**Streamlit SOP**

This explanation will help us understand the purpose and functionality of each part of the code.

**Sidebar Navigation and Data Initialization**

1. Lines 1-5

python

import streamlit as st

import pandas as pd

import matplotlib.pyplot as plt

Sidebar for navigation

sidebar\_option = st.sidebar.selectbox('Select the Page', ['Regional Summary','Client Plan Coverage Today','Reviewing Revenue and RWA Growth','Client Plan Alignment to Strategic Priorities'])

Line 1: Import the Streamlit library, which is essential for creating the web application.

Line 2: Import the Pandas library for data manipulation.

Line 3: Import Matplotlib for plotting graphs.

Line 4: Create a sidebar with a select box for page navigation. The user can choose between different pages.

Page 1: Regional Summary

2. Lines 6-22

python

if sidebar\_option == 'Regional Summary':

data = {

"Metric": ["NDI $mm", "Credit RWA $mm", "Total RWA $mm"],

"2023 Actual": [200, 500, 600],

"2024 Projection": [180, 450, 500]

}

df = pd.DataFrame(data)

df['YoY %'] = ((df['2024 Projection'] - df['2023 Actual']) / df['2023 Actual']) \* 100

st.title("Client Plan Forecasts Through 2024")

st.subheader("Regional Summary")

st.markdown(" Trend ")

col1, col2, col3 = st.columns(3)

Line 6: Check if the selected page is 'Regional Summary'.

Lines 7-12: Create a dictionary with metrics and their respective actual and projected values.

Line 14: Convert the dictionary into a DataFrame.

Line 15: Calculate the Year-over-Year (YoY) percentage change.

Lines 17-19: Display the title, subheader, and a markdown text on the Streamlit app.

Line 21: Create three columns in the Streamlit layout.

3. Lines 23-29

python

with col1:

st.dataframe(df)

with col2:

for index, row in df.iterrows():

st.write(f"{row['Metric']}: {row['YoY %']:.2f}%")

with col3:

fig, ax = plt.subplots()

ax.plot(df['Metric'], df['2023 Actual'], label='2023 Actual')

ax.plot(df['Metric'], df['2024 Projection'], label='2024 Projection')

ax.set\_title('Metrics Comparison')

ax.set\_xlabel('Metric')

ax.set\_ylabel('$mm')

ax.legend()

st.pyplot(fig)

Line 23: Use the first column to display the DataFrame.

Lines 25-27: Use the second column to display the YoY percentage change for each metric.

Lines 29-38: Use the third column to plot the actual and projected values using Matplotlib, then display the plot in Streamlit.

Page 2: Client Plan Coverage Today

4. Lines 40-68

python

elif sidebar\_option == 'Client Plan Coverage Today':

st.title("Client Plan Coverage Today")

st.subheader("33% of our clients are covered by a valid plan however these reflect almost 85% of Revenues and RWAs in the entire portfolio")

df\_page2 = pd.read\_excel('Report1.xlsx')

client\_counts = df\_page2['category'].value\_counts().to\_dict()

grouped\_df\_page2 = df\_page2.groupby('Category').agg({'Client Name': 'count', 'NDI': 'sum', 'Credit RWA': 'sum', 'Total RWA': 'sum'}).reset\_index()

grouped\_df\_page2.rename(columns={"Client Name": "Clients"}, inplace=True)

totals = pd.DataFrame({

'Category': ['All (Active Clients)'],

'Clients': [df\_page2['Client Name'].count()],

'NDI': [df\_page2['NDI'].sum()],

'Credit RWA': [df\_page2['Credit RWA'].sum()],

'Total RWA': [df\_page2['Total RWA'].sum()]

})

summary\_df = pd.concat([grouped\_df\_page2, totals], ignore\_index=True)

summary\_df = summary\_df.round(2)

st.dataframe(summary\_df)

fig\_page2, ax = plt.subplots(figsize=(4,3))

categories = summary\_df['Category'].tolist()

values = summary\_df['Clients'].tolist()

colors\_list = ['#FFBB78', '#FF9896', '#98DF8A', '#D62728']

if len(categories) > len(colors\_list):

blank = [0] \* (len(categories) - len(colors\_list))

colors\_list.extend(colors\_list)

values.extend(blank)

categories.extend(blank)

ax.barh(categories, values, color=colors\_list[:len(categories)])

for i in range(len(values)):

ax.text(values[i], i, values[i], ha='center', va='center')

ax.set\_title("Client Plan Coverage")

ax.set\_xlabel("Number of Clients")

st.pyplot(fig\_page2)

Line 40: Check if the selected page is 'Client Plan Coverage Today'.

Lines 41-42: Display the title and a subheader on the Streamlit app.

Line 44: Read data from an Excel file into a DataFrame.

Line 45: Count the number of clients in each category.

Line 46: Group data by category and aggregate the sum of NDI, Credit RWA, and Total RWA.

Line 47: Rename the column 'Client Name' to 'Clients'.

Lines 49-54: Create a DataFrame for the total of all active clients.

Lines 56-57: Concatenate the grouped DataFrame with the totals DataFrame and round the values to two decimal places.

Line 59: Display the summary DataFrame in the Streamlit app.

Lines 61-62: Create a horizontal bar chart using Matplotlib.

Lines 64-65: Set the chart's title and x-axis label, then display the plot in Streamlit.

Page 3: Reviewing Revenue and RWA Growth

5. Lines 70-118

```python

elif sidebar\_option == 'Reviewing Revenue and RWA Growth':

st.title("Reviewing Revenue and RWA Growth")

st.subheader("Review of revenue and RWA growth")

df\_page3 = pd.read\_excel('data.xlsx')

yoy\_ndi = (df\_page3['NDI\_Last\_Year'] - df\_page3['NDI\_Last\_Year']) / df\_page3['NDI\_Last\_Year'] \* 100

yoy\_credit\_rwa = (df\_page3['Credit\_RWA\_Last\_Year'] - df\_page3['Credit\_RWA\_Last\_Year']) / df\_page3['Credit\_RWA\_Last\_Year'] \* 100

st.write(f"YoY NDI: {yoy\_ndi:.2f}%")

st.write(f"YoY Credit RWA: {yoy\_credit\_rwa:.2f}%")

increasing\_credit\_rwas = [200, 300, 250]

decreasing\_credit\_rwas = [180, 150, 200]

fig\_page3, ax1 = plt.subplots(1, 2, figsize=(12, 4))

ax1[0].barh(["A", "B", "C"], increasing\_credit\_rwas, color='green', edgecolor='none')

ax1[0].text(increasing\_credit\_rwas[0] + 20, 0, f"${increasing\_credit\_rwas[0]:,}", va='center')

ax1[0].text(increasing\_credit\_rwas[1] + 20, 1, f"${increasing\_credit\_rwas[1]:,}", va='center')

ax1[0].text(increasing\_credit\_rwas[2] + 20, 2, f"${increasing\_credit\_rwas[2]:,}", va='center')

ax1[0].set\_title("$500m incremental revenue from plans with increasing credit RWAs")

ax1[0].set\_xlabel("Credit RWA ($mm)")

ax1[0].set\_yticklabels(["A", "B", "C"])

ax1[1].barh(["A", "B", "C"], decreasing\_credit\_rwas, color='red', edgecolor='none')

ax1[1].text(decreasing\_credit\_rwas[0] + 20, 0, f"${decreasing\_credit\_rwas[0]:,}", va='center')

ax1[1].text(decreasing\_credit\_rwas[1] + 20, 1, f"${decreasing\_credit\_rwas[1]:,}", va='center')

ax1[1].text(decreasing\_credit\_rwas[2] + 20, 2, f"${decreasing\_credit\_rwas[2]:,}", va='center')

ax1[1].set\_title("$350m incremental revenue from plans with decreasing credit RWAs")

ax1[1].set\_xlabel("Credit RWA ($mm)")

ax1[1].set\_yt

st.pyplot(fig\_page3)

Line 70 : Check if the selected page is 'Reviewing Revenue and RWA Growth'.

Lines 71-72 : Display the title and a subheader on the Streamlit app.

Line 74 : Read data from an Excel file into a DataFrame.

Lines 75-76 : Calculate the YoY change for NDI and Credit RWA.

- Lines 78-79 : Display the YoY changes in the Streamlit app.

- Lines 81-82 : Define the data for increasing and decreasing credit RWAs.

- Line 84 : Create a subplot with two horizontal bar charts.

- Lines 86-90 : Create the first bar chart for increasing credit RWAs and annotate the bars.

- Lines 92-93 : Set the title and x-axis label for the first bar chart.

- Lines 95-99 : Create the second bar chart for decreasing credit RWAs and annotate the bars.

- Lines 101-102 : Set the title and x-axis label for the second bar chart.

- Line 104 : Display the subplot in the Streamlit app.

# Page 4: Client Plan Alignment to Strategic Priorities

6. Lines 120-142

python

elif sidebar\_option == 'Client Plan Alignment to Strategic Priorities':

st.title("Client Plan Alignment to Strategic Priorities")

st.subheader("Review of how client plans align with strategic priorities")

data = {

"Priority": ["Priority 1", "Priority 2", "Priority 3"],

"Clients Aligned": [10, 20, 15]

}

df\_page4 = pd.DataFrame(data)

st.dataframe(df\_page4)

fig\_page4, ax4 = plt.subplots()

ax4.bar(df\_page4['Priority'], df\_page4['Clients Aligned'], color=['blue', 'green', 'red'])

ax4.set\_title('Client Plan Alignment to Strategic Priorities')

ax4.set\_xlabel('Priority')

ax4.set\_ylabel('Clients Aligned')

st.pyplot(fig\_page4)

```

Line 120: Check if the selected page is 'Client Plan Alignment to Strategic Priorities'.

Lines 121-122: Display the title and a subheader on the Streamlit app.

Lines 124-127: Create a dictionary with priorities and the number of clients aligned to each priority.

Line 129: Convert the dictionary into a DataFrame.

Line 131: Display the DataFrame in the Streamlit app.

Lines 133-137: Create a bar chart to visualize the alignment of client plans to strategic priorities and display the plot in Streamlit.