**For my Portfolio**

*(In the end of this notes, have given the Prompts used to get answer from ChatGPT Plus)*

**Problem Statement:**

"Design and implement an **Interactive Stock Portfolio Dashboard** that enables users to analyze the performance of their stock portfolio by integrating real-time data, historical trends, and financial metrics. The application should provide interactive visualizations, allow for dynamic input of stock tickers and date ranges, and offer actionable insights for portfolio optimization and decision-making."

*If someone gives you the above problem statement, your portfolio format should be structured to reflect a professional and organized approach to solving the problem. Here's a recommended structure:*

**Portfolio Format**

**1. Introduction**

* Restate the problem statement briefly.
* Explain the purpose of the project and its significance in the real world.
* Mention the tools and technologies you will use (e.g., Python, Streamlit, yfinance).

**2. Objective**

* Clearly define the key objectives of the project.
  + Example: "The goal is to provide an interactive dashboard where users can analyze stock performance, visualize trends, and gain insights into portfolio performance."

**3. Approach / Thought Process**

* Outline the step-by-step thought process for solving the problem. This helps showcase your planning and problem-solving skills.
  + **Understanding the Problem**: What does the user need? (E.g., real-time stock prices, performance analysis, etc.)
  + **Identifying the Features**: List the desired features. For instance:
    - Input for stock tickers.
    - Historical price trends visualization.
    - Portfolio performance metrics.
    - Real-time data updates.
  + **Data Source**: Mention the reliance on yfinance for fetching data.
  + **User Interface Design**: Detail how Streamlit will help create an intuitive and interactive UI.

**4. Workflow**

* **Step 1**: Define the Inputs
  + User inputs (stock tickers, date range, etc.).
* **Step 2**: Fetch Data
  + Use yfinance to retrieve stock data (historical and real-time).
* **Step 3**: Process Data
  + Perform data cleaning, calculations (e.g., returns, averages), and transformations.
* **Step 4**: Visualize Data
  + Use Streamlit to create line charts, bar plots, and tables for insights.
* **Step 5**: Build the Dashboard
  + Integrate all components into a cohesive Streamlit application.

**5. Implementation**

* Provide an overview of your coding process, broken into modules:
  + **Data Retrieval**: Code to fetch data using yfinance.
  + **Data Processing**: Data manipulation and calculations.
  + **Visualization**: Code snippets for creating interactive charts.
  + **Integration**: Combining functionality into the Streamlit app.

**6. Key Features**

* List the core features of your application. For example:
  + View historical price trends for multiple stocks.
  + Analyze portfolio performance over a specified date range.
  + Real-time stock price updates.
  + Downloadable reports of the portfolio.

**7. Challenges and Solutions**

* Discuss any challenges you faced (e.g., handling API rate limits, data cleaning) and how you overcame them.

**8. Results**

* Include screenshots or GIFs of your working dashboard.
* Highlight specific use cases, such as analyzing a sample portfolio.

**9. Future Scope**

* Suggest possible extensions or improvements for the project, such as:
  + Adding advanced financial indicators (e.g., RSI, MACD).
  + Incorporating machine learning to predict stock trends.
  + Enabling multi-user access with a login system.

**10. Conclusion**

* Summarize your learnings from the project.
* Reflect on how the project adds value to your portfolio and skillset.

This structure ensures that your portfolio is comprehensive, logical, and showcases both your technical and problem-solving abilities effectively. Let me know if you'd like to explore any specific section in detail!

**Interactive Stock Portfolio Dashboard**

In today’s dynamic financial markets, investors need tools to monitor and analyze their investments effectively. This project aims to design and develop an **Interactive Stock Portfolio Dashboard** that allows users to input stock tickers, specify a date range, and visualize portfolio performance. By integrating Python, Streamlit, and the yfinance library, the application will provide users with real-time and historical data, helping them make informed decisions.

The dashboard will serve as a practical tool for understanding stock trends, evaluating performance metrics, and identifying key financial indicators. This project not only enhances technical skills but also provides valuable experience in building a data-driven application.

**Objective of the Project**

The primary objective of this project is to create a user-friendly **Interactive Stock Portfolio Dashboard** that empowers users to monitor and analyze the performance of their stock investments.

The key goals of the project are:

1. **Provide Historical Analysis**
   * Allow users to input stock tickers and fetch historical price data for a specified time range.
   * Visualize trends and patterns in stock prices over time.
2. **Enable Portfolio Performance Evaluation**
   * Calculate and display key performance metrics such as returns, volatility, and cumulative growth.
   * Compare the performance of multiple stocks in a single view.
3. **Offer Real-Time Insights**
   * Integrate real-time stock prices to keep the portfolio up-to-date.
   * Highlight significant changes and market events affecting the portfolio.
4. **Interactive Visualizations**
   * Use intuitive charts and graphs to represent data in a visually appealing manner.
   * Enable users to interact with the dashboard by dynamically changing inputs such as stock tickers and date ranges.
5. **Enhance Decision-Making**
   * Provide actionable insights through data-driven analysis.
   * Help users identify potential opportunities and risks in their investments.

By achieving these objectives, the project will serve as an essential tool for both novice and experienced investors, demonstrating the practical application of data science in finance.

**Step 1. Understanding the Problem**

The primary problem is to design a dashboard that meets the needs of Indian investors to track and analyze their stock portfolio. It should include:

* Historical price trends for specific stocks listed on the **Indian Stock Market** (e.g., NSE and BSE).
* Real-time stock price updates and portfolio performance metrics.
* An intuitive and interactive interface to make the application user-friendly for diverse audiences.

**Step 2. Identifying the Features**

Based on the problem, the following features will be included:

1. **User Input:**
   * Allow users to input multiple stock tickers (e.g., "TCS", "RELIANCE").
   * Specify the date range for historical data analysis.
2. **Data Retrieval:**
   * Fetch stock data (historical and real-time) for Indian stock market tickers.
   * Calculate metrics like daily returns, cumulative returns, and volatility.
3. **Visualizations:**
   * Line charts for price trends.
   * Bar charts for portfolio allocation and returns.
   * Data tables for detailed stock information.
4. **Insights and Alerts:**
   * Provide summary statistics like average return, highest/lowest price.
   * Highlight significant stock movements or portfolio changes.

**Step 3. Data Source**

To work with Indian Stock Market data, we will rely on the following options:

* **yfinance (Yahoo Finance, we will focus on this source only for this project due to its ease of use and comprehensive data)**:
  + yfinance supports Indian stock tickers listed on NSE and BSE by appending .NS (NSE) or .BO (BSE) to the ticker symbol.
    - Example: "TCS.NS" for Tata Consultancy Services (NSE).
  + Historical and real-time data can be fetched seamlessly using the yfinance API.
* **Alternative Data Sources (is possible, but in this case, we will not be using it):**
  + **NSEpy**: A Python library specifically designed for retrieving NSE stock data.
  + **Alpha Vantage** or **Quandl**: For broader access to stock market data, though free APIs may have limitations.

**Step 4. User Interface Design**

The application will use **Streamlit** to create a clean, intuitive, and interactive user interface. Key design elements include:

1. **Sidebar Inputs:**
   * Input fields for stock tickers and date range.
   * A dropdown for selecting visualization types (e.g., line chart, bar chart).
2. **Dashboard Layout:**
   * **Top Section**: Overview of the portfolio (e.g., total returns, allocation breakdown).
   * **Middle Section**: Interactive visualizations (e.g., stock price trends, performance comparison).
   * **Bottom Section**: Detailed data tables and insights.
3. **Interactivity:**
   * Real-time updates when users change inputs.
   * Downloadable reports for portfolio performance analysis.

**Code, but, without Interactive Graphs version 1**

**import streamlit as st**

**import yfinance as yf**

**import pandas as pd**

**import matplotlib.pyplot as plt**

**# Set Streamlit Page Configuration**

**st.set\_page\_config(page\_title="Stock Portfolio Dashboard", layout="wide")**

**# Dashboard Title**

**st.title("Stock Portfolio Analysis Dashboard")**

**st.write("Analyze the performance of stocks from the Indian stock market with daily and cumulative returns.")**

**# Input Section**

**st.sidebar.header("User Inputs")**

**tickers = st.sidebar.text\_input("Enter stock tickers (comma-separated):", "TCS.NS, INFY.NS")**

**start\_date = st.sidebar.date\_input("Start Date", value=pd.to\_datetime("2023-01-01"))**

**end\_date = st.sidebar.date\_input("End Date", value=pd.to\_datetime("2023-12-31"))**

**# Fetch and Process Data**

**if st.sidebar.button("Analyze"):**

**if tickers:**

**tickers\_list = [ticker.strip() for ticker in tickers.split(",")]**

**stock\_data = {}**

**processed\_data = {}**

**st.sidebar.write("Fetching data...")**

**for ticker in tickers\_list:**

**try:**

**# Fetch Data**

**data = yf.download(ticker, start=start\_date, end=end\_date)**

**if 'Adj Close' in data.columns:**

**data['Price'] = data['Adj Close']**

**elif 'Close' in data.columns:**

**data['Price'] = data['Close']**

**else:**

**st.warning(f"No valid price data found for {ticker}. Skipping.")**

**continue**

**# Process Data**

**data['Daily Return'] = data['Price'].pct\_change()**

**data['Cumulative Return'] = (1 + data['Daily Return']).cumprod()**

**stock\_data[ticker] = data**

**processed\_data[ticker] = data**

**except Exception as e:**

**st.error(f"Error fetching or processing data for {ticker}: {e}")**

**# Visualization**

**if processed\_data:**

**st.header("Stock Performance")**

**for ticker, data in processed\_data.items():**

**st.subheader(ticker)**

**# Price Trend Plot**

**fig, ax = plt.subplots(figsize=(10, 5))**

**ax.plot(data.index, data['Price'], label='Price')**

**ax.set\_title(f"{ticker} Price Trend")**

**ax.set\_xlabel("Date")**

**ax.set\_ylabel("Price (INR)")**

**ax.legend()**

**st.pyplot(fig)**

**# Cumulative Return Plot**

**fig, ax = plt.subplots(figsize=(10, 5))**

**ax.plot(data.index, data['Cumulative Return'], label='Cumulative Return', color='green')**

**ax.set\_title(f"{ticker} Cumulative Return")**

**ax.set\_xlabel("Date")**

**ax.set\_ylabel("Cumulative Return")**

**ax.legend()**

**st.pyplot(fig)**

**else:**

**st.warning("Please enter valid stock tickers.")**

**# Footer**

**st.sidebar.write("Developed by Manas Bhise - Stock Portfolio Analysis")**

**How It Works**

1. **Defining Inputs:**
   * Users input stock tickers and specify date ranges in the Streamlit sidebar.
2. **Fetching Data:**
   * The app fetches stock data using yfinance for the specified tickers and dates.
3. **Processing Data:**
   * Calculates daily returns and cumulative returns for each stock.
4. **Visualizing Data:**
   * Displays interactive charts showing historical prices and portfolio performance.
5. **Building the Dashboard:**
   * Integrates all elements into a Streamlit app for an interactive experience.

**What’s Included in the Dashboard?**

1. **Title and Description:**
   * A clear introduction to the dashboard functionality.
2. **User Inputs:**
   * A sidebar for entering stock tickers, start date, and end date.
   * Button to trigger the analysis.
3. **Data Fetching and Processing:**
   * Uses yfinance to fetch stock data.
   * Calculates daily returns and cumulative returns.
4. **Visualization:**
   * Displays two plots for each stock:
     + Price Trend.
     + Cumulative Return Trend.
5. **Interactive Elements:**
   * Real-time analysis based on user inputs.
6. **Error Handling:**
   * Skips tickers with invalid data and provides warnings.

Following modifications done and code updated to make the graphs interactive using matplotlib and streamlit-multiapp library to allow users to zoom in and interact with the plots.

**Steps to Make Graphs Interactive:**

1. Use st.plotly\_chart to integrate Plotly-based interactive plots.
2. Allow users to zoom in and pan around using Plotly's interactive features.

**Updated Code to Add Interactivity, version 2**

import streamlit as st

import yfinance as yf

import pandas as pd

import matplotlib.pyplot as plt

import plotly.graph\_objects as go

# Set Streamlit Page Configuration

st.set\_page\_config(page\_title="Stock Portfolio Dashboard", layout="wide")

# Dashboard Title

st.title("Stock Portfolio Analysis Dashboard")

st.write("Analyze the performance of stocks from the Indian stock market with daily and cumulative returns.")

# Input Section

st.sidebar.header("User Inputs")

tickers = st.sidebar.text\_input("Enter stock tickers (comma-separated):", "TCS.NS, INFY.NS")

start\_date = st.sidebar.date\_input("Start Date", value=pd.to\_datetime("2023-01-01"))

end\_date = st.sidebar.date\_input("End Date", value=pd.to\_datetime("2023-12-31"))

# Fetch and Process Data

if st.sidebar.button("Analyze"):

if tickers:

tickers\_list = [ticker.strip() for ticker in tickers.split(",")]

stock\_data = {}

processed\_data = {}

st.sidebar.write("Fetching data...")

for ticker in tickers\_list:

try:

# Fetch Data

data = yf.download(ticker, start=start\_date, end=end\_date)

if 'Adj Close' in data.columns:

data['Price'] = data['Adj Close']

elif 'Close' in data.columns:

data['Price'] = data['Close']

else:

st.warning(f"No valid price data found for {ticker}. Skipping.")

continue

# Process Data

data['Daily Return'] = data['Price'].pct\_change()

data['Cumulative Return'] = (1 + data['Daily Return']).cumprod()

stock\_data[ticker] = data

processed\_data[ticker] = data

except Exception as e:

st.error(f"Error fetching or processing data for {ticker}: {e}")

# Visualization

if processed\_data:

st.header("Stock Performance")

for ticker, data in processed\_data.items():

st.subheader(ticker)

# Price Trend Plot

fig, ax = plt.subplots(figsize=(10, 5))

ax.plot(data.index, data['Price'], label='Price')

ax.set\_title(f"{ticker} Price Trend")

ax.set\_xlabel("Date")

ax.set\_ylabel("Price (INR)")

ax.legend()

st.plotly\_chart(fig, use\_container\_width=True)

# Cumulative Return Plot

fig, ax = plt.subplots(figsize=(10, 5))

ax.plot(data.index, data['Cumulative Return'], label='Cumulative Return', color='green')

ax.set\_title(f"{ticker} Cumulative Return")

ax.set\_xlabel("Date")

ax.set\_ylabel("Cumulative Return")

ax.legend()

st.plotly\_chart(fig, use\_container\_width=True)

else:

st.warning("Please enter valid stock tickers.")

# Footer

st.sidebar.write("Developed by Vijay Joshi - Stock Portfolio Analysis")

**Explanation of Changes:**

1. **Interactive Plots with Plotly:**
   * Replaced matplotlib.pyplot with plotly.graph\_objects to enable interactive plots.
   * Used st.plotly\_chart to render the interactive charts.
2. **Zooming and Panning:**
   * Plotly provides zooming, panning, and other interactive features for time series plots.
   * use\_container\_width=True ensures the plots are responsive.
3. **Graph Interactivity:**
   * Allows users to zoom in, pan, and interact with the plots via a user-friendly interface.

**Result:**

* When you run the app, the graphs will be interactive.
* Users can zoom in, hover over the data points, and explore specific periods of the stock performance more closely.

**Prompts given to ‘ChatGPT Plus’ to get the Portfolio**

1. What is Yfinance library used with streamlit
2. What objects can be created using yfinance
3. Lets dive more in detals for each objects and its parameters
4. I would like to make a project using yfinance, streamlit aand Python for my students portfolio. It should be a practical use case and intermediate level complexity. Now initially in this prompt, just give me a problem statement for such a portfolio with a brief description of 4 lines suggesting what we are expected or plan to do in it.
5. OK, great, so what could be the problem statement in our portfolio mentioned as.
6. In my portfolio, after the above problem statement, what should be my actual format of portfolio. Like should we put down our thought process about how we are going to approach the problem and solve it or the work flow in brief? Please recommend how should i go about if someone gives me above problem statement.
7. Great, now given that i have to work on the above 10 points one by one. Please give me solution for above format. Let us start from step 1 only first and once satisfied and understood well, i will prompt you for the next step. We will go one step at a time.
8. Great job, now let us work on step 2, the 'Objective' part. Please assist me on it.
9. We are doing a great job. Now assist me on step 3, the Approach / Thought Process part considering the 4 items viz. Understanding the problem, identifying the features, data Source and User Interface Design. Also please note here, can we get Data Source from India Stock market. If yes, that would be great as i am residing in India and we will work on that Stock Market.
10. That was great and am happy with our pace of working and the details provided. Can we proceed to step 4 now. Will you be giving me the actual workable code that will answer the 5 steps in our workflow. The defining inputs, fetching data, processing data , visualizing data and building the dashboard.. The complete workable model.
11. I did follow the above steps, but getting following error, when running using VS Code. "pandas\\\_libs\\hashtable\_class\_helper.pxi", line 7089, in pandas.\_libs.hashtable.PyObjectHashTable.get\_item KeyError: 'Adj Close'
12. That was great, can we proceed to our next step now.
13. Can we make the graphs interactive, so that we can zoom in and rest.
14. That was a great project. Now one more thing i need help on. Since i am a student and not much familiar with stock market and its working. I would like to learn more about the Stock Market working related with above project only. Please start from scratch, first give me an overview about Stock Market, related to Stocks only. Subsequently i will prompt you to give more details on individual topics.
15. Which are the most commonly used and monitored stock index in India, please explain how they are composed. Which body decides on its composition and monitors it.
16. List the Ticker codes for the top 30 companies, which we can use in our project to monitor or check.
17. Why in place a company goes public? Why do they start as a private limited and then later only go public? What is its benefit in going Public?
18. How do a person makes money in Shares Trading as an Long time Investor and Short Time speculator? Explain by giving a fictious person trading and investing in a fictitous company.