

```
import streamlit as st
import pandas as pd
import plotly.express as px
import plotly.graph_objects as go
from datetime import datetime
from supabase import create_client
import openai

# ページ設定(この行はスクリプト最上部でなければならない)
st.set_page_config(page_title="財務ダッシュボード", layout="wide")

# タイトル表示
st.title("📊 財務データアップロード & 可視化アプリ")

# Supabase 接続設定
@st.cache_resource
def init_supabase():
    url = st.secrets["supabase"]["url"]
    key = st.secrets["supabase"]["key"]
    return create_client(url, key)

supabase = init_supabase()
openai.api_key = st.secrets["openai"]["api_key"]

# 定数
TABLE_NAME = "monthly_pl"
DATE_COLUMN = "date"
SALES_COLUMN = "sales"

# GPT による売上アドバイス
def generate_sales_advice(df: pd.DataFrame, sales_col: str):
    if df.empty:
        return "データが存在しないため、分析できません。"

    csv_data = df[['year_month', sales_col]].to_csv(index=False)
    prompt = f"""

あなたは財務分析の専門家です。
以下の CSV データは、企業の 3 年分の月次売上データです。
売上の傾向、注意点、改善アドバイスを 500 文字以内で日本語で要約してください。

データ:
{csv_data}
"""

    response = openai.Completion.create(
        engine="text-davinci-003",
        prompt=prompt,
        max_tokens=500,
        n=1,
        stop=None,
        temperature=0.5
    )
    return response.choices[0].text
```

```

"""
try:
    response = openai.ChatCompletion.create(
        model="gpt-4",
        messages=[{"role": "user", "content": prompt}],
        temperature=0.7,
        max_tokens=500
    )
    return response.choices[0].message.content.strip()
except Exception as e:
    return f"GPT API エラー: {e}"

# Supabase からデータ取得
@st.cache_data(ttl=600)
def fetch_supabase_data():
    try:
        res = supabase.table(TABLE_NAME).select("*").order(DATE_COLUMN,
desc=False).execute()
        return pd.DataFrame(res.data)
    except Exception as e:
        st.error(f"X Supabase 取得エラー: {e}")
        return pd.DataFrame()

# データ処理関数
def process_data(df):
    if df.empty:
        return df
    df[DATE_COLUMN] = pd.to_datetime(df[DATE_COLUMN], errors="coerce")
    df[SALES_COLUMN] = pd.to_numeric(df[SALES_COLUMN], errors="coerce")
    df.dropna(subset=[DATE_COLUMN, SALES_COLUMN], inplace=True)
    df["year"] = df[DATE_COLUMN].dt.year
    df["year_month"] = df[DATE_COLUMN].dt.strftime('%Y-%m')
    latest_year = df["year"].max()
    return df[df["year"].isin([latest_year, latest_year - 1, latest_year - 2])]

# ローカル CSV アップロードと可視化
st.title("📁 ローカル CSV での財務分析")
uploaded_local = st.file_uploader("ローカル CSV をアップロード", type=["csv"], key="local_csv")
if uploaded_local:
    df = pd.read_csv(uploaded_local)
    df["date"] = pd.to_datetime(df["date"], errors='coerce')
    df.dropna(subset=["date"], inplace=True)

```

```

df["変動費"] = df["outsourcing_costs"] + df["commissions_fees"]

固定費列 = [
    "executive_compensation", "salaries", "bonuses", "rent_payments",
    "welfare_expenses", "employee_welfare", "supplies",
    "utilities", "communication", "transportation", "communication_expenses",
    "advertising", "entertainment", "training_expenses", "miscellaneous_expenses"
]
df["固定費"] = df[[col for col in 固定費列 if col in df.columns]].sum(axis=1)

if "gross_profit" in df.columns:
    df["粗利益"] = df["gross_profit"]
elif "cost_of_sales" in df.columns:
    df["粗利益"] = df["sales"] - df["cost_of_sales"]
else:
    df["粗利益"] = df["sales"] - df["変動費"] - df["固定費"]

df["経常利益"] = df["ordinary_profit"] if "ordinary_profit" in df.columns else df["粗利益"]

df_summary = df[["date", "sales", "変動費", "固定費", "粗利益", "経常利益"]].copy()
df_summary = df_summary.rename(columns={"sales": "売上高"})

# 📊 財務指標別 三期比較グラフ(1月はじまり)
st.subheader("📊 財務指標別 三期比較グラフ(1月はじまり)")

selected_metric = st.selectbox("表示する財務指標を選んでください", ["売上高", "変動費", "固定費", "粗利益", "経常利益"])

df_summary["月ラベル"] = pd.to_datetime(df_summary["date"]).dt.strftime("%m月")
df_summary["年度"] = pd.to_datetime(df_summary["date"]).dt.year

latest_year = df_summary["年度"].max()
target_years = [latest_year - 2, latest_year - 1, latest_year]
df_summary = df_summary[df_summary["年度"].isin(target_years)]

year_label_map = {
    target_years[0]: "前々期(R4)",
    target_years[1]: "前期(R5)",
    target_years[2]: "当期(R6)",
}

```

```

df_summary["年度ラベル"] = df_summary["年度"].map(year_label_map)

month_order = [f"{i:02d}月" for i in range(1, 13)]
df_summary["月ラベル"] = pd.Categorical(df_summary["月ラベル"], categories=month_order,
ordered=True)

df_plot = df_summary[["月ラベル", "年度ラベル", selected_metric]].dropna()
df_plot = df_plot.sort_values(["月ラベル", "年度ラベル"])

df_pivot = df_plot.pivot(index="月ラベル", columns="年度ラベル",
values=selected_metric).reset_index()

fig = go.Figure()
colors = {
    "前々期(R4)": "darkgreen",
    "前期(R5)": "deepskyblue",
    "当期(R6)": "lightgreen"
}

for col in ["前々期(R4)", "前期(R5)", "当期(R6)"]:
    if col in df_pivot.columns:
        fig.add_trace(go.Bar(
            x=df_pivot["月ラベル"],
            y=df_pivot[col],
            name=col,
            text=df_pivot[col].apply(lambda x: f"{x:.0f}" if pd.notnull(x) else ""),
            textposition="auto",
            marker_color=colors[col]
        ))
    )

fig.update_layout(
    title=f"{selected_metric} 三期比較グラフ(1月～12月)",
    xaxis_title="月",
    yaxis_title="金額(円)",
    barmode="group",
    height=550,
    legend_title="年度"
)
st.plotly_chart(fig, use_container_width=True)

```