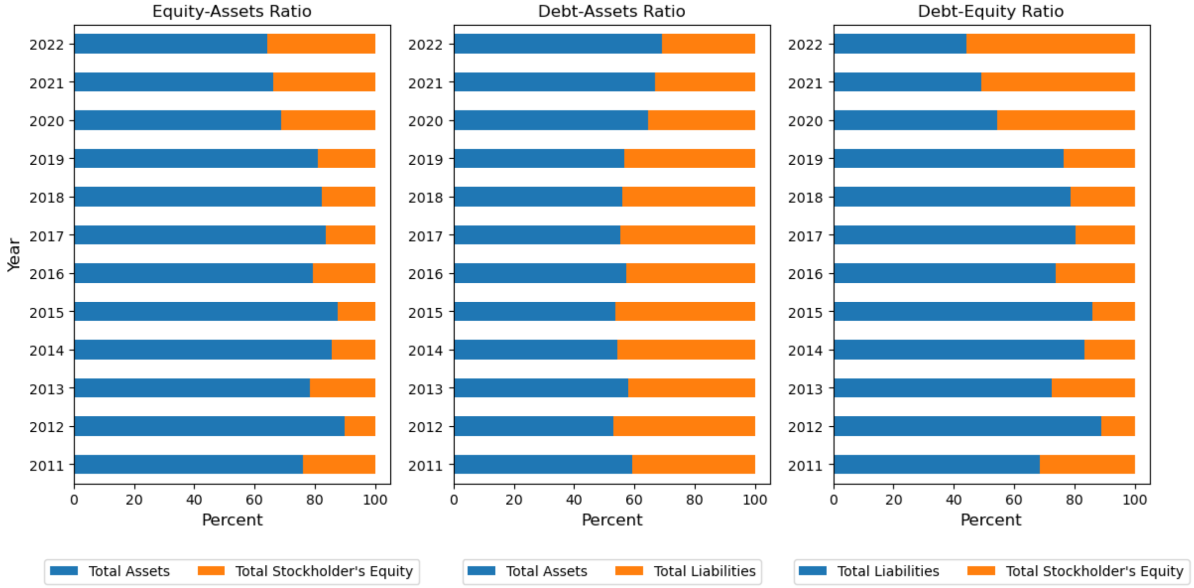


Liquidity Ratios

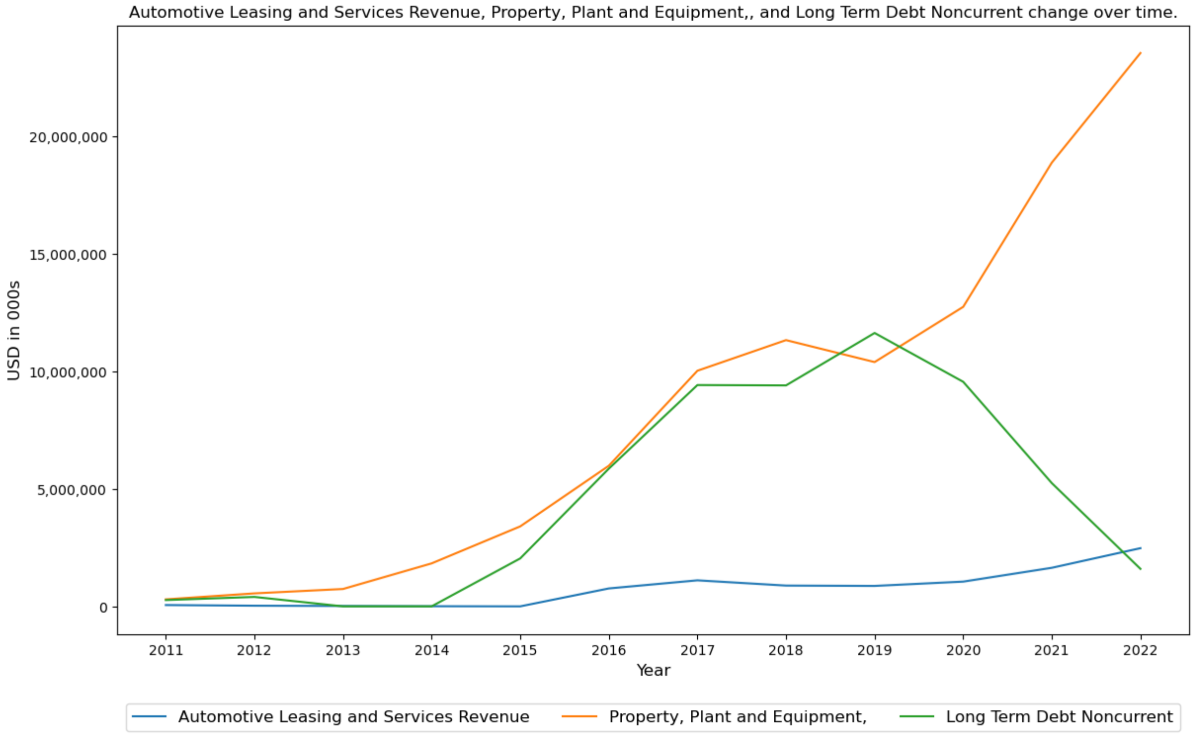




Solvency Ratios



Account Trend(s) Graph

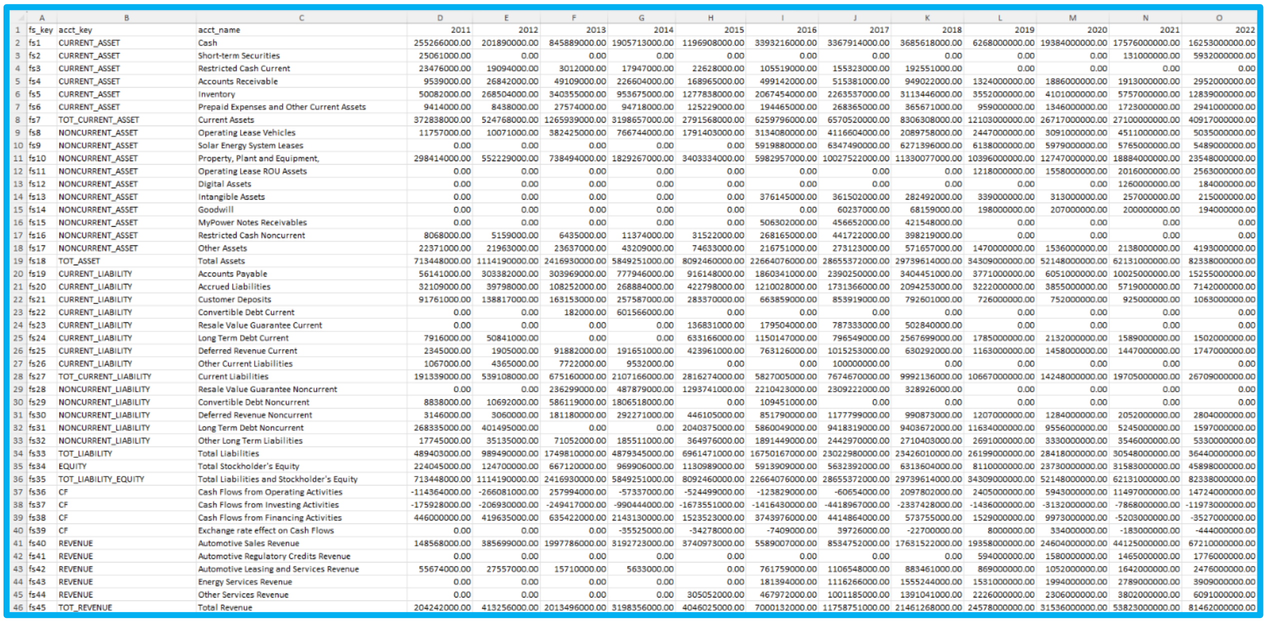


Trial Balance Format

- First column is the **fs_key**
  - Used by the program as a primary key when a unique identifier is required.
  - Must be unique and begin with a letter.
  - Sequence does not matter, only used here for sorting in Excel.
  - Other columns determine placement of account information.

- Second column is the **acct_key**
  - Used for account placement in the financial statements as well as for selecting accounts for the account composition charts and financial ratio visualizations.
  - Must use the **acct_key**s presented for program to work.
  - If correct **acct_key**s are used, account mapping does not matter and is up to discretion of the user.

- Third column is the **acct_name**
  - Used for display only.
  - Does not matter what the user puts here as long as the **acct_name**s are:
    - Not blank.
    - Begin with a letter.



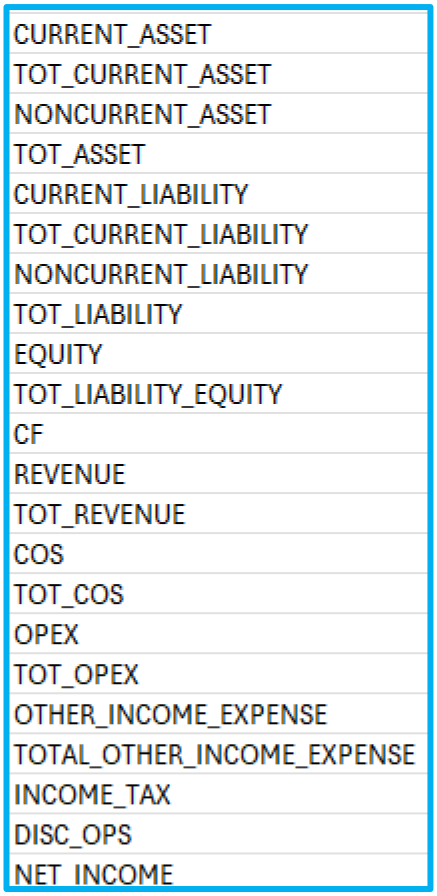
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	fs_key	acct_key	acct_name	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020	2021	2022
2	fs1	CURRENT_ASSET	Cash	255266000.00	201890000.00	845889000.00	1905713000.00	1196908000.00	3393216000.00	3367914000.00	3685618000.00	6268000000.00	19384000000.00	17576000000.00	16253000000.00
3	fs2	CURRENT_ASSET	Short-term Securities	25061000.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	131000000.00	5932000000.00
4	fs3	CURRENT_ASSET	Restricted Cash Current	23476000.00	19094000.00	3012000.00	17947000.00	22628000.00	105519000.00	155323000.00	192551000.00	0.00	0.00	0.00	0.00
5	fs4	CURRENT_ASSET	Accounts Receivable	9539000.00	26842000.00	49109000.00	226604000.00	168965000.00	499142000.00	515381000.00	949022000.00	1324000000.00	1886000000.00	1913000000.00	2952000000.00
6	fs5	CURRENT_ASSET	Inventory	50082000.00	268504000.00	340355000.00	953675000.00	1277838000.00	2067454000.00	2263537000.00	3113446000.00	3552000000.00	4101000000.00	5757000000.00	12839000000.00
7	fs6	CURRENT_ASSET	Prepaid Expenses and Other Current Assets	9414000.00	8438000.00	27574000.00	94718000.00	125229000.00	194465000.00	268365000.00	365671000.00	959000000.00	1346000000.00	1723000000.00	2941000000.00
8	fs7	TOT_CURRENT_ASSET	Current Assets	372838000.00	524768000.00	1265939000.00	3198657000.00	2791568000.00	6259796000.00	6570520000.00	8306308000.00	12103000000.00	26717000000.00	27100000000.00	40917000000.00
9	fs8	NONCURRENT_ASSET	Operating Lease Vehicles	11757000.00	10071000.00	382425000.00	766744000.00	1791403000.00	3134080000.00	4116604000.00	2089758000.00	2447000000.00	3091000000.00	4511000000.00	5035000000.00
10	fs9	NONCURRENT_ASSET	Solar Energy System Leases	0.00	0.00	0.00	0.00	0.00	5919880000.00	6347490000.00	6271396000.00	6138000000.00	5979000000.00	5765000000.00	5489000000.00
11	fs10	NONCURRENT_ASSET	Property, Plant and Equipment,	298414000.00	552229000.00	738494000.00	1829267000.00	3403334000.00	5982957000.00	10027522000.00	11330077000.00	10396000000.00	12747000000.00	18884000000.00	23548000000.00
12	fs11	NONCURRENT_ASSET	Operating Lease ROU Assets	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1218000000.00	1558000000.00	2016000000.00	2563000000.00
13	fs12	NONCURRENT_ASSET	Digital Assets	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1260000000.00	184000000.00
14	fs13	NONCURRENT_ASSET	Intangible Assets	0.00	0.00	0.00	0.00	0.00	376145000.00	361502000.00	282492000.00	339000000.00	313000000.00	257000000.00	215000000.00
15	fs14	NONCURRENT_ASSET	Goodwill	0.00	0.00	0.00	0.00	0.00	0.00	60237000.00	68159000.00	198000000.00	207000000.00	200000000.00	194000000.00
16	fs15	NONCURRENT_ASSET	MyPower Notes Receivables	0.00	0.00	0.00	0.00	0.00	506302000.00	456652000.00	421548000.00	0.00	0.00	0.00	0.00
17	fs16	NONCURRENT_ASSET	Restricted Cash Noncurrent	8068000.00	5159000.00	6435000.00	11374000.00	31522000.00	268165000.00	441722000.00	398219000.00	0.00	0.00	0.00	0.00
18	fs17	NONCURRENT_ASSET	Other Assets	22371000.00	21963000.00	23637000.00	43209000.00	74633000.00	216751000.00	273123000.00	571657000.00	1470000000.00	1536000000.00	2138000000.00	4193000000.00
19	fs18	TOT_ASSET	Total Assets	713448000.00	1114190000.00	2416930000.00	5849251000.00	8092460000.00	22664076000.00	28655372000.00	29739614000.00	34309000000.00	52148000000.00	62131000000.00	82338000000.00
20	fs19	CURRENT_LIABILITY	Accounts Payable	56141000.00	303382000.00	303969000.00	777946000.00	916148000.00	1860341000.00	2390250000.00	3404451000.00	3771000000.00	6051000000.00	10025000000.00	15255000000.00
21	fs20	CURRENT_LIABILITY	Accrued Liabilities	32109000.00	39798000.00	108252000.00	268884000.00	422798000.00	1210028000.00	1731366000.00	2094253000.00	3222000000.00	3855000000.00	5719000000.00	7142000000.00
22	fs21	CURRENT_LIABILITY	Customer Deposits	91761000.00	138817000.00	163153000.00	257587000.00	283370000.00	663859000.00	853919000.00	792601000.00	726000000.00	752000000.00	925000000.00	1063000000.00
23	fs22	CURRENT_LIABILITY	Convertible Debt Current	0.00	0.00	182000.00	601566000.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
24	fs23	CURRENT_LIABILITY	Resale Value Guarantee Current	0.00	0.00	0.00	0.00	136831000.00	179504000.00	787333000.00	502840000.00	0.00	0.00	0.00	0.00
25	fs24	CURRENT_LIABILITY	Long Term Debt Current	7916000.00	50841000.00	0.00	0.00	633166000.00	1150147000.00	796549000.00	2567699000.00	1785000000.00	2132000000.00	1589000000.00	1502000000.00
26	fs25	CURRENT_LIABILITY	Deferred Revenue Current	2345000.00	1905000.00	91882000.00	191651000.00	423961000.00	763126000.00	1015253000.00	630292000.00	1163000000.00	1458000000.00	1447000000.00	1747000000.00
27	fs26	CURRENT_LIABILITY	Other Current Liabilities	1067000.00	4365000.00	7722000.00	9532000.00	0.00	0.00	100000000.00	0.00	0.00	0.00	0.00	0.00
28	fs27	TOT_CURRENT_LIABILITY	Current Liabilities	191339000.00	539108000.00	675160000.00	2107166000.00	2816274000.00	5827005000.00	7674670000.00	9992136000.00	10667000000.00	14248000000.00	19705000000.00	26709000000.00
29	fs28	NONCURRENT_LIABILITY	Resale Value Guarantee Noncurrent	0.00	0.00	236299000.00	487879000.00	1293741000.00	2210423000.00	2309222000.00	328926000.00	0.00	0.00	0.00	0.00
30	fs29	NONCURRENT_LIABILITY	Convertible Debt Noncurrent	8838000.00	10692000.00	586119000.00	1806518000.00	0.00	109451000.00	0.00	0.00	0.00	0.00	0.00	0.00
31	fs30	NONCURRENT_LIABILITY	Deferred Revenue Noncurrent	3146000.00	3060000.00	181180000.00	292271000.00	446105000.00	851790000.00	1177799000.00	990873000.00	1207000000.00	1284000000.00	2052000000.00	2804000000.00
32	fs31	NONCURRENT_LIABILITY	Long Term Debt Noncurrent	268335000.00	401495000.00	0.00	0.00	2040375000.00	5860049000.00	9418319000.00	9403672000.00	11634000000.00	9556000000.00	5245000000.00	1597000000.00
33	fs32	NONCURRENT_LIABILITY	Other Long Term Liabilities	17745000.00	35135000.00	71052000.00	185511000.00	364976000.00	1891449000.00	2442970000.00	2710403000.00	2691000000.00	3330000000.00	3546000000.00	5330000000.00
34	fs33	TOT_LIABILITY	Total Liabilities	489403000.00	989490000.00	1749810000.00	4879345000.00	6961471000.00	16750167000.00	23022980000.00	23426010000.00	26199000000.00	28418000000.00	30548000000.00	36440000000.00
35	fs34	EQUITY	Total Stockholder's Equity	224045000.00	124700000.00	667120000.00	969906000.00	1130989000.00	5913909000.00	5632392000.00	6313604000.00	8110000000.00	23730000000.00	31583000000.00	45898000000.00
36	fs35	TOT_LIABILITY_EQUITY	Total Liabilities and Stockholder's Equity	713448000.00	1114190000.00	2416930000.00	5849251000.00	8092460000.00	22664076000.00	28655372000.00	29739614000.00	34309000000.00	52148000000.00	62131000000.00	82338000000.00
37	fs36	CF	Cash Flows from Operating Activities	-114364000.00	-266081000.00	257994000.00	-57337000.00	-524499000.00	-123829000.00	-60654000.00	2097802000.00	2405000000.00	5943000000.00	11497000000.00	14724000000.00
38	fs37	CF	Cash Flows from Investing Activities	-175928000.00	-206930000.00	-249417000.00	-990444000.00	-1673551000.00	-1416430000.00	-4418967000.00	-2337428000.00	-1436000000.00	-3132000000.00	-7868000000.00	-11973000000.00
39	fs38	CF	Cash Flows from Financing Activities	446000000.00	419635000.00	635422000.00	2143130000.00	1523523000.00	3743976000.00	4414864000.00	573755000.00	1529000000.00	9973000000.00	-5203000000.00	-3527000000.00
40	fs39	CF	Exchange rate effect on Cash Flows	0.00	0.00	0.00	-35525000.00	-34278000.00	-7409000.00	39726000.00	-22700000.00	8000000.00	334000000.00	-183000000.00	-444000000.00
41	fs40	REVENUE	Automotive Sales Revenue	148568000.00	385699000.00	1997786000.00	3192723000.00	3740973000.00	5589007000.00	8534752000.00	17631522000.00	19358000000.00	24604000000.00	44125000000.00	67210000000.00
42	fs41	REVENUE	Automotive Regulatory Credits Revenue	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	594000000.00	1580000000.00	1465000000.00	1776000000.00
43	fs42	REVENUE	Automotive Leasing and Services Revenue	55674000.00	27557000.00	15710000.00	5633000.00	0.00	761759000.00	1106548000.00	883461000.00	869000000.00	1052000000.00	1642000000.00	2476000000.00
44	fs43	REVENUE	Energy Services Revenue	0.00	0.00	0.00	0.00	0.00	181394000.00	1116266000.00	1555244000.00	1531000000.00	1994000000.00	2789000000.00	3909000000.00
45	fs44	REVENUE	Other Services Revenue	0.00	0.00	0.00	0.00	305052000.00	467972000.00	1001185000.00	1391041000.00	2226000000.00	2306000000.00	3802000000.00	6091000000.00
46	fs45	TOT_REVENUE	Total Revenue	204242000.00	413256000.00	2013496000.00	3198356000.00	4046025000.00	7000132000.00	11758751000.00	21461268000.00	24578000000.00	31536000000.00	53823000000.00	81462000000.00

Note: The least recent fiscal year needs to be the first column after **acct_name**, then going in chronological order from left to right with the most recent fiscal year being the furthest column to the right.

Trial Balance Format (cont'd)

List of all unique acct_keys used:

- If an **acct_key** is not present, the program compiles correctly but without any output for the feature areas associated with the aforementioned acct_key



CURRENT_ASSET
TOT_CURRENT_ASSET
NONCURRENT_ASSET
TOT_ASSET
CURRENT_LIABILITY
TOT_CURRENT_LIABILITY
NONCURRENT_LIABILITY
TOT_LIABILITY
EQUITY
TOT_LIABILITY_EQUITY
CF
REVENUE
TOT_REVENUE
COS
TOT_COS
OPEX
TOT_OPEX
OTHER_INCOME_EXPENSE
TOTAL_OTHER_INCOME_EXPENSE
INCOME_TAX
DISC_OPS
NET_INCOME

Please find the application code on the remaining pages.

```
#!/usr/bin/env python
# coding: utf-8

# ## Table of contents
#
# ###### ctrl+f a line of text below to navigate to the associated area
#
# #### 1* plot account trend graphs
# #### 2* plot account composition charts and financial ratio visualizations graphs
# #### 3* plot cash flow composition chart
# #### 4* create document and add selected visualizations to it
# #### 5* create document for financial statements
# #### 6* produce (save) document locally
# #### 7* create and (when called) open EDWARD GUI
# #### 8* Open E.D.W.A.R.D.
#

# In[63]:


# import os (operating system) library
import os

# import sys (system) library
import sys

# import pandas library as pd
import pandas as pd


# import numpy library as np
import numpy as np

# import these modules from datetime to store and compare dates
from datetime import datetime, date, timedelta

# import time for delay
import time
import pygame

# from tkinter import all standard modules with * (this is for the gui)
from tkinter import *
# from tkinter 'specifically' import messagebox, ttk as they are not standard
modules uploaded with *
from tkinter import messagebox, ttk
from tkinter import filedialog as fd

# import imageTk and Image for picture use in gui
from PIL import ImageTk as itk, Image
```

```
# import various matplotlib modules to create plots and then draw those plots in
the gui
import matplotlib
import matplotlib.pyplot as plt
from matplotlib.figure import Figure
import matplotlib.ticker as mtick
from matplotlib.backends.backend_tkagg import (FigureCanvasTkAgg,
NavigationToolbar2Tk)

from docx import Document
from docx.shared import Inches, Cm
from docx.shared import Pt
from docxcompose.composer import Composer

from docxtpl import DocxTemplate
import jinja2

import random
import ipyplot

# import webbrowser for hyperlink use
import webbrowser

from docx2pdf import convert
from tkPDFViewer import tkPDFViewer as pdf

from io import StringIO
from io import BytesIO


# import IPython display for wider coding screen (not required to run program)
from IPython.display import display, HTML
display(HTML("<style>.jp-Cell { width: 120% !important; }</style>"))


# #### 1* plot account trend graphs

# In[64]:


def plotAccountGraph(df, acclist):

    plotdf = df.iloc[:,2:].set_index('acct_name')

    fig, ax = plt.subplots(figsize=(14, 8))


    plot1df = plotdf.loc[acclist[0],:]

    ax.plot(plot1df)
    ax.get_yaxis().set_major_formatter(matplotlib.ticker.FuncFormatter(lambda x, p:
format(int(x/1000), ',')))


    ax.get_yaxis().set_major_formatter(matplotlib.ticker.FuncFormatter(lambda x, p:
format(int(x/1000), ',')))
    ax.set_ylabel("USD in 000s", fontsize="12")
    ax.set_xlabel('Year', fontsize="12")


    if len(acclist) == 2:
        plot2df = plotdf.loc[acclist[1],:]
        ax.plot(plot2df)

    if len(acclist) == 3:
        plot2df = plotdf.loc[acclist[1],:]
        ax.plot(plot2df)
        plot3df = plotdf.loc[acclist[2],:]
        ax.plot(plot3df)

    if len(acclist) == 1:
        titlestr = acclist[0] + " change over time."
    elif len(acclist) == 2:
        titlestr = acclist[0] + " and " + acclist[1] + " change over time."
    elif len(acclist) == 3:
        titlestr = acclist[0] + ", " + acclist[1] + ", and " + acclist[2] + "
change over time."

    ax.set_title(titlestr)

    ax.legend(acclist, ncol=len(acclist), loc="upper center", bbox_to_anchor=(0.5,
-0.1),
          fancybox=True, fontsize="12")

    memfile = BytesIO()


    plt.savefig(memfile, bbox_inches="tight")

    plt.close(fig)


    return memfile





# #### 2* plot account composition charts and financial ratio visualizations graphs
# In[65]:


def plotAccountCharts(df1,df2,df3,str1,str2,str3,vh,n):

    if n == 2:
        fig, axes = plt.subplots(1,2, figsize=(14, 6))
    if n == 3:
        fig, axes = plt.subplots(1,3, figsize=(14, 6))
        axes[2].set_title(str3)


    if vh == 'v':

        df1.plot.bar(ax=axes[0], stacked=True)
        axes[0].set_ylabel("USD in 000s", fontsize="12")

        df2.plot.bar(ax=axes[1], stacked=True)

        for i in range(2):

axes[i].get_yaxis().set_major_formatter(matplotlib.ticker.FuncFormatter(lambda x,
p: format(int(x/1000), ',')))
            #axes[i].set_xlabel('Year', fontsize="12")
            axes[i].legend(ncol=2, loc="upper center", bbox_to_anchor=(0.5, -0.15),
              fancybox=True, fontsize="9")

        if n == 3:
            df3.plot.bar(ax=axes[2], stacked=True)

axes[2].get_yaxis().set_major_formatter(matplotlib.ticker.FuncFormatter(lambda x,
p: format(int(x/1000), ',')))
            #axes[2].set_xlabel('Year', fontsize="12")
            axes[2].legend(ncol=2, loc="upper center", bbox_to_anchor=(0.5, -0.1),
              fancybox=True, fontsize="9")


    elif vh == 'h':

        df1.plot.barh(ax=axes[0], stacked=True)
        axes[0].set_ylabel("Year", fontsize="12")

        df2.plot.barh(ax=axes[1], stacked=True)

        for i in range(2):
            axes[i].set_xlabel('Percent', fontsize="12")
            axes[i].legend(ncol=2, loc="upper center", bbox_to_anchor=(0.5, -0.15),
              fancybox=True, fontsize="10")
        if n == 3:
            df3.plot.barh(ax=axes[2], stacked=True)
            axes[2].set_xlabel('Percent', fontsize="12")
            axes[2].legend(ncol=2, loc="upper center", bbox_to_anchor=(0.5, -0.15),
              fancybox=True, fontsize="10")


    axes[0].set_title(str1)
    axes[1].set_title(str2)


    memfile = BytesIO()

    # memlist = [' ']*2
    # memlist[0] = liqmemfile
    plt.savefig(memfile, bbox_inches="tight")

    plt.close(fig)

    return memfile



# #### 3* plot cash flow composition chart

# In[66]:


def plotCF(df, years):

    cfdict = df.to_dict('list')

    # cash flow composition graph
    cflist = [0]*len(cfdict)
    counter = 0

    for key in cfdict:
        cflist[counter] = cfdict[key]
        counter += 1

    data = np.array(cflist)

    data_shape = np.shape(data)

    # Take negative and positive data apart and cumulate
    def get_cumulated_array(data, **kwargs):
        cum = data.clip(**kwargs)
        cum = np.cumsum(cum, axis=0)
        d = np.zeros(np.shape(data))
        d[1:] = cum[:-1]
        return d

    cumulated_data = get_cumulated_array(data, min=0)
    cumulated_data_neg = get_cumulated_array(data, max=0)

    # Re-merge negative and positive data.
    row_mask = (data<0)
    cumulated_data[row_mask] = cumulated_data_neg[row_mask]
    data_stack = cumulated_data

    width = 0.5

    fig, ax = plt.subplots(1,1, figsize=(14, 7))


    cfkeylist = list(cfdict.keys())

    for i in np.arange(0, data_shape[0]):
        ax.bar(years, data[i], bottom=data_stack[i], label=cfkeylist[i])

    # Shrink current axis's height by 10% on the bottom
    box = ax.get_position()
    ax.set_position([box.x0, box.y0 + box.height * 0.1,
                     box.width, box.height * 0.9])

    ax.get_yaxis().set_major_formatter(matplotlib.ticker.FuncFormatter(lambda x, p:
format(int(x/1000), ',')))
    ax.set_ylabel("USD in 000s", fontsize="12")
    ax.set_xlabel('Year', fontsize="12")

    ax.set_title("Cash Flow Composition")
    ax.legend(ncol=2, loc="upper center", bbox_to_anchor=(0.5, -0.1),
              fancybox=True, fontsize="10")



    #plt.show()

    memfile = BytesIO()

    # memlist = [' ']*2
    # memlist[0] = liqmemfile
    plt.savefig(memfile, bbox_inches="tight")

    plt.close(fig)

    return memfile
```

```
# #### 4* create document and add selected visualizations to it

# In[67]:


def createCharts(ac,fr,at,df,acclist):

    memlist = [0]*10
    assetfile = ''
    liabfile = ''
    aleisfile = ''
    ilefile = ''
    cffile = ''
    pmfile = ''
    liqfile = ''
    solvfile = ''
    trendfile = ''


    tbdf = df
    years = tbdf.columns[3:].tolist()

    if ac:
        # assets composition
        cadf = tbdf[(tbdf['acct_key'] ==
'CURRENT_ASSET')].iloc[:,2:].set_index('acct_name').T
        ncadf = tbdf[(tbdf['acct_key'] ==
'NONCURRENT_ASSET')].iloc[:,2:].set_index('acct_name').T

        str1 = 'Current Assets Composition'
        str2 = 'Noncurrent Assets Composition'

        assetfile = plotAccountCharts(cadf,ncadf,None,str1,str2,None,'v',2)


        # liabilities composition
        cldf = tbdf[(tbdf['acct_key'] ==
'CURRENT_LIABILITY')].iloc[:,2:].set_index('acct_name').T
        ncldf = tbdf[(tbdf['acct_key'] ==
'NONCURRENT_LIABILITY')].iloc[:,2:].set_index('acct_name').T

        str1 = 'Current Liabilities Composition'
        str2 = 'Noncurrent Liabilities Composition'

        liabfile = plotAccountCharts(cldf,ncldf,None,str1,str2,None, 'v',2)


        # balance sheet and income statement composition
        alelist = ['TOT_ASSET', 'TOT_LIABILITY', 'EQUITY']
        aledf =
tbdf[tbdf['acct_key'].isin(alelist)].iloc[:,2:].set_index('acct_name').T

        islist = ['TOT_REVENUE', 'TOT_COS', 'TOT_OPEX']
        isdf =
tbdf[tbdf['acct_key'].isin(islist)].iloc[:,2:].set_index('acct_name').T

        str1 = 'Balance Sheet Composition'
        str2 = 'Income Statement Composition'

        aleisfile = plotAccountCharts(aledf,isdf,None,str1,str2,None, 'v',2)


        # income and loss and expense composition
        revdf = tbdf[tbdf['acct_key'] == 'REVENUE'].iloc[:,2:]

        oiedf = tbdf[(tbdf['acct_key'] == 'OTHER_INCOME_EXPENSE') ].iloc[:,2:]
        oidf = oiedf.copy()
        oidf[oidf[years] < 0] = 0

        explist = ['COS', 'OPEX']
        expdf = tbdf[tbdf['acct_key'].isin(explist)].iloc[:,2:]

        gaindf = expdf.copy()

        expdf[expdf[years] < 0] = 0
        gaindf[gaindf[years] > 0] = 0

        gaindf[gaindf.select_dtypes(include=['number']).columns] =
gaindf[gaindf.select_dtypes(include=['number']).columns].abs()

        oedf = oiedf.copy()
        oedf[oedf[years] > 0] = 0
        oedf[oedf.select_dtypes(include=['number']).columns] =
oedf[oedf.select_dtypes(include=['number']).columns].abs()

        revdf = pd.concat([revdf, gaindf, oidf])
        revdf = revdf[(revdf.iloc[:,2:].T != 0).any()].set_index('acct_name').T

        expdf = pd.concat([expdf, oedf])
        expdf = expdf[(expdf.iloc[:,2:].T != 0).any()].set_index('acct_name').T

        str1 = 'Income Composition'
        str2 = 'Loss and Expense Composition'

        ilefile = plotAccountCharts(revdf,expdf,None,str1,str2,None, 'v',2)


        cfdf = tbdf[tbdf['acct_key'] == 'CF'].iloc[:,2:].set_index('acct_name').T

        cffile = plotCF(cfdf, years)
```

```
    if fr:

        ## profit margin ratios
        pmlist = ['TOT_REVENUE', 'TOT_COS', 'TOT_OPEX']
        pmdf = tbdf[tbdf['acct_key'].isin(pmlist)].iloc[:,2:]
        # operating margin
        opmargindf = pmdf.set_index('acct_name')
        # gross margin
        gpmargindf = pmdf.set_index('acct_name').loc[['Total Revenue','Total Cost
of Revenues'],:]

        #Total sum per row:
        opmargindf.loc['Total',:] = opmargindf.sum(axis=0)
        gpmargindf.loc['Total',:] = gpmargindf.sum(axis=0)

        # create new rows for each original row as a percent of the new total row
        opmargindf.loc['Revenue Percent',:] = opmargindf.loc['Total
Revenue',:]/opmargindf.loc['Total',:]*100
        opmargindf.loc['COS Percent',:] = opmargindf.loc['Total Cost of
Revenues',:]/opmargindf.loc['Total',:]*100
        opmargindf.loc['OPEX Percent',:] = opmargindf.loc['Total Operating
Expenses',:]/opmargindf.loc['Total',:]*100

        gpmargindf.loc['Revenue Percent',:] = gpmargindf.loc['Total
Revenue',:]/gpmargindf.loc['Total',:]*100
        gpmargindf.loc['COS Percent',:] = gpmargindf.loc['Total Cost of
Revenues',:]/gpmargindf.loc['Total',:]*100

        # just keep percent rows
        opmargindf = opmargindf.loc[['Revenue Percent','COS Percent','OPEX
Percent'],:].T

        gpmargindf = gpmargindf.loc[['Revenue Percent','COS Percent'],:].T

        str1 = 'Operating Margin Ratio'
        str2 = 'Gross Margin Ratio'

        pmfile = plotAccountCharts(opmargindf,gpmargindf,None,str1,str2,None,
'h',2)


        ## Liquidity ratios

        # current ratio
        currentdf = tbdf[(tbdf['acct_key'] == 'TOT_CURRENT_ASSET') |
(tbdf['acct_key'] == 'TOT_CURRENT_LIABILITY')].iloc[:,2:].set_index('acct_name')
        currentdf.loc['Total',:] = currentdf.sum(axis=0)
        currentdf = currentdf.div(currentdf.loc['Total',:])*100
        currentdf = currentdf.loc[~currentdf.index.isin(['Total'])].T
        # quick ratio
        quickdf = tbdf[(tbdf['acct_key'] == 'CURRENT_ASSET') | (tbdf['acct_key'] ==
'TOT_CURRENT_LIABILITY')].iloc[:,2:].set_index('acct_name')
        quickdf = quickdf.loc[~quickdf.index.isin(['Inventory','Prepaid Expenses
and Other Current Assets'])]
        quickdf.loc['Total',:] = quickdf.sum(axis=0)
        quickdf = quickdf.div(quickdf.loc['Total',:])*100
        quickdf = quickdf.loc[~quickdf.index.isin(['Total'])].T

        str1 = 'Current Ratio'
        str2 = 'Quick Ratio'

        liqfile = plotAccountCharts(currentdf,quickdf,None,str1,str2,None, 'h',2)


        ## Solvency ratios

        aedf = tbdf[(tbdf['acct_key'] == 'TOT_ASSET') | (tbdf['acct_key'] ==
'EQUITY')].iloc[:,2:].set_index('acct_name')
        aedf.loc['Total',:] = aedf.sum(axis=0)
        aedf = aedf.div(aedf.loc['Total',:])*100
        aedf = aedf.loc[~aedf.index.isin(['Total'])].T

        aldf = tbdf[(tbdf['acct_key'] == 'TOT_ASSET') | (tbdf['acct_key'] ==
'TOT_LIABILITY')].iloc[:,2:].set_index('acct_name')
        aldf.loc['Total',:] = aldf.sum(axis=0)
        aldf = aldf.div(aldf.loc['Total',:])*100
        aldf = aldf.loc[~aldf.index.isin(['Total'])].T

        dedf = tbdf[(tbdf['acct_key'] == 'TOT_LIABILITY') | (tbdf['acct_key'] ==
'EQUITY')].iloc[:,2:].set_index('acct_name')
        dedf.loc['Total',:] = dedf.sum(axis=0)
        dedf = dedf.div(dedf.loc['Total',:])*100
        dedf = dedf.loc[~dedf.index.isin(['Total'])].T

        str1 = 'Equity-Assets Ratio'
        str2 = 'Debt-Assets Ratio'
        str3 = 'Debt-Equity Ratio'

        solvfile = plotAccountCharts(aedf,aldf,dedf,str1,str2,str3,'h',3)


    if at:

        trendfile = plotAccountGraph(df, acclist)

    document= Document()
    sections = document.sections
    for section in sections:
        section.top_margin = Cm(1)
        section.bottom_margin = Cm(1)
        section.left_margin = Cm(2)
        section.right_margin = Cm(2)

    if ac:

        document.add_heading("Account Composition Charts")

        p1 = document.add_paragraph("Asset Composition")
        p1.style = document.styles['Normal']
        r1 = p1.add_run()
        r1.add_picture(assetfile, width = Inches(7.5))

        p2 = document.add_paragraph("Liabilities Composition")
        p2.style = document.styles['Normal']
        r2 = p2.add_run()
        r2.add_picture(liabfile, width = Inches(7.5))

        p3 = document.add_paragraph("Balance Sheet and Income Statement
Composition")
        p3.style = document.styles['Normal']
        r3 = p3.add_run()
        r3.add_picture(aleisfile, width = Inches(7.5))

        p4 = document.add_paragraph("Income and Loss and Expense Composition")
        p4.style = document.styles['Normal']
        r4 = p4.add_run()
        r4.add_picture(ilefile, width = Inches(7.5))

        p5 = document.add_paragraph("Cash Flow Composition")
        p5.style = document.styles['Normal']
        r5 = p5.add_run()
        r5.add_picture(cffile, width = Inches(7.5))

    if fr:

        if ac:
            document.add_page_break()

        document.add_heading("Financial Ratio Visualizations")

        p6 = document.add_paragraph("Profit Margin Ratios")
        p6.style = document.styles['Normal']
        r6 = p6.add_run()
        r6.add_picture(pmfile, width = Inches(7.5))

        p7 = document.add_paragraph("Liquidity Ratios")
        p7.style = document.styles['Normal']
        r7 = p7.add_run()
        r7.add_picture(liqfile, width = Inches(7.5))

        p8 = document.add_paragraph("Solvency Ratios")
        p8.style = document.styles['Normal']
        r8 = p8.add_run()
        r8.add_picture(solvfile, width = Inches(7.5))

    if at:

        if ac or fr:
            document.add_page_break()

        document.add_heading("Account Trend(s) Graph")

        p9 = document.add_paragraph("\n")
        p9.style = document.styles['Normal']
        r9 = p9.add_run()
        r9.add_picture(trendfile, width = Inches(7.5))


    #document.save("chartstest.docx")

    return document




# #### 5* create document for financial statements

# In[68]:


def createFS(df, name):

    #create financial statement df with just first and last two columns
    fsdf = df.iloc[:,list(range(0,3,1)) + list(range(-2,0,1))]
    cy = fsdf.iloc[:,list(range(-2,0,1))].columns.tolist()[1]
    py = fsdf.iloc[:,list(range(-2,0,1))].columns.tolist()[0]
    #format last two columns for currency
    fsdf.iloc[:,-1]=fsdf[cy].div(1000).apply('{:,.0f}'.format)
    fsdf.iloc[:,-2]=fsdf[py].div(1000).apply('{:,.0f}'.format)
    # keep only the accounts where both balances are nonzero
    fsdf = fsdf[~((fsdf[cy] == '0') & (fsdf[py] == '0'))]
    # replace 0 strings with '-'
    fsdf.loc[fsdf[cy] == '0', cy] = '-'
    fsdf.loc[fsdf[py] == '0', py] = '-'

    # create sub dictionaries for each multi fs line account group and add to fs
dictionary
```

```
    fsdict = {}
    fsdict['CURRENT_ASSET'] = fsdf.query("acct_key ==
'CURRENT_ASSET'").set_index('fs_key').T.to_dict('list')
    fsdict['NONCURRENT_ASSET'] = fsdf.query("acct_key ==
'NONCURRENT_ASSET'").set_index('fs_key').T.to_dict('list')
    fsdict['CURRENT_LIABILITY'] = fsdf.query("acct_key ==
'CURRENT_LIABILITY'").set_index('fs_key').T.to_dict('list')
    fsdict['NONCURRENT_LIABILITY'] = fsdf.query("acct_key ==
'NONCURRENT_LIABILITY'").set_index('fs_key').T.to_dict('list')
    fsdict['REVENUE'] = fsdf.query("acct_key ==
'REVENUE'").set_index('fs_key').T.to_dict('list')
    fsdict['COS'] = fsdf.query("acct_key ==
'COS'").set_index('fs_key').T.to_dict('list')
    fsdict['OPEX'] = fsdf.query("acct_key ==
'OPEX'").set_index('fs_key').T.to_dict('list')
    fsdict['OTHER_INCOME_EXPENSE'] = fsdf.query("acct_key ==
'OTHER_INCOME_EXPENSE'").set_index('fs_key').T.to_dict('list')
    fsdict['CF'] = fsdf.query("acct_key ==
'CF'").set_index('fs_key').T.to_dict('list')

    # create dictionary for remaining single fs line items
    onefslinedict =
fsdf[~fsdf["acct_key"].isin(fsdict.keys())].copy().set_index('acct_key').T.to_dict(
'list')

    if 'DISC_OPS' not in onefslinedict:
        onefslinedict['DISC_OPS'] = ['fsDO', 'Discontinued Operations', '-', '-']

    if 'INCOME_TAX' not in onefslinedict:
        onefslinedict['INCOME_TAX'] = ['fsITE', 'Income Tax Expense', '-', '-']


    # combine single and mutli fs line dictionaries
    fsdict = fsdict | onefslinedict

    # create additional fs line subtotals
    pyglfromop = float(fsdict['TOT_REVENUE'][2].replace(',','')) -
float(fsdict['TOT_COS'][2].replace(',','')) -
float(fsdict['TOT_OPEX'][2].replace(',',''))
    cyglfromop = float(fsdict['TOT_REVENUE'][3].replace(',','')) -
float(fsdict['TOT_COS'][3].replace(',','')) -
float(fsdict['TOT_OPEX'][3].replace(',',''))
    pyglb4tax = pyglfromop +
float(fsdict['TOTAL_OTHER_INCOME_EXPENSE'][2].replace(',',''))
    cyglb4tax = cyglfromop +
float(fsdict['TOTAL_OTHER_INCOME_EXPENSE'][3].replace(',',''))

    if fsdict['INCOME_TAX'][3].replace(',','').isnumeric():
        cyglfromcop = cyglb4tax - float(fsdict['INCOME_TAX'][3].replace(',',''))
    else:
        cyglfromcop = cyglb4tax

    if fsdict['INCOME_TAX'][2].replace(',','').isnumeric():
        pyglfromcop = pyglb4tax - float(fsdict['INCOME_TAX'][2].replace(',',''))
    else:
        pyglfromcop = pyglb4tax

    # add in additional fs line subtotals
    fsdict['CYGLFROMOP'] = '{:,.0f}'.format(cyglfromop)
    fsdict['PYGLFROMOP'] = '{:,.0f}'.format(pyglfromop)
    fsdict['CYGLB4TAX'] = '{:,.0f}'.format(cyglb4tax)
    fsdict['PYGLB4TAX'] = '{:,.0f}'.format(pyglb4tax)
    fsdict['CYGLFROMCOP'] = '{:,.0f}'.format(cyglfromcop)
    fsdict['PYGLFROMCOP'] = '{:,.0f}'.format(pyglfromcop)

    fsdict['cy'] = 'Fiscal Year '+cy
    fsdict['py'] = 'Fiscal Year '+py

    fsdict['name'] = name

    return fsdict



# #### 6* produce (save) document locally

# In[73]:


def produceDocument(fs,ac,fr,at,df,name,acclist):

    f = ''
    fsdict = {}
    fsdoc = ''
    chartdoc = ''

    if fs:

        fsdict = createFS(df,name)

        fsdoc = DocxTemplate('fs_template.docx')
        fsdoc.render(fsdict)


    if any([ac,fr,at]):

        chartdoc = createCharts(ac,fr,at,df,acclist)
    f = fd.asksaveasfilename(defaultextension=".docx", title="Select name and
path.")
    #print(f)
    if f:

        try:

            if fs and any([ac,fr,at]):

                composer = Composer(fsdoc)
                composer.append(chartdoc)
                composer.save(f)

            elif fs and not any([ac,fr,at]):

                fsdoc.save(f)

            elif any([ac,fr,at]):

                chartdoc.save(f)

            webbrowser.open_new(f)


        except IOError as e:

            messagebox.showinfo(title='ERROR:', message='File to be replaced is
open. Could not save document.\nPlease close and try again. '+str(e))



# #### 7* create and (when called) open EDWARD GUI

# In[70]:


def openEDWARD():

    def selectFile():

        filetypes = (("CSV Files","*.csv"),)

        filename = fd.askopenfilename(
            title='Select Data File',
            initialdir='/',
            filetypes=filetypes)

        if filename:

            messagebox.showinfo(title='Data File:', message=filename)
            global dataFile
            dataFile = filename
            df = pd.read_csv(dataFile).fillna(0)

            listdf = df[df['acct_key'] != 'CF']

            acctList = list(listdf['acct_name'].values)
            acctList.sort()

            account1['values'] = acctList
            account2['values'] = acctList
            account3['values'] = acctList


    def getChoices():

        xmlList = ['\"', "'", '<', '>', '&']

        escape = False

        name = nameEntry.get()

        for i in xmlList:

            if i in name:
                escape = True

        if not name:

            messagebox.showwarning("Name field incomplete:", "Please enter a
company name.")

        elif escape:

            messagebox.showwarning("XML character detected:", "Please do not use
XML characters (\", ', <, >, &).")

        else:

            fs = fsVar.get()
            ac = acVar.get()
            fr = frVar.get()
            at = atVar.get()

            at1 = account1.get()
            at2 = account2.get()
            at3 = account3.get()

            acclist = []
            if at1:
                acclist.append(at1)
            if at2:
                acclist.append(at2)
            if at3:
                acclist.append(at3)

            if at and not any([at1,at2,at3]):

                messagebox.showwarning("No trend account selected:", "If you want
to produce an account trend report\nplease select at least one trend account.")

            else:

                global dataFile
                df = pd.read_csv(dataFile).fillna(0)

                produceDocument(fs,ac,fr,at,df, name, acclist)



    global dataFile

    edWindow = Tk()
    edWindow.geometry('900x850')
    edWindow.config(bg='black')

    edFrame = Frame(edWindow, bg='black', relief='ridge')
    edFrame.grid(row = 0)

    for i in range(12):
        edFrame.grid_rowconfigure(i, weight=1)

    for i in range(3):
        edFrame.grid_columnconfigure(i, weight=1)

    edlogo_img = Image.open("edbgpic.png").resize((710, 185))
    edlogo_tkimg = itk.PhotoImage(edlogo_img)

    edlogo = Label(edFrame, image = edlogo_tkimg, bg='gray', bd=5, relief='sunken')
    edlogo.image = edlogo_tkimg
    edlogo.grid(row = 0, column=0, columnspan=3, pady=5)

    fileButton = Button(edFrame, text = 'Select Data File', command = selectFile,
font=('OCR A Extended',15),
                       activeforeground = 'cyan', activebackground='black',
bg='gray', width = 26)
    fileButton.grid(row = 1, column = 1, pady=20, ipady=3)
    nameLabel = Label(edFrame, text = 'Please enter company name:', width = 24,
font=('System',10), bg='black', fg='white')
    nameLabel.grid(row = 2, column = 1, pady = 5, ipady=2, sticky='s')
    nameEntry = Entry(edFrame, borderwidth=5, relief="ridge", width = 24,
font=('System',10))
    nameEntry.grid(row = 3, column = 1, pady = 10, ipady=2)


    fsVar = IntVar()
    acVar = IntVar()
    frVar = IntVar()
    atVar = IntVar()

    fsButton = Checkbutton(edFrame, text = "Balance Sheet & Income Statement",
                          variable = fsVar,
                          onvalue = 1,
                          offvalue = 0,
                          height = 2,
                          relief='groove',
                          font=('System',10, 'bold'),
                          activeforeground = 'cyan',
                          activebackground='black',
                          bd=5, bg='gray', width = 30, anchor="w")

    acButton = Checkbutton(edFrame, text = "Account Composition Charts",
                          variable = acVar,
                          onvalue = 1,
                          offvalue = 0,
                          height = 2,
                          relief='groove',
                          font=('System',10, 'bold'),
                          activeforeground = 'cyan',
                          activebackground='black',
                          bd=5, bg='gray', width = 30, anchor="w")

    frButton = Checkbutton(edFrame, text = "Financial Ratios",
                          variable = frVar,
                          onvalue = 1,
                          offvalue = 0,
                          height = 2,
                          relief='groove',
                          font=('System',10, 'bold'),
                          activeforeground = 'cyan',
                          activebackground='black',
                          bd=5, bg='gray', width = 30, anchor="w")

    atButton = Checkbutton(edFrame, text = "Account Trend Graphs",
                          variable = atVar,
                          onvalue = 1,
                          offvalue = 0,
                          height = 2,
                          relief='groove',
                          font=('System',10, 'bold'),
                          activeforeground = 'cyan',
                          activebackground='black',
                          bd=5, bg='gray', width = 30, anchor="w")

    fsButton.grid(row = 4, column = 1, pady=5)
    acButton.grid(row = 5, column = 1, pady=5)
    frButton.grid(row = 6, column = 1, pady=5)
    atButton.grid(row = 7, column = 1, pady=5)

    acct1Lab = Label(edFrame, text='Trend Account 1', font=('System',8, 'bold'),
fg='white', bg='black')
    acct1Lab.grid(row = 8, column = 0, pady=3, sticky ='s')
    acct2Lab = Label(edFrame, text='Trend Account 2', font=('System',8, 'bold'),
fg='white', bg='black')
    acct2Lab.grid(row = 8, column = 1, pady=10, sticky ='s')
    acct3Lab = Label(edFrame, text='Trend Account 3', font=('System',8, 'bold'),
fg='white', bg='black')
    acct3Lab.grid(row = 8, column = 2, pady=3, sticky ='s')

    account1 = ttk.Combobox(edFrame, values=[], width = 40)
    account1.grid(row = 9, column = 0, pady=5, sticky ='n')
    account2 = ttk.Combobox(edFrame, values=[], width = 40)
    account2.grid(row = 9, column = 1, pady=5, sticky ='n')
    account3 = ttk.Combobox(edFrame, values=[], width = 40)
    account3.grid(row = 9, column = 2, pady=5, sticky ='n')

    docButton = Button(edFrame, text = 'Produce Document', command = getChoices,
font=('OCR A Extended',15),
                       activeforeground = 'cyan', activebackground='black',
bg='gray', width = 26)
    docButton.grid(row = 10, column = 1, pady=20, ipady=3)

    closeButton = Button(edFrame, text = 'Close E.D.W.A.R.D.', command =
edWindow.destroy, font=('OCR A Extended',15),
                         activeforeground = 'cyan', activebackground='black',
bg='gray', width = 26)
    closeButton.grid(row = 11, column = 1, pady=15, ipady=3)

    edWindow.grid_rowconfigure(0, weight=1)
    edWindow.grid_columnconfigure(0, weight=1)

    edWindow.title('E.D.W.A.R.D.')
    edWindow.mainloop()


# ### 8* Open E.D.W.A.R.D.
# In[74]:


openEDWARD()
```