



FinDash

Product _ “FinDash

Product Goals & Challenges

- To build a **window** into the world of S&P 500.
- To build a **one stop** solution for stock market enthusiasts. All important data is present in the dashboard.
- To present stock market data in an **interactive** dashboard.
- To help investors view the **trends** in their portfolio.
- To help investors see **predictions** and VAR of their stock.

Select a ticker

AAPL

Select tab

Summary

Chart

Statistics

Financials

Analysis

Monte Carlo Simulation

Your Portfolio's Trend

Summary

Select ticker on the left to begin

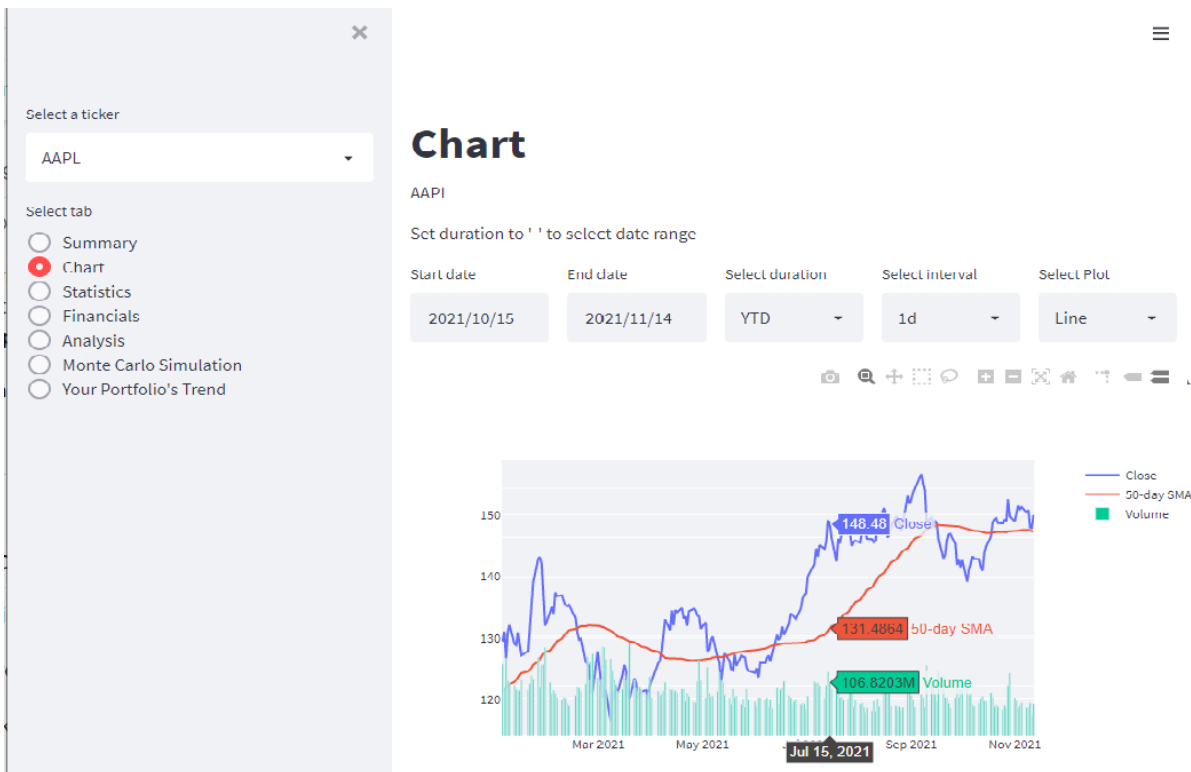
AAPL

	value		value
Previous Close	147.87	Market Cap	2.461T
Open	148.43	Beta (5Y Monthly)	1.21
Bid	149.70 x 2900	PE Ratio (TTM)	26.74
Ask	149.73 x 1000	EPS (TTM)	5.61
Day's Range	147.48 - 150.40	Earnings Date	Jan 25, 2022 - Jan
52 Week Range	112.59 - 157.26	Forward Dividend & Yield	0.88 (0.59%)
Volume	63635596.0	Ex-Dividend Date	Nov 05, 2021
Avg. Volume	75722023.0	1y Target Est	168.45



Tab 1

- View important statistics
- View trend of close price for selected period



Tab 2

- View three plots in one figure
- Choose your desired parameters
- See datapoints as you hover

Select a ticker

AAPL

Select tab

☐ Summary

☐ Chart

☒ Statistics

☐ Financials

☐ Analysis

☐ Monte Carlo Simulation

☐ Your Portfolio's Trend

Statistics

AAPL

Valuation Measures

Market Cap (intraday) 5	2.46T
Enterprise Value 3	2.52T
Trailing P/E	26.74
Forward P/E 1	26.81
PEG Ratio (5 yr expected) 1	3.22
Price/Sales (ttm)	6.91
Price/Book (mrq)	39.00
Enterprise Value/Revenue 3	6.90
Enterprise Value/EBITDA 7	20.49

Financial Highlights

Fiscal Year

	Value
Fiscal Year Ends	Sep 24, 2021
Most Recent Quarter (mrq)	Sep 24, 2021

Profitability

	Value
Profit Margin	25.88%
Operating Margin (ttm)	29.78%

Management Effectiveness

Trading Information

Stock Price History

	Value
Beta (5Y Monthly)	1.21
52-Week Change 3	24.68%
S&P500 52-Week Change 3	29.11%
52 Week High 3	157.26
52 Week Low 3	112.59
50-Day Moving Average 3	146.52
200-Day Moving Average 3	141.23

Share Statistics

	Value
Avg Vol (3 month) 3	75.72M
Avg Vol (10 day) 3	58.21M
Shares Outstanding 5	16.41B
Implied Shares Outstanding 6	nan
Float 8	16.39B
% Held by Insiders 1	0.07%
% Held by Institutions 1	58.95%
Shares Short (Oct 28, 2021) 4	100.5M
Short Ratio (Oct 28, 2021) 4	1.34
Short % of Float (Oct 28, 2021) 4	0.61%
Short % of Shares Outstanding (Oct 28, 2021) 4	0.61%
Shares Short (prior month Sep 29, 2021) 4	101.11M

Select a ticker

AAPL

Select tab

☐ Summary

☐ Chart

☐ Statistics

☒ Financials

☐ Analysis

☐ Monte Carlo Simulation

☐ Your Portfolio's Trend

Financials

AAPL

Show

Income Statement

Period

Yearly	2021-09-30 00:00:00	2020-06-30 00:00:00	2019-06-30 00:00:00	18-09-30 00:00:00
research&Development	100000000	97,000,000.0000	64,000,000.0000	72,700,000.0000
effectOfAccountingCharges	<NA>	<NA>	<NA>	<NA>
incomeBeforeTax	1212000000	811,000,000.0000	608,000,000.0000	705,400,000.0000
minorityInterest	57000000	61,000,000.0000	65,700,000.0000	68,800,000.0000
netIncome	939000000	612,000,000.0000	430,000,000.0000	575,200,000.0000
sellingGeneralAdministrative	1279000000	1,283,000,000.0000	815,000,000.0000	784,200,000.0000
grossProfit	2732000000	2,591,000,000.0000	1,799,000,000.0000	1,856,800,000.0000
ebit	1438000000	1,246,000,000.0000	1,098,000,000.0000	1,042,900,000.0000
operatingIncome	1438000000	1,246,000,000.0000	1,098,000,000.0000	1,042,900,000.0000
otherOperatingExpenses	-85000000	-35,000,000.0000	-178,000,000.0000	-43,000,000.0000
interestExpense	-157000000	-207,000,000.0000	-208,000,000.0000	-210,000,000.0000
extraordinaryItems	<NA>	<NA>	<NA>	<NA>
nonRecurring	<NA>	<NA>	<NA>	<NA>
otherItems	<NA>	<NA>	<NA>	<NA>
incomeTaxExpense	261000000	187,000,000.0000	172,000,000.0000	118,800,000.0000
totalRevenue	12861000000	12,468,000,000.0000	9,458,000,000.0000	9,319,100,000.0000
totalOperatingExpenses	11423000000	11,222,000,000.0000	8,360,000,000.0000	8,276,200,000.0000
costOfRevenue	10129000000	9,677,000,000.0000	7,659,000,000.0000	7,462,300,000.0000
totalOtherIncomeExpenseNet	-226000000	-435,000,000.0000	-490,000,000.0000	-337,500,000.0000
discontinuedOperations	<NA>	-8,000,000.0000	1,000,000.0000	1,000,000.0000
netIncomeFromContinuingOps	951000000	624,000,000.0000	436,000,000.0000	586,600,000.0000
netIncomeApplicableToCommonShares	937000000	612,000,000.0000	429,000,000.0000	573,900,000.0000

Analysis

Currency in USD

AAPL

Earnings Estimate	Current Qtr. (Dec 2021)	Next Qtr. (Mar 2022)	Current Year (2022)	Next Year (2023)
0 No. of Analysts	24,0000	23,0000	34,0000	29,0000
1 Avg. Estimate	1.8800	1.3200	5.7500	6.1500
2 Low Estimate	1.7500	1.1600	5.2000	5.2600
3 High Estimate	1.9700	1.5200	6.2700	6.8200
4 Year Ago EPS	1.6800	1.4000	5.6100	5.7500

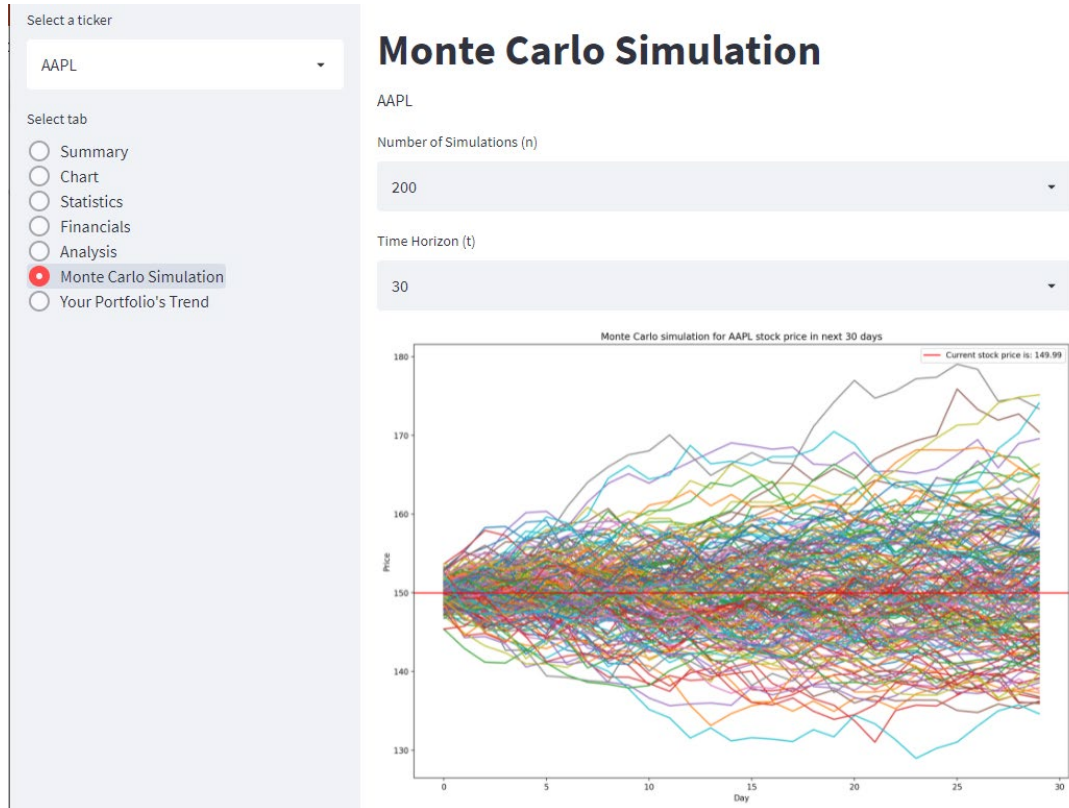
Revenue Estimate	Current Qtr. (Dec 2021)	Next Qtr. (Mar 2022)	Current Year (2022)	Next Year (2023)
0 No. of Analysts	21	21	33	27
1 Avg. Estimate	117.97B	90.41B	381.66B	398.36B
2 Low Estimate	111.81B	82.17B	359.16B	300.5B
3 High Estimate	121.3B	98.28B	405.44B	445.58B
4 Year Ago Sales	<NA>	<NA>	365.82B	381.66B
5 Sales Growth (year(est)	<NA>	<NA>	4.30%	4.40%

Earnings History	12/30/2020	3/30/2021	6/29/2021	9/29/2021
0 EPS Est.	1.41	0.99	1.01	1.24
1 EPS Actual	1.68	1.4	1.3	1.24
2 Difference	0.27	0.41	0.29	0
3 Surprise %	19.10%	41.40%	28.70%	0.00%

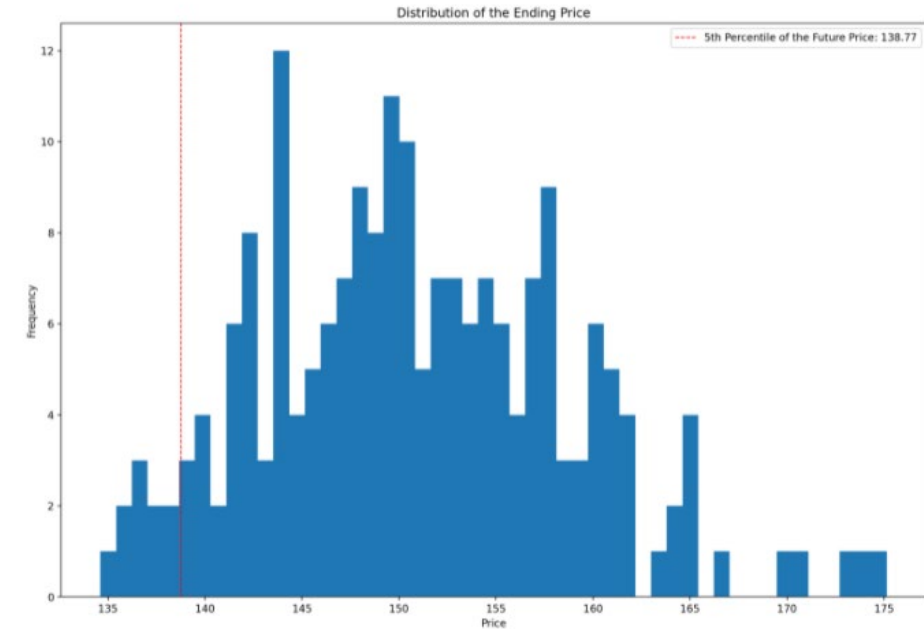
EPS Trend	Current Qtr. (Dec 2021)	Next Qtr. (Mar 2022)	Current Year (2022)	Next Year (2023)
0 Current Estimate	1.8800	1.3200	5.7500	6.1500

Tab 3, 4 & 5

- View clearly labelled statistics
- View all financial statements with the option for period selection
- View all analysis information



Value at Risk (VaR)



VaR at 95% confidence interval is: 11.22 USD

Tab 6

- View stock price prediction
- Select between time horizon and number of intervals
- Value at risk calculated for you

Select a ticker

AAPL

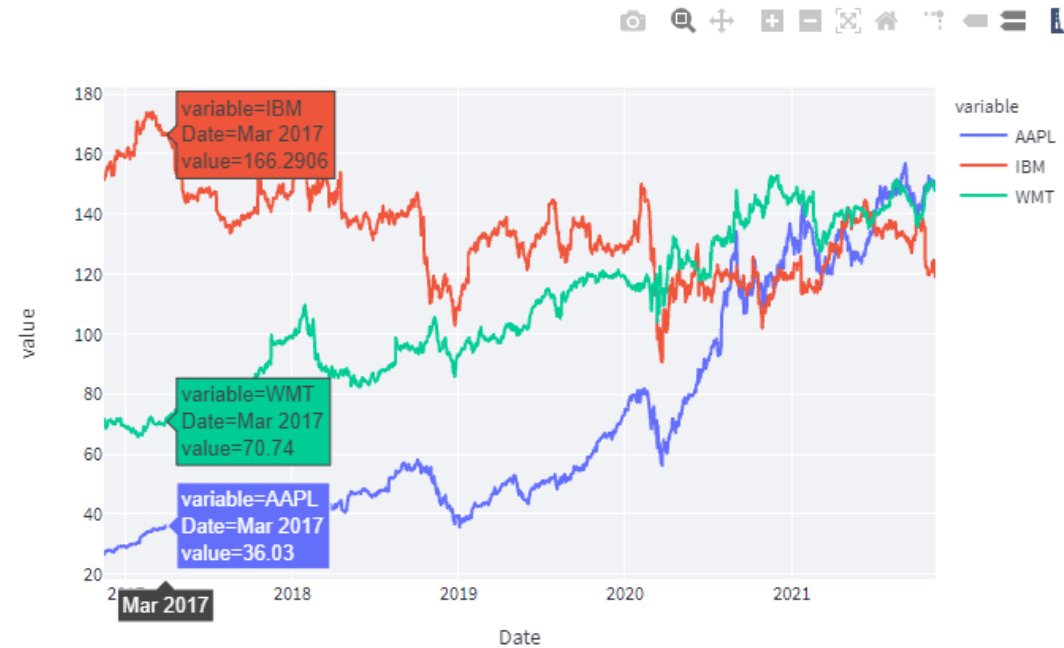
Select tab

- ☐ Summary
- ☐ Chart
- ☐ Statistics
- ☐ Financials
- ☐ Analysis
- ☐ Monte Carlo Simulation
- ☒ Your Portfolio's Trend

Your Portfolio's Trend

Select tickers in your portfolio

AAPL X IBM X WMT X



Tab 7

- View price trend for all the stocks in your portfolio

How was the product built?

- Data was scraped from Yahoo Finance website using Python libraries **yahoo_fin.stock_info** and **Yfinance**.
- Dashboard was built using **Spyder** & **Streamlit**.
- Data was processed using Python libraries **Pandas**, **Numpy**, and **Datetime**.
- Data visualization was done using Python libraries **Matplotlib** and **Plotly**.


```
def tab1():

    st.title("Summary")
    st.write("Select ticker on the left to begin")
    st.write(ticker)

    #The code below gets the quota table from Yahoo Finance. The streamlit page
    #is divided into 2 columns and selected columns are displayed on each side of the page.

    def getsummary(ticker):
        table = si.get_quote_table(ticker, dict_result = False)
        return table

    c1, c2 = st.columns((1,1))
    with c1:
        if ticker != '-':
            summary = getsummary(ticker)
            summary['value'] = summary['value'].astype(str)
            showssummary = summary.iloc[[14, 12, 5, 2, 6, 1, 16, 3],]
            showssummary.set_index('attribute', inplace=True)
            st.dataframe(showsummary)

    with c2:
        if ticker != '-':
            summary = getsummary(ticker)
            summary['value'] = summary['value'].astype(str)
            showssummary = summary.iloc[[11, 4, 13, 7, 8, 10, 9, 0],]
            showssummary.set_index('attribute', inplace=True)
            st.dataframe(showsummary)
```

How to scrape and display a table

```
#The code below uses the yahoofinance package to get all the available stock
#price data. Plotly is then used to visualize the data. An interesting feature
#from plotly called range selector is also used. A list of dictionaries
#is added to range selector to make buttons and identify the periods.
#References:
#https://plotly.com/python/range-slider/

@st.cache
def getstockdata(ticker):
    stockdata = yf.download(ticker, period = 'MAX')
    return stockdata

if ticker != '-':
    chartdata = getstockdata(ticker)

    fig = px.area(chartdata, chartdata.index, chartdata['Close'])

    fig.update_xaxes(
        rangeselector=dict(
            buttons=list([
                dict(count=1, label="1M", step="month", stepmode="backward"),
                dict(count=3, label="3M", step="month", stepmode="backward"),
                dict(count=6, label="6M", step="month", stepmode="backward"),
                dict(count=1, label="YTD", step="year", stepmode="todate"),
                dict(count=1, label="1Y", step="year", stepmode="backward"),
                dict(count=3, label="3Y", step="year", stepmode="backward"),
                dict(count=5, label="5Y", step="year", stepmode="backward"),
                dict(label = "MAX", step="all")
            ])
        )
    )
    st.plotly_chart(fig)
```

Chart with buttons for selecting period

Sample Codes

```
#The code below first obtains all the data using the download option from yahoo finance.
#It then creates a column for the simple moving average, makes the date index into a column
#and then subsets the dataframe to get just the date and and SMA column.
#Then if a duration is selected from the dropdown, data for that duration is downloaded
# and the SMA column is merged to the dataframe. If a duration is not selected then
#automatically the specified date range is used to get the data and that is also merged
#with the SMA column
#References:
#https://towardsdatascience.com/data-science-in-finance-56a4d99279f7
```

```
@st.cache
def getchartdata(ticker):
    SMA = yf.download(ticker, period = 'MAX')
    SMA['SMA'] = SMA['Close'].rolling(50).mean()
    SMA = SMA.reset_index()
    SMA = SMA[['Date', 'SMA']]

    if duration != '-':
        chartdata1 = yf.download(ticker, period = duration, interval = inter)
        chartdata1 = chartdata1.reset_index()
        chartdata1 = chartdata1.merge(SMA, on='Date', how='left')
        return chartdata1
    else:
        chartdata2 = yf.download(ticker, start_date, end_date, interval = inter)
        chartdata2 = chartdata2.reset_index()
        chartdata2 = chartdata2.merge(SMA, on='Date', how='left')
        return chartdata2
```

Data Manipulation

```
#The code below uses plotly to visualize the data. Subplots from plotly is used to make 2 y axis.
#First y axis shows the stock close price and SMA and the second is used to show volume.
#Plotly graph objects are used to add graphs to the axes.The range for the y axis for
#volume is manipulated so that the bars appear small.
#References:
#https://plotly.com/python/multiple-axes/

if ticker != '-':
    chartdata = getchartdata(ticker)

    fig = make_subplots(specs=[[{"secondary_y": True}]])

    if plot == 'Line':
        fig.add_trace(go.Scatter(x=chartdata['Date'], y=chartdata['Close'], mode='Lines',
                                name = 'Close'), secondary_y = False)
    else:
        fig.add_trace(go.Candlestick(x = chartdata['Date'], open = chartdata['Open'],
                                     high = chartdata['High'], low = chartdata['Low'], close = chartdata['Close'], name = 'Candle'))

    fig.add_trace(go.Scatter(x=chartdata['Date'], y=chartdata['SMA'], mode='Lines', name = '50-day SMA'), secondary_y = False)

    fig.add_trace(go.Bar(x = chartdata['Date'], y = chartdata['Volume'], name = 'Volume'), secondary_y = True)

    fig.update_yaxes(range=[0, chartdata['Volume'].max()*3], showticklabels=False, secondary_y=True)

    st.plotly_chart(fig)
```

Multiple graphs in same figure

Sample Codes
