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)

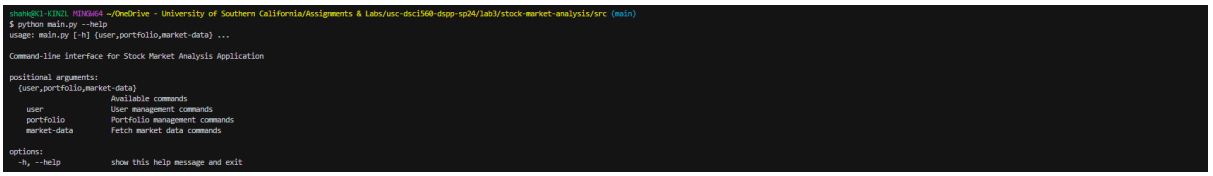
- Creates, removes, and lists user portfolios.
- Adds and removes stocks from portfolios.
- Manages stocks associated with each portfolio.
- Stores data in MongoDB collections.

About the Code

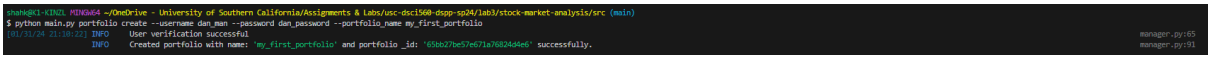
- **db.py:** This file contains code related to database connectivity and configuration using MongoDB. It defines a PyObjectId class for handling ObjectIds and creates cursors for different collections such as users, tickers_info, and portfolios.
- **main.py:** The main.py file is the entry point for 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

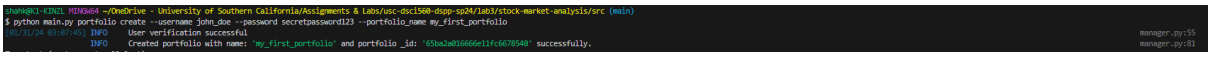


Create a new user



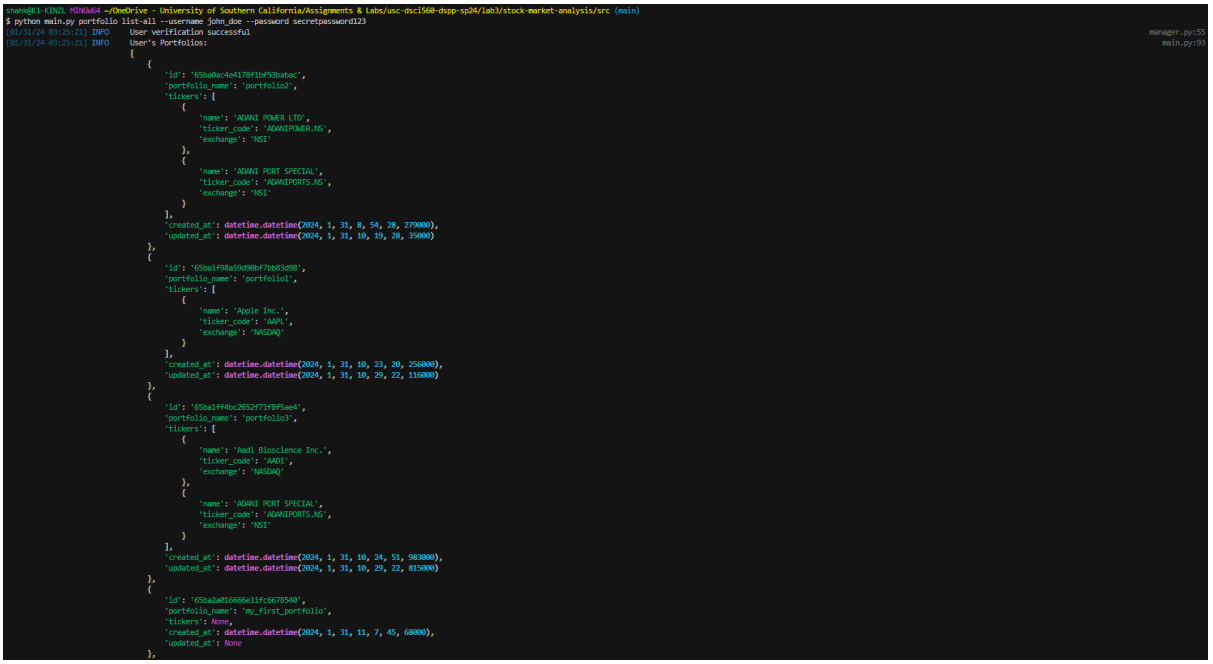
Get user information

Create Portfolio

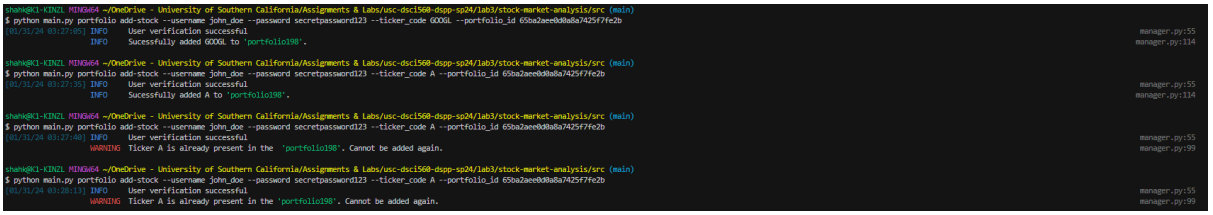


Create portfolio with same name

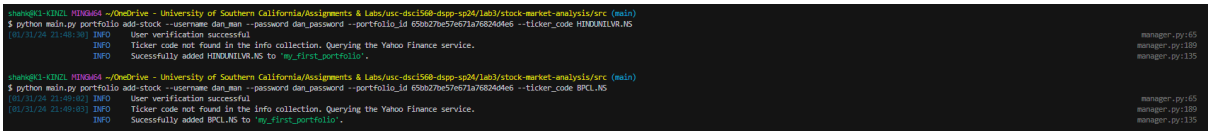
Get the List of Portfolios for the User



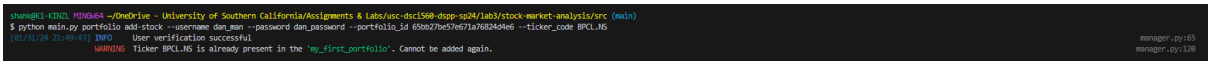
Add Stock to Portfolio



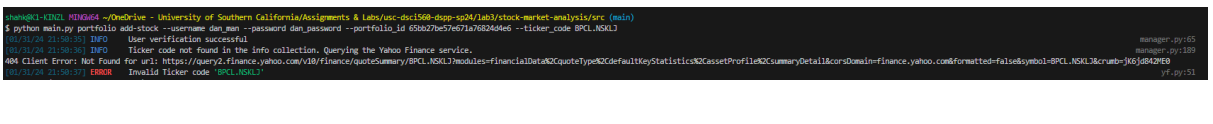
Adding stocks not in ticker list



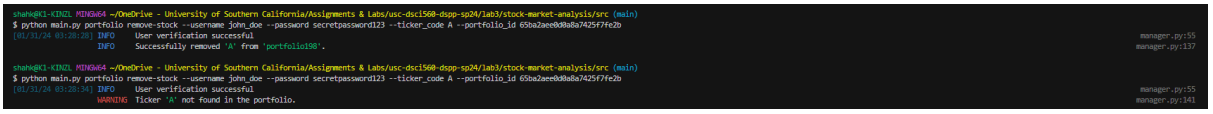
Adding the same stock



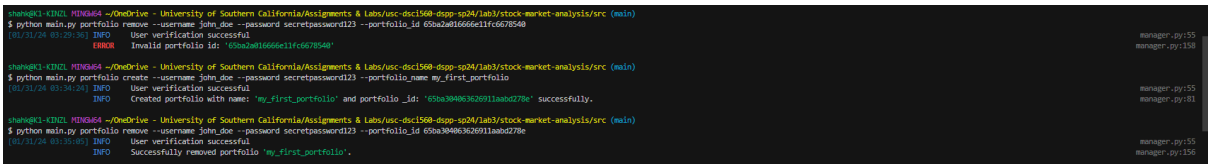
Adding stock that doesn't exist



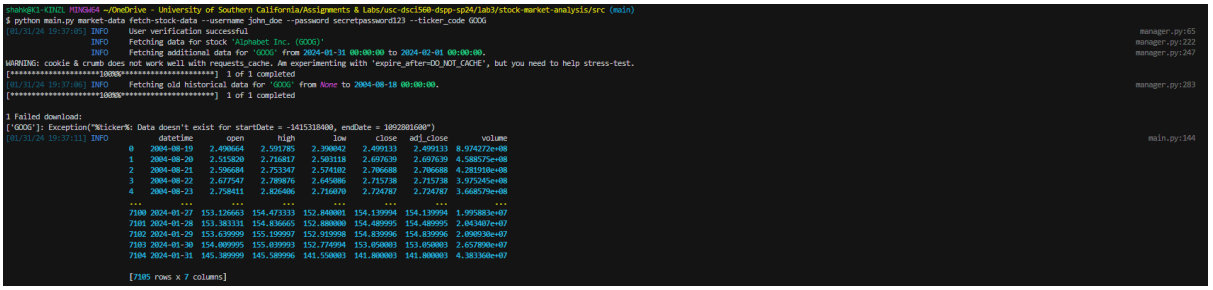
Remove Stock from Portfolio



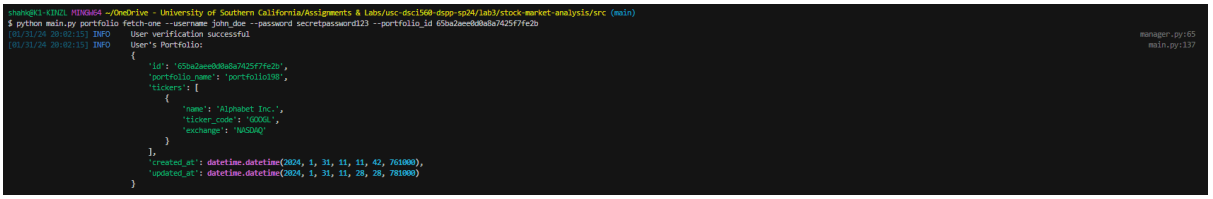
Remove Portfolio



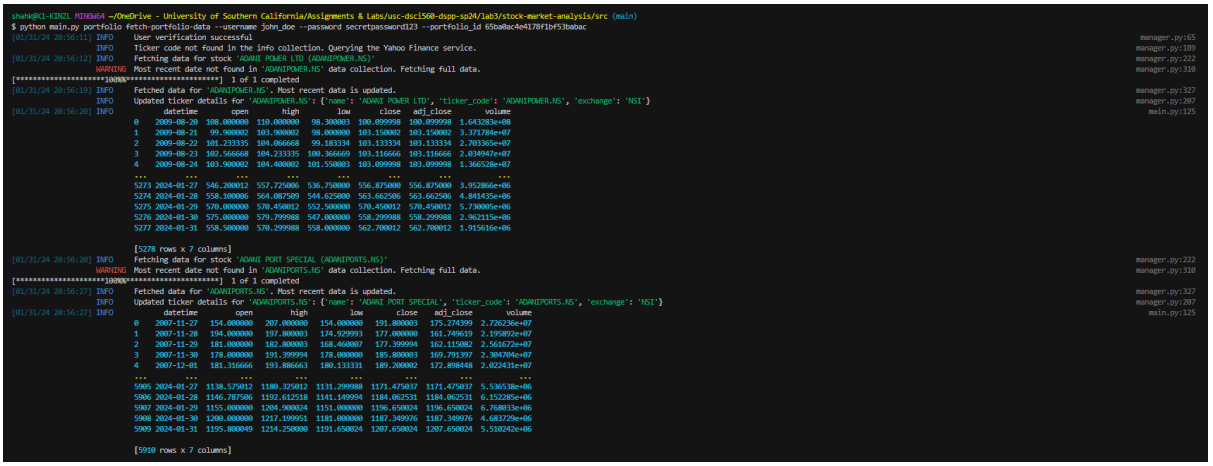
Fetch OHLC Data for One Stock

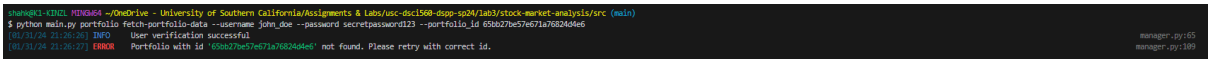


Fetch Portfolio by ID



Fetch data for all stocks by portfolio 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