XP Analytics

A trading app that shows “experienced” analytics for the stocks / portfolio

Date 10/20/2022

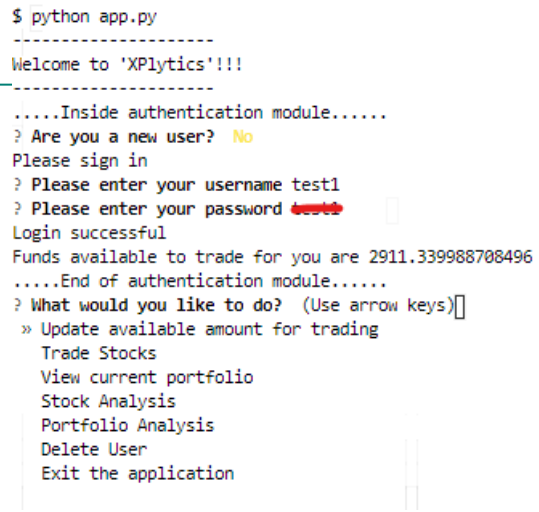
XP Analytics

Executive summary

Main conclusion

Goal of the Project

- To apply all the modules learned so far in Fintech Bootcamp
- To be able to trade stocks, perform stock / portfolio analysis and generate relevant reports
- Idea is to generate a trading app that helps user look at the history, present risk/ratio metrics and future possibilities for individual stocks as well the user portfolio
- App would enable user to look at analytics and make informed decision on his/her stock investment choices



XP Analytics

Team

Awesome team members

Team members

- Pravin Patil
- Esteban Lopez
- Jorge Villacreses
- Liset Lopez
- William Wolfenbarger

XP Analytics

Data Collection

Decided not to use alpaca since it only retrieves up to 3 years data only.

Data Input

- Project used Fire and Questionary libraries to collect inputs from user via command line interface. For instance,

```
# function to load user options
def load_user_options():
    signed_in_user_choices = [
        'Update available amount for trading',
        'Trade Stocks',
        'View current portfolio',
        'Stock Analysis',
        'Portfolio Analysis',
        'Delete User',
        'Exit the application'
    ]

    user_choice = qs.select(
        "What would you like to do?",
        choices=signed_in_user_choices
    ).ask()
```

Market Data Collection

- Code uses 'YFinance' API to download market data.

```
## Use Yahoo finance to retrieve 5 yrs of price data
portfolio = yf.Tickers(tickers)
portfolio_history_df = portfolio.history(period = "5y")
prices_df = portfolio_history_df["Close"]
```

XP Analytics

Data Cleanup & Exploration

Subtitle

Data Cleanup

- DataFrame function '.dropna()' was used for data cleanup (especially on returns data).

Data Exploration

- On multiple occasions, following tactics were used to filter out irrelevant data:
  - _df.drop(columns = ["x"])
  - _df[x']
- Local functions were defined to get data out of panel widgets
  - _wd = pn.widgets.Select(options=
  - def _name(_wd):
  - name = pn.pane.Markdown("# " + yf.Ticker(_wd).info['longName'])
  - return name
- SQL Lite database (with 'sqlalchemy' library) was used to interact with database

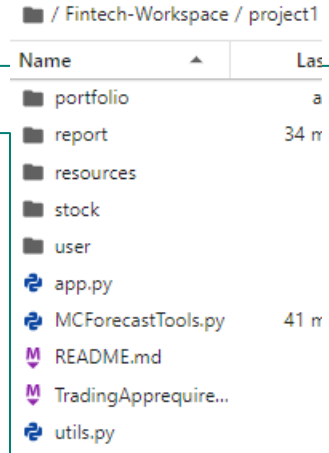
```
# retrieve user portfolio details
user_portfolio_query = f"""
    SELECT
        ticker, number_of_shares
    FROM
        portfolio
    WHERE user_name ='{user_df['user_name'].iloc[0]}'
"""
portfolio_df = pd.read_sql_query(user_portfolio_query, db_engine)
```

Approach
SDLC

Approach

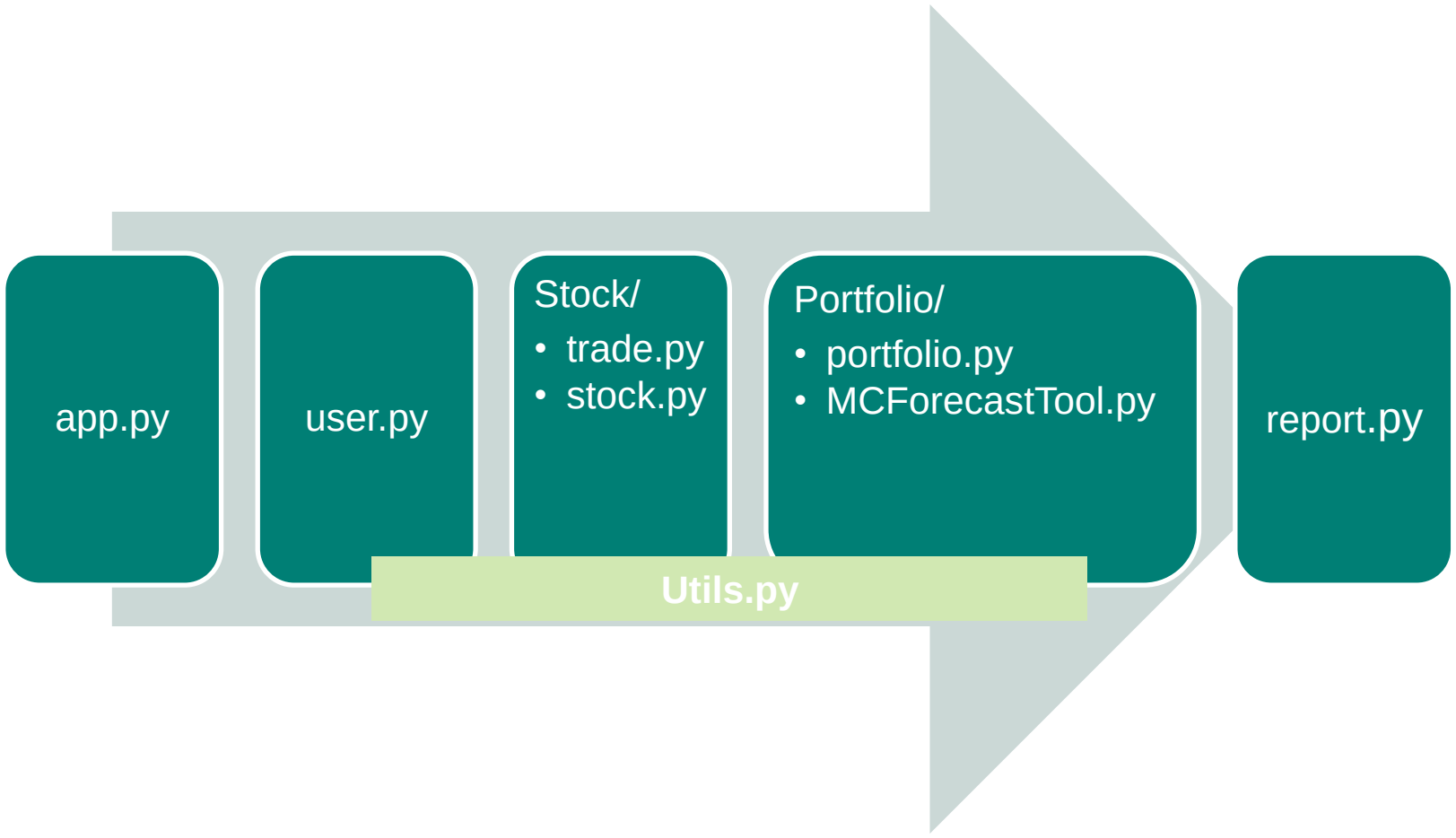
- We used typical software development lifecycle methodology
  - Defined requirements (user stories, technology details, user interface)
  - Post definition of requirements, various components were envisaged. For instance:
    - user module
    - trade module
    - stock and portfolio analysis module
    - and visualization or reports module
  - These modules were divided amongst members and assigned one project manager to ensure integration of all components

- After individual components were built, we integrated the code into the app
- Next step was code review, QA testing
- And finally, documentation (readme file)



Code

Code Flow



Code
Details

Libraries

- 'Pandas' library to work with dataframes and analyse timeseries data.
- 'YFinance' API to download market data.
- 'Numpy' library in integration with 'Pandas' to manipulate financial data.
- 'MCForecastTools' file for Monte Carlo Simulations.
- 'Panel' library for creating an interactive web dashboard.
- 'Fire' and 'Questionary' library for Command Line Interface.
- 'SQLAlchemy' toolkit for application development.
- 'hvplot' and 'matlabplot' for data visualization.
- 'statsmodels' for linear regression to calculate stock alphas

Github project

- https://github.com/prpercy/project1

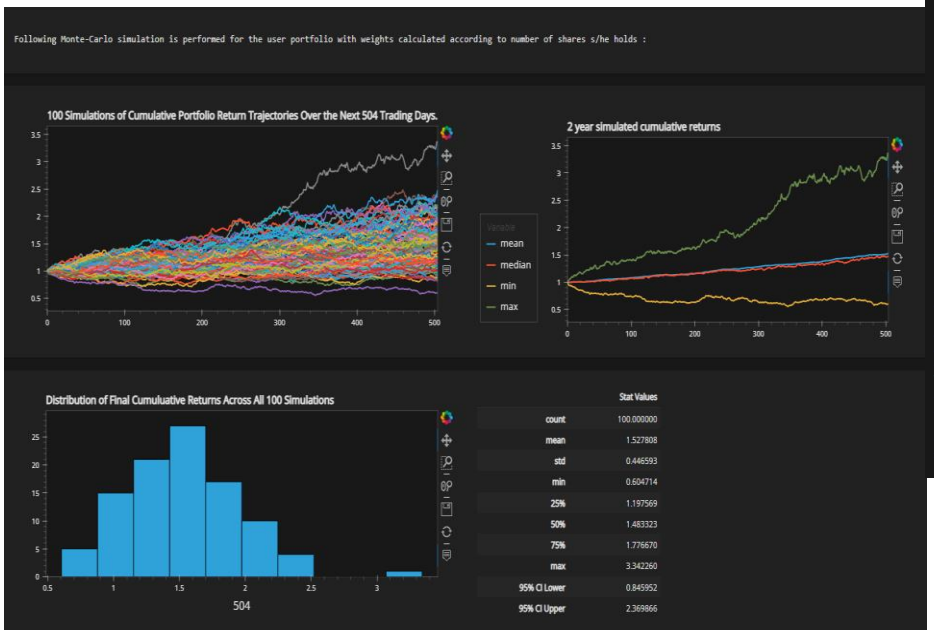
XP Analytics

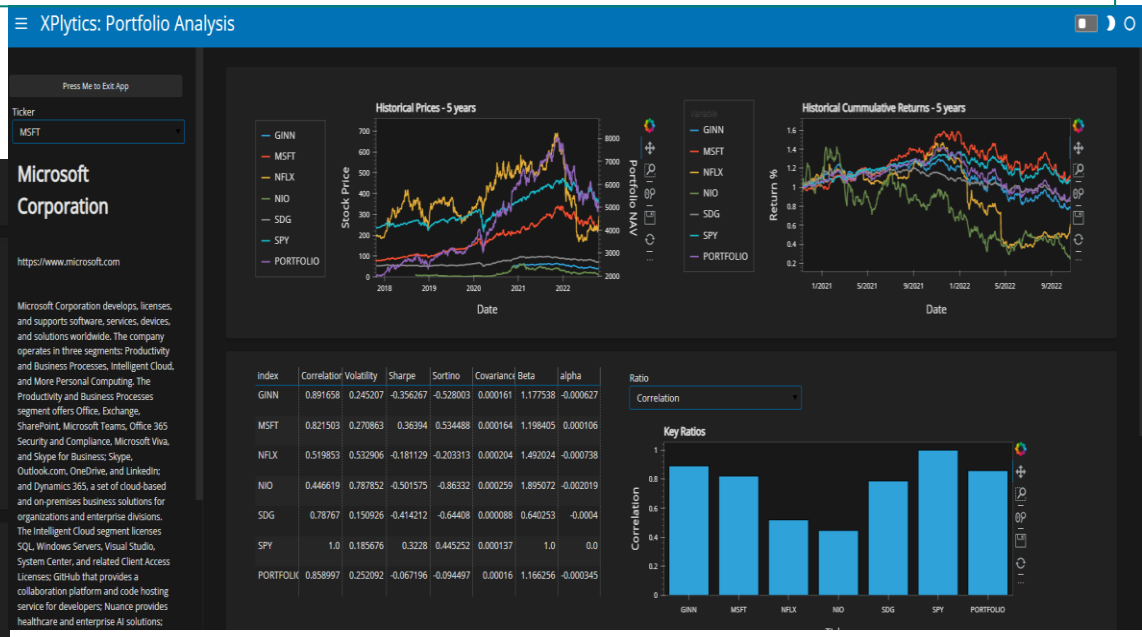
Results

App produces fantastic interactive report using Panel template

Interactive report

- Report is intuitive, interactive and responsive to user selections
- Report shows past, present and future perspectives providing deep understanding
- Report also allows user to explore more through interactive options to analyze or run a deep dive





Next Steps

Next Steps

- App could be further developed to add:
  - proper user interface
  - and actual market interface to perform actual trading