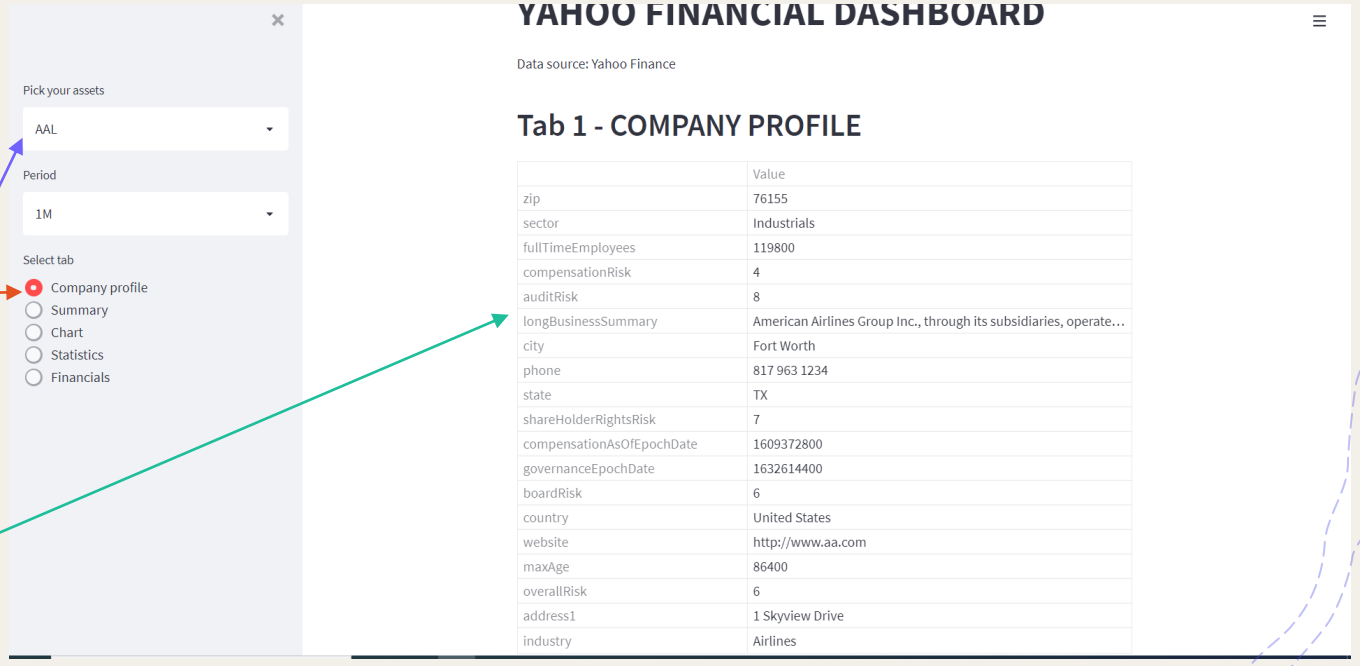
FINANCIAL ANALYSIS PRESENTATION

BY

ENITA OMUVWIE

Company Profile

- This is the first page of my application
- The first thing you need to do is to select a ticker
- Then it gives you everything about the company with the ticker assigned

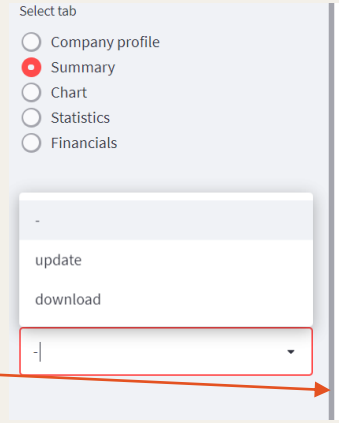


Summary

- This is the second page of the application
- This contains all attributes of current day trading from Volume, Day's range, Open price, Market Cap e.t.c
- You can download and create file name to directly download to your system
- It has a feature of update as well to get information updated

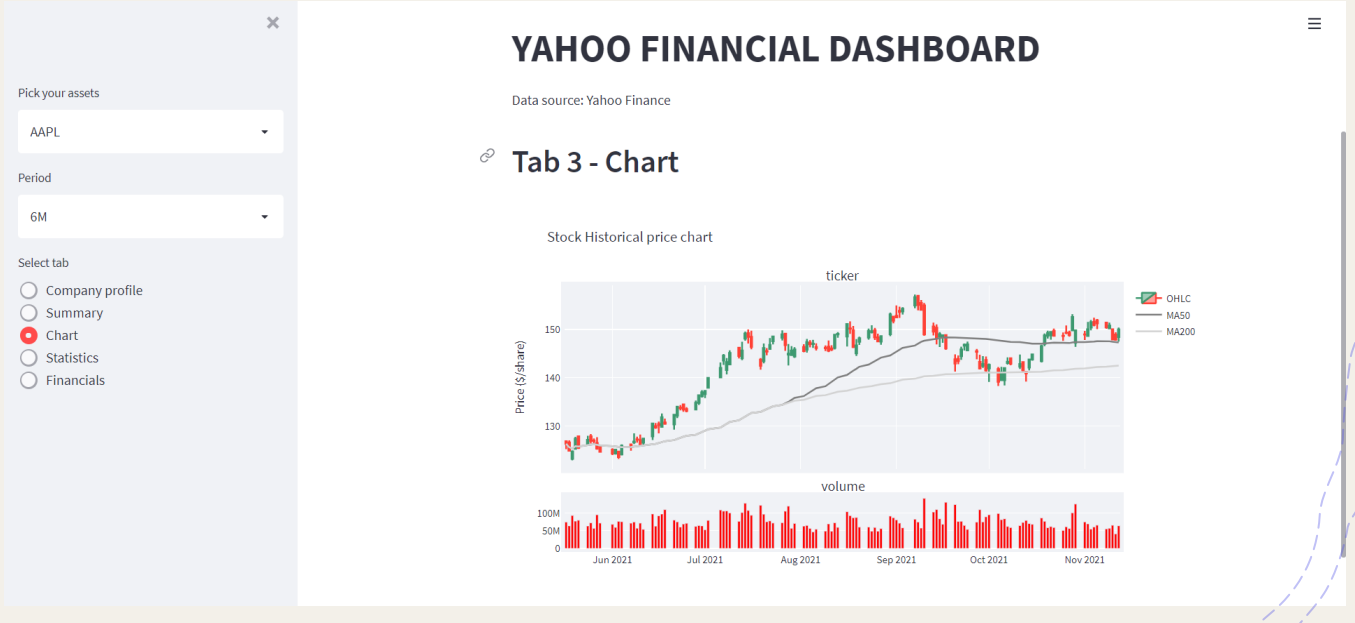
Summary

+ This is also on the second page
+ It gives you a line graph containing Low, High, Volume e.t.c of a selected stock for a selected period or interval



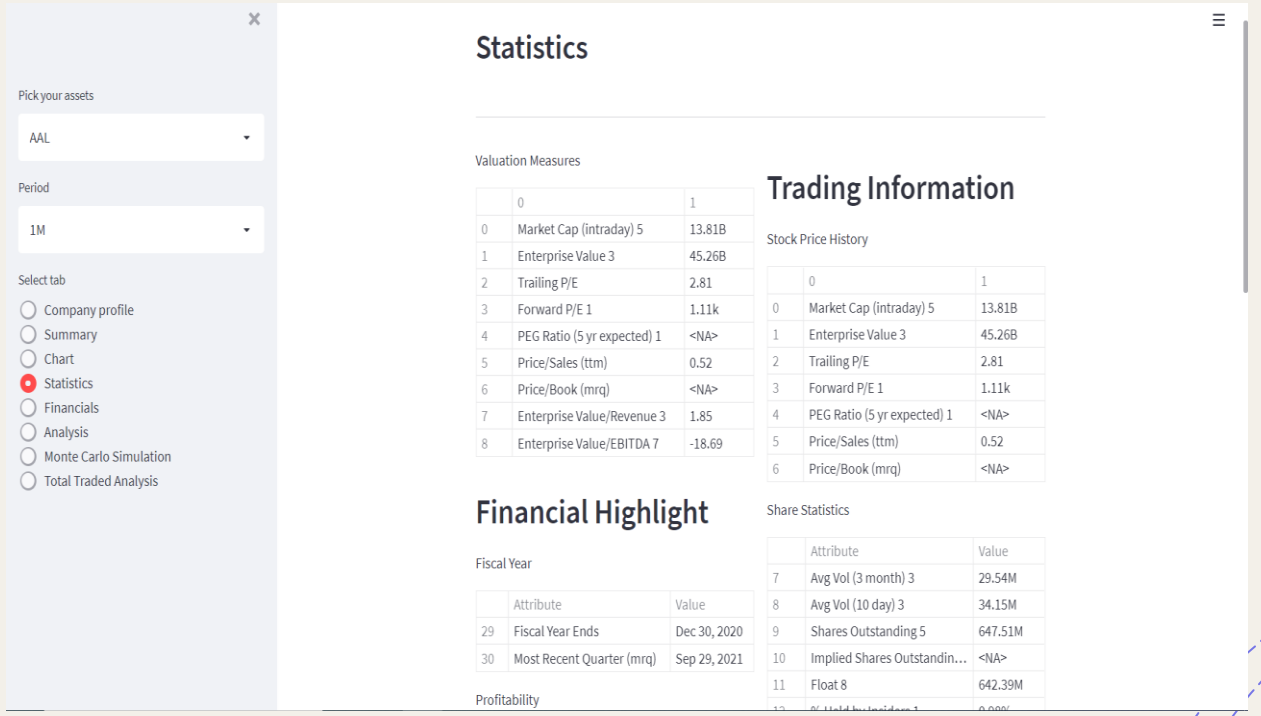
Chart

- This is the third page of the application containing a graph showing the high, low, open, close, volume of the selected stock in a specified period or time frame
- This graph consists of three different plots: Candlestick, Line and Bar graph
- The bar holds the volume, the ohlc holds the high, low, closing and opening prices of the stock
- The Line graph shows the moving average which helps is basically the calculates the average for a given time and it uses in smoothening out the variations to indicate unknown patterns in the data
- Reference of code: https://asxportfolio.com/shares-python-for-finance-plotting-stock-data



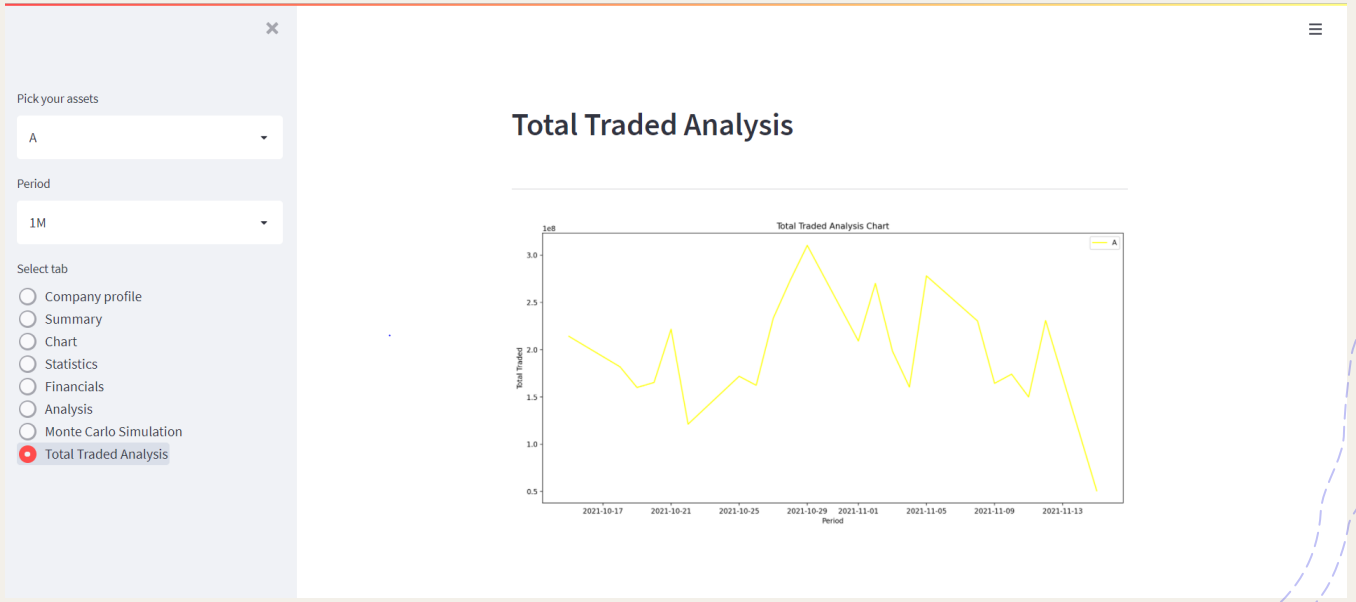
Statistics

- This tab shows us the statistics of each stock within a given period
- This shows the Valuation Measures, Financial Highlights and Trading Information of a selected stock



Total Traded Analysis

- This tab shows us the total traded price of each stock within a given period
- It helps us analyse the trading volume for a stock. Where the volume is used as a unit of measurement for market's activity during a period of time
- Reference of code https://www.youtube.com/watch?v=57qAxRV577c&t=267s



Body of Code for Libraries

This code contains all the library functions that was used for the entire project

```
from os import stat
from attr import attrib
import streamlit as st
from streamlit.type_util import Key
import yfinance as yf
import pandas as pd
import yahoo_fin.stock_info as si
from datetime import datetime, timedelta
import plotly.graph_objects as go
from plotly.subplots import make_subplots
import matplotlib.pyplot as plt
import numpy as np
from pandas_datareader import data as pdr


```

Body of Code for Time Frames

- This contains the code for the interval periods for the stock trade
- I also imported the relative delta library in order to compute for the dates
- The current_date is meant to fetch the date as today's date(present date)
- Then the if-elif loop is meant to loop through the time intervals stated in the timeframe in months and display the data for the selected period

```python
# main date function to compute for intervals
def date_function(n):
    from dateutil.relativedelta import relativedelta
    current_date = datetime.today()
    print('Current Date: ', current_date)
    timeframe = 0
    if(n == '1M'):
        timeframe = 1
    elif(n == '3M'):
        timeframe = 3
    elif(n == '6M'):
        timeframe = 6
    elif(n == 'YTD'):
        timeframe = datetime.now().month
    elif(n == '1Y'):
        timeframe = 12
    elif(n == '3Y'):
        timeframe = 36
    elif(n == '5Y'):
        timeframe = 60
    past_date = current_date - relativedelta(months=timeframe)
    # Convert datetime object to string in required format
    date_format = '%Y/%m/%d'
    past_date_str = past_date.strftime(date_format)
    print('Date (as string) - 20 months before current date: ', past_date_str)
    return past_date_str

```

Body of Code for Selection Bars

- The ticker list basically contains all the sp5000 tickers on the yahoo finance
- Ticker item is called with the streamlit sidebar+selectbox function to choose the stock of choice on the menu sidebar
- The range array consist of the time intervals in a list
- The period is also assigned with a streamlit sidebar+select box for the time frames
- The key helps to pair values with requirements that keys returned are unique

```
# set default ticker to sp500
ticker_list = ['-'] + si.tickers_sp500()
# create ticker items dropdown
ticker_item = st.sidebar.selectbox(
    'Pick your assets', ticker_list, key="menu_sidebar")

# duration of time
range_array = ['1M', '3M', '6M', 'YTD', '1Y', '3Y', '5Y', 'MAX']
# create sidebar dropdown for duration
period = st.sidebar.selectbox('Period', range_array, key="time-period-tab1")
```

Body of Code for Tab 1

- Defining tab1 function with the first few streamlit functions to give us the Title and Header of the page
- The function GetCompanyInfo is a function that returns the company information of any given ticker
- The get_company_info helps to fetch the company information from yahoo finance
- The if loop holds the tickers and once a ticker is selected the information of the company will be displayed

```
57  def tab1():
58      # Add dashboard title and description
59      st.title("YAHOO FINANCIAL DASHBOARD")
60      st.write("Data source: Yahoo Finance")
61      st.header('Tab 1 - COMPANY PROFILE')
62
63      @st.cache
64      def GetCompanyInfo(ticker_item):
65          return si.get_company_info(ticker_item)
66
67      # show warning when stock isn't selected
68      if(ticker_item == '-'):
69          st.warning('Select a stock to see data')
70
71      if ticker_item != '-':
72          info = GetCompanyInfo(ticker_item)
73          info['Value'] = info['Value'].astype(str)
74          st.dataframe(info, height=2000)
75
```

```python
def tab2():
    st.header('Summary')
    st.write('*****************************************************************')
    # set sidebar header
    st.sidebar.header('Select Tickers to Get Started')

    # Add table to show stock data

    @st.cache
    def GetCompanyInfo(ticker_item):
        return si.get_quote_table(ticker_item)

    # show warning when stock isn't selected
    if(ticker_item == '-'):
        st.warning('Select a stock to see data')

    if ticker_item != '-':
        info = GetCompanyInfo(ticker_item)
        data = pd.DataFrame.from_dict(info, orient='index').astype(str)
        st.dataframe(data, height=2000)

    start = pd.to_datetime(date_function(period))
    today = pd.to_datetime('today')

    # stock chart
    df = yf.download(ticker_item, start, today)
    st.line_chart(df)

    # update button to download and update the data
    st.sidebar.write('Select an option to either download or update')

    options = ['-', 'update', 'download']
    selected_option = st.sidebar.selectbox('Select an action', options)

    # download dataframe as csv after adding filename (filename optional. it would result .de
    def download_as_csv(filename):
        if not filename:
            df.to_csv('default.csv', index=False)
        else:
```