**1. Setting Up the UI**

Use Qt Designer to design the UI with the following components:

* 4 QPushButtons (Bar Chart, Line Chart, Pie Chart, Exit)
* 1 QVBoxLayout (verticalLayoutPlot) for displaying charts
* Menu bar and Status bar

**2. Integrating Matplotlib & Seaborn**

In MainWindowEx.py, import the required libraries:

import sys

import pandas as pd

import seaborn as sns

from matplotlib import pyplot as plt

from matplotlib.backends.backend\_qt5agg import FigureCanvasQTAgg as FigureCanvas

from matplotlib.backends.backend\_qt5agg import NavigationToolbar2QT as NavigationToolbar

from MainWindow import Ui\_MainWindow

**FigureCanvasQTAgg**: Converts Matplotlib figures into PyQt6 widgets.

**NavigationToolbar2QT**: Adds navigation controls for zooming, panning, and saving plots.

**3. Implementing the Chart Display**

Create a MainWindowEx class that extends Ui\_MainWindow:

class MainWindowEx(Ui\_MainWindow):

def \_\_init\_\_(self):

pass

def setupUi(self, MainWindow):

super().setupUi(MainWindow)

self.MainWindow = MainWindow

self.setupPlot() # Initialize the chart display area

# Connect buttons to their respective functions

self.pushButtonBarChart.clicked.connect(self.showBarChart)

self.pushButtonLineChart.clicked.connect(self.showLinePlotChart)

self.pushButtonPieChart.clicked.connect(self.showPieChart)

self.pushButtonExit.clicked.connect(self.processExit)

def show(self):

self.MainWindow.show()

**4. Configuring the Chart Display Area**

def setupPlot(self):

self.figure = plt.figure() # Create an empty figure

self.canvas = FigureCanvas(self.figure) # Convert figure into a PyQt6 widget

self.toolbar = NavigationToolbar(self.canvas, self.MainWindow) # Add a navigation toolbar

# Add the toolbar and canvas to the layout

self.verticalLayoutPlot.addWidget(self.toolbar)

self.verticalLayoutPlot.addWidget(self.canvas)

**plt.figure():** Creates an empty Matplotlib figure.

**FigureCanvas:** Converts the figure into a PyQt6-compatible widget.

**NavigationToolbar**: Enables zoom, pan, and save functions inside the UI.

**5. Implementing Chart Display Functions**

**Displaying a Bar Chart**

def showBarChart(self):

df = pd.read\_csv('./data/NetProfit.csv') # Load dataset

self.figure.clear() # Clear previous content

ax = self.figure.add\_subplot(111) # Create a subplot

ax.ticklabel\_format(useOffset=False, style="plain") # Format axis labels

ax.grid() # Enable grid for better readability

ax.bar(df["Year"], df["VNM"]) # Plot bar chart

ax.set\_title("Bar Chart") # Set title

self.canvas.draw() # Render the new plot

.**clear():** Ensures old plots don’t overlap with new ones.

**.add\_subplot(111):** Creates a subplot inside the figure.

**.draw():** **Very important! Without this, the new plot won't update.**

**Displaying a Line Chart:**

def showLinePlotChart(self):

df = pd.read\_csv('./data/NetProfit.csv') # Load dataset

self.figure.clear()

ax = self.figure.add\_subplot(111) # Create a subplot

ax.ticklabel\_format(useOffset=False, style="plain")

ax.grid()

sns.lineplot(data=df, x="Year", y="VNM", marker='o', color='orange') # Plot a line chart

ax.set\_xlabel("Year")

ax.set\_ylabel("VNM")

ax.set\_title("Seaborn Line Plot")

ax.legend(loc='lower right')

self.canvas.draw()