Software Requirements Documentation (SRD)

Cwacos (Cloudy With A Chance Of Stonks)

10 March 2021

Fig.1 - Cwacos the stonks loving Quakka

Prepared for: **Ike Quigley - UNCG CSC 340**

1.0 Introduction

1.1 Team Members:

1.1.1 Project Name

Cwacos (Cloudy With A Chance Of Stonks)

1.1.2 Team Name

Buy GME

1.1.3 Date

3/15/2021

1.1.4 Team Members

Anthony Mesa - *GitHub Administrator*
Abdoul Djido - *General Code Wrangler*
Hyoungjin(Alan) Choi - *Persistent Data storage*
Jack Fink - *UI/Front-End Developer*
Michael Leonard - *API Connection*

1.1.5 Stakeholders/Company - Honor Code

We will not share user information with any third party organizations. We will not install any third party software outside of the designed application. Our application will not mislead nor mis-inform our users.

1.2 Table of Contents:

1.3 Purpose:

- To provide a platform that allows users to view and save current stock and cryptocurrency index data as well as tools to analyze the market data collected, providing metric guided insight for possible buy/sell opportunities in the future.

1.4 Documentation Conventions:

- Source code and github commits will all be properly commented and explained. We will use the documentation techniques outlined in the UNCG CSC 340 style guide provided by Prof. Ike Quigley.

1.5 Intended Audience:

- Any entry level stock enthusiast with access to a computer capable of running Java (who loves quokkas). There are no age, gender, or income constraints on our intended audience.

1.6 Definitions (Jargon):

- **Stonks:** Stocks
- **Quokka:** A small macropod about the size of a domestic cat, also known as the short-tailed scrub wallaby.

1.7 Project Scope:

- Our Software will be a portable Java application utilizing the JavaFX UI framework. It will use AlphaVantage API to connect to current stock and crypto-currency information, as well as the RandomFacts API for Quokka facts. These APIs were chosen for their affordability and ease of use. Data will be stored in and retrieved from text-based documents, through Java's file reading libraries to avoid the complications of an SQL database.

1.8 Technical Challenges

- The use of the Random Facts API is limited to 5 free calls per day. We will only make up to 5 calls per day using this API and store the facts locally for future use.
- The Alpha Vantage API is also limited to only 5 calls per minute and 500 calls per day. We will limit our users' ability to make API calls so this limit is not reached.
- Due to the fact that locally stored text files are used for the application's persistent database, there is the possibility of permission errors.

1.9 References

- API Documentation:
    - AlphaVantage - https://www.alphavantage.co/documentation/
    - Random Facts - https://fungenerators.com/api/facts/

2.0 Overall Description

2.1 Product Features

- Search, view, and save current stock and crypto index data
- Generate candlestick graph reports from most recently downloaded stock and crypto data
- Analyze stored intraday information to find maximum possible profit within given time frames using an efficient divide and conquer algorithm.
- Persistent storage of current session data for later viewing

2.2 User characteristics

- Loves quokkas
- Likes to keep up with stock and crypto currency

2.3 Operating Environment

- Any modern desktop OS with Java installed and access to WIFI

2.4 Design and Implementation Constraints

- The AlphaVantage free-use tier only allows for 5 requests per minute at a maximum of 500 requests per day. Given this, these constraints will also be applied to the user's market data requests within the application.
- The AlphaVantage API's crypto-currency requests do not include intraday requests, so crypto data requests will only provide new market info every 24 hours.
- Given that this application does not implement individual users, local file(s) will be stored on the user's computer to provide persistent storage.

2.5 Assumptions and Dependencies

- To compile this project from source, these libraries are required:
    - JavaFX
    - JSON parser: org.json.jar

- For users to use this application in the way that it was intended, it is assumed of the user and/or their desktop environment that:
    - Java 15 JRE is installed and PATH’ed correctly
    - An internet connection (either wired or wireless) is available
    - Local storage exists (i.e. SSD, USB) and can be written to
    - Program is run from local storage with read/write permission
    - User can both read and count
    - User is familiar with candlestick charts

3.0 Function Requirements

3.1 Primary

- Stock will be interacted with via text windows, buttons, and graphs in a javaFX based UI.
    - User will input via keyboard a stock symbol composed of alphabetical characters with a maximum length no longer than 5, to define desired stocks.
- Input from UI will be transferred to the corresponding Classes for API calls and data storage.
    - Validity of inputs - such as stock symbols longer than 5 characters, or the existence of numbers in the input - will be checked at this level and handled with proper exceptions.
- Outputs will be displayed to the screen using the javaFX UI.
    - Raw text based output such as stock intraday information can be viewed in output boxes in the program’s main window.
    - Intraday information about stocks can be viewed as a candle-stick graph.
    - Intraday information can also be analyzed by the MaxProfit algorithm to access the largest possible profit margin from trading within a given period as well as buy and sell points.
- Error Handling will be taken care of using popup dialogue windows over the main UI window.
- Upon application start, a random Quokka fact will open in a dialogue box that the user must close before using the program.

3.2 Secondary

- The MaxProfit Algorithm will take an array of data composed of a stock’s value at different points in time. This data is retrieved by the API class and is then the data is passed to the MaxProfit Algorithm class. The array given must contain only doubles and all values must be sorted by time in ascending order. The algorithm uses a divide-and-conquer design to determine the buying and selling points that result in the largest net profit. This profit is then finally displayed in a UI window when the algorithm is used. The algorithm has a runtime of O(nlogn).

4.0 Technical Requirements

4.1 Operating Systems/Compatibility

- Windows
- OSX
- Linux

4.2 Interface Requirements

4.2.1 User Interface

- JavaFX

4.2.2 Hardware Interface

- 2GB Ram
- 256MB Storage
- Networking Hardware

4.2.3 Software Interface

- Java version 15

4.2.4 Communications Interface

- HTTPS verification

5.0 Nonfunctional Requirements

5.1 Performance Requirements

- Algorithms and logic designs must be composed so that runtimes are reasonable (no exponential runtimes) for the size of inputs they will be given. This is done for a user friendly design with fast load times.

5.2 Safety/Recovery Requirements

- Beyond providing a way for the user to save market data to local storage, no other recovery options will be provided. It is therefore upon the user to ensure that data obtained using Cwacos is stored in a secure location consistently.

5.3 Security Requirements

- Data downloaded is publicly available market data and as such, security should not be a concern in regards to downloaded data.
- Locally saved user session data is not encrypted, and thus it is upon the user to secure their locally saved data in any way they see fit, noting though that said data must return to its original form for Cwacos to load it correctly.

5.4 Policy Requirements

- This software has no policy requirements.

5.5 Software Quality Attributes

5.5.1 Availability

- This software is only available for download in the Cwacos github repo. All information gathered is at the mercy of AlphaVantage and Random Facts API's and thus if they stop working, Cwacos stops working.

5.5.2 Correctness

- Current crypto-currency data is updated every 24 hours, whereas stock data is updated every 24 hours but also with intra-day information being updated throughout normal market operating hours. Because of this limitation, we cannot guarantee the accuracy of this data. To be clear, this app should not be used for up-to-date market insight for trading stocks, and we are not liable for any poor gambling choices made using it.

5.5.3 Maintainability

- This application will not be maintained.

5.5.4 Reusability

- This program is portable and saves files in a local folder defined by the user, so you can reuse this bad baby anywhere your heart desires.

5.5.5 Portability

- This application is 100% portable - but only if the host has a JRE installed.

5.6 Process Requirements

5.6.1 Development Process Used

- Waterfall

5.6.2 Time Constraints

- Must be done by April 30th, 2021.

5.6.3 Cost and Delivery Date

- Cost of the software is naught but a little bit of sanity, and sleep or the lack thereof.