

LAB 3 Part 1 REPORT

Team Details

Team Name: SSK

Name	USC ID
Shreyansh Baredia	3739756887
Soma Meghana Prathipati	4226812081
Kayvan Shah	1106650685

Table of Contents

Rationale.....	4
Implementation.....	5
Tools and Libraries.....	5
Data Collection.....	6
Yahoo Finance API.....	6
User-Provided Ticker List.....	6
Data Storage.....	6
MongoDB.....	6
MongoDB collections.....	6
Data Processing and Models.....	6
Pydantic Models.....	6
Data Flow.....	6
Data Retrieval and Processing (in yf.py).....	6
User Management (in main.py and manager.py).....	7
Portfolio Management (in main.py and manager.py).....	7
About the Code.....	7
Snapshots.....	8
Command Line Toolkit.....	8
Create a new user.....	8
Get user information.....	8
Create Portfolio.....	8
Get the List of Portfolios for the User.....	9
Add Stock to Portfolio.....	9
Remove Stock from Portfolio.....	10
Remove Portfolio.....	10
Fetch OHLC Data for One Stock.....	10
Fetch Portfolio by ID.....	10
Fetch data for all stocks by portfolio ID.....	11
Individual Contributions.....	12

Rationale

In this lab assignment, we will be learning how to query stock market data from the web, process it, and store it in the database before querying and retrieving the required data.

Stock market data is generally considered volatile as it keeps changing rapidly. A flexible database that can store and retrieve data with minimal latency would be required to store such data.

We decided to use a NoSQL database to store the OHLC data of different stocks for the following reasons:

- **Flexible Schema Design**
MongoDB's flexible schema allows you to store data in a JSON-like format, making it easy to adapt to changes in your data structure without requiring a predefined schema. This flexibility is beneficial for handling various types of financial data.
- **Rich Query Language**
MongoDB's query language supports many queries, including complex ones needed for financial analysis. This allows you to retrieve and analyze specific subsets of data efficiently.
- **Aggregation Framework**
MongoDB's aggregation framework allows you to perform complex data transformations and analysis directly within the database. This can be beneficial for generating OHLC aggregates or other derived metrics from raw stock data.
- **Document-Oriented Storage**
MongoDB stores data in BSON (Binary JSON) format, a binary representation of JSON documents. This document-oriented storage is well-suited for financial data, where each stock's OHLC data can be represented as a document.

Implementation

Tools and Libraries

1. **Motor:** Asynchronous driver for MongoDB.
2. **Numpy:** Numerical computing library for handling large, multi-dimensional arrays.
3. **Pandas:** Data manipulation and analysis library, providing data structures for efficient manipulation and cleaning of structured data. The pandas library is used extensively in the code to work with financial data and manage portfolios.
4. **Pandas-datareader:** A remote data access library fetching financial data from Yahoo Finance. It is used to retrieve stock price data.
5. **Pyarrow:** A library for efficient columnar in-memory analytics used to handle data efficiently.
6. **Pydantic:** Data validation and settings management using Python type annotations. Used for defining Pydantic models to validate and structure data. Pydantic models represent users, portfolios, tickers, etc.
7. **Pydantic-settings:** Extends Pydantic to support configuration settings. Used for managing configuration settings in a Pydantic model. Helps in handling application configurations.
8. **Pymongo:** MongoDB driver for Python. Used for interacting with MongoDB databases. It is used for database connectivity, creating cursors, and managing collections.
9. **Pymongo[srv]:** Additional package for MongoDB, specifically for connecting to MongoDB Atlas, MongoDB's cloud service. It indicates using the Pymongo package with support for connecting to MongoDB Atlas.
10. **Pyrate-limiter:** Rate-limiting library. Used for rate-limiting requests to external services, such as the Yahoo Finance API. It helps in managing the number of requests made to prevent being blocked.
11. **Python-dotenv:** Loads environment variables from a file. Used for loading environment variables from a .env file. It helps in managing sensitive information like API keys.
12. **Requests:** HTTP library for making requests. Used for making HTTP requests, for example, when fetching data from external APIs like Yahoo Finance.
13. **Requests-cache:** Library for caching HTTP responses. Used for caching HTTP responses when fetching data. It helps in reducing the number of requests made and improving performance.
14. **Requests-rate limiter:** Rate limiting for HTTP requests. Like pyrate-limiter, it helps rate-limiting HTTP requests to prevent excessive usage and potential blocking.
15. **YFinance:** Python wrapper for Yahoo Finance API. Used for fetching financial data (stock prices, tickers) from Yahoo Finance. It is a key library for retrieving stock-related information.

Data Collection

Yahoo Finance API

- **Tool:** yfinance, pandas-datareader
- **Process**
 - The code fetches financial data (stock prices) from Yahoo Finance using the yfinance library.
 - pandas-datareader is used to retrieve data from Yahoo Finance API.

User-Provided Ticker List

- **Tool:** pandas
- **Process**
 - Reads a CSV file (us_symbols.csv) containing a list of stock ticker symbols.
 - The file is read using the pandas library, and the data is manipulated for further use.

Data Storage

MongoDB

- **Tool:** pymongo
- **Process**
 - MongoDB is used as the database to store user-related data, ticker information, and portfolios.

MongoDB collections

- **users_collection:** Stores user details.
- **tickers_info_collection:** Stores information about stock tickers.
- **portfolios_collection:** Stores user portfolios.

Data Processing and Models

Pydantic Models

- **Tool:** pydantic
- **Process:**
 - Pydantic models are used for defining the structure and validation of various data entities, such as users, tickers, portfolios, etc.
 - These models are used to ensure that the data conforms to a predefined structure before being stored or processed.

Data Flow

Data Retrieval and Processing (in yf.py)

- Fetches data from Yahoo Finance API for stock tickers.
- Processes the fetched data using functions like clean_ticker_data, resample, and basic_preprocess.
- Stores ticker information in the tickers_info_collection MongoDB collection.

User Management (in main.py and manager.py)

- Creates and manages user accounts using the UserManager class.
- Validates and verifies user credentials.
- Manages user portfolios using the PortfolioManager class.

Portfolio Management (in main.py and manager.py)

- Creates, removes, and lists user portfolios.
- Adds and removes stocks from portfolios.
- Manages stocks associated with each portfolio.
- Stores data in MongoDB collections.

About the Code

- **db.py:** This file contains code related to database connectivity and configuration using MongoDB. It defines a PyObjectId class for handling ObjectIds and creates cursors for different collections such as users, tickers_info, and portfolios.
- **main.py:** The main.py file is the entry point for the stock market analysis application's command-line interface (CLI). It uses argparse to define and parse command-line arguments for user and portfolio management actions, such as creating users and portfolios and managing stocks within portfolios.
- **manager.py:** This file contains classes for managing users (UserManager) and portfolios (PortfolioManager). It handles user creation, verification, and portfolio management actions like creating, removing, and listing portfolios.
- **models.py:** The models.py file defines Pydantic models representing various entities in the application, such as users, tickers, portfolios, and OHLC (Open-High-Low-Close) data. It also includes a TickerInfoManager class for retrieving detailed information about stock tickers.
- **settings.py:** This file contains configurations for the application, including the MongoDB URI, yfinance cache file location, and password hashing settings. It also provides a function to get a logger and caches the application settings.
- **yf.py:** The yf.py file contains code for fetching stock information using the Yahoo Finance API. It defines a CachedLimiterSession class for rate-limiting requests and caching responses. The get_ticker_info function retrieves detailed information about a stock ticker.
- **manager.py:** The manager.py file contains classes for managing users (UserManager) and portfolios (PortfolioManager). It handles user creation, verification, and portfolio management actions like creating, removing, and listing portfolios.

Snapshots

Command Line Toolkit

```
shah@KI-KINDL MINGW64 ~/OneDrive - University of Southern California/Assignments & Labs/usc-dsci560-dspp-sp24/lab3/stock-market-analysis/src (main)
$ python main.py --help
usage: main.py [-h] (user,portfolio,market-data) ...

Command-line Interface for Stock Market Analysis Application

positional arguments:
  (user,portfolio,market-data)
  user                  Available commands
  portfolio             User management commands
  market-data          Portfolio management commands
                       Fetch market data commands

options:
  -h, --help            show this help message and exit
```

Create a new user

```
shah@KI-KINDL MINGW64 ~/OneDrive - University of Southern California/Assignments & Labs/usc-dsci560-dspp-sp24/lab3/stock-market-analysis/src (main)
$ python main.py portfolio create --username dan_man --password dan_password --portfolio_name my_first_portfolio
[01/31/24 21:38:22] INFO User verification successful
[01/31/24 21:38:22] INFO Created portfolio with name: 'my_first_portfolio' and portfolio_id: '60a627be576d71a78b2464e6' successfully.
```

manager.py:85
manager.py:81

Get user information

```
shah@KI-KINDL MINGW64 ~/OneDrive - University of Southern California/Assignments & Labs/usc-dsci560-dspp-sp24/lab3/stock-market-analysis/src (main)
$ python main.py portfolio get-info --username john_doe --password secretpassword123
[01/31/24 00:22:40] INFO User verification successful
[01/31/24 00:22:40] INFO { 'username': 'john_doe', 'name': { 'first_name': 'John', 'last_name': 'Doe' }, 'password': '$2b$12$HfZG64Jkz09KCTD1lGQp7Gul8yPG43UyYbA3Dx0Ge/I7nc0X6', 'created_at': datetime.datetime(2024, 1, 30, 9, 7, 23, 978000)}
[01/31/24 00:22:40] INFO User Information: { 'username': 'john_doe', 'name': { 'first_name': 'John', 'last_name': 'Doe' }, 'created_at': '2024-01-30T09:07:23.978000' }
```

Create Portfolio

```
shah@KI-KINDL MINGW64 ~/OneDrive - University of Southern California/Assignments & Labs/usc-dsci560-dspp-sp24/lab3/stock-market-analysis/src (main)
$ python main.py portfolio create --username john_doe --password secretpassword123 --portfolio_name my_first_portfolio
[01/31/24 01:07:40] INFO User verification successful
[01/31/24 01:07:40] INFO Created portfolio with name: 'my_first_portfolio' and portfolio_id: '60a627be576d71a78b2464e6' successfully.
```

manager.py:85
manager.py:81

Create portfolio with same name

```
shah@KI-KINDL MINGW64 ~/OneDrive - University of Southern California/Assignments & Labs/usc-dsci560-dspp-sp24/lab3/stock-market-analysis/src (main)
$ python main.py portfolio create --username dan_man --password dan_password --portfolio_name my_first_portfolio
[01/31/24 01:31:41] INFO User verification successful
[01/31/24 01:31:41] ERROR Portfolio cannot have the same name: 'my_first_portfolio'. Please use a different name
NoneType: None
```

manager.py:85
manager.py:82

```

user@kali:~/KDDI_HDD464-0-jupyter$ python main.py portfolio list-all --username John_doe --password secretpassword123
[08/31/24 03:25:21] INFO User verification successful
[08/31/24 03:25:21] INFO User's Portfolios:
{
  {
    'id': '650a8c4e417bf1f5330abc',
    'portfolio_name': 'portfolio2',
    'tickers': [
      {
        'name': 'ADANI POWER LTD.',
        'ticker_code': 'ADANIPOWER.NS',
        'exchange': 'NSEI'
      },
      {
        'name': 'ADANI PORT SPECIAL',
        'ticker_code': 'ADANIPORTS.NS',
        'exchange': 'NSEI'
      }
    ],
    'created_at': datetime.datetime(2024, 1, 31, 8, 54, 28, 279000),
    'updated_at': datetime.datetime(2024, 1, 31, 10, 10, 28, 30000)
  },
  {
    'id': '650a1f9a0900e479a83580',
    'portfolio_name': 'portfolio1',
    'tickers': [
      {
        'name': 'Apple Inc.',
        'ticker_code': 'AAPL',
        'exchange': 'NASDAQ'
      }
    ],
    'created_at': datetime.datetime(2024, 1, 31, 10, 23, 30, 250000),
    'updated_at': datetime.datetime(2024, 1, 31, 10, 29, 22, 110000)
  },
  {
    'id': '650a1f94bc2052f71f9f5ae4',
    'portfolio_name': 'portfolio3',
    'tickers': [
      {
        'name': 'Aadi Bioscience Inc.',
        'ticker_code': 'AADI',
        'exchange': 'NASDAQ'
      },
      {
        'name': 'ADANI PORT SPECIAL',
        'ticker_code': 'ADANIPORTS.NS',
        'exchange': 'NSEI'
      }
    ],
    'created_at': datetime.datetime(2024, 1, 31, 10, 24, 51, 983000),
    'updated_at': datetime.datetime(2024, 1, 31, 10, 29, 22, 815000)
  },
  {
    'id': '650a2a016666ac1fc6678540',
    'portfolio_name': 'my_first_portfolio',
    'tickers': None,
    'created_at': datetime.datetime(2024, 1, 31, 11, 7, 45, 68000),
    'updated_at': None
  },
  {
    'id': '650a2a0e080ba7a25ff6e2b',
    'portfolio_name': 'portfolio100',
    'tickers': None,
    'created_at': datetime.datetime(2024, 1, 31, 11, 11, 42, 761000),
    'updated_at': None
  }
]

```

```

shawn@KING1: /NDIR64 --OndrE - University of Southern California/Assignments & Labs/usc-dcc1568-dspp-sp24/lab3/stock-market-analysis/src (main)
$ python main.py portfolio add-stock --username john doe --password secretpassword123 --ticker-code XXXX --portfolio_id 69ba2ae8dbaba7425f7c2b
(01/21/24 03:27:03) INFO User verification successful
(01/21/24 03:27:03) INFO Successfully added XXXX to 'portfolio1090'.
manager.py:55 manager.py:114

shawn@KING1: /NDIR64 --OndrE - University of Southern California/Assignments & Labs/usc-dcc1568-dspp-sp24/lab3/stock-market-analysis/src (main)
$ python main.py portfolio add-stock --username john doe --password secretpassword123 --ticker-code A --portfolio_id 69ba2ae8dbaba7425f7c2b
(01/21/24 03:27:35) INFO User verification successful
(01/21/24 03:27:35) INFO Successfully added A to 'portfolio1090'.
manager.py:55 manager.py:114

shawn@KING1: /NDIR64 --OndrE - University of Southern California/Assignments & Labs/usc-dcc1568-dspp-sp24/lab3/stock-market-analysis/src (main)
$ python main.py portfolio add-stock --username john doe --password secretpassword123 --ticker-code A --portfolio_id 69ba2ae8dbaba7425f7c2b
(01/21/24 03:27:40) INFO User verification successful
WARNING Ticker A is already present in the 'portfolio1090'. Cannot be added again.
manager.py:55 manager.py:99

shawn@KING1: /NDIR64 --OndrE - University of Southern California/Assignments & Labs/usc-dcc1568-dspp-sp24/lab3/stock-market-analysis/src (main)
$ python main.py portfolio add-stock --username john doe --password secretpassword123 --ticker-code A --portfolio_id 69ba2ae8dbaba7425f7c2b
(01/21/24 03:27:43) INFO User verification successful
WARNING Ticker A is already present in the 'portfolio1090'. Cannot be added again.
manager.py:55 manager.py:99

```

```

$ awscli -tD01_HD646 --qdevfs - University of Southern California/Assignments & Labs/usc-61568-dp24/lab3/stock-market-analysis/src (main)
$ python main.py portfolio add-stock --username dan_mun --password dan_password --portfolio 6156802756767a7682d464e6 --ticker-code HENXUNMLVR-NS
(07/31/24 21:48:30) INFO User verification successful
INFO Ticker code not found in the info collection. Querying the Yahoo Finance service.
INFO Successfully added HENXUNMLVR-NS to 'my_first_portfolio'.
manager.py:65
manager.py:130
manager.py:135

$ awscli -tD01_HD646 --qdevfs - University of Southern California/Assignments & Labs/usc-61568-dp24/lab3/stock-market-analysis/src (main)
$ python main.py portfolio add-stock --username dan_mun --password dan_password --portfolio 6156802756767a7682d464e6 --ticker-code BPCL-NS
(07/31/24 21:49:02) INFO User verification successful
INFO Ticker code not found in the info collection. Querying the Yahoo Finance service.
INFO Successfully added BPCL-NS to 'my_first_portfolio'.
manager.py:65
manager.py:130
manager.py:135

```

```

$ cd /Users/robert/PycharmProjects/stock_market_analysis/src
$ python main.py portfolio add stock --email dan@dan.com --password dan_password --portfolio_id 6900270c37671a708264Ae6 --ticker_code BPCL.NS
(09/03/24, 21:48:08) INFO User verification successful.
WARNING Ticker BPCL.NS is already present in the my_first_portfolio. Cannot be added again.
manager.py:45
manager.py:165

```

```
shawn@KI-KIND: ~/NINJA64 -/OneDrive - University of Southern California/Assignments & Labs/usac-dci568-sdp2-sg24/lab3/stock-market-analysis/src (main)
$ python main.py portfolio add-stock --username dan_mun --password dan_password --portfolio_id 636267567c7610a7b6826466 --ticker-code BPCL.NSLQ
400 [2024-02-21 19:50:55] INFO Use verification service
400 [2024-02-21 19:50:55] INFO Ticker code not found in the info collection. Querying the Yahoo Finance service.
400 [2024-02-21 19:50:57] ERROR 404 Client Error: Not Found for url: https://query2.finance.yahoo.com/v8/finance/quoteSummary/BPCL.NSLQ?modules=financialData&quoteType=price&defaultStatisticsContext=Profile&quoteSummaryDetail=on&includeDomain=finance.yahoo.com&formatted=false&symbol=BPCL.NSLQ&crumb=jx61642698
400 [2024-02-21 19:50:57] ERROR Invalid ticker code "BPCL.NSLQ"
y.f.py:5 manager.py:65
y.f.py:139
```


Remove Stock from Portfolio

```
shah@KI-KING: /Ondrive - University of Southern California/Assignments & Labs/usc-dsci560-dpp-sp24/lab3/stock-market-analysis/src (main)
$ python main.py portfolio remove-stock --username john_doe --password secretpassword123 --ticker_code A --portfolio_id 65ba2aeb08da7425f7fe2b
[05/31/24 03:28:53] INFO User verification successful. manager.py:155
INFO Successfully removed 'A' from portfolio198. manager.py:137

shah@KI-KING: /Ondrive - University of Southern California/Assignments & Labs/usc-dsci560-dpp-sp24/lab3/stock-market-analysis/src (main)
$ python main.py portfolio remove-stock --username john_doe --password secretpassword123 --portfolio_id 65ba2aeb08da7425f7fe2b
[05/31/24 03:28:54] INFO User verification successful. manager.py:155
WARNING Ticker 'A' not found in the portfolio. manager.py:161
```

Remove Portfolio

```
shah@KI-KING: /Ondrive - University of Southern California/Assignments & Labs/usc-dsci560-dpp-sp24/lab3/stock-market-analysis/src (main)
$ python main.py portfolio fetch-stock-data --username john_doe --password secretpassword123 --portfolio_id 65ba2aeb08da7425f7fe2b
[05/31/24 03:29:56] INFO User verification successful. manager.py:155
ERROR Invalid portfolio id: '65ba2aeb08da7425f7fe2b' manager.py:158

shah@KI-KING: /Ondrive - University of Southern California/Assignments & Labs/usc-dsci560-dpp-sp24/lab3/stock-market-analysis/src (main)
$ python main.py portfolio create --username john_doe --password secretpassword123 --portfolio_name my_first_portfolio
[05/31/24 03:30:04] INFO User verification successful. manager.py:155
INFO Created portfolio with name: 'my_first_portfolio' and portfolio_id: '65ba304063626911aabd278e' successfully. manager.py:181

shah@KI-KING: /Ondrive - University of Southern California/Assignments & Labs/usc-dsci560-dpp-sp24/lab3/stock-market-analysis/src (main)
$ python main.py portfolio remove --username john_doe --password secretpassword123 --portfolio_id 65ba304063626911aabd278e
[05/31/24 03:30:05] INFO User verification successful. manager.py:155
INFO Successfully removed portfolio 'my_first_portfolio'. manager.py:156
```

Fetch OHLC Data for One Stock

```
shah@KI-KING: /Ondrive - University of Southern California/Assignments & Labs/usc-dsci560-dpp-sp24/lab3/stock-market-analysis/src (main)
$ python main.py market-data fetch-stock-data --username john_doe --password secretpassword123 --ticker_code GOO
[05/31/24 10:57:05] INFO User verification successful. manager.py:65
INFO Fetching data for stock 'Alphabet Inc. (GOO)'. manager.py:222
INFO Fetching additional data for 'GOO' from 2004-01-31 00:00:00 to 2004-01-01 00:00:00. manager.py:247
WARNING: cookie & crumb does not work well with requests_cache. An experimenting with 'expire_after=00_NOT_CACHE', but you need to help stress-test.
[*****] 1 of 1 completed
[05/31/24 10:57:06] INFO Fetching old historical data for 'GOO' from None to 2004-01-18 00:00:00. manager.py:283
[*****] 1 of 1 completed

1 failed download:
[*****] Exception("Wicker%: Data doesn't exist for startDate = -1415318400, endDate = 1092081600")

[05/31/24 10:57:11] INFO
datetime open high low close adj_close volume
0 2004-01-19 2.49864 2.59185 2.30892 2.49913 2.49913 8.97472e+08
1 2004-01-20 2.515820 2.716817 2.503118 2.697639 2.697639 4.588575e+08
2 2004-01-21 2.596684 2.753347 2.574102 2.706688 2.706688 4.281918e+08
3 2004-01-22 2.677547 2.782676 2.654886 2.715736 2.715736 3.973345e+08
4 2004-01-23 2.758411 2.826486 2.716070 2.724787 2.724787 3.668579e+08
...
7180 2004-01-27 153.126653 154.473333 152.840003 154.139994 154.139994 1.995883e+07
7181 2004-01-28 153.383331 154.836665 152.880000 154.489995 154.489995 2.043407e+07
7182 2004-01-29 153.139999 155.199997 152.919998 154.839996 154.839996 2.009538e+07
7183 2004-01-30 154.489995 155.839993 152.774994 153.620003 153.620003 2.653708e+07
7184 2004-01-31 145.389999 145.589996 141.550003 141.800003 141.800003 4.383368e+07

[7185 rows x 7 columns]
```

Fetch Portfolio by ID

```
shah@KI-KING: /Ondrive - University of Southern California/Assignments & Labs/usc-dsci560-dpp-sp24/lab3/stock-market-analysis/src (main)
$ python main.py portfolio fetch-one --username john_doe --password secretpassword123 --portfolio_id 65ba2aeb08da7425f7fe2b
[05/31/24 20:00:15] INFO User verification successful. manager.py:65
INFO User's Portfolio: main.py:137
{
  'id': '65ba2aeb08da7425f7fe2b',
  'portfolio_name': 'portfolio198',
  'tickers': [
    {
      'name': 'Alphabet Inc.',
      'ticker_code': 'GOO',
      'exchange': 'NASDAQ'
    }
  ],
  'created_at': datetime.datetime(2024, 1, 31, 11, 42, 761000),
  'updated_at': datetime.datetime(2024, 1, 31, 11, 28, 781000)
}
```

Fetch data for all stocks by portfolio ID

```
shah@C1-KDZL MING64 ~/OneDrive - University of Southern California/Assignments & Labs/usc-dsci560-spp-sp24/lab3/stock-market-analysis/src (main)
$ python main.py portfolio fetch-portfolio-data --username john_doe --password secretpassword123 --portfolio_id 650a8ace4178f1b53babac
[01/31/24 20:56:11] INFO User verification successful.
INFO Ticker code not found in the info collection. Querying the Yahoo Finance service.
[01/31/24 20:56:12] INFO Fetching data for stock 'ADAM POWER LTD (ADAMPWR.LS)'
WARNING Most recent date not found in 'ADAMPWR.LS' data collection. Fetching full data.
[*****] 1 of 1 completed
[01/31/24 20:56:19] INFO Fetched data for 'ADAMPWR.LS'. Most recent data is updated.
INFO Updated ticker details for 'ADAMPWR.LS': {'name': 'ADAM POWER LTD', 'ticker_code': 'ADAMPWR.LS', 'exchange': 'NSI'}
[01/31/24 20:56:20] INFO
datetime      open      high      low      close      adj_close      volume
0 2009-08-26  108.000000  110.000000  98.300000  108.059998  108.059998  1.545283e+06
1 2009-08-27  99.500000  101.500000  98.000000  101.150000  101.150000  1.321756e+07
2 2009-08-28  101.233335  104.066668  99.183334  103.133334  103.133334  2.783365e+07
3 2009-08-29  100.566668  104.233335  100.366669  103.116666  103.116666  2.818947e+07
4 2009-08-30  103.000000  104.400000  101.520000  103.800000  103.800000  1.365258e+07
...
...
...
5273 2024-01-27  546.200012  557.720006  536.750000  550.870000  550.870000  8.952866e+06
5274 2024-01-28  558.100006  564.087500  544.625000  563.662506  563.662506  4.861675e+06
5275 2024-01-29  570.000000  570.450012  552.500000  570.450012  570.450012  5.730005e+06
5276 2024-01-30  575.000000  579.799988  567.000000  578.299988  578.299988  2.962115e+06
5277 2024-01-31  558.500000  570.250008  558.000000  562.700012  562.700012  1.915610e+06
[5278 rows x 7 columns]
[01/31/24 20:56:20] INFO Fetching data for stock 'ADAM PORT SPECIAL (ADAMPRTS.LS)'
WARNING Most recent date not found in 'ADAMPRTS.LS' data collection. Fetching full data.
[*****] 1 of 1 completed
[01/31/24 20:56:27] INFO Fetched data for 'ADAMPRTS.LS'. Most recent data is updated.
INFO Updated ticker details for 'ADAMPRTS.LS': {'name': 'ADAM PORT SPECIAL', 'ticker_code': 'ADAMPRTS.LS', 'exchange': 'NSI'}
[01/31/24 20:56:27] INFO
datetime      open      high      low      close      adj_close      volume
0 2007-11-27  154.000000  207.000000  154.000000  191.800003  175.274399  2.726236e+07
1 2007-11-28  194.000000  197.000003  174.925993  177.000000  161.749619  2.195892e+07
2 2007-11-29  181.000000  182.000003  168.460007  177.999994  162.111582  2.561672e+07
3 2007-11-30  178.000000  191.399994  178.000000  185.800003  169.791397  2.304704e+07
4 2007-12-01  181.316666  193.886663  180.133331  189.200002  172.898448  2.822431e+07
...
...
...
5905 2024-01-27  1138.575012  1180.325012  1131.299988  1171.475017  1171.475017  5.536538e+06
5906 2024-01-28  1146.787500  1150.612518  1141.149994  1184.062531  1184.062531  6.152285e+06
5907 2024-01-29  1152.000000  1204.000024  1151.000000  1196.630024  1196.630024  6.708032e+06
5908 2024-01-30  1200.000000  1217.199951  1181.000000  1187.349976  1187.349976  4.681729e+06
5909 2024-01-31  1195.800049  1214.250000  1191.650024  1207.650024  1207.650024  5.518242e+06
[5910 rows x 7 columns]

shah@C1-KDZL MING64 ~/OneDrive - University of Southern California/Assignments & Labs/usc-dsci560-spp-sp24/lab3/stock-market-analysis/src (main)
$ python main.py portfolio fetch-portfolio-data --username john_doe --password secretpassword123 --portfolio_id 650b27be5e7671a7682d4d4e
[01/31/24 21:05:50] INFO User verification successful.
[01/31/24 21:06:07] ERROR Portfolio with id '650b27be5e7671a7682d4d4e' not found. Please retry with correct id.
```

Individual Contributions

Meghana

- Integration of Yahoo Finance API
- Cleaning & Pre-processing of the retrieved OHLC data
- Readme for setup and usage of the command line toolkit

Kayvan

- Data Validation and Data Modelling
- Portfolio Management
- Stocks data management
 - Maintenance of ticker metadata, and stocks data
 - Fall back strategy to Yahoo Finance API if data is missing from the database
- Develop the command line toolkit

Shreyansh

- MongoDB Backend & Setup
- Data Retrieval from Yahoo finance and Ingestion to MongoDB
- User Credentials Validation for Portfolio Management