Requirements Analysis

Jay Ng, Suzanne Candanedo, Adam Dodson, Heather Clark, Amber Rosevear, Ash Brent-Carpenter

I. Preface

This document details requirements from both the point of view of the customer and the team that will be managing, developing, testing and maintaining the project. It aims to define in detail what the software should do to fulfill the needs of the customer along with both the goals and constraints of the system.

II. Introduction

Chatbots, or virtual assistants, are a new and emerging technology and when employed effectively can increase productivity. The task detailed in this document is to build a chatbot which can assist traders at Deutsche Bank. The system will collect data on the news and FTSE 100, respond to relevant queries, and will employ artificial intelligence for personalisation.

III. Team Organisation

Each member of the team has been assigned a specific role defining their specialisation within the project. Those with an applicable focus will be responsible for addressing the relevant requirements, and then taking the lead on designing and implementing the solution that stems from them.

- Adam Dodson, Business Analyst - interacts with the customer
- Heather Clark, Project Manager - oversees the planning and organisation of the team
- Amber Rosevear, Lead Developer - focuses on the chatbot and AI technology
- Jay Ng, Software Developer and Financial Analyst - focuses on data collection
- Suzanne Candanedo, Quality Assurance and Software Testing - supports the lead developer
- Ash Brent-Carpenter, Software Developer and Webmaster - focuses on the application backend

The team holds weekly meetings in which there is a set agenda and recorded minutes. During these meetings the completed work is reviewed and new work is assigned. Due to the small nature of the team decisions are democratically decided where possible otherwise the Project Manager has the final say.

IV. Stakeholders

The primary stakeholders of this project are those whom the outcome will affect, chiefly those directly engaged with the creation of the software such as engineers and testers, which in this instance is the team. Other stakeholders are also the managers responsible for setting the task which in this project is Deutsche Bank and the university. In this case, these identified parties are also the customer as well.

V. Requirements Specification

Requirements have been categorised from most important to least: must have: [M], should have: [S], could have: [C] and will not have: [W].

A. Functional

Requirement	Customer	Developer	Cat.
R1 Accessing and interfacing with FTSE 100 stock data			
R1.1 Simple stock data	The real time price, and daily change of stocks can be returned when requested. Basic information about each of the companies on the index including sector and subsector are also accessible.	Maintain a database which contains all 100 FTSE 100 companies. For each of these companies the full name, ticker symbol and sector, including subsector information must be stored. As the companies included in the index are reviewed quarterly this list must update at least that often. Use an API to access the changing stock data.	[M]
R1.2 Specific stock data	For every stock on the FTSE 100 more detailed information is requestable, including high/low along with the open/close price, the percentage change in stock price and the volume of the stock traded. For specifiable range of periods of time including the current day and the year.	Use an API to access the requested stock data and allow for data to be retrieved repeatedly with small interval. Interval must be within 5 minutes, could be up to every 1 minute. Allow the user to specify the time range that the data applies to.	[S]

R1.3 Multi-stock support	Allow stocks to be separated by their sector and be kept in a portfolio. Relevant data (R1.1-2) should be requestable concerning these stocks.	Database of stocks supports query grouping by sector. Allow for stocks to be placed in a separate list or table to enable portfolio functionality.	[C]
R1.4 Graphs and diagrams	Visualisations of the collected data detailed in R1.1-2 can be drawn. Allow axis to be specified by the user (e.g. time intervals). Should have the option to superimpose graphs on one another for comparison.	Store the retrieved data in a format supported by the graphing package. Graphs should be separate part of the UI that freely supports movement.	[C]
R1.5 Other indexes and currencies	Be able to request both the current and historical exchange rate of currencies such as GBP, Euro, USD and Yen. Access data, as detailed in R1.1-4 for other indexes such as the Dow Jones.	Use an API to request currency data and other indexes data for any time period specified by the user.	[C]
R1.6 Further stock calculations	Perform Alpha and Beta data calculations.	Accurately perform requested calculation on the requested data set.	[C]
R2 Receive and respond to queries from the user			
R2.1 Preset queries	Frequently asked and user-defined queries should be saved and their answers immediately available.	AI should automate queries on specific stocks and store a cache of the data. AI should store a history/heatmap of queries from the user to weight the likelihood of repeat queries	[C]
R2.2 Queries on stock data	The chatbot should be able to answer queries about any piece of accessible data detailed in R1.1-2.	The chatbot needs to parse input from the user and identify the stock, then perform an API request to look up the corresponding data to return to the user.	[M]
R2.2.1 Group stock data queries	This is dependent on R1.3. The chatbot should be able to recognize stocks by sector, subsector, and portfolio. Be able to identify if the group and which individual stocks are rising or falling.	Perform calculations on groups of multiple stocks for detecting increasing or decreasing value over a set time period. As with R2.2 the chatbot should use an API to lookup corresponding data for each stock in the group.	[C]
R2.2.2 Currency queries	This is dependent on R1.5. The chatbot should have support for questions that ask about the data collected about currencies as detailed in R1.5.	The chatbot should an API to lookup the data on the currency requested by the user. Any calculations necessary should be performed and the result should be returned.	[C]
R2.2.3 News queries	The chatbot should be able to recover news from feeds on both companies and sectors and determine if that news is positive or negative.	The chatbot should use an API to request news stories about a user-defined company or sector. Sentiment analysis should be carried out the stories.	[M]
R3 Chatbot with AI capabilities			
R3.1 Prediction	Provide suggestions for buying, selling, and trading by analysing current stock trends and analyst ratings. Monitor for boosts in news coverage concerning sectors or companies that the user is likely to be interested in.	Provide automated notifications at set time periods or triggered by news related to a company or sector defined by the user.	[C]
R3.1.1 Summary	On startup present a summary digest on stocks defined by the user that gives information on the opening and closing of trading. Should highlight any news or notable changes that happened overnight.	Automated Query for important stocks/commodities/currencies over the interval of trading day closing in the UK to trading day opening in the UK.	[S]
R3.2 Language Processing	User should be able to communicate with program in simple sentences and the program should understand what their intent is.	The AI should decompose the natural language inputs into their smallest syntactical units. The AI should be able to map natural language inputs into a form it can interpret.	[M]
R3.2.1 Simple statements processing	The user should be able to enter a query and have the most correct answer returned. The AI will interact with the user in British English and will understand basic finance jargon.	Input analysis should consider context and place heavier probabilistic weights on financial and calculation phrases when interpreting the input. The AI should interpret well formed queries by identifying informative keywords, and request the desired information and relay it back to the user when received.	[M]

R3.2.2 Advanced statement processing	The AI should be able to parse grammatically incorrect queries and suggest the closest alternative.	Individual word mistakes should be resolved by finding the word most likely to be correct based on the syntax and grammar of the input.	[S]
R3.3 System must learn about the trader	The AI must individualise its suggestions and trading advice to its main user.	Store queries frequently asked by the user to allow prefetching of requested information. Track common mistakes and make automatic corrections without needing to be asked.	[M]
R3.3.1 Portfolio system	AI should learn the user's investment goals, either by simple input or monitoring usage. Consider the trader's investment goals when making response suggestions for possible changes in value within the portfolio.	Support R3.2.1-2 for the trader's entire portfolio; extracting what investments the traders holds and automatically checking for information concerning them.	[C]
R3.3.2 Favourites system	A user-selected collection of stocks should have their information without asking directly. A preferred time period should be selectable to avoid specifying the time interval every query; defaulting to on the hour when a favourite stock opens/closes for the day.	Prefetch data on a number of user defined stocks to have their information immediately available to the trader.	[C]

B. Non Functional

Requirement	Customer	Developer	Cat.
R4 Suitability for non-technical users			
R4.1 Intuitive Design	A non-technical user should be able to use the system with minimal instruction and where to find any help they require should be obvious.	Provide clear labelling and descriptive titles to all elements. To describe more complex functionality pop out information boxes should be used.	[M]
R4.2 Follow UI conventions	The layout should be logically designed and similar to that of a existing applications that users are likely to be familiar with.	Interactive elements should be large and minimal in number. These elements should be organised into logical group; keeping those with related functionality closest.[1]	[S]
R4.3 Documentation	Provide help documentation that gives written tutorials on how to use the chatbot.	Providing an easily accessible "Help" documentation that explains all features clearly and concisely, making use of relevant diagrams and pictures. On the first start up the user should be directed to this documentation.	[C]
R5 Optimal System performance			
R5.1 Your system must remain responsive	Aim for 99% of queries to be returned within 5 seconds. The while idle the system should always be ready to accept a query.	Make use of pipelining and start returning information as fast as possible to the user. The application should not require any manual refreshing.	[S]
R5.1 System uptime	System should be operations at least 99% of the time. Any connection issues should be communicated to the user.	Systems chosen to support the chatbot must have their reliability evaluated. Any problems with the information gathering and processing chain, or the internet connection, should be shown clearly on the interface to alert the user.	[S]
R6 Using appropriate input-output			
R6.1 Text	Support text based input and output for the chatbot	Interface needs a free text input panel with another larger panel to display the output text. These should be the main components of the interface.	[M]
R6.2 Audio	Support for both voice input and audio output. The chatbot can be activated by speaking a certain phrase.	Use the device's microphone to listen to a user's commands, process the query, and then continue the same as text based input. Use the device's speakers and text to speech processing to output the results of the query.	[C]
R6.3 Visual	Graphs to help display bulk data for a quick overview.	Interface should use a popup style panel or box to display graphs or diagrams.	[C]

R7 Interface Design and Usability	The system should provide feedback when the AI is processing information or when it does not understand a given input. Application should be minimalistic and non-intrusive; design should increase rather than obstruct workflow.	The interface should be small with minimal wasted space but this should not impede usability.	[M]
R8 Cross-device compatibility	Application should work across multiple platforms including mobile and desktop. Should support different display resolutions and orientations when viewing on mobile devices.	System should be reactive and have support for different operating systems including Windows, Linux, macOS on desktop and Android and iOS on mobile.	[S]
R9 Freedom from legal, social, ethical and professional issues	Other companies property must be used in legal and ethically compliant manner.	All developers should have their work fairly recognised. Usage policies for any existing technologies that are used must be respected.	[M]
R10 Multiple Users	Chatbot will only support a single user.	Separate profiles or accounts are not required, all settings and preferences are global.	[W]
R11 Security	Security features are not being considered.	Defending against any potential threats or exploits is not required.	[W]

VI. System Evolution

As this is a proof of concept it is focused on what can currently be created for the given application, and it is not intended to be supported in its current form for any significant length of time. Requirements have been specified that detail how it is expected that this application should be compatible with a range of technologies so it will have a high chance of being continually supported into the future.

References

[1] Laws of ux. [Online]. Available: https://lawsofux.com/