



EEE521 Final Year Project Report

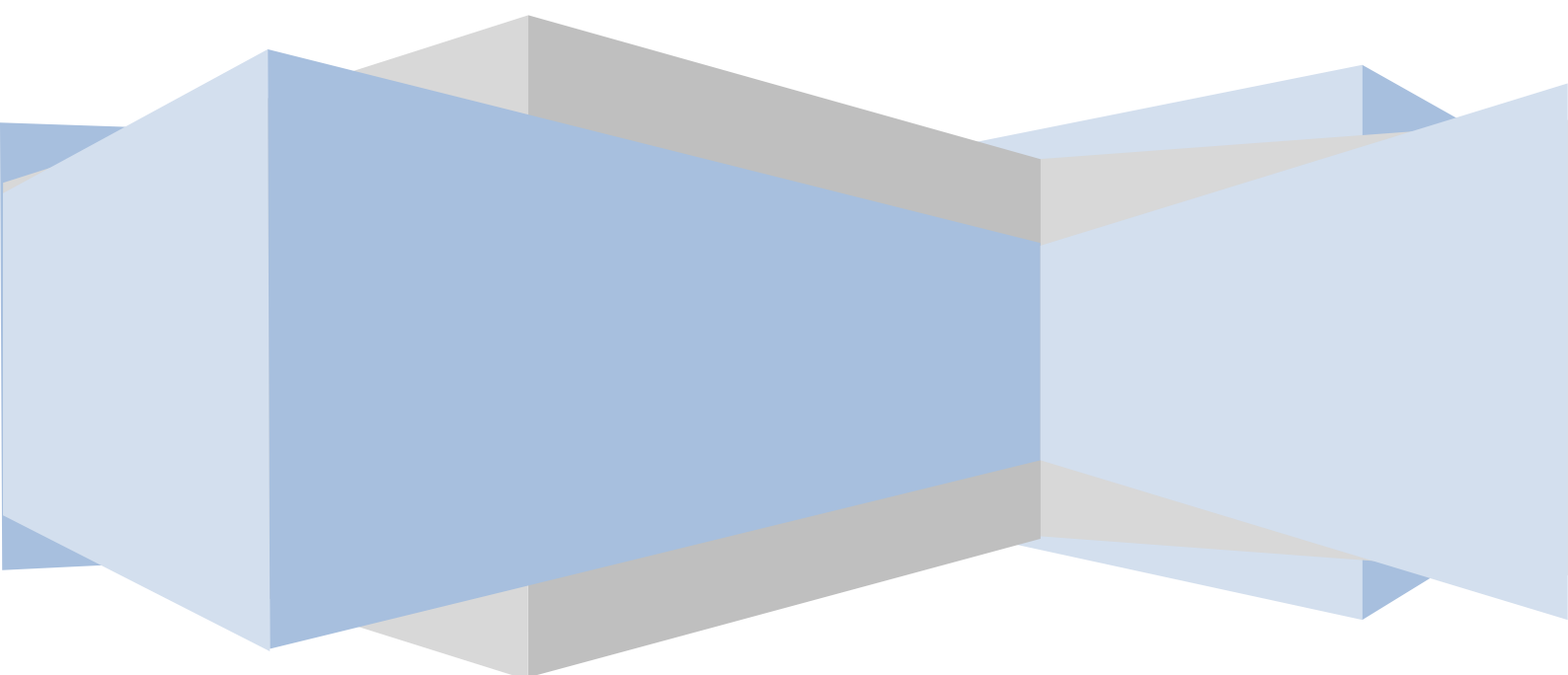
School of Computing, Engineering &
Intelligent Systems

Dana Doherty

BSc Hons Computer Science

Task-based Interaction Chatbot

Supervisor Dr. Kevin Curran
Second Marker Dr. Daniel Kelly



Plagiarism Statement

I declare that this is all my own work and does not contain unreferenced material copied from any other source. I have read the University's policy on plagiarism and understand the definition of plagiarism. If it is shown that material has been plagiarised, or I have otherwise attempted to obtain an unfair advantage for myself or others, I understand that I may face sanctions in accordance with the policies and procedures of the University. A mark of zero may be awarded and the reason for that mark will be recorded on my file.

I confirm that the Originality Score provided by TurnItIn for this report is _____.

[your signature]

Acknowledgements

I would like to take this opportunity to thank my project supervisor Dr Kevin Curran for his continuous guidance and encouragement throughout the entirety of the project. I would also like to thank Dr Daniel Kelly for his feedback and guidance during the development stage. Finally I would to thanks Dr Michael McTear for his guidance and valuable and critical input on the Testing and Evaluation Chapters.

Contents

1. INTRODUCTION	1
1.1 AIMS & OBJECTIVE	1
1.2 LITERATURE REVIEW	2
1.3 CHATBOTS IN INDUSTRY	3
1.3.3 CHATBOTS	6
1.4 NATURAL LANGUAGE UNDERSTANDING ENGINE	7
1.5 ARTIFICIAL INTELLIGENCE.....	8
1.5.1 ARTIFICIAL INTELLIGENCE METHODS	8
1.6 CHATBOT ARCHITECTURE	10
1.7 SUMMARY OF REPORT	10
1.8 CONCLUSION.....	11
2. REQUIREMENTS SPECIFICATION AND ANALYSIS.....	11
2.1 PROBLEM STATEMENT	11
2.2 PROPOSED SOLUTION	12
2.3 SOFTWARE DEVELOPMENT METHODOLOGIES	12
2.3.1 WATERFALL METHODOLOGY	13
2.3.2 INCREMENTAL MODEL.....	13
2.3.3 CHOSEN METHODOLOGY	14
2.4 INFORMATION GATHERING TECHNIQUES	14
2.4.1 CHOSEN INFORMATION GATHERING TECHNIQUE	16
2.5 FUNCTIONAL AND NON-FUNCTIONAL REQUIREMENTS.....	17
2.5.1 FUNCTIONAL REQUIREMENTS.....	18
2.5.2 NON-FUNCTIONAL REQUIREMENTS	18
2.6 SOFTWARE AND HARDWARE SPECIFICATION.....	19
2.6.1 SOFTWARE REQUIREMENTS	19
2.6.2 HARDWARE REQUIREMENTS.....	19
2.7 PROJECT PLANNING	19

2.7.1 PROJECT MILESTONES AND DELIVERABLES	20
2.7.2 DEVELOPMENT ITERATION BREAKDOWN.....	21
2.7.3 RISK MANAGEMENT.....	22
2.7.4 PROJECT MANAGEMENT.....	23
3. DESIGN.....	25
3.1 ARCHITECTURAL DESIGN.....	25
3.2 DATABASE DESIGN.....	29
3.3 CLASS DIAGRAM.....	30
3.4 SPEECH RECOGNITION	32
3.5 ACTIVITY DIAGRAMS	33
3.6 SEQUENCE DIAGRAMS	38
3.7 CONVERSATIONAL USER INTERFACE DESIGN	40
3.8 DIALOG DESIGN.....	42
4. IMPLEMENTATION AND TESTING	46
4.1 CONFIGURING THE DEVELOPMENT ENVIRONMENT:	46
4.2 AUTHENTICATION:	48
4.3 DATABASE DEVELOPMENT.....	48
4.4 GOOGLE TWO-FACTOR AUTHENTICATION:	49
4.5 OPEN BANKING API:	51
4.6 WEBHOOK DEVELOPMENT:	56
4.7 CHATBOT WEB CLIENT.....	63
4.8 GIT VERSION CONTROL.....	64
5. TESTING.....	64
5.1 SIMULATED CONVERSATIONAL TESTING	65
5.2 WEBHOOK TESTING	73
5.3 USER TESTING:.....	74
6. EVALUATION.....	75
6.1 CONCLUSION.....	79
6.2 FUTURE ADVANCEMENTS.....	79

6. REFERENCES	2
HTTPS://WWW.BBA.ORG.UK/WP- CONTENT/UPLOADS/2014/06/BBA_COMPETITION_REPORT_23.06_WEB_2.0.PDF ..	4
7.APPENDICES	8
7.1 APPENDIX A QUESTIONNAIRE RESULTS USER REQUIREMENTS	9

Abstract

~~People interact with systems more and more through voice assistants and chatbots. The days of solely engaging with a service through a keyboard are over. These new modes of user interaction are aided in part by~~ This research will investigate how ~~aa~~ advancements in Artificial Intelligence and Machine Learning ~~technology.~~

~~are being used to improve many services. In particular it will look at the development of chatbots as a channel for information distribution.~~

This project aimed to implement a web-based chatbot to assist with online banking, using tools that expose artificial intelligence methods such as natural language understanding. Allowing users to interact with the chatbot using natural language input and to train the chatbot using appropriate methods so ~~it~~ it will be able to generate a response. The chatbot will allow users to view all their personal banking information all from within the chatbot.

The produced prototype was found to be a very ~~The TrueLayer API will be used to retrieve account information which has just recently been made accessible to the public. This is part of the Open Bank Standard which allows developer's access banking data through the API Rest client. This came about through the bank as a platform (BaaP) standard.~~

useful tool to justify the need of a modern method of interaction to be integrated within many services offered by banks and financial businesses. In an industry with low user satisfaction rates and limited technology to increase accessibility. It is clear the chatbot overcomes the challenges banks face to increase the use of their services and gain a competitive edge over leading competitors.

With many people adopting Smart Assistant Devices such as Google Home or Amazon's Alexa. The chatbot was tested across a range of devices such as Google Home and Assistant on android devices to outline the key differences between the two modes of interaction , spoken and text dialog. These test were carried out to identify the value in integrating such technology surrounding the recent interest in

chatbots and conversational interfaces. Proving chatbots can be applied to a specific domain to enhance accessibility, reaffirming that they are more than just a passing fad and have a viable use.

1. Introduction

“Digitalisation, the surge of mobile and internet connected devices has revolutionised the way people interact with one another and communicate with businesses” (Eeuwen, ~~M.V.~~ (2017)). Millennials are accepting and supporting new technology into the routine of their everyday life, this is becoming more ~~is becoming more~~ prevalent as technology companies are streamlining Artificial

Intelli~~Intelli~~gence (AI) into the products they offer, such as; Google Assistant, Google

Home and Amazon Alexa. The new and upcoming generation are expected to be

critical and game changing customers for businesses. “They demand effortless experiences, answers within seconds, not minutes and more intelligent self-service options” (Teller Vision, ~~.,~~ (2017)).

The banking and the financial service industry was one of the first industries to adopt technology. This integration has grown massively, helping banks reach a wider customer base enabling them to perform their banking conveniently (Baptista and, G. ~~and~~ Oliveira, ,T. (2015)).

Banks are becoming ever more competitive with each other to adopt the newest advancements in technology to provide an improved delivery service to satisfy customers. Ulster Bank, Deloitte, AIB and PTSB are wanting to focus on integrating new technology to improve the speed at which transactions are acknowledged (Global Banking News, ~~.,~~ (2017)). With this in mind the relationship with the customer is always evolving due to the growth of technology.

Banks are now enabling the use of technology so customers can perform more tasks online, such as; cheque image clearing to allow the payment of cheques remotely and intelligent chatbots to increase customer service and assist employees. A chatbot is a “simple software program that can respond to customer prompts i.e. what’s my bank balance?” (Entrepreneur, ~~.,~~ (2016)). Mastercard has launched Kai an artificial intelligent chatbot and other bots for financial services. They can handle customer queries such as: ‘what is APR?’, requests, look at spending habits and solve problems. This in turn enables financial institutions to provide a new, engaging experience and strengthen their relationship with the customer, with the aid of natural language used by bots to establish a more personal and contextual conversation (Wire, ~~.,~~ (2016)). The focus of this project is to implement these new technologies to create an intelligent chatbot to enable banks to appeal to millennials and potentially gain a lifelong customer.

1.1 Aims & Objective

This work aims to provide a fast and convenient way to manage your banking. The online banking chatbot will help facilitate the user with queries and assist with personal banking.

The application will allow users to check

Check their balance.

View account details & s

View Transactions. There will be integration of SMS, currency conversions, two factor authentication and the ability to interact with the service through a chatbot.

- SMS confirmations and email mini statement schedule. The Starling API will be used to access the users banking information requested through the chatbot.
- Currency conversions
- Ask queries through chatbot and get appropriate and immediate responses
- Improve customer service through conversation using the chatbot

The chatbot will be implemented using the Laravel Framework and Dialogflow. The Dialogflow will be utilised as the NLU to perform artificial intelligence methods such as; Natural Language Processing (NLP), POS tagging, and entity recognition to analyse text and carry out the appropriate actions. Dialogflow is a Natural Language Understanding Engine (NLU) used for the extraction of entities and intent from a user's message.

Google Two-Factor Authentication will be implemented as an extra measurement of security for the customer. This ensures that no one else can access their account and view their personal banking information. A unique code will be generated and sent to their phone from scanning a QR code displayed after a successful login. Users will download the Google Authenticator app onto their mobile device to receive the unique code generated which will grant access to the chatbot.

1.2 Literature Review

The banking industry has multiple electronic delivery channels in use to distribute technology assets and services for the benefit of their customers. Online banking is a commodity of commerce within

financial services as well as banking industries ([Ajimon and G. G.S. Gireesh K. \(George and Kumar, 2013\)](#)). Advancements in technology has transformed many of our services into the digital era and the banking industry is one of the primary industries to avail of these advancements to improve their services. Currently within the UK two paradigms are available for online banking. One of which is an integrated internet bank which still operates through the branch but has an online presence. The other, a stand-alone internet bank, that operates completely independently and its only existence is solely through the internet (MarketLine, 2017).

Banks implement technology to strengthen their processing capacity, acquire a larger customer base and expand the services they could offer (Consoli [et al. \(2005\)](#)). The use of internet banking has grown in demand enormously in the last decade. “15% of branch customers use online banking once a day, 59% once a week, 77% at least once a month and 53% were confident in carrying out the best part of their banking online” ([Barty, J. and Recketts, T. BBA, \(2014\)](#)). Online banking has become more popular as it negates the need for customers to visit their local branch as they can manage their finances on the go to meet the demand of modern life. This is evident as branchless banks are now emerging from the industry such as Atom Bank and many banks now closing some of their branches. This is evident with the recent closure of 11 Ulster Bank Branches in NI due to the increased number of customers performing their banking online (The Belfast Telegraph).

~~However~~However, a recent study by Ling et al., (2016) [et al.](#) notes that most internet banking service providers struggle to get many of their customers to use their service. They identify lack of customer satisfaction when using online banking services to be a major cause. “Service quality, web design and content, security, privacy, speed and convenience” (Ling et al., 2016) are ~~stated~~ [identified](#) as the top factors influencing customer satisfaction This suggests that there is a lack of technology in place to enhance the customer online banking experience which could be improved by integrating a chatbot to provide an efficient, convenient and personal service.

1.3 Chatbots in Industry

Most businesses and organisations are understanding the potential benefits of machine learning and artificial intelligence to have a positive change on how they perform business. Artificial intelligence has progressed to allow the development of more sophisticated chatbots. Organisations are focusing on specific areas of user engagement that take up a lot of time but can be replaced through the use of a chatbot. Chatbots can understand what the customer needs from a single text instead of the customer having to follow a process of multiple steps.

Chatbots are used to automate customer service and reduce manual tedious tasks performed by employees so they can spend their time more productively on higher priority tasks. Establishments that regularly deal with its customers have discovered the potential of chatbots as a channel to distribute

more efficient and immediate information to customers in comparison to a customer service representative regarding queries and issues (Onufreiv, ~~YY.~~, 2017).

HDFC Bank has merged with Niki.ai an artificial intelligence company to develop a state of the art conversational banking chatbot (~~(HDFC Bank.,-20177)~~). The chatbot is accessible through the banks Facebook Messenger allowing users to utilise e-commerce and banking transactions all within the chatbot.

There is also a chatbot integrated within the HDFC login page to assist in online banking. The chatbot was developed as a concierge service, this is definitely one approach that can increase customer satisfaction within online banking enabling banks to develop a better relationship with its customers.

Chatbots will renew and modernise the customer service industry and the main sectors outlined including the banking and healthcare sector (Newswire, ~~.-~~2017). Industries such as; retail, healthcare, e-commerce and banking, are expected to achieve considerable savings from integrating a chatbot. Between 2017 and 2020 chatbots are estimated to make savings of \$8 billion per annum for businesses. The majority of savings will be made through customer service as customers can now ask queries about banking through the chatbot rather than having to call the bank, allowing banks to save on call centre staff. The study predicted that the integration of chatbots within the banking industry would rise from 12% to 90% by 2022.

Most online banking services would benefit from having a chatbot integrated into their services. The use of bots help internet banking service providers establish a better relationship with its customers. Customers can get answers to query's immediately, conduct e-commerce and banking operations all from within the one bot conversation. This is another benefit of using an online banking system with and integrated chatbot as "87% customers that bank online prefer to execute their personal finance operations within a single site" (Dauda, et al.,-2015).

~~W.B.~~King, W.B. (2017), records a survey conducted by Personetics which identified "40% of its clients are planning to integrate a bot into its services within the next 2 to 3 years and 70% said AI was an exciting opportunity". This clearly identifies the demand and need for a chatbot within the financial services Industry.

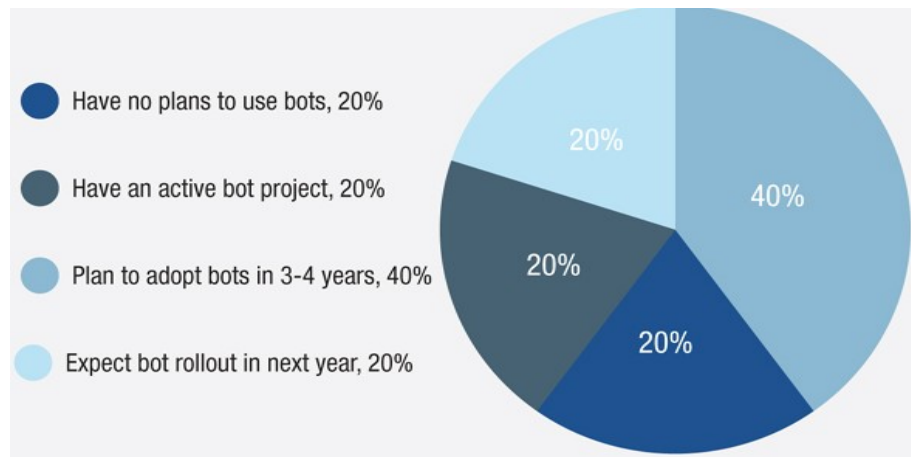


Figure 1: ~~Figure 1.3.1~~ Interest in chatbots within the Financial Industry (Source: ~~W.B. King, W.B~~ 2017)

~~Mark Zuckerberg CEO of Facebook, believes~~ Chatbots provide more personal and immediate ways for customers to communicate with an organisation allowing them to provide consistent and appropriate customer service at scale (Conversational Business, 2017). Facebook has now released its own Messenger Bots, which allows organisations to develop their own Chatbot through Messenger to deliver their services.

Facebook are also developing AI methods to obtain specific information related to users across relevant topics. The acquired information is used as a dataset to develop and improve their Artificial Intelligence so they can analyse, group and rate each user post (Rossmann et al., 2017).

“44% of US consumers prefer to interact with a chatbot compared to a human in relation to customer queries” (Wellers et al., 2017). This is evident as Chatbots are also becoming more prevalent in other industries such as; insurance, recruitment, media and pharmaceuticals.



Figure 1.3.2: Industries piloting chatbots (Source: Etlinger, E., 2017)

Figure 1.3.2_2 displays the tasks that can be replaced with the integration of chatbots into an organisation, with the goal of advancing access and use over time.

1.3.3 Chatbots

A chatbot is a software tool that utilises natural language processing (NLP) for human machine interaction (HMI) and Machine Learning (ML). “The complexity of a chatbot is directionally proportional to the scope of the domain”. An open domain requires a larger knowledge base, whereas, a closed domain has a more specific knowledge base that was developed to achieve a specific goal (Gregori, E., 2017).

Chatbot technology initially began in the 1960s to determine whether a chatbot could be portrayed as a human. Throughout the 1980s there was an elevated amount research carried out on natural language interfaces which lead to the development of sophisticated chatbot architectures such as A.L.I.C.E. This chatbot architecture is one of the earlier chatbots developed in 1995 by Dr Wallace which is now open-source, the acronym stands for Artificial Linguistic Internet Computer Entity. This is a chatbot you can create through interaction as it will learn from previous interactions to create its knowledge base. Its knowledge is saved in AIML (Artificial Intelligent Mark-up Language) files which evolved from the Extensible Mark-up Language (XML) (Shawar, ~~B.A~~ and ~~and~~ Atwell, ~~E~~, 2007).

Chatbots are developed using two approaches; a rule based approach where chatbots operate by moving through branches of a tree diagram of an expert system. The second approach involves advanced artificial intelligence and machine learning, thus the chatbot can learn from the conversations, generating appropriate responses to continuously improve over time (Watson, ~~A~~, 2017).

There are two modes in which chatbots can simulate a conversation with users which include :

~~S~~system-initiated chatbots where- they commence the conversation with the user and

User-initiated chatbots where- the user directs the conversation instead.

Systems that incorporate the two methods of initiation are known as mixed initiative systems (Duijst, ~~D~~, 2017).

1.4 Natural Language Understanding Engine

The chatbot engine is thought of as one of the most critical elements of a chatbot, alias “Natural Language Understanding (NLU) engine” (-Kar, ~~R~~ and Halder, ~~R~~, 2016). The NLU holds liability for the translation of conversational dialogs to actions which are understood by the machine. NLU engines use a variety of artificial intelligence methods to understand the natural language used in conversational interfaces such as chatbots. These methods consist of: Natural Language Processing (NLP) and Machine Learning (ML) (Kar, ~~R~~ and Halder, ~~R~~, 2016).

Googles Dialogflow, previously known API.ai, is a natural language understanding engine that identifies the intent and context from the natural language in user supplied utterances. These concepts are used to develop the behaviour of the chatbot and how coherently it interacts with the user. Intents are used to establish a connection between the user input and the appropriate action to be executed by the chatbot in order for the user to achieve their goal. Contexts are utilised to differentiate and understand user input which may have an alternative meaning depending upon the current conversational context (Gregori, ~~E~~, 2017).

1.5 Artificial Intelligence

“Artificial Intelligence is neither a new technology nor a machine”. Artificial intelligence is the recognition of outcome-direction which is the rapid analysis of live data to achieve the expected goal. Outcome-directed thinking splits from the confines of the rule-directed approach that is accomplished through artificial intelligence.

The generalised practice of AI can be broken down into a straightforward process which does not require an experienced level of proficiency to understand. First of all, a numerical representation is established for the target or outcome. Specific data is then associated with the target is gathered and conditions and behaviours are investigated to increase the likelihood of achieving the expected target. Multiple aspects can determine the outcome. The weight of each aspects effect is computed. “AI uses the relative weighting of each aspect to create a prediction (evaluation) formula” (Yano, ~~K~~, 2017). Lastly, the formula devised from the weighted aspects are employed to business decisions (Yano, ~~K~~, 2017). AI can be classified into four groups: “systems that think like humans, systems that act like humans, systems that think rationally and systems that act rationally” ((Russell, ~~S.J~~ and Norvig, ~~P~~, 1995).

AI is generally categorised as strong and weak AI: strong AI is the production of human-like intelligent systems. Weak AI would be the integration of intelligent algorithms embedded within a system. “Machine learning, deep-learning, natural language processing and neural networks are often summarised under the term of AI” (Exner-Stöhr, et al., 2017)

1.5.1 Artificial Intelligence Methods

“Natural Language Processing (NLP) is a theory motivated range of computational techniques, for the automatic analysis and representation of human language” (Jurafsky, ~~D~~ and Martin, ~~J.H~~, 2017). Natural Language Processing technology has made great advancements in machine learning based systems to be able to extract meaning from natural language utterances also known as sentiment analysis (Cambria and White, 2014).

~~There are many~~ techniques used in NLP for the analysis of natural language include Entity recognition: which Entity Recognition is a technique which recognises entities in text. The most common entities include; nouns, organisations, people and places.

Named Entity Recognition (NER) is the task of finding every instance of a named entity in text and label its type in order to classify it correctly (Jurafsky, ~~D~~ and Martin, ~~J.H~~, 2017).

“Entities are domain specific information extracted from the utterance that maps the natural language phrases to their canonical phrases in order to understand the intent. They help in identifying the parameters which are required to take specific action” (~~Kar, R and Haldar, R. Kar and Haldar,~~ 2016).

Establishing the context of the of the users message is a vital feature that allows the chatbot to deal with situations that it may not be able to carry out a specific action for. This is due to the user input being very vague or may have an alternative meaning. The context is the capacity of a chatbot to sustain its state, also referred to as the number of user supplied input (utterances) when the context is extracted and the appropriate intent is paired to conduct the desired action for the user.

Intents are the core backbone of conversational interfaces. The intents represent what the customer is wanting to achieve such as ‘show me my balance’. The text sent by the user in natural language is analysed to identify the corresponding intent of the text. This requires matching a specific user supplied phrase with an appropriate action to be executed by the system. The chat bot would then return appropriate dialog in order for the user to achieve their goals.

Actions are the processes or steps executed by the system when the intent of a message is identified, they contain parameters which categorise and define its- properties (~~Kar, R and Haldar, R. Kar and Haldar,~~ 2016).

Sentiment analysis incorporates multiple natural language processing techniques in order to extract meaning and polarity from text. Polarity detection classifies text to be either positive or negative and measures the intensity of the overall polarity detected. Sentiment analysis achieves a more in depth understanding of the contextual role of each concept within a given piece of text (Cambria,~~E.~~ -and White,~~B.~~ 2017).

Part of Speech (POS) Tagging: this involves assigning a label to each word in the user input with its part of speech (e.g. noun, verb, adjective, etc.). The labels or tags can be rule-based (a manually developed set of rules is defined to establish part of speech for ambiguous words provided in the conversational context). The labels can also be developed utilising advanced models that are trained using input labelled with the appropriate POS. This is additionally used in response generation in order to outline the POS object type of the expected response made by the chat bot (Cahn, 2017).

1.6 Chatbot Architecture

Previously chatbots solely supported a single adjacency pair, also known as a one-shot conversation. However, modern chatbots can sustain multiple adjacency pairs, remembering states and contexts between conversations and have the capability to associate data in different adjacency pairs which is related. This is the bots ability to preserve the conversation. A chatbot consists of four main parts: front-end, knowledge-base, back-end and corpus which is the training data. The front end is accountable for enabling communication between the bot and the user. The NLU utilises Artificial intelligence methods to identify the intent and context of the user input. An appropriate response is generated from the users' intent. The knowledge base defines the chatbots knowledge, which is created within the NLU and supported by the back-end, the back-end applies the domains corpus to produce the knowledge base (Gregori, [E.](#) 2017).

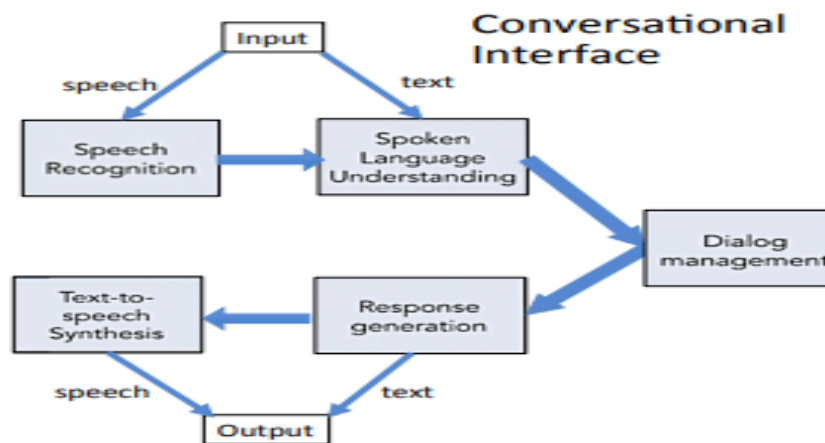


Figure 3.1.6.1: illustrates a standard chatbot architecture (source: Michael-McTear, (2018)).

Input can be supplied to the chatbot in the form of text or speech. The Input is sent to the dialog management system which is the NLU in this case, which determines an appropriate response and amends the chatbots state accordingly to carry out the required action. The chatbot will produce text and speech responses in the form of both text and speech.

1.7 Summary of Report

The remainder of this report will focus on:

- Analysis - This section will include various software methodologies; quantifiable feedback will be collected from questionnaires which will be analysed to identify the requirements of the chatbot. The internal and external hardware and system requirements are outlined. A Gantt chart will be provided outline the project schedule and when each milestone should be met.

- **Design** - This section will cover the overall design of the chatbot; UI diagrams/story boards will be included which detail how the GUI will appear to the user. The technical architecture will be designed and displayed in graphical form such as an activity diagram. It will outline how each component interacts with each other. The design for the database and the relationships between tables will be provided.
- **Implementation and testing** - The section will document the implementation process to develop the solution, including; the code used to develop compelling features. Define what was learnt during each iteration and document each prototype. Once implementation is complete the chatbot will be tested using unit testing and appropriate test cases. Any errors or faults identified will be fixed to ensure maximum quality.
- **Evaluation and Reflection** - An evaluation will be carried on the completed project to determine its quality and value. It will detail how the overall project was developed. Reflecting upon the overall experience, highlighting any areas that could be improved for future work and what well throughout the project.

1.8 Conclusion

There is clear evidence based on the research conducted that there will be a drastic increase in the number of chatbots being implemented within the financial service industry. The vast amount of research that has been carried out, and currently ongoing, within the artificial intelligence field has led to the rise of more sophisticated and intellectual chatbots. This will prove to be immensely beneficial in providing convenient and accessible customer service at a rapid scale.

2. Requirements Specification and Analysis

This section provides a clear analysis for the potential solution, to outline and breakdown the necessary requirements to implement the chatbot. This includes:

- the problem the project is overcoming
- the design methodologies considered for implementation
- functional and non-functional requirements
- software and hardware utilised to implement the project

2.1 Problem Statement

It is evident from the research carried out in the literature review that modern financial

services are constantly seeking to expand their technologies, both to improve customer service and increase delivery of services through the advancements in technology. This is to gain a competitive edge over other banks for financial benefits and to expand its customer base. A domain specific chatbot will be implemented to assist users with their banking. In order to overcome the user satisfaction issues associated with online banking services. The chatbot will provide personal and efficient communication between the user and their bank in order to manage their finances and get assistance when needed, such as; answering any queries and booking appointments. The chatbot will allow users to feel confident and comfortable when using this service regardless of the user's computer literacy due to the natural language used in messages. It also provides a very accessible and efficient service as all interactions will take place within the one chat conversation negating the need for the user to navigate through a site.

2.2 Proposed solution

The proposed solution is to create a chatbot to simulate a human conversation to assist users with their banking needs and to provide a more personal experience. Advancements in artificial Intelligence, machine learning techniques, improved aptitude for decision making, larger availability of domains and corpus, have increased the practicality of integrating a chat bot into applications (Dole et al., 2015).

Users will be able to ask any banking related queries in natural language that they are comfortable using such as; view account information, transactions and check balance. The chatbot will identify and understand what the user is asking and generate an appropriate response based on the conversational context. Immediate responses will be provided by the chatbot to redeem the need for the user to have to call or visit- their local banks branch and wait in queue in order to get through to an advisor for assistance.

In order to make the application more secure Googles 2 Factor Authentication will be integrated to increase security ensuring only registered users can gain access to their account preventing the risk of fraud.

2.3 Software Development Methodologies

Deciding upon an appropriate methodology is vital for the overall development of any software application to ensure a realistic timeframe is established for each stage of the project and requirements are clearly outlined. Various development methodologies will be discussed and considered for the development and design of this software. This section will highlight the development methodology that is best suited to this project.

2.3.1 Waterfall Methodology

This is a very traditional methodology, which is usually introduced when you initially learn about software development. The waterfall model is a very predictive approach to software development that consists of 5 stages to include; requirements gathering, analysis, design, implementation and testing. Each stage is completed subsequently of one another. A major drawback of the waterfall model is that it is very inflexible, as the project is broken up into phases. Each phase is given a deadline in order for a deliverable to be produced at the end of each phase to adhere to the overall project schedule. The success and progression of the project is measured from the project deliverables, design documents and test plans. As each phase of the project is outlined at the beginning of the project lifecycle and targets have been set it's difficult to integrate new requirements or a change in requirements that may be identified at a later stage as it would adversely affect the overall project schedule. The waterfall model moves a lot of the more high risk and difficult components towards the end of the project life cycle (Nat Laundry, ~~N.~~ 2011).

2.3.2 Incremental Model

This software methodology evolved from the waterfall model. The application is designed, developed and tested using iterative incremental build stages. At the end of each build a subsystem or feature will be created. The project will progress in complexity as new requirements are likely to be discovered and implemented in each

incremental build, developing on top of the functionality from the last build leading to the overall development of the application.

It is very common for software to be released in stages, it is critical that component versions utilised within the software are managed throughout the entire lifecycle using version control tools such as GitHub. Each build will only last a few weeks to produce a baseline version of the application. Feedback can be given on any requirement errors or faults found in the application. Distributing the development of the project over various build cycles can lower the risks associated with development to a more manageable level as requirements are broken down into smaller functionality to be implemented at the end of each build.

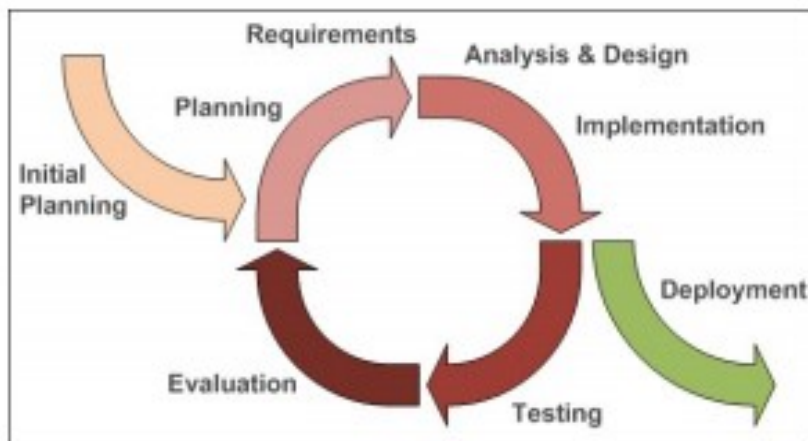


Figure 4.2.3.2.1 - Displaying the Incremental Model (Source: Moniruzzaman and Hossain, 2013)

Testing and debugging are made easier as faults are identified early on during small manageable cycles. This methodology is flexible during implementation as new requirements identified are easily integrated at each build and an updated version is released (Khan et al., 2011).

2.3.3 Chosen Methodology

The incremental model is the most suitable development methodology to implement for this project. The flexibility of the incremental model makes it ideal for this project as it is likely new requirements will be identified during the later stages of development and each iterative build makes it easy to implement new requirements throughout the development process.

2.4 Information Gathering Techniques

To determine the requirements of an application, information gathering must be carried out in order to know what the end user wants the system to be able to do and what the user hopes to achieve through using the application. There are many information gathering techniques, these include:

Questionnaires/Surveys: this method of information gathering is efficient in collecting

data of a large group quickly. Surveys consist of a range of questions such as closed ended questions i.e. where the user is prompted to select a choices/boxes. Open-ended questions allow user to express their opinion or provide whatever feedback they feel is suitable. Other types of questions can be used such as ranges and scales which can be used to gauge how strongly a user may feel. Surveys can be distributed through email, online, paper-based and telephone surveys. The questions used must be clear and precise. Surveys should be tested on a small focus group of user to get feedback on the survey this ensures the questions are unambiguous and unbiased (Jonathan Lazar, J., 2001).

Interviews :- interviews are a structured approach to information gathering. They are conducted face to face to allow the interviewer to ask more detailed open-ended questions and follow up questions to elaborate on certain points to capture the user needs. This method allows the interviewer to take control of the interview and direct it in a way so they can focus on more important aspects of the proposed system. This ensures the most important aspects of the system development are identified within a limited time constraint. One interview tactic interviewers use to determine they are gathering as much information as possible is to ensure the interviewee feels comfortable and relaxed in the interview environment. Open ended questions can be asked resulting in more detailed responses. Follow up questions can be used to ensure a previous response is fully understood. The interviewer can take notes during this process to reflect on and analyse the content later (Graham Oakes, G., 2008).

Discovery Prototyping: this involves developing a low fidelity throwaway prototype of the proposed system. This approach can be useful in situations where the requirements are vague. This allows the end user to experience and get an impression of the system and determine whether it meets their needs. Developers will meet with the user to identify the system requirements and investigate what requirements that can be established. Once the requirements are identified the prototype is developed and given to user to assess whether or not it meets the requirements previously discussed. This gives the user an opportunity to be more involved in the project and provide feedback on the prototype that will be used to identify the requirements for the

final software to be developed. The prototyping approach allows the developer to get a deeper understanding of the requirements through continuous communication and each prototype iteration or cycle ([Tata McGraw-Hill Publishing Instructional Software Research and Development](#), 2007).

2.4.1 Chosen Information Gathering Technique

The most efficient way to gather information for this project would be the use of a Questionnaire, distributed online to the general public. This will reach a large user group, producing a diverse and varied result set for analysis providing detailed scope on what users from different age groups and backgrounds are expecting from an application such as this. It will give valuable insight into the age group, how often the user utilises online banking, if they would prefer assistance through an advisor on phone or using a friendly chatbot service, level of computer literacy and their expectations from the application.

These questionnaires will be completed using Survey Monkey, which provides free cloud-based expert custom templates for surveys and robust analytical features with a UI to display data analysis in a number of formats that make it straight forward to read and analyse the data to identify requirements.

The structured questionnaire will be made up of open and close ended questions as a convenient means of extracting information from users. There will be clear descriptions throughout the online questionnaire making it easy to navigate through and use, increasing the likelihood of participation from users. Clear and concise information will also be given on how the user should respond to each question accordingly- ([M. McGuirk, P.M](#) and [P. O'Neill, P.](#) 2016).

The questionnaire will be well presented with a consistent colour theme throughout provided by Survey Monkey. A range of different questioning techniques will be used such as point scales, ranges, open and closed question. The questionnaire elicit the requirements for the chatbot can be found in the appendix.

This questionnaire will provide a great understanding on how the public would respond to using a chatbot to assist them with their banking.

-As the purpose of this project is to provide a chatbot to assist with online banking, the question relating to users who have had past experiences in interacting with a chatbot, [-see question 5 figure 7.1.1](#) of appendix [A](#) indicates valuable insight into whether or not it was an efficient and enjoyable experience for the users.

It's clear from analysing the responses of this question that a small number of the respondents have interacted with a chatbot before as only 11.54 % had previously interacted with a chatbot. That the majority of users that

have interacted with a chatbot, even if the interaction did not go seamlessly had a very positive opinion of chatbots in general and actually felt as though they were communicating with a human as the conversation felt very natural. This is due to the chatbot having the ability to understand the natural language used in messages and generate an appropriate response through various artificial intelligence methods such as natural language processing. Users found it to be very informative and efficient when replying to their queries-. Although some respondents found that it could take a long time to reply, or provide very simplistic responses to more complex tasks and become unresponsive. This reaffirms the need for the chatbot to have a method that will give the user guidance if it does not understand the context of the conversation or if it cannot assist the user any further. This may be in the form of a link in the chat conversation to get assistance from a human or a list of instructions the chatbot can understand. —

The results from the question- 4 of appendix A 'Do you feel online banking is secure?' are displayed in figure 7.1.2 of appendix outlining there is only a 31.028.34% difference in the amount of users that feel online banking is secure. Out of 264 responses 6558.383% feel online banking is secure however a lot of users also worry about their personal information being compromised, stolen or account being hacked and a criminal gaining access to their financial information and committing fraud. Due to the security concerns most people have in regards to about online banking the chatbot will have 2-Factor Authentication implemented. This feature will reassure users that their account will be protected, even if a criminal gains access to a user personal information they would need the users mobile phone in order to successfully access their account as a unique code is sent to the users phone each time upon requesting to login, the code is validated to allow users access to their account.

2.5 Functional and Non-Functional Requirements

Functional and Non-functional requirements are identified through the analysis of the data collected from the survey. Functional requirements are the features and functionality that the system must have or be able to perform whereas non-functional requirements define the manner or characteristic the system must have such as: performance, usability, modifiability, maintainability, security, scalability, reliability,

availability, configurability and design constraints (Frank Tsui et al., 2016)

2.5.1 Functional Requirements

- Allow unregistered users to register on the application and save their details to the database.
- Provide confirmations notifications through emailSMS.
- Registered users will be able to login, once login details are submitted to database the user will be presented with a QR code implemented through Google's Two-Factor Authentication and a unique code will be generated and sent to the user's mobile device.
- The chat-bot must allow users to view information about accounts held by them i.e. savings, loans, current account.
- The Chat-bot will allow users to view their transactions through a transactions statement sent to the users email.
- The Chat-bot will integrate with the TrueLayer Sterling API which will return data about the users' bank account.
- The Chat-bot will assist users with their queries and carry out appropriate actions such as scheduling appointments, with finance consultants.
- Users will be able to converse with the Chat-bot through voice or text commands and it will understand what the user is saying through natural language understandingprocessing provided through the integration of Dialogflow API.
- The chat-bot should be able to maintain the conversational state when the context may be unclear through previous messages and conversations.
- Provide text and audio responses.

2.5.2 Non-Functional Requirements

- The chatbot must be efficient with very little lag in response time for instance no longer than 5 seconds to reply to a user message.
- The chatbot must be reliable with next to no faults or bugs
- The database must be scalable to adopt to a growing number of users
- The chatbot must be secure as sensitive data is being used, Googles 2 Factor Authentication will be implemented as an extra security feature
- Comply with data protection laws such as the Data Protection Act 1198
- The use of natural language used to interact with the chatbot promotes human computer interaction.
- Provide accurate responsesaccurate responses to input

- Appropriate handling of unexpected input ~~&, and~~ correctly inform the user if it cannot provide assistance

2.6 Software and Hardware Specification

The software and hardware requirements necessary to implement the chatbot are stated below.

2.6.1 Software Requirements

- Oracle VM VirtualBox
- Vagrant
- Laravel Homestead
- Nginx Server
- MySQL DB
- PHP 7.1
- JavaScript
- Sublime Text 2
- Laravel 5.6
- Dialogflow
- Ngrok

2.6.2 Hardware Requirements

- PC running Linux-Ubuntu to act as server to host chatbot locally
- Processor – intel core i5
- Android Device

2.7 Project Planning

A set of project milestones and deliverables are produced illustrating the project breakdown to create a management plan to adhere to throughout the project lifecycle, however there may be slight changes in each iteration as some requirements may require modification. The steps required in each iteration are outlined to meet each project deliverable. Produce risk management plan and a project plan displayed through the use of a Gantt chart showing the projects tasks for each semester.

2.7.1 Project Milestones and Deliverables

The project milestones will be used to evaluate the projects progression, setting dates for tasks to be completed and identifying which objectives have been satisfied.

Table 1: Project Milestones.

Milestone	Milestone Description	Completion Date
1	Literature Review	06/11/2017
2	Analysis & Requirements Specification	21/11/2017
3	Complete Interim Report	24/11/2017
4	Design	20/01/2018
5	Iteration 1	26/01/2018
6	Iteration 2	15/02/2018
7	Iteration 3	9/03/2018
8	Iteration 4 –final build	26/03/2018
9	Testing	03/04/2018
10	Evaluation	05/04/2018

Table 1: Project Milestones

The project deliverables are outlined to construe the end result of this project. The deliverables are shown in table 2 below:

Table 2: Project Deliverables

Deliverable	Deliverable Description
1	Interim Report
2	Web based Chatbot
3	Final Report
4	Viva Demonstration

Table 2: Project Deliverables

2.7.2 Development Iteration Breakdown

When using the incremental model its vital to establish what functionality is required in each iteration to lead to the successful development of the project. If any new requirements are discovered during the development process they will either be added to an existing iteration if small or a new iteration will be added to the project plan.

Table 3: Project Iterations

<u>Iteration</u>	<u>Functionality to be implemented</u>
<u>Iteration 1</u>	<ul style="list-style-type: none"> • <u>Install and configure VirtualBox VM and Vagrant</u> • <u>Install Laravel Homestead</u> • <u>Connect to database</u> • <u>Add Login functionality using Auth library</u> • <u>Implement Google Two-Factor Authentication</u>
<u>Iteration 2</u>	<ul style="list-style-type: none"> • <u>Integrate Starling API</u> • <u>Have chatbot make calls to Starling API to return customer data when paired with the appropriate user intent and action</u> • <u>Integrate Fixer.io API to return conversions when paired with appropriate user intent and action</u> • <u>Implement Email transactions</u>
<u>Iteration 3</u>	<ul style="list-style-type: none"> • <u>Implement the Dialogflow API</u> • <u>Train Chatbot using intents – expected questions made by users</u> • <u>Train chatbot to hold a conversation with user</u> • <u>Create webhook to receive incoming messages from the user and extract entities</u> • <u>Implement Twilio API for appointment confirmation</u> • <u>Implement speech recognition and synthesis</u>
<u>Iteration 4</u>	<ul style="list-style-type: none"> • <u>Implement Chabot UI</u>
<u>Final Build</u>	<ul style="list-style-type: none"> • <u>Integrate app to google assistant and Google Home</u>

<u>Iteration</u>	<u>Functionality to be implemented</u>
<u>Iteration 1</u>	<ul style="list-style-type: none"> • <u>Install and configure VirtualBox VM and Vagrant</u>

	<ul style="list-style-type: none"> ● Install Laravel Homestead ● Connect to database ● Add Login functionality using Auth library ● Implement Google Two Factor Authentication
Iteration 2	<ul style="list-style-type: none"> ● Integrate Starling API ● Have chatbot make calls to Starling API to return customer data when paired with the appropriate user intent and action ● Integrate Fixer.io API to return conversions when paired with appropriate user intent and action ● Implement Email transactions
Iteration 3	<ul style="list-style-type: none"> ● Implement the Dialogflow API ● Train Chatbot using intents—expected questions made by users ● Train chatbot to hold a conversation with user ● Create webhook to receive incoming messages from the user and extract entities ● Implement Twilio API for appointment confirmation ● Implement speech recognition and synthesis
Iteration 4 Final Build	<ul style="list-style-type: none"> ● Implement Chabot UI ● Integrate app to google assistant and Google Home

When each iteration is completed and the finished solution developed. The project will be evaluated to highlight what was learnt and reflect upon the entire project life cycle. This gives me the opportunity to discuss what was achieved and what steps or features I would do differently in future projects from what was learnt during this experience.

2.7.3 Risk Management

Identifying and managing the risks and implications associated with any project is a vital step in reducing the risks impact on the overall progression of the project.

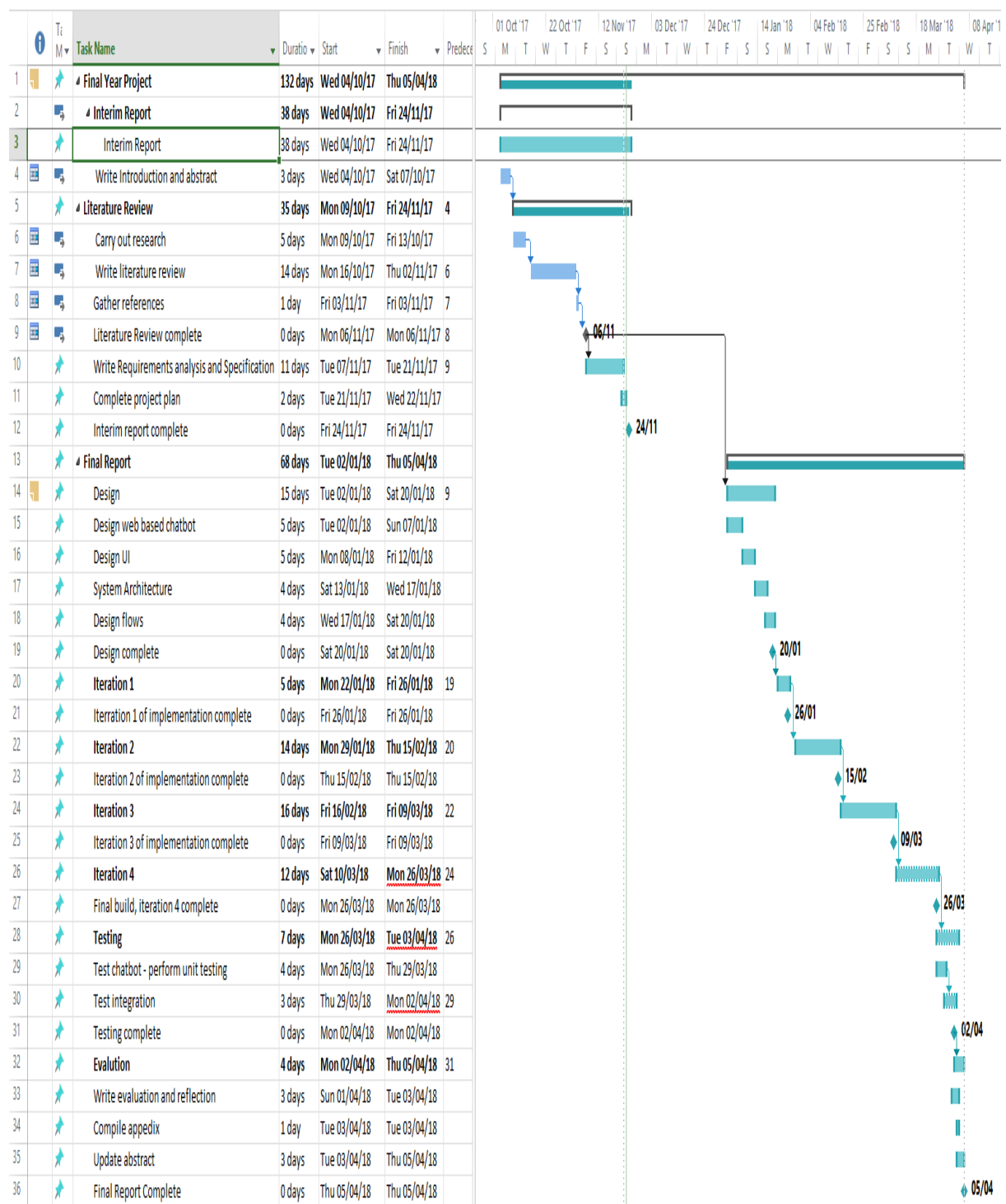
Identifying the risks implications early on in the project allows preventative actions to be put in place the minimise the likelihood of the risks occurring. The following actions

will be taken to reduce risks in this project; Github to backup code and prevent loss, use own initiative to keep on top of project plan, regularly make backups of project report and important documents, if an unplanned day off occurs make up for work lost.

2.7.4 Project Management

A Gantt chart created in MS Project displaying the project timeline ~~as is~~ shown below in figure ~~5.4.4.1~~. Gantt charts are used to identify when the project milestones should be completed so the project is delivered on time. A breakdown of the tasks, the order in which they will be completed and their time allocation is shown below.

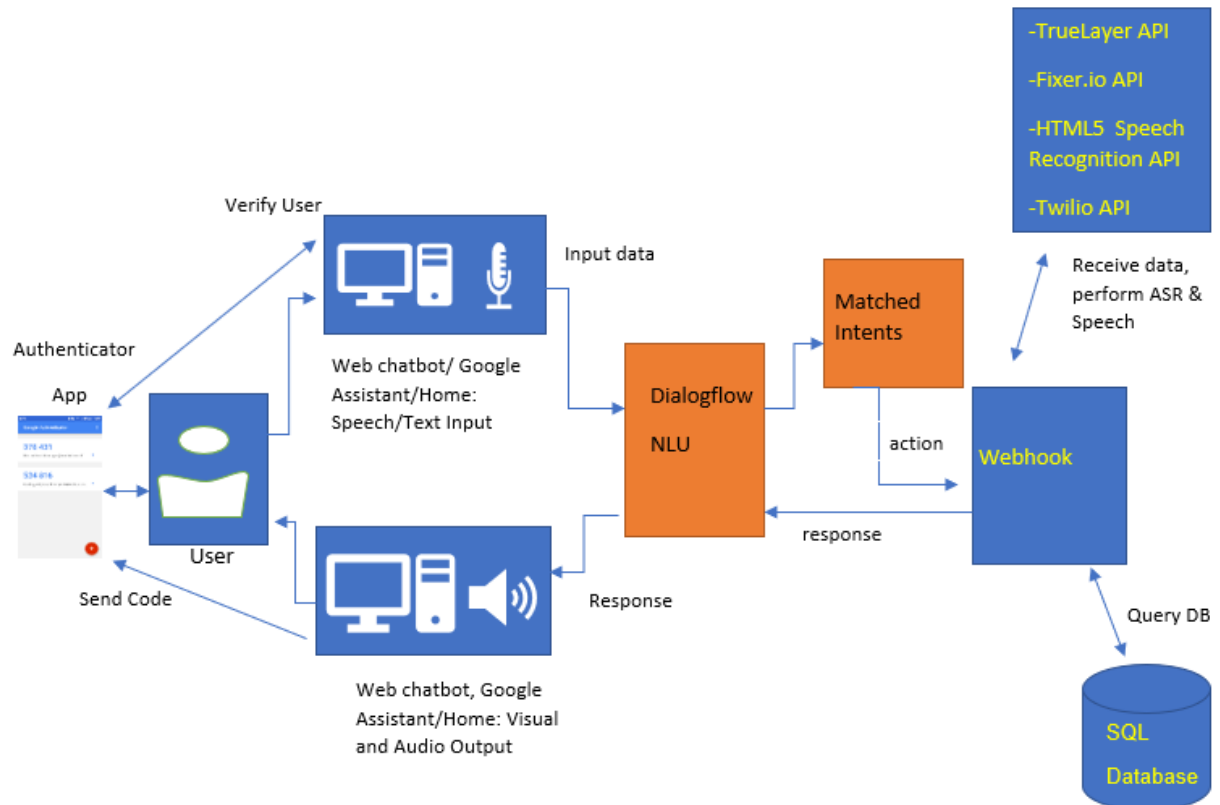
Figure 5: 2.7.4.1—Gantt chart showing project plan

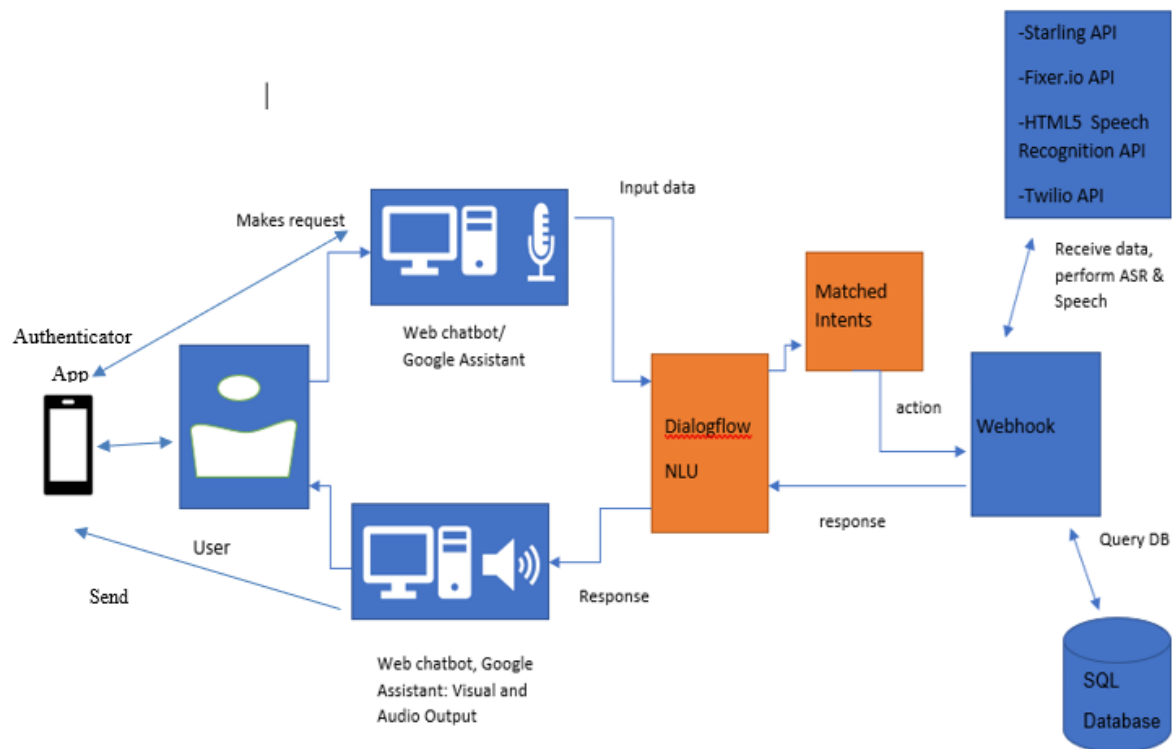


3. Design

3.1 Architectural Design

Figure 63.1 provides a clear and concise abstract architectural design of the proposed chatbot.





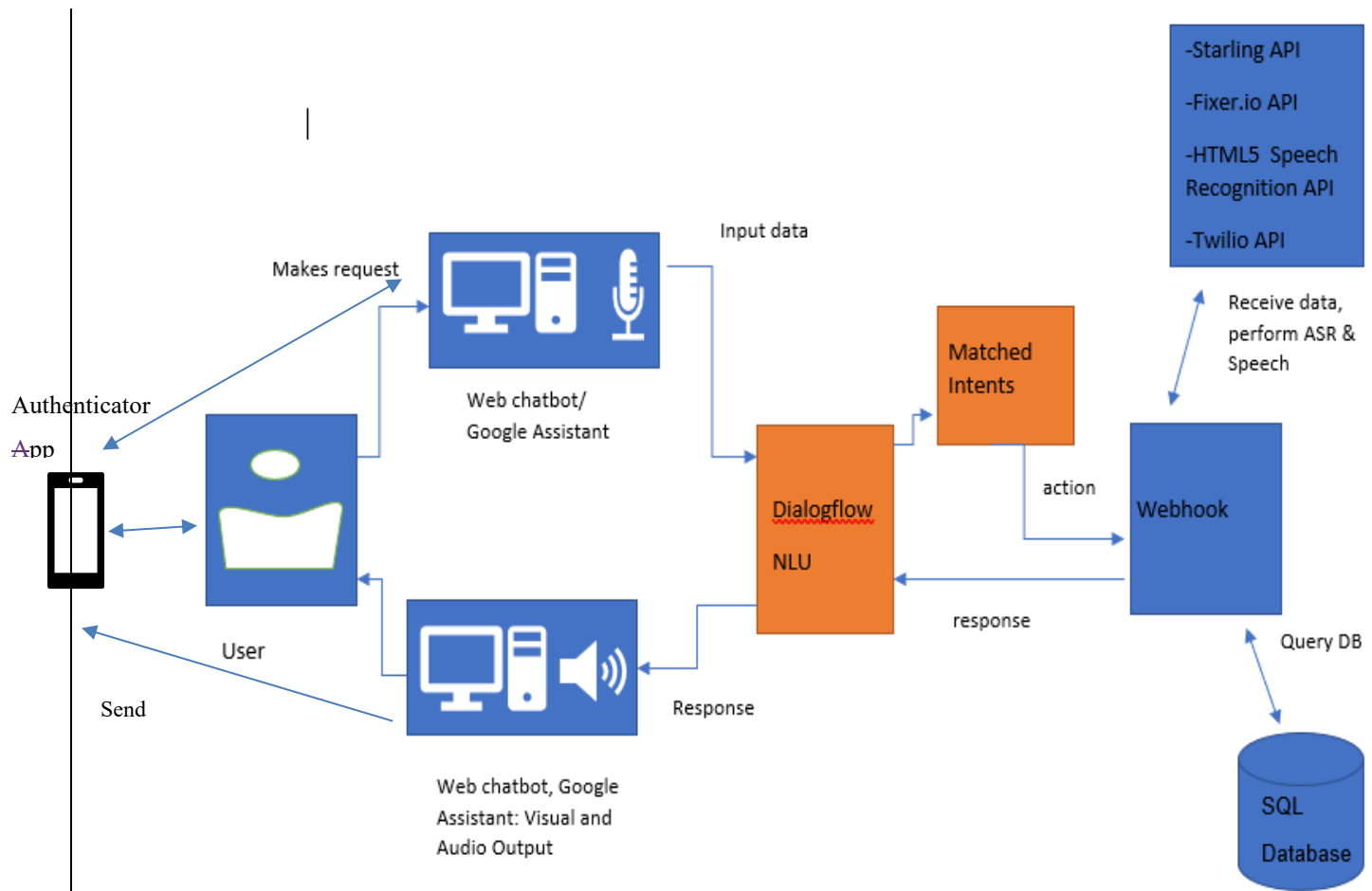


Figure 6.3.1.1 Chatbot Architectural Design.

Users will interact with the chatbot from the web client and Google assistant app for Android devices. Users will carry out their interaction with the chatbot in the form natural language voice or text-based phrases. With the Google assistant integration, users will receive rich responses, such as; images, and- cards thus improving ease of use and interaction experience for users as less typing and effort will be required when interacting with the chatbot mainly through the use of voice commands. The google assistant integration means the chatbot is easily be extended by Google Home.

The client side of the application is developed using [HTML5](#), CSS, JavaScript and Laravel Blade.

This is a templating engine built into the Laravel framework which follows the MVC architectural pattern. Laravel Blade compiles and caches view files to improve the performance of the application. Blade template supports separation of view files to be split up

_into sections using the '@section' annotation, which enhances the readability of frontend code, as HTML, PHP and JavaScript can be clearly separated into particular sections of the view and the '@yield' annotation is utilised to render the content for sections of the view.

Blade utilises template inheritance to allow views to derive the base layout of another view to modify and reuse UI components dynamically without replacing the code in the view it is inheriting from. Blade also has its own annotations for control structures which may be required in certain view files, for instance; the '@foreach' annotation (Laravel, 2018). The client side also avails of the chart.js CDN (Content Delivery Service) to display the user's spending details in a graphical format so they can easily see where they spend the most of their money.

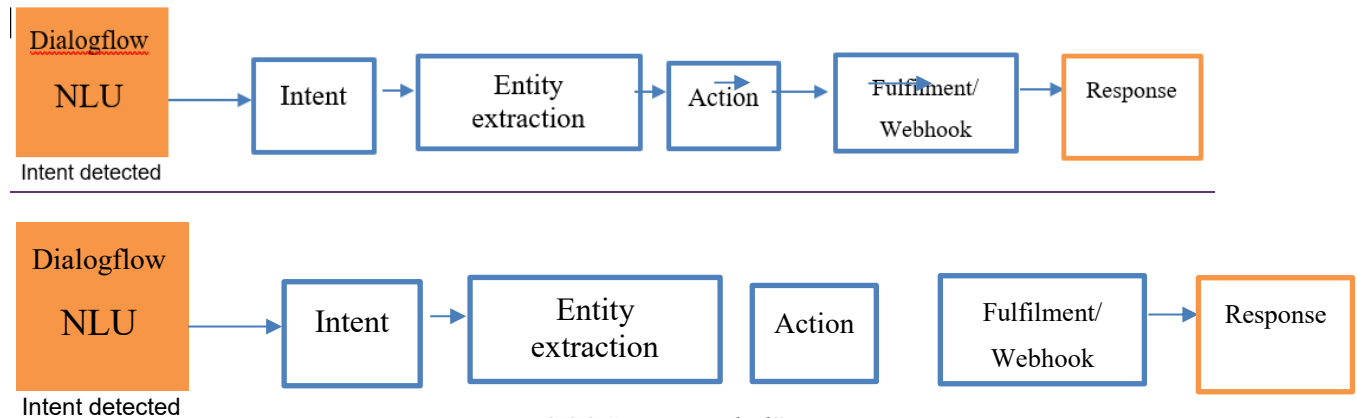
The Laravel framework is also used to implement the Web client and Webhook which receives the JSON data from the NLU via HTTPS POST requests. The Webhook is implemented as a controller class so the corresponding web route ~~can be is~~ defined as an endpoint, where the NLU posts real-time payload to the Webhook. The Webhook receives data from the NLU when it detects an intent has been triggered by the user from the client-side, Google Assistant or Home devices.

A webhook receives data posted to it from another service when an update or event occurs whereas an API follows a RESTful design approach that makes requests to retrieve the data (elastic.io, 2018).

Dialogflow is utilised as the natural language understanding engine (NLU), which allows the chatbot to be trained to recognise entities and intents in a user utterance. The Chatbot will be designed using intent mapping on the Dialogflow console to route user utterances to a collection of phrases.

~~The F~~figure 73.1.2 represents the flow of data as an intent is invoked by a user: Dialogflow recognises if an intent was triggered through its training using a collection of typical user phrases. This data is then posted to the fulfilment webhook. The required entities/parameters are extracted along with the action and a custom response is generated and returned to the

user either visually or spoken from the webhook. The Webhook parses and validates the JSON from the payload it receives and encodes the JSON response generated to be sent back to the user for both text and voice interaction.



The Webhook parses and validates the JSON from the payload it receives and encodes the JSON response generated to be sent back to the user for both text and voice interaction.

3.2 Database Design

Laravel also comes with Eloquent ORM, which “provides an ActiveRecord implementation for working with database objects” (Laravel, 2018). The data is stored locally using a MySQL Workbench database, it is small-scale to conserve memory. Every table within the database is represented using a ‘model’ which allows logic and relationships between database objects to be defined and manipulated.

The database model for the proposed chatbot is outlined in Figure 8.2.1 with defined multiplicity. The users table contains data about the users of the application and whether or not they are verified through google 2 Factor Authentication. The appointments table holds data about the user’s appointments such as ‘topic’. This holds information specified by the user about what the appointment is for. The ‘booked’ attribute is set to a Boolean value to determine whether or not the user already has an appointment. The accounts and transactions

table holds data regarding the ~~users~~user's bank accounts, for instance they may have multiple account types with the one bank such as current and savings, however the users online banking credentials they possess for each bank are not saved.

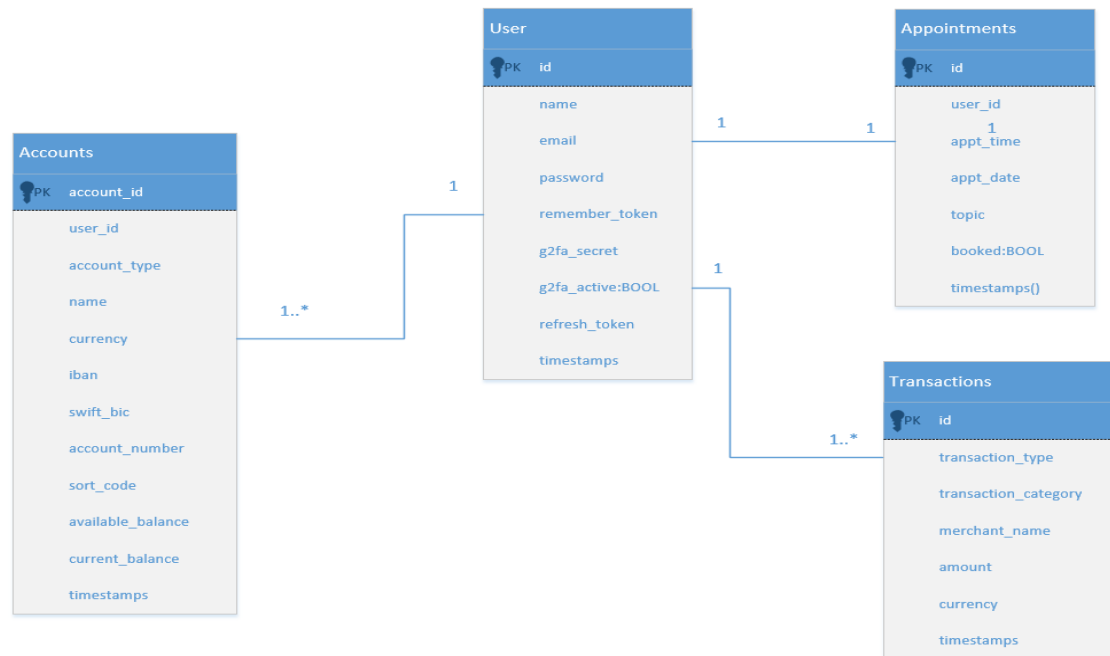


Figure 8: ~~3.2.1~~UML Database Model.

3.3 Class Diagram

The class diagram shown in Figure ~~9 3.3.1~~ illustrates the classes required to implement the chatbot. This will act as a guide for implementing the chatbot however other classes may be identified or modified during development to meet the requirements.

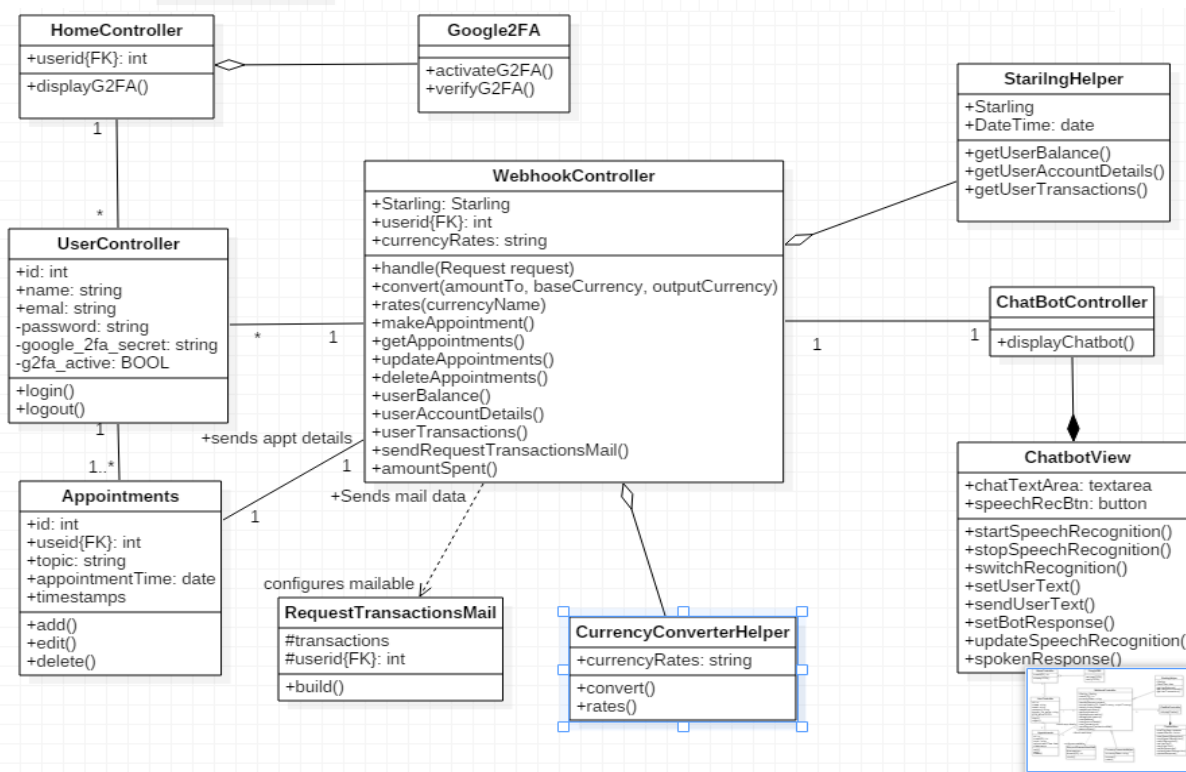
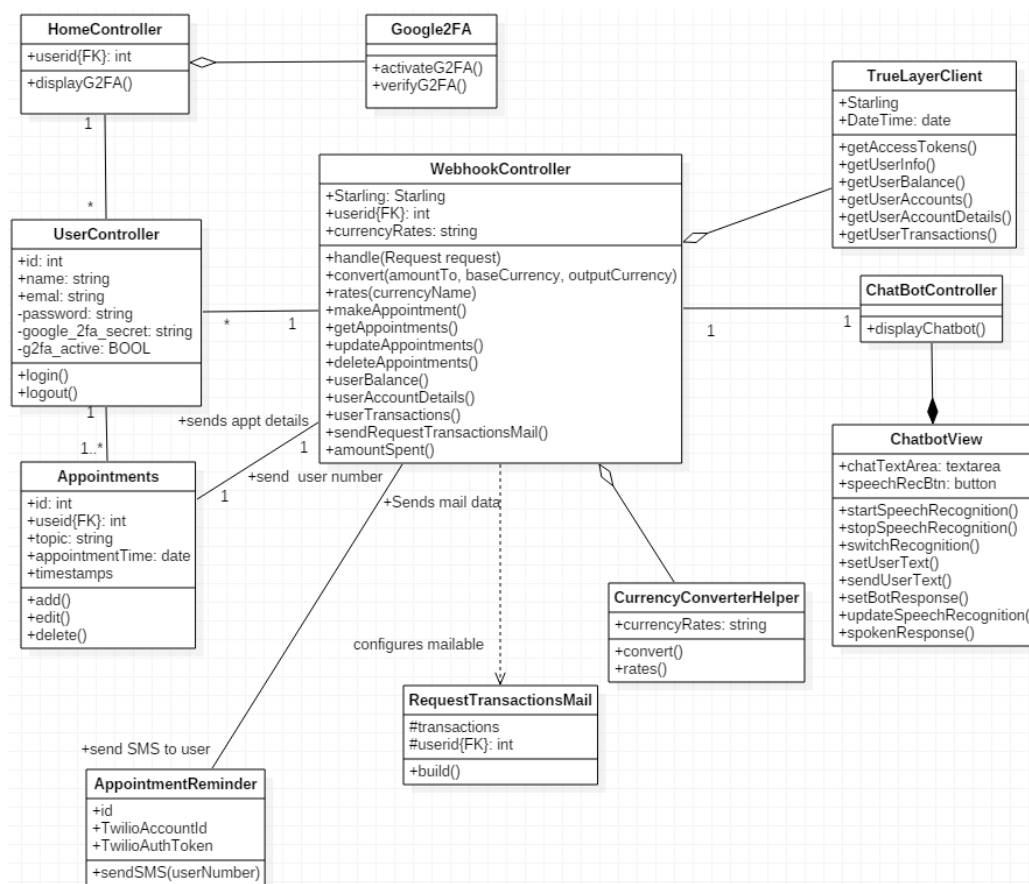


Figure ~~9.3.3.1~~ UML Class Diagram.

3.4 Speech Recognition

Users are able to interact with the chatbot using voice commands~~speaking to the chatbot~~ through the ~~built in~~built-in microphone on the chatbot device.

The HTML5 speech Recognition API is implemented within the web app for speech recognition and synthesis.

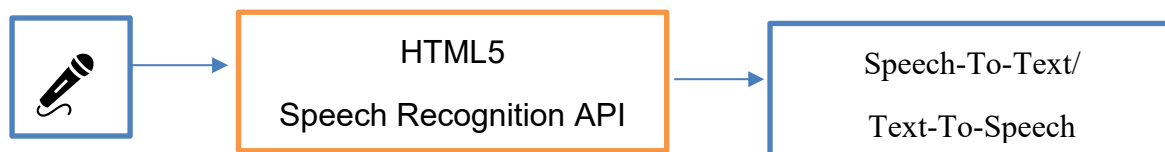


Figure 3.4.1 “Speech Recognition”

Dialogflow will then take the deciphered speech and convert it into a structured JSON object it can analyse, this is used as the text input. The JSON response is given an entity matching score also known as confidence score. This score expresses how well the NLU engine matched the user input to an intent defined on the console (DialogFlow, 2015). Scores range between 0-1, 1 being an exact match. Below is a snippet of the JSON object response. User says: “Who are you?”.

```

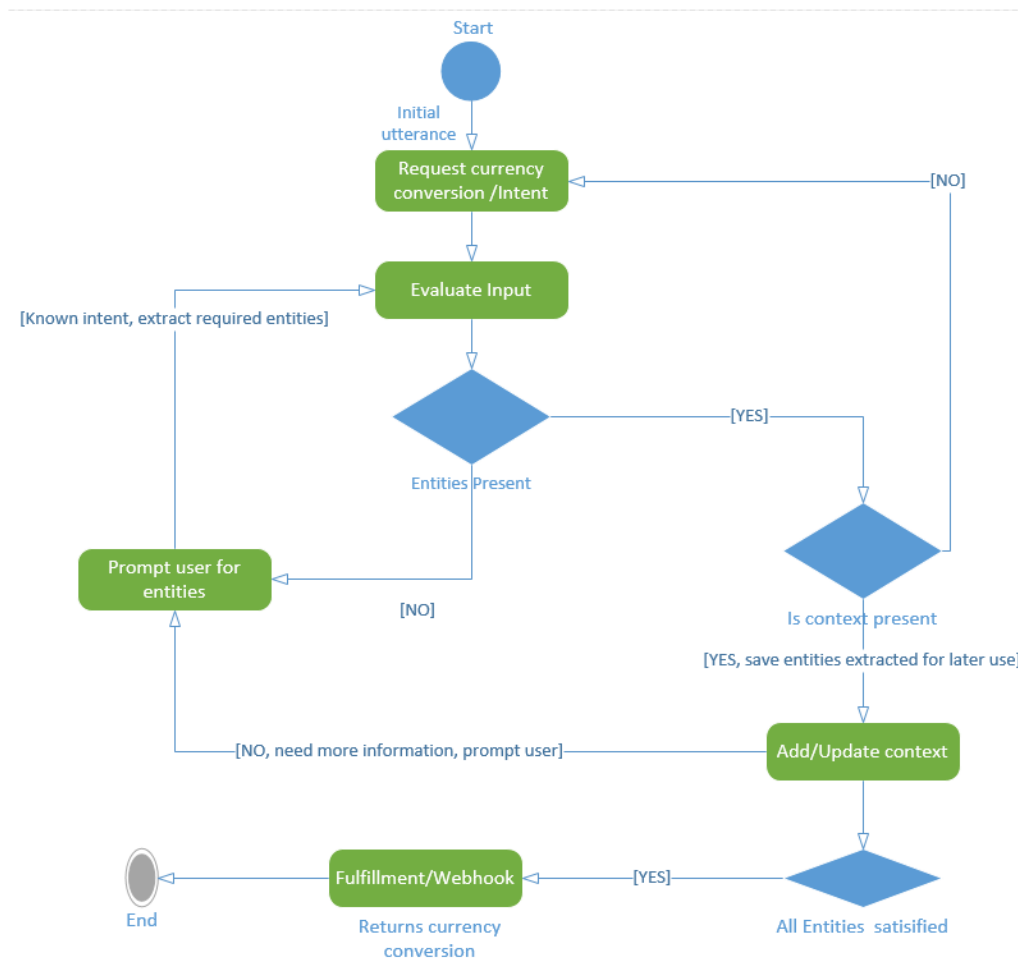
{
  "fulfilment": {
    "speech": "I am your smart banking bot, use me to get your account balance quickly, view transactions, debits, account details and receive currency conversions.",
    "messages": [
      {
        "type": 0,
        "speech": "I am your smart banking bot, use me to get your account balance quickly, view transactions, debits, account details and receive currency conversions."
      }
    ]
  },
  "score": 1 - Exact Match
}
  
```

says: *“Who are you?”*.

3.5 Activity Diagrams

Activity diagrams are drawn to specify the informational flow of the data and outline the processes that are involved in the chatbot. They represent the users' behaviour when interacting with the chatbot and how it responds to these actions through processes. These will breakdown the distinct functionality of the chatbot.

Figure ~~10.3.5.1~~ describes the chatbot logic when a user requests currency conversion. Once the NLU evaluates the user input and determines a context, the user utterance is passed to the Fulfilment webhook. The webhook utilises POST endpoints and accepts JSON returned from the chatbot NLU. The webhook then determines the associated action of an intent the user supplied utterance is mapped to; the action of an intent was defined in the Dialogflow console during training. If the action is present it will extract the entities from the given utterance and _call the Fixer.io API and return the appropriate response encoded in JSON format. If the NLU cannot recognise the intent of the user supplied utterance it will prompt the user for more information to try and recognise the intent and context to provide fulfilment for the user.



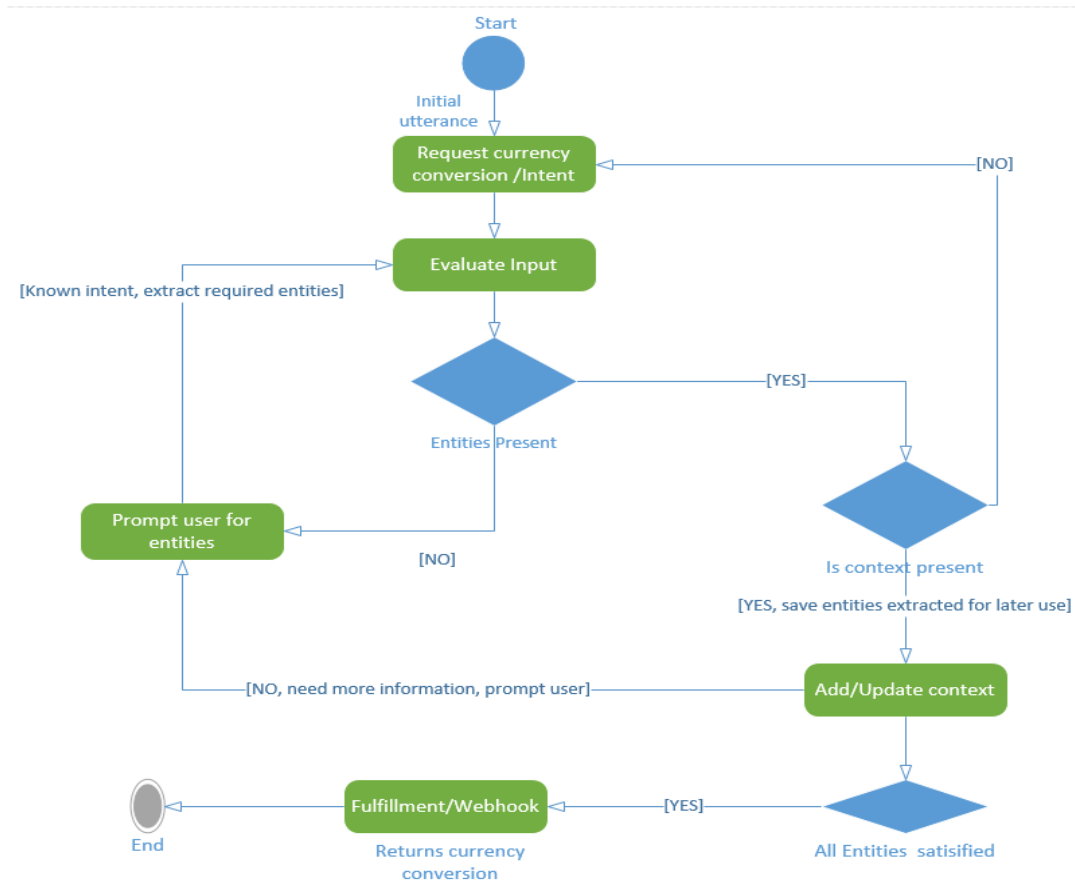
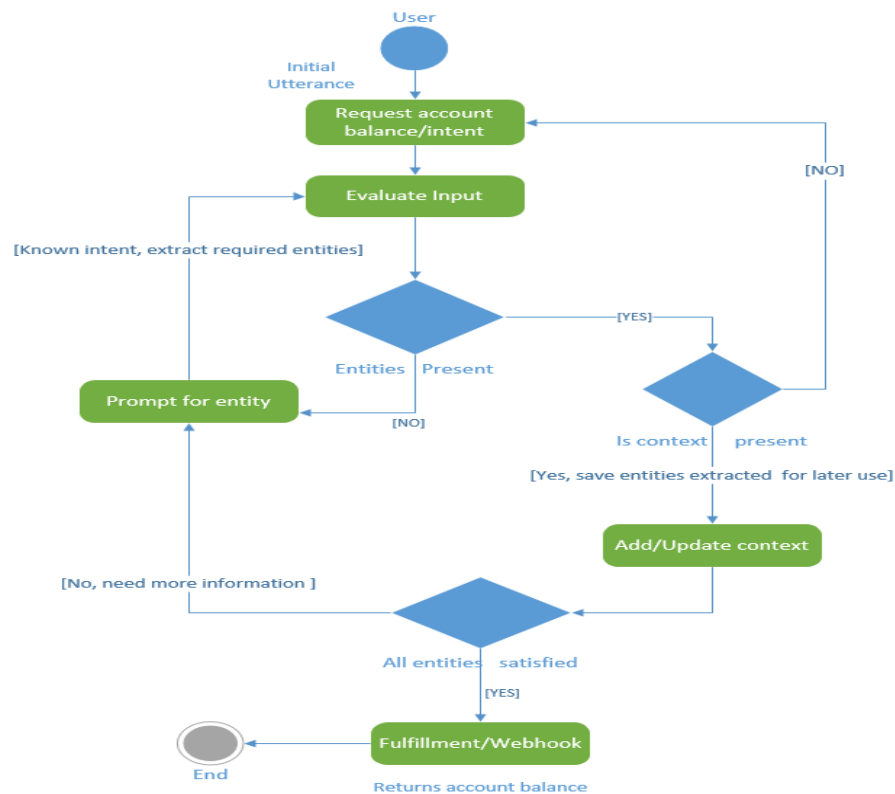
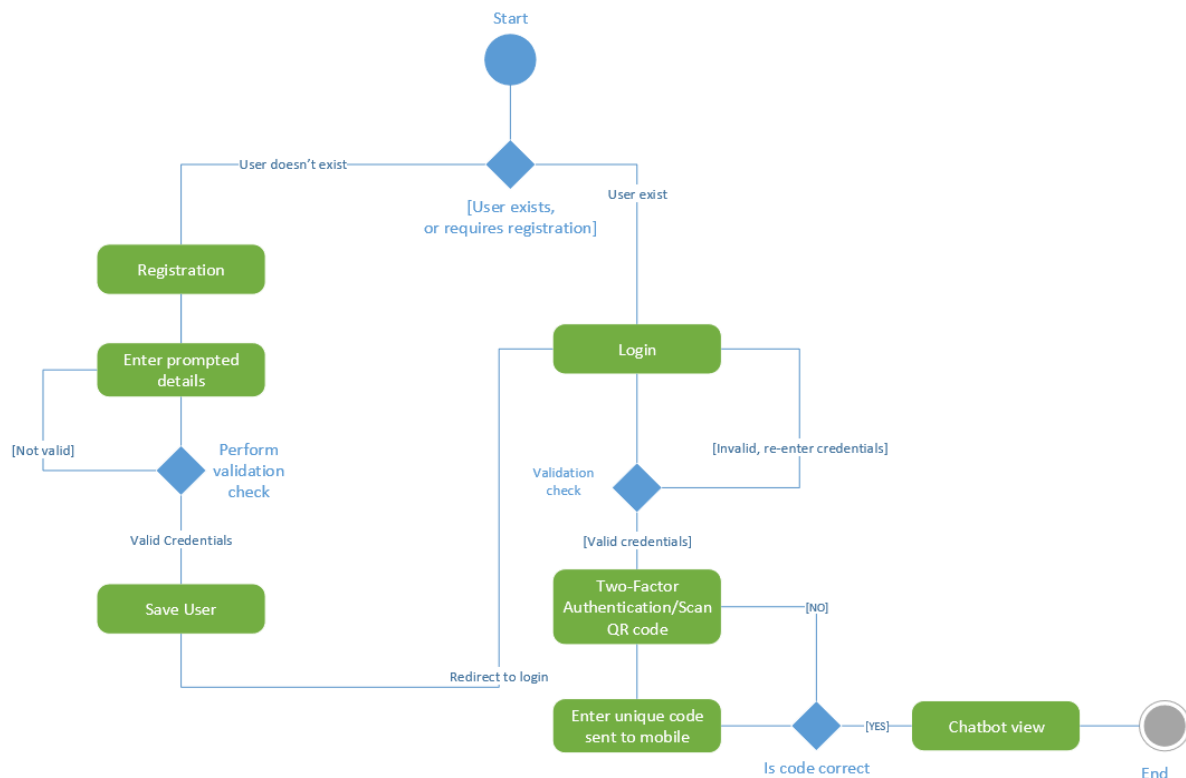


Figure 10: 3.5.1- "Currency conversion" UML activity diagram.



~~Figure 11.3.5.2~~ “Account balance” UML activity diagram.

The diagram shown in Figure [113.5.2](#) outlines the flow of data when a user requests their account balance. The NLU determines the context and presence of required entities in user _input that was matched to an intent. The webhook then receives the user input in JSON



format.

Figure 12: “Registration and Login” UML activity diagram.

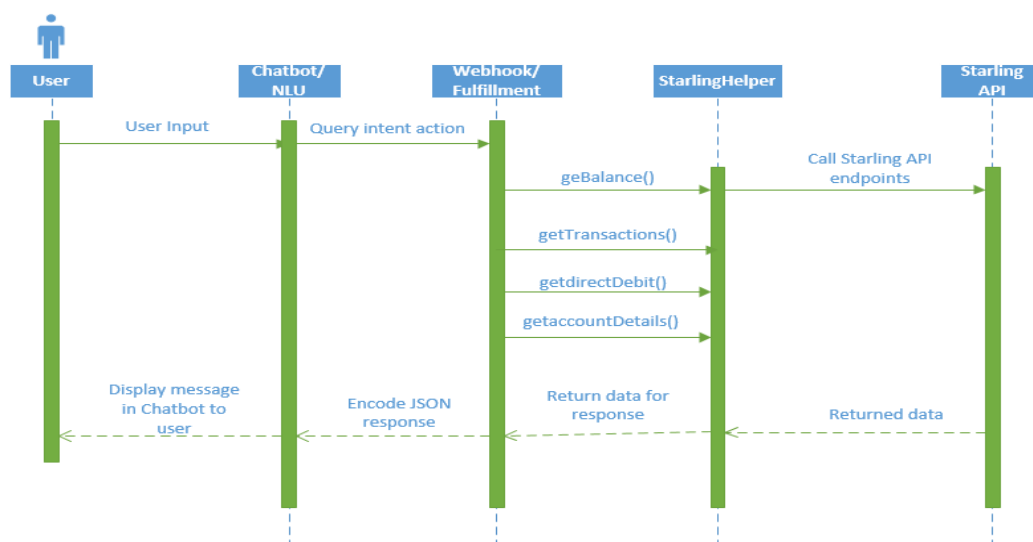
The webhook is utilised to deliver appropriate custom fulfilment to users by extracting the action of the intent it receives and checking whether the action is present in the intent. It then calls the appropriate function based on the current action. If the action is present the function `getBalance()` is called. This method calls the TrueLayer API which uses REST endpoints. The response is then encoded in JSON format and sent to the chatbot NLU. The Laravel framework comes with an authentication package to handle authentication of the application. Figure [123.5.3](#) outlines the process for login and registration. Users must first register on the application to set up an account. If the user attempts to login before registering they will not be allowed access the chatbot view. Upon requesting to login, a database check is performed to determine the validity of the user’s login details. If the details are valid the user will be redirected to the Two-Factor Authentication view. In this view the user will be required to scan a QR code displayed on screen to receive a unique code which must be

entered to verify their account, if the code is correct the user is redirected to chatbot view.

Figure 12:3.5.3 “Registration and Login” UML activity diagram

3.6 Sequence Diagrams

The sequence diagram shown in Figure 13 3.6.1 highlights the operation of the user requesting their banking information through the chatbot, outlining the fundamental process involved. The webhook receives the user input from the Chatbot NLU engine (Dialogflow) in the form of JSON. The webhook determines the action of the defined intent posted from the NLU by parsing the JSON it receives. If the action matches a defined banking operation, the webhook will make the appropriate method ~~method~~ call to the TrueLayerClient~~StarlingHelper~~ Class, which calls the TrueLayer ~~Starling~~ API and returns the data to the webhook to generate a for



the response. Once the webhook receives the data returned by the API it encodes the data to JSON which is as the response sent back to the NLU to be displayed to the user through the Chatbot view.

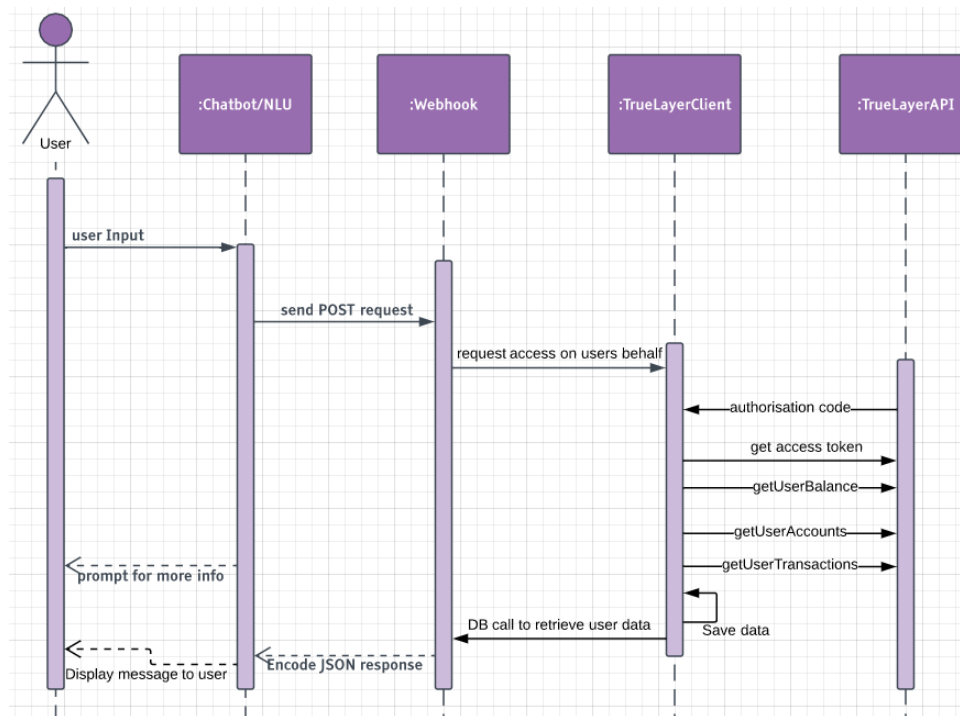


Figure 3.6.13: "Request banking Information" sequence diagram.

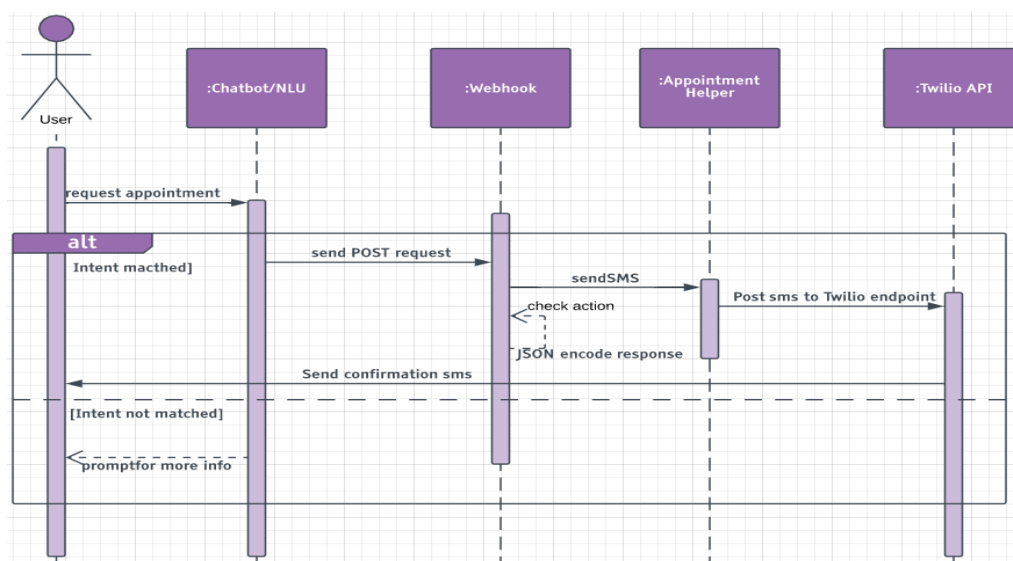


Figure 14.3.6.2 "Appointment & confirmation"

In order to implement a successful chatbot the user group that will be interacting with it is carefully thought through. The target audience for the chatbot consists of different age groups with varying technical ability, as mobile banking is relevant across a broad range of age groups. To appeal to all age groups the chatbot will utilise simplistic and straightforward responses. Google assistant allows developers to determine a voice for their bot that matches the personality and utilise rich responses (Actions on Google, 2018). However the majority

of users of an application like this already carry out the most part of their online interaction through some messaging platform, so the use of such an application should come with ease (Interactions.acm.org, 2017).

3.7 Conversational User Interface Design

Interaction will take place within the chatbot in the form of natural language and this will be the focus of design in regards to conversational user interfaces (CUI). “Conversation as a method of interaction is often referred to as the new UI” (Actions on Google, 2018). (cs.hmc.edu, 2018) define a set of design patterns to adhere to when designing Conversational User Interfaces.

Table 4: CUI Design Principles.

PRINCIPLES		DESCRIPTION
1.	Design by personality	The bot personality should provide a natural and unique interaction experience for users. The personality should be designed to guide the users through the interaction which may include being aware of what the user may say or want. The personality is the style and manner the bot holds throughout the interaction.
2.	Flexibility in response	Flexible responses should be provided to user input, the bot should have various/random responses to the same user input.
3.	Text vs Custom Buttons	It should be clear to the user what input is expected to carry on in the conversation. A fallback response should be used in situations where the input is not recognised such as suggesting other operations that are available. Instead of solely relying on natural text commands from the user, a good UI should render structured options to the user in form of buttons or text.
4.	Simplicity in Interaction	Be concise and clear in response to questions. The conversation flow should be

		straight and not branch out to complex paths.
5.	Conversation Flow	The conversation flow is important to ensure that the user is not irritated in situations where the bot may not be able to provide an appropriate answer immediately. The user would then have to undergo a conversation in order to determine the answer to their query.
6.	Tasks and duty specifications	The bot should deliver explicit tasks for a particular domain.
7.	Rigid syntax, then NLP	People are prone to spelling mistakes or using colloquial phrases which may not be understood by the bot.
8.	Empathy and emotional state	It is important that the bot builds a rapport with the user and convey emotion in responses based on certain user input.
9.	Keep conversation short	Users want to interact with the bot to receive answers or reach a goal quickly. Avoid long winded conversations as it will make the interaction feel burdensome. This helps avoid ambiguity.
10.	Triggers and actions	The bot needs to persuade users and seek ways to encourage users to carry out specific actions through the bot.
11.	Predict and personalise	Bots can analyse previous input from users for important parameters, in order to anticipate a user's predisposition. It will then provide an answer that would be unique to that particular user.
12.	Fully/partially automated	NLP should be used for automated tasks.
13.	Providing a way out	Allow user to begin the conversation again or exit the conversation.
14.	User boredom	The conversation should engage the user throughout the interaction.

Table 3: "CUI Design Principles"

A set of design patterns where outlined for domain specific chatbots including those utilised within a finance industry.

Table 5: CUI Design Patterns.

DESIGN PATTERNS		DESCRIPTION
1.	Follow-up questions	Follow-up questions should be specific in the context of the conversation. An example of this is when the user wishes to request a currency conversion. The chatbot would ask; “what is the amount and currency currency you wish to convert to?”
2.	Cognitive load	Chatbots should present coherent choices to users to reduce the amount of effort needed for the user to interact with it. This can be achieved through the use of buttons and images.

Table 4: “CUI Design Patterns”

3.8 Dialog Design

As main focus of interaction is between the user and the chatbot, which is carried out through natural language, the conversation design is a crucial aspect to consider when designing the chatbot. To achieve this; appropriate dialog will be designed through the Dialogflow console using follow-up and fall-back intents. A particular intent will be set, that once invoked by the user, the chatbot will end the conversation effectively regardless of the conversational context. The dialog is the collection of words and phrases used to respond to user input. The dialog will

User: “please book me an appointment”

be designed with human like phrases to respond to users with in order to sound more natural. Another design aspect will include fall-back intents. If a user utterance cannot be matched to an intent, the chatbot will respond with a message stating it did not match the input to an intent and prompt the user to repeat the query or provide more details. The dialog is also designed using follow-up intents, these provide a more natural conversational flow when interacting with the chatbot. Follow-up intents extract more information from users as and when required, for instance; a user may say;

User: “Please Book me an appointment”

The chatbot is aware it needs more information from the user such as; date, time and phone number and will therefore use follow-up intents to extract the required information from the

user. Dialogflow recommend designing a linear dialog for interactions where data present in the conversation is extracted to help the users achieve their goal, this is applied to the design specification is applied to the chatbot as it is an task-based interaction chatbot that consist of banking domain knowledge. The user utterance can be said in many different ways:

User: "Show me my balance?"

User: "What is my balance?"

User: "Tell me my account balance?"

In this example the chatbot will extract the required entity parameters 'balance' and 'account' in order to return a custom response to the user (Dialogflow, 2018).

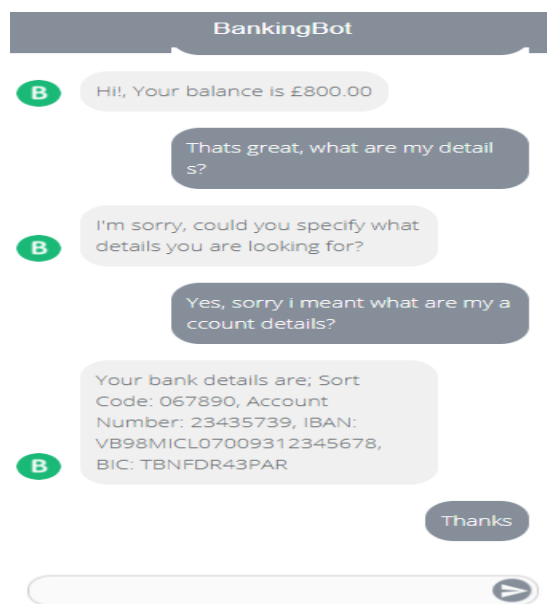


Figure 1453-8.1: "Banking dDialog"
conversion dialog"

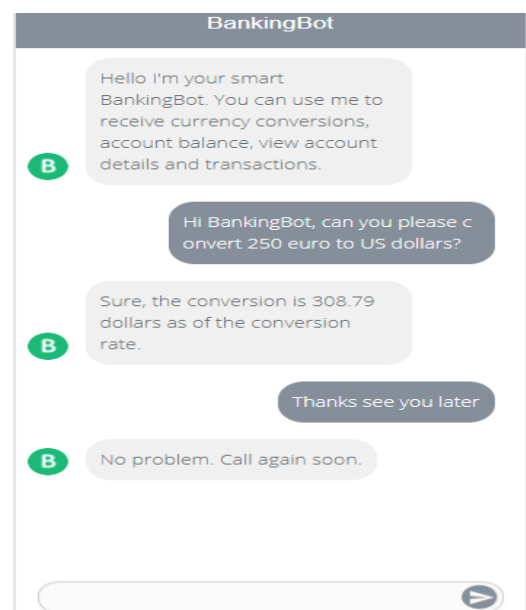


Figure-1563-8.2: "Ceurrency
conversion dialog"

The Ffigures 14 5s 3.8.1 and 15 6-3.8.2 illustrate show the design for the Google assistant application. It follows a

simplistic layout so it is intuitive and efficient for users, in comparison to other forms of traditional UI (DZone, 2018). Users can directly ask a question to the chatbot using the input box or tapping the devices microphone. Once the icon is tapped it will activate the devices microphone and make the user aware it has started recording and the chatbot response will be displayed on screen instantly. Google assistant uses specific colours for each message. The

use of different colours will be utilised to distinguish between the chatbot response and the user input.

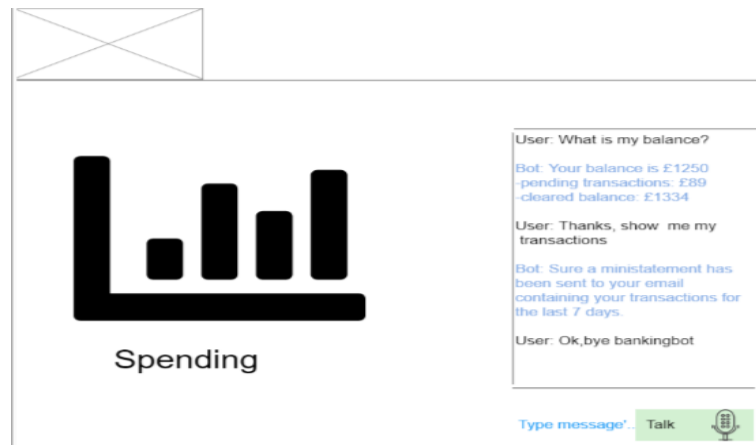
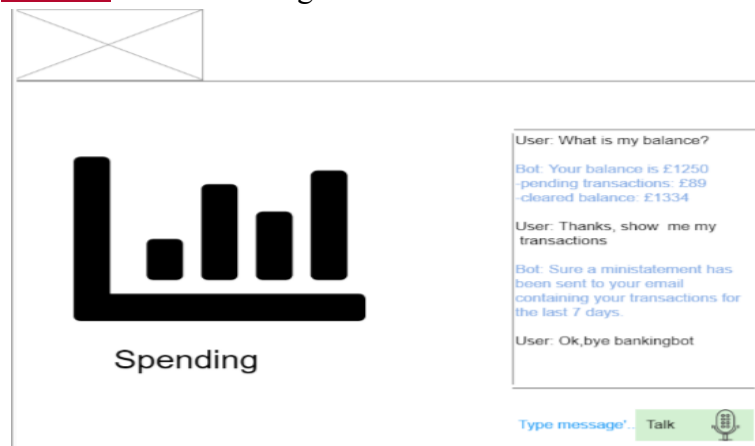


Figure 1673.8.3 “Web-based chatbot”

Figure 1673.8.3 outlines shows the design for the web-based chatbot view. Users can see how




much they


spend and what they spend their money on in a graphical format, this will help users identify over expenditures, for instance; they may realise they spend far too much on coffee each week. It is made visually clear to the user what the bot’s response is, as each individual message has an associated title and colour depending on whether or not it was sent by the user or the bot.

Figure 1783.8.4 shows the QR code displayed to users for them to scan using the authenticator app on their device. Users will not be able to access the chatbot view without verifying their account first as users will be prompted to enter a unique code after each login. Once the QR code is scanned a unique 6 digit number will be sent to the user’s mobile device

_for them to enter. Descriptive text will be used in order to guide the user through the process of downloading the authenticator app.



Verify your Account



Please scan the above QR code using the authenticator app on your mobile device

Please enter the 6 digit unique code to verify your account

BankingBot

Below is a mini statement of your transactions for the last 7 days

Currency	Amount	Direction	Narrative	Source	Balance
GBP	-67	OUTBOUND	Tesco	MASTER_CARD	931.24
GBP	-40	OUTBOUND	Topshop	MASTER_CARD	998.24
GBP	-15	OUTBOUND	Easons	MASTER_CARD	1038.24
GBP	300	INBOUND	Work	FASTER_PAYMENTS_IN	1053.24
GBP	-12	OUTBOUND	cafe	VISA_DEBIT	753.24
GBP	-34.76	OUTBOUND	BIKE PRO	MASTER_CARD	765.24
GBP	800	INBOUND	Student Finance	FASTER_PAYMENTS_IN	800

Thanks,
BankingBot

[Return to chat](#)

~~Figure 178: 3-8-4~~ "Verify Account View"

~~Figure 189: 3-8-4~~ "Transaction Email"

Users will be able to view their transactions on a regular basis without the need to navigate through various web-pages. Once the user requests to view their transactions an email will be sent to their email address that was provided at registration.

Design Conclusion

4. Implementation and Testing

The main functionality of the chatbot is outlined. The chatbot was implemented using the MVC approach, and adheres to Laravel blade template stated in section 3.1 of the design phase chapter. Chatbot specific testing is carried out and the results are outlined and analysed.

4.1 Configuring the development environment:

Laravel homestead was chosen used for the virtual development environment. This is essentially a

vagrant wrapper for Laravel. It comes with all the required dependencies to create a Laravel application, to include; PHP, MySQL, Nginx Server, Node, Git, Ubuntu and ngrok (Laravel, 2018). This saves a lot of time as it avoids the need to manually install any required packages, in order to set up the ideal development environment for PHP projects. However, Homestead requires installation prerequisites to include; Vagrant and VirtualBox. Laravel

was also added globally to the root directory in order to utilise the framework and create a project. Figure ~~19204.1.1~~ shows the command ~~which was executed used~~ to install the Laravel\Homestead vagrant box

```
C:\Users\dana\Documents\bot\bankbot>composerrequire laravel/homestead --dev
```

in the project directory.

Figure ~~1920:4.1.1~~ “Adding Homestead vagrant box to project”

To configure Homestead, the command ~~presentedshown~~ in figure ~~2014.1.2~~ was ~~utilisedapplied~~ to initialise the environment inside the project directory.

```
C:\Users\dana\Documents\bot\bankbot>vendor\bin\homestead make
```

Figure ~~2014.1.2~~: “Configure Homestead command”

The homestead command ~~showndepicted~~ in figure ~~2014.1.2~~ creates a Homestead.yaml file to map to the applications folders in the local directory to the appropriate location on the vagrant box ~~so~~. The virtual server can then determine the location of the project. Once the Laravel/Homestead vagrant box is configured, the vagrant box can host the chatbot on an Nginx server. The project Homestead.yaml file, ~~is shown in~~ figure ~~212,4.1.3~~ outlinesing the applications location on the vagrant box.

```
1 ip: 192.168.10.10
2 memory: 2048
3 cpus: 1
4 provider: virtualbox
5 authorize: ~/.ssh/id_rsa.pub
6 keys:
7   - ~/.ssh/id_rsa
8 folders:
9   -
10     map: 'C:\Users\dana\Documents\bot\bankbot'
11     to: /home/vagrant/code
12 sites:
13   -
14     map: homestead.test
15     to: /home/vagrant/code/public
16 databases:
17   - bankbot
18   name: bankbot
19   hostname: bankbot
20
```

Figure ~~2124.1.3~~: “Homestead.yaml file”

Note the IP address at the top of the file ~~showndisplayed~~ in figure ~~2124.1.3~~ is the IP for the Vagrant box

and the connected database is ~~specifieddeclared~~ under ‘databases:’

To serve the project the “vagrant up” command shown in figure ~~2234.1.1~~, ~~this~~ is ~~executeddeployedrun withinin~~ the

project directory through the terminal.

```
C:\Users\dana\Documents\bot\bankbot>vagrant up
```

Figure ~~223~~4.1.4: "Vagrant up command"

The vagrant box also contains the operating system required for this project which is Ubuntu Linux. In order to connect to the virtual machine an SSH terminal command is used to establish a secure connection to the vagrant box.

```
C:\Users\dana\Documents\bot\bankbot>vagrant ssh
```

Figure ~~234~~4.1.5: "Vagrant ssh command"

4.2 Authentication:

An authentication package is included with the Laravel framework that handles authentication. ~~The Laravel framework comes with an authentication package to handle authentication.~~ The command defined/outlined in figure 2454.2.1 shown below generates the foundation for authentication. This creates the controllers, views and routes to support authentication.

```
vagrant@bankbot:~/code$ php artisan make:auth
```

Figure ~~245~~4.2.1: "php artisan auth command"

4.3 Database Development

Once the authentication package was ~~added~~installed, the database was subsequently migrated to include/add the users_-table ~~to the database~~. Laravel comes with built in support for creating migrations to define the database schema in order to create the required tables. Migrations allow database tables to be updated or rolled back to a previous version if necessary. The artisan command shown in figure ~~256~~4.3.1 ~~is used to~~migrates the database tables defined in newly added migration files.

```
vagrant@bankbot:~/code$ php artisan migrate
```

Figure ~~256~~~~4.3.1~~: “php artisan migrate command”

The migration command outlined in figure ~~267~~~~4.3.2~~ creates a new migration file to update the users table to store their Google 2-factor authentication credentials.

```
vagrant@bankbot:~/code$ php artisan make:migration AddGoogle2faToUsers
```

Figure: ~~4.3.2~~~~67~~: “php artisan make migration command”

4.4 Google Two-Factor Authentication:

~~As an extra layer of security~~To further enhance security, Google Two-Factor Authentication was implemented given the

context of the chatbot. The application uses an authentication package to implement Google Two-Factor Authentication. The package was installed through composer using the Following commands:

```
C:\Users\dana\Documents\bot\bankbot>Composer require pragmarx/google2fa-laravel
```

```
C:\Users\dana\Documents\bot\bankbot>Composer require bacon/bacon-qr-code
```

```
public function G2FA()
{
    $user = Auth::user();

    $google2fa = new Google2FA(); //G2FA instance
    $user->google_2fa_secret = Google2FA::generateSecretKey();
    $user->save();

    $google2fa_url = Google2FA::getQRCodeGoogleUrl(
        'BankingBot',
        $user->email,
        $user->google_2fa_secret
    );
    return view('home', compact('user', 'google2fa_url'));
}
```

Figure ~~27~~~~84.4.1~~: “G2FA function”

The G2FA() function ~~presented defined~~ in figure ~~278~~~~4.4.1~~, displays the code used to implement 2-Factor

authentication. The authenticated user is obtained and an instance of the Google2FA facade is created. A unique secret key is ~~added~~appended to the user’s record and saved to the database. The

getQRCodeGoogleUrl ~~methodfunction~~ of the facade creates an Image URL for the QR code, ~~thus~~

allowing users to scan the QR code using their authenticator app. ~~The users email is sent along with the secret key, and the application name (BankingBot), to their authenticator app~~The users email along with

~~the secret key;~~ Users must manually enter the ~~which they are~~ required ~~keyto enter~~ upon verifying their account. ~~is sent to their~~

~~authenticator app along with the name of the application (BankingBot), this is to identify~~

~~which key should be used for their account.~~The QR image URL is then passed to the view to display the QR code.

```
public function verify_g2fa(Request $request){
    $user = Auth::user();

    $secret = $request->input('secret');
    $window = 8; /* 8 keys (respectively 4 minutes) past and future.
                  Time window in which the key will remain valid until
                  a new key is generated.
                  */
    $valid = Google2FA::verifyKey($user->google_2fa_secret, $secret, $window);

    if ($valid) {
        $user->g2fa_active = true;
        $user->save();
        $request->session()->put('g2fa_authorized', true);
    }

    return redirect('chatbot');
}
```

Figure 289-4.2: “verify_g2fa function”

Figure 289-4.2 shows ~~depicts the~~ verify_g2fa() controller function which handles verification of the unique

secret key submitted by the user. ~~This particularthe_secret~~ key is obtained from the incoming request ~~which~~

holding the posted data. The window variable specifies the time frame for ~~how longwhich~~ the key ~~remainsis~~

valid. A secret key is created every 30 seconds. If the key is valid, a database operation is performed to set the boolean user attribute g2fa_active to true in the database, ~~save it to the database~~ and save the verified user ~~add it to the HTTP~~

session to determine if the user is verified between session states. ~~and the user is redirected to the chatbot view.~~

The Laravel blade file, which defines the frontend for the account verification view, is shown in figure 29304.4.3, the image URL for the QR code is used to display the image within a HTML

image tag. The user is able to submit their secret key via the form input and Laravel comes with built in support to prevent csrf vulnerabilities. The input is posted to the corresponding HTTP route outlined defined in figure 3044.4.4.

```
@extends('layouts.app')

@section('content')
<div class="container">
  <div class="col-md-8 col-md-offset-2">
    <div class="panel panel-default panel-info">
      <div class="panel-heading"><h2 class="pull-right"><center>Verify your Account! <i class="fa fa-check-cir
    <div class="panel-body">
      @if (session('status'))
        <div class="alert alert-success">
          {{ session('status') }}
        </div>
      @endif
      <center>{{ Html::image($google2fa_url) }}</center>
      {!! Form::open() !!}
      <div class="form-group">
        <label class="control-label">Enter Verification Code</label>
        {{ csrf_field() }}
        {!! Form::text('secret', null, ['class'=>'form-control']) !!}
      </div>
      <small>Please verify your account by scanning the QR code using the Google Authenticator app
      <small>(The app can be downloaded from the App Store or Google Play Store)</small>
      <center><button type="submit" class="btn btn-md btn-success">Verify</button></center>
      {!! Form::close() !!}
    </div>
  </div>
</div>
@endsection
```

Figure 29304.4.3: "Verify Account view"

The web routes used to handle account verification are shown in the below snippet:

```
Route::get('/home', 'HomeController@index')->name('home');
Route::post('/home', 'HomeController@verify_g2fa')->name('home');
```

Figure 3044.4.4: "Web Routes"

An authorisation link is embedded within the controller class which in turn redirects the user's browser to the TrueLayer's Authorisation Server. A list of real world banks is presented where the user can select their bank and login with their online banking credentials.

4.5 Open Banking API:

The TrueLayer API was integrated in order to access the user's banking data enabling the chatbot to conform to Open Banking standards. This API allows users to select from a wide

range of banks in which they have a pre-existing bank account with. Users can view their banking information from various accounts from within the chatbot.

After successfully logging into their bank account and permitting the application access to their sensitive data. TrueLayer API authenticates users through the OAuth2 RFC6749 flow to grant the application access to the users' personal banking data on their behalf once redirected back to the web application. This is achieved by obtaining an access and refresh token through a TLS encrypted connection, once the user is authorised and redirected back to the chatbot application. The refresh token is stored in the database, to uniquely identify users upon each request. The method used to retrieve the access token is shown in figure 3124.5.1

~~The TrueLayer API was integrated in order to access the user's banking data enabling the chatbot to conform to Open Banking standards. This API allows users to select from a wide range of banks in which they have a pre-existing bank account with. Users can view their banking information from various accounts from within the web application. TrueLayer API authenticates users through the OAuth2 flow to grant the application access to the user personal banking data once redirected back to the web application. This is achieved by obtaining an access and refresh token once the user is redirected back to the chatbot application after logging into their bank account using their online banking credentials. The refresh token is stored in the database, to uniquely identify users upon each request. The method used to retrieve the access token is shown in figure~~

```

public function getAccessToken(Request $request)
{
    $user = Auth::user();
    $client = new Client();
    $code = $request->code;

    $response = $client->request('POST', 'https://auth.truelayer.com/connect/token',[ /*obtain access token
    through code exchange

    'form_params'=>[
        'grant_type'=>'authorization_code',
        'client_id'=>'bankingbot-36xh',
        'client_secret'=>'cq8ih0hphvuhph5bo95klt',
        'redirect_uri'=>'http://localhost:8000/truelayer',
        'code'=>$code
    ]
    ]);
    $body = $response->getBody();
    $objectResponse = json_decode($body);
    $this->access = $objectResponse->access_token;

    $result = $client->request('GET','https://api.truelayer.com/data/v1/info',[ //retrieve user info
    'headers'=> [
        'Authorization'=> 'Bearer ' . $access //request headers sent on each API request
    ]
    ]->getBody()->getContents();

    return $this->access;
}

```

Figure 324.5.1: “getAccessToken Method”

The TrueLayer API is A RESTful API which returns JSON that is parsed and stored in the database. The API is queried using HTTP requests to specified endpoints. Guzzle is integrated -to send HTTP requests to the API endpoints. A HTTPS POST request is sent to the token endpoint of the Truelayer API containing the code retrieved from incoming request after the user is redirected back to the web application after successfully logging into their bank.

```

public function getAccounts($access)
{
    $user = Auth::user();
    $client = new Client();
    $result = $client->request('GET', 'https://api.truelayer.com/data/v1/accounts', //accounts
    [
        'headers'=>[
            'Authorization' => 'Bearer ' . $access //request headers
        ]
    ]
    )->getBody()->getContents();
    $accounts = json_decode($result,true);

    foreach ($accounts['results'] as $account) { //save user accounts to db
        $user_accounts = $user->accounts()->updateOrCreate(['user_id'=>$user->id,
        'account_id'=> $account['account_id'],
        'account_type'=>$account['account_type'],
        'name'=> $account['display_name'],
        'currency'=>$account['currency'],
        ]);
    }
    return $accounts;
}

```

Figure 324.5.2: “getAccounts method”

To retrieve the ~~user~~ accounts a user they may hold with their bank the TrueLayer data API endpoint is

queried as outlined figure 323. This returns a JSON object containing the account details, for instance a user may

have a
savings
The
then
and the
saved
the

```
public function getAccountDetails($access)
{
    $user = Auth::user();
    $accountId = $user->accounts()->account_id; //current account ID
    $client = new Client();
    $result = $client->request('GET', 'https://api.truelayer.com/data/v1/accounts/' . $accountId,
    [
        'headers'=>[
            'Authorization'=> 'Bearer ' . $access,
            'Content'=>'application/json'
        ]
    ]->getBody()->getContents();
    $accountDetails = json_decode($result, true);

    $user = Auth::user();
    foreach ($accountDetails['results'] as $account) {
        $user->accounts()->update([
            'iban'=>$account['account_number']['iban'],
            'swift_bic'=>$account['account_number']['swift_bic'],
            'number'=>$account['account_number']['number'],
            'sort_code'=>$account['account_number']['sort_code'],
        ]);
    }

    return $accountDetails;
}
```

current and
account.
JSON is
decoded
details are
to
database.

Figure 4.5.334: “getAccountDetails function”

The users account Id is appended to the end of the request URL to return specific information regarding the account specified on the API call. The users IBAN, BIC, Account Number and Sort code are saved to the database.

```

public function getBalance($access)
{
    $accountId = $user->accounts->id;
    $user = Auth::user();
    $client = new Client();
    $result = $client->request('GET', 'https://api.truelayer.com/data/v1/accounts/' . $accountId . '/balance');
    [
        'headers'=>[
            'Authorization'=> 'Bearer ' . $access,
            'Content'=>'application/json'
        ]
    ]->getBody()->getContents();
    $balance = json_decode($result,true);

    $user->accounts()->updateOrCreate(['available_balance' => $balance['results']['available'],
                                     'current_balance'=>$balance['results']['current']
    ]);
    return $balance;
}

```

Figure 3454.5.4: “getBalance function”

An important feature is to retrieve the users balance, to ensure users can instantly view their account balance information. Figure 345 4.5.4 highlights shows the *getBalance()* method function, used to retrieve the users balance. This function makes a call to the defined endpoint specified by Truelayer Data API to retrieve a user’s balance information such as available and current balance. Firstly the result returned from the request are decoded into a PHP array and then parsed to extract the users balance which is then stored in the database.

```

public function getTransactions($access)
{
    $user = Auth::user();
    $accountId = $user->accounts->account_id;

    $client = new Client();
    $result = $client->request('GET', 'https://api.truelayer.com/data/v1/accounts/' . $accountId . '/transactions',
    [
        'headers'=>[
            'Authorization'=> 'Bearer ' . $access,
            'Content'=>'application/json'
        ]
    ]->getBody()->getContents();
    $transactions = json_decode($result,true);

    foreach ($transactions['results'] as $transaction) {
        if(array_key_exists('merchant_name', $transaction)){ //check key exists, some result
            $transaction = str_replace("-", "", $transaction); // remove '-' char before saving
            //var_dump($test);
            $user->transactions()->insert([ //store transaction history to database
                'user_id'=>$user->id,
                'transaction_type'=>$transaction['transaction_type'],
                'transaction_category'=>$transaction['transaction_category'],
                'merchant_name'=>$transaction['merchant_name'],
                'currency'=>$transaction['currency'],
                'amount'=>$transaction['amount']
            ]);
        }
    }
}

```

Figure 3564.5.5: “getTransactions function”

Figure 3564.5.5 outlines the function which retrieves the user’s transactions from the

API for a specific account. The transactions object returned from the API is filtered by the 'merchant_name' index as not all transactions returned contained the index 'merhant_name', each transaction is then saved to the database.

4.6 Webhook Development:

The webhook is developed to implement custom fulfilment for the chatbot. Dialogflow posts data to the chatbot through HTTPS requests. In order to generate a publicly accessible URL, the tunnelling tool known as

ngrok was used to expose the~~create a tunnel to~~ localhost fulfilment web route endpoint. This is

```
vagrant@bankbot:~/code$ share bankingbot.test
```

achieved

through the following

command:

The webhook was implemented as a controller class ~~that expose the web route as an endpoint~~. The webhook receives the data from Dialogflow NLU and parses the JSON to extract the entities present in the user utterance to provide a custom response to the user.

```
public function handle(Request $request){
    $request = json_decode(file_get_contents("php://input"));
```

Figure 3674.6.1: "Webhook handle function"

The PHP json_decode() function is used to convert the incoming JSON Object from the NLU into a PHP variable. The code shown in figure 3784.6.2 is a snap shot of the webhook which returns a custom response when a user requests a currency conversion. Firstly the appropriate entities are extracted from the request posted to the Webhook then a method call is invoked to the function that will query the Fixer.io API using the Currency Rates library installed and the response is returned to Dialogflow in JSON format. The response is defined for both speech and text in the response array. Each response is sent with the header Content type: application/json which will display the JSON in plain text to the user through the chatbot.

```

if($request->result['action'] == 'convert') //if action is convert extract appropriate entities
{
    $currency = $request->result['parameters']['amountToConvert'];

    $amountto = $currency['amount'];
    $baseCurrency = $currency['currency'];

    $outputCurrency = $request->result['parameters']['outputCurrency'];

    //check if input currency is the same as output currency
    if($baseCurrency == $outputCurrency)
    {
        $responseText = "The currency " . $baseCurrency . " is the same as " . $outputCurrency;
        $response = array("speech" => $responseText,
                        "displayText" => $responseText
                        );
        header('Content-Type: application/json');
        echo json_encode($response);
    }else {
        //query fixer.io API
        $result= $this->convert($amountto, $baseCurrency, $outputCurrency);
        $response = array("speech" => $result,
                        "displayText"=>$result );
        header('Content-Type: application/json');
        echo json_encode($response);
    }
}
}

```

Figure 3784.6.2: “Webhook”

In order to implement currency conversion functionality a PHP library called “CurrencyRates” was integrated into the project using the following command:

```
vagrant@bankbot:~/code$ composer require ultraleet/currency-rates
```

Figure 3894.6.3 shows the convert function which implements the library to call the Fixer.io API.

The required entities are extracted out of the user response and passed as parameters to the convert function within the webhook and the conversion is returned in a string format.

```

public function convert($amountto, $baseCurrency, $outputCurrency)
{
    // Set the amount by chaining in an amount() call
    $result = $this->currencyRates->driver('fixer')->base($baseCurrency)
    |->amount($amountto)->target($outputCurrency)->get();

    // Get the converted values
    $values = $result->getConverted(); // returns an array of values

    // You can also access the results as a property:
    $value = $result->converted[$outputCurrency]; // returns 120.07

    return "The conversion is " . $value . " as per conversion rate";
}

```

Figure 4-6.389: “convert function”


```

if($request->result['action'] == 'balance')
{
    $balance = $this->usersBalance();

    $response = array(
        "speech"=>$balance,
        "displayText"=>$balance,
    );
    header('Content-Type: application/json');
    echo json_encode($response);
}

```

Figure 3940.6.4: “Webhook balance response”

When the incoming POST request is received by the webhook, the action of the request is determined in order to return an appropriate response as outlined in figure 3940.6.4. When the user requests to view their balance, the entity is recognised by the NLU where appropriate actions were defined for each corresponding entity created within the DialogFlow Developer console. The if statement within the webhook evaluates whether the requested action of the user supplied utterance matches the action defined for the balance entity.

Figure 401.6.5 shows the Dialogflow console recognising developer defined entities from an example user training phrase used to develop the chatbot’s knowledge base.

“ show me my account balance ”

Figure 401.6.5: “Example training phrase, entity recognition”

```

public function usersBalance()
{
    $user = Auth::user();
    $user_balance = $user->accounts;

    foreach ($user_balance as $balance) {
        $available = $balance->available_balance;
        $current = $balance->current_balance;
    }

    $responseText = "Your balance is: " . "Available balance " . $available . " \n " .
        "current balance " . $current;
    return $responseText;
}

```

Figure 412.6.6: “usersBalance function”

The `usersBalance()` function illustrated in figure 41function obtains an instance of the authenticated user, through the Auth middleware applied to the corresponding route. A collection of the user’s accounts is retrieved using the ‘accounts’ function defined within the user model. The users available and current balance are returned to the webhook.

The chatbot was trained to recognise custom entities relevant to the banking domain. This allows the chatbot to recognise the entities in the user utterance to deliver a relevant response.

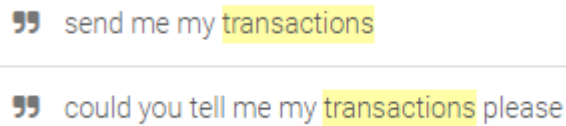


Figure 423-6.7: “User training phrases”

The `retrieveTransactions()` function will retrieve the users transactions from the database using the `transactions()` function implemented in the user model. Laravel’s `Mail::to` function is utilised to send the transactions email to the user.

```
public function retrieveTransactions()
{
    $user = Auth::user();
    $transactions = $user->transactions;
    \Mail::to($user_email->email)->send(new RequestTransactions($transactions));
    return $transactions;
}
```

Figure 434-6.8: “retriveTransactions() webhook function”

The `amount_spent()` function is shown in figure 44,54-6.9 which iterates through each transaction

and adds each transaction item to an array to work out the total. The sum is calculated for every outbound transaction which is the amount the user spent over the last 7 days.

```
public function amount_spent()
{
    $user = Auth::user();
    $transactions = $user->transactions;

    $amountSpent = [];

    foreach ($transactions as $spent) {
        $amountSpent[$spent['amount']] = $spent['amount'];
    }
    return array_sum($amountSpent);
}
```

Figure 445-6.9: “amount_spent webhook function”

An email is sent containing the user transactions for the last seven days using the `Mail::to` command as part of the mail facade. An instance of the logged in user is obtained and the mail is sent using an instance of the Mailable class created `RequestTransactions`, the

transactions array is passed to the mailable class which is used to construct the email.

The following command creates the Mailable class:

```
vagrant@bankbot:~/code$ php artisan make:mail RequestTransaction --markdown=ema
ils.requests.ministatement
Mail created successfully.
vagrant@bankbot:~/code$
```

The mailable class is used to configure and build the email that is sent to the user. The transactions object is initialised and passed through to the mail view in the build function shown in figure 456.6.10.

```
public function __construct($transactions)
{
    $this->transactions = $transactions;
}

/**
 * Build the message.
 *
 * @return $this
 */
public function build()
{
    return $this->markdown('emails.requests.ministatement')
        ->with(['transactions'=>$this->transactions]);
}
```

Figure 456.6.10: "Laravel mailable class"

The Blade file used to format the transactions email sent to users is outlined ~~outlined~~ in figure 467.6.11.

It consists of markdown and HTML to display the users transactions in a table format to replicate the design of a traditional paper based statement. Markdown components are used

to render certain UI elements such as buttons and message component which defines the message to be displayed to users.

```
@component('mail::message')
<h3>Hello, below is your mini statement of your transactions for the last 7 days</h3>

<table class="table table-striped">
<thead>
    <th width="10%">Currency</th>
    <th width="10%">Amount</th>
    <th width="10%">Direction</th>
    <th width="10%">Narrative</th>
    <th width="10%">Source</th>
    <th width="10%">Balance</th>
</thead>

@foreach($transactions as $transaction)
<tr>
    <td>{{ $transaction['currency'] }}</td>
    <td><strong>{{ $transaction['amount'] }}</strong></td>
    <td>{{ $transaction['direction'] }}</td>
    <td><strong>{{ $transaction['narrative'] }}</strong></td>
    <td>{{ $transaction['source'] }}</td>
    <td><strong>{{ $transaction['balance'] }}</strong></td>
</tr>

@endforeach
</table>

@component('mail::button', ['url' => '/chatbot'])
Return to chatbot
@endcomponent

Thanks,<br>
{{ config('app.name') }}
@endcomponent
```

Figure 467-6-11: "Markdown Laravel mailable view"

```
public function make_appointment($time, $topic, $date,$phoneNumber)
{
    $user = Auth::user();
    $user_id = User::find($user->id);

    $set_date = implode("", $date);

    if($user->appointments()->booked == NULL){
        $appoint= DB::table('appointments')->insert([
            'user_id'=>$user_id,
            'time'=>$time,
            'topic'=>$topic,
            'date'=>$date,
        ]);

        $notify = new AppointmentReminder();
        $notify->sendMessage($phoneNumber, $date, $time);

        return "Your appointment successfully made for " . $time . " on " . $date .
            "\n" . "a text reminder has been sent to phone";
    }else {
        return "I'm sorry you already have an appointment";
    }
}
```

Figure 478-6-12 "make_appointment function"

The code ~~defined~~ shown in figure 478-6-12 shows the `make_appointment()` function which obtains an

instance of the authenticated user in order to make an appointment. The

required parameters used to make the appointment are passed from the webhook using a

method call defined in figure 4.6.13. In the *make_appointment()* function an instance of the AppointmentReminder class is created in order to send a text notification of their appointment. The users phone number and appointment details are passed as parameters.

```
if($request->result['action'] == 'appointment')
{
    $time = $request->result['parameters']['time'];
    $topic = $request->result['parameters']['topic'];
    $date = $request->result['parameters']['date'];
    $phoneNumber = $request->result['parameters']['phone-number'];
    $appointment = $this->make_appointment($time, $topic, $date, $phoneNumber);
    $response = array("speech"=>$appointment,
        "displayText"=>$appointment);
    header('Content-Type: application/json');
    echo json_encode($response);
}
```

Figure 4.6.13: "Webhook code snippet"

```
function __construct()
{
    //$this->appointments = Appointments::appointmentsDue()->get();

    $twilioConfig = config('services.twilio');
    $id = $twilioConfig['twilio_account_sid'];
    $token = $twilioConfig['twilio_auth_token'];
    $this->twilioClient = new Client($id, $token);

    //$this->sendMessage($twilioClient);
}

public function sendMessage($userNumber, $date, $time)
{
    $num = (string)$userNumber;
    $this->twilioClient->messages->create(
        $num,
        array(
            "from" => '+447533032850',
            "body" => "FROM BankBot: Your appointment has been confirmed " . "for " . $time . "on " . $date,
        )
    );
}
```

Figure 4.6.14: "AppointmentReminder Class"

Figure 4.6.14 outlines the AppointmentReminder Class which makes use of the Twilio API to

send SMS messages to users to notify them the confirmation details regarding their appointment. The Twilio client credentials are set in the ENV file of the project to ensure they cannot be easily accessed.

4.7 Chatbot Web Client

4.7 Chatbot Web Client:

The frontend of the chatbot makes use of the official JavaScript/HTML5 Dialogflow SDK. The SDK was built upon to add speech synthesis and custom styling using CSS and HTML5. An Ajax POST request sends the user utterance to the '/query' endpoint of the Dialogflow API as shown in figure ~~501~~ 4.7.15 to query the user utterance, where it is matched to an intent and sent to the webhook to supply the user with a response. The Webhook endpoint is defined within the Dialogflow developer console which uses entity recognition to extract meaning from text among other AI and ML methods.

```
function send() {
  var text = $("#input").val();
  conversation.push("Me: " + text + '\r\n');
  $.ajax({
    type: "POST",
    url: baseUrl + "query?v=20150910",
    contentType: "application/json; charset=utf-8",
    dataType: "json",
    headers: {
      "Authorization": "Bearer " + accessToken
    },
    data: JSON.stringify({ query: text, lang: "en", sessionId: "somerandomthing" }),
    success: function(data) {
      var respText = data.result.fulfillment.speech;
      console.log("Respuests: " + respText);
      setResponse(respText);
      speak(respText);
      $("#response").scrollTop($("#response").height());
    },
    error: function() {
      setResponse("Internal Server Error");
    }
  });
}
```

Figure ~~4.7.15~~ 4.7.1: "AJAX post request to Dialogflow API"

```
function speak(text){
  var utterResponse = new SpeechSynthesisUtterance(text);
  utterResponse.rate = 0.7;
  utterResponse.pitch = 1;

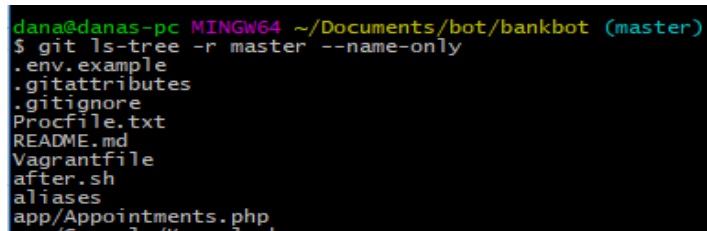
  window.speechSynthesis.speak(utterResponse);
}
```

Figure ~~5124.7.1~~ 5124.7.1: "Speech Synthesis function"

Figure ~~5124.7.1~~ 5124.7.1 shows the function that will produce a speech synthesis of the bots response to the user in order to provide audio responses reducing the amount of time required to interact

with the chatbot. The HTML5 Speech API is implemented for speech recognition and synthesis. Dialogflow comes with support for integration of the Google Assistant, supported across an array of devices including google Home and Android devices. Dialogflow allows developers to extend the Actions on Google (AoG) console to test the chatbot across multiple devices. The Google assistant integration is shown below.

4.8 Git Version Control



```

dana@danas-pc MINGW64 ~/Documents/bot/bankbot (master)
$ git ls-tree -r master --name-only
.env.example
.gitattributes
.gitignore
Procfile.txt
README.md
Vagrantfile
after.sh
aliases
app/Appointments.php

```

Figure 523: “GitHub master branch”

Throughout the development lifecycle GitHub was integrated as the chosen method of version control for the project. As the application was updated during each iteration, and these changes were committed to the GitHub repository to ensure the codebase for the application was managed effectively. Figure 523 outlines the use of the Git Bash terminal that was used to push changes to the master branch of the project.

5. Testing:

A crucial part of any software development lifecycle is testing. This involves carrying out certain procedures and operations to understand the limitations of the software. It is evident that with testing the constraints of the application that particular bugs and errors are picked up and documented through test cases. This will improve the overall standard and quality of the chatbot and enhance the user experience.

Various testing methods were carried out in order to measure the overall effectiveness of the chatbot. ~~The dialog was tested to measure the efficiency of the chatbot which includes measuring how well the chatbot can understand a user supplied utterance, even if miss spelt. Identifying if the intent was recognised, with average response times between text and voice~~

interactions are also included in said response. The chatbot was tested in a specific manner to record the performance metrics.

5.1 Simulated Conversational Testing

The dialog was tested to measure the efficiency of the chatbot which includes measuring how well the chatbot understood ~~can understand the~~ users supplied utterance for both voice and text interactions, even if miss-spelt. Identifying if the intent was recognised, with average response times between text and voice interactions ~~are~~ also included in said response. The chatbot was tested in a specific manner to record the performance metrics.

A Dialogflow command line tool called dialogflow-cli, written in JavaScript, was installed into the project directory for the purpose of undertaking these tests. This was employed to replicate how users would interact with the chatbot and metrics were recorded for each simulated interaction.

To initialise the mock interaction for testing the tool fires a welcome event to the Dialogflow API once instantiated in the command line, it then ~~and~~ displays the response returned from the API as the result ~~returned through the~~ command prompt.

Each type of use case was tested through the command line and the response was returned in JSON format.

A total of 20 simulated utterances were tested and the results are shown in *table 5*. Each utterance tested was uniquely phrased including variations of the same utterance to ~~accumulate~~ accumulate a range of results ~~so and thoroughly classify~~ the chatbots understanding can be thoroughly classified.

From the results shown in the table 65 it can be determined how well the chatbot understood the sentence and typical response times are displayed to measure the overall performance of the chatbot. This also outlines how it handles unexpected input, such as misspelt words. The ~~ec~~ confidence score is returned along with the intent the test user phrase was matched to.

Table 6: “Simulated User Phrases: Test Results”.

Simulated User Utterance <i>(including misspelt phrases)</i>	Confidence Score	Matched Intent(s)	Unrecognised Intent(s)	Response Time <i>(ms)</i>
Hello bankingbot, what is my balance?	1	✓	✗	3480
Hello bankingbot, please show me my balance?	0.9999	✓	✗	3414
Display my blance	1	✗	✓	N/A
Show me my account details please?	1	✓	✗	3446
“accont dctails” (account details)	1	✓	✗	2181
Can I see my account details	0.8700	✓	✗	3480
How much money have I spent this week?	0.8299	✓	✗	3649
What hove I spunt	0.3000	✓	✗	3308
Wat have I spnt this weke	0.3300	✓	✗	2504
Can you send me my transactions for the last seven _days?	1	✓	✗	7339
tamsucyions	1	✗	✓	N/A
how much is 5000 Canadian _dollars in Japanese yen	0.8900	✓	✗	3228
Convert 600 euro to Japanese yen	0.9499	✓	✗	3182
and 700 euro	1	✓	✗	1613
and 800 pounds	1	✓	✗	3381
what about 500 US dollar	1	✓	✗	3128
Can ypu please book me an appointment please?	0.6200	✓	✗	3380
buuk me apptmnt	1	✓	✗	3345

Book me an appqentmttt for sqivings on the 23/04/2018	1	✓	✗	3322
Ok, my phone number is 089456787642	1	✓	✗	4212

Table 5: "Simulated User Phrases: Test Results"

Table 65 displays the results from simulated "typical" interactions with the chatbot through the command line tool. The simulated phrases were derived from the user questionnaires and observing the users interacting with the chatbot, the dialog for the chatbot was devised during the design stage. Every utterance is given an "Confidence Score", rated on a scale of 0.0 - 1, with 1 being a complete match and 0 meaning the utterance was not successfully matched to any intent. However, the NLU has also given words or utterances, where no intent was recognised, a complete confidence score of 1 as it was matched to the default fall-back intent to handle unexpected input from the user with ease. It is clear that the response times do vary depending on the underlying functionality required for the user to achieve their goal, for instance: requesting their transactions via email or the chatbot sending text confirmation of their appointment.

Overall, the chatbot is able to match intents with the majority of the utterances with only 10% of the utterances failing to match. This is taking spelling mistakes into account also. The total number of successful matched intents out of 20 user phrases amounted to 80%. These results indicate that the chatbot can understand most phrases including spelling mistakes reflecting the quality of the dialog. The phrase "wat have I spnt this weke" (meaning "what have I spent this week") obtained a low confidence score of 0.33 a perfect 1, however the chatbot was able to understand the intent of the utterance and identify the action on the understanding score as shown in figure 534re... even though misspelt.

```

wat have i spnt this weke
[2018-04-15 18:16:24.575] [DEBUG]
id: a962582a-973d-4d74-a116-e143c499fb2b
timestamp: 2018-04-15T18:16:20.988Z
lang: en
result:
  source: agent
  resolvedQuery: wat have i spnt this weke
  action: spent
  actionIncomplete: false
  parameters:
    date-period:
      spend:
  contexts: []
  metadata:
    intentId: 2194539a-bb3d-47f6-8b7b-e12fe1af8feb
    webhookUsed: true
    webhookForSlotFillingUsed: false
    webhookResponseTime: 2584
    intentName: bot-spent
  fulfillment:
    speech: You have spent -190.36 this week
    displayText: You have spent -190.36 this week
    messages:
      - type: 0
        speech: You have spent -190.36 this week
    score: 0.33000001311302185
  status:
    code: 200
    errorType: success
    webhookTimedOut: false
  sessionId: 56038af0-40d3-11e8-bb3e-57c75b419c46
You have spent -190.36 this week

```

Figure 54-534: "Matched Intent to misspelt phrase"

Table 7: "Average Test Results"

Average Understanding Score	Average Response Time	Dialog Duration
17.7897/20= 0.8894	60594 (ms) 60.594(sec) / 20 = 3.02 sec	3 minutes and 26 seconds

Table 6: "Average Test Results"

The average response time has been rounded to 3 seconds, which is quite a rapid response time. This suggests that users will be able to perform their necessary tasks through interacting with the chatbot relatively quickly which is also reflective on the overall dialog duration as shown in table 76, this represents the overall interaction time. The results will be further elaborated upon in the Evaluation, ~~with user testing results in contrast to the simulated results~~ phrases.

Both the Google Assistant and Home agent were tested through the AoG Simulator on the Actions on Google console (AoG). The Google Assistant integration was tested through the simulator using text input while the Google Home was tested through voice interaction on the AoG simulator. The simulator returns a JSON object holding information regarding the chatbots understanding level identifying whether the utterance was matched to an intent phrase. Similar phrases were used across devices to outline any variations in the NLU understanding.

Table 8: "Simulated Device testing".

<u>Intent</u>	<u>Google Assistant</u> <u>(Text)</u>	<u>Matched</u> <u>Intent(s)</u>	<u>Google Home</u> <u>(voice)</u>	<u>Matched</u> <u>Intent(s)</u>	<u>Web APP</u> <u>(Voice)</u>	<u>Matched</u> <u>Intent(s)</u>
1.	<u>Correctly spelt</u> <u>phrase:</u> 1.What is my balance	✓	<u>Correctly pronounced phrase:</u> 1.What is my balance	✓	<u>Correctly pronounced phrase:</u> 1.What is my balance	✓
	<u>Incorrectly spelt</u> <u>phrases:</u> 1.What is my balan 2. Wat is my bolonce	✗ ✓	<u>Incorrectly pronounced phrases:</u> 1.What is my balan 2.Wat is my bolonce <u>Detected as:</u> 1.What is my balance 2.What is my balance	✓ ✓	<u>Incorrectly pronounced phrases:</u> 1.What is my balan 2.Wat is my bolonce <u>Detected as:</u> 1.What is my balance 2.What is my bowl	✓ ✗
2.	<u>Correctly spelt</u> <u>phrase:</u> 1.Show me my transactions	✓	<u>Correctly pronounced phrase:</u> 1.Show me my transactions	✓	<u>Correctly pronounced phrase:</u> 1.Show me my transactions	✓
	<u>Incorrectly spelt</u> <u>phrases:</u> 1.Shw me mi tronsoctions	✗	<u>Incorrectly pronounced phrases:</u> 1.Show me my Tronsoctions <u>Detected as:</u> 1.Show me my Transactions	✓	<u>Incorrectly pronounced phrases:</u> 1.Show me my tronsoctions <u>Detected as:</u> 1.Show me my Transactions	✓
3.	<u>Correctly spelt</u> <u>phrase:</u> 1.Convert 200 Autralian dollars to pounds	✓	<u>Correctly pronounced phrase:</u> 1.Convert 200 Autralian dollars to pounds	✓	<u>Correctly pronounced phrase:</u> 1.Convert 200 Australian dollars to pounds	✓

	<u>Incorrectly spelt phrases:</u> <u>1. Convert 200 pound to yoyo</u>	<u>x</u>	<u>Incorrectly pronounced phrases:</u> <u>Convert 200 pound to yoyo</u> <u>Detected as:</u> <u>Convert 200 pond to euro</u>	<u>✓</u>	<u>Incorrectly pronounced phrases:</u> <u>Convert 200 pound to yoyo</u> <u>Detected as:</u> <u>Convert 200 pound to euro</u>	<u>✓</u>

Intent	Google Assistant (Text)	Matched Intent(s)	Google Home (voice)	Matched Intent(s)	Web APP (Voice)	Matched Intent(s)
1.	<u>Correctly spelt phrase:</u> <u>1. What is my balance</u>	<u>✓</u>	<u>Correctly pronounced spelt phrase:</u> <u>1. What is my balance</u>	<u>✓</u>	<u>Correctly pronounced spelt phrase:</u> <u>1. What is my balance</u>	<u>✓</u>
	<u>Incorrectly spelt phrases:</u> <u>1. What is my balan</u> <u>2. Wat is my bolonce</u>	<u>*</u> <u>✓</u>	<u>Incorrectly pronounced spelt phrases:</u> <u>1. What is my balan</u> <u>2. Wat is my bolonce</u> <u>Detected as:</u> <u>1. What is my balance</u> <u>2. What is my balance</u>	<u>✓</u> <u>✓</u> <u>✓</u> <u>✓</u>	<u>Incorrectly pronounced spelt phrases:</u> <u>1. What is my balan</u> <u>2. Wat is my bolonce</u> <u>Detected as:</u> <u>1. What is my balance</u> <u>2. What is my bowl</u>	<u>✓</u> <u>*</u> <u>✓</u> <u>*</u>
2.	<u>Correctly spelt phrase:</u> <u>1. Show me my transactions</u>	<u>✓</u>	<u>Correctly pronounced spelt phrase:</u> <u>1. Show me my transactions</u>	<u>✓</u>	<u>Correctly pronounced spelt phrase:</u> <u>1. Show me my transactions</u>	<u>✓</u>

	Incorrectly spelt phrases: 1. Shw me mi tronsoctions	*	Incorrectly pronounced spelt phrases: 1. Show me my Tronsoctions Detected as: 1. Show me my Transactions	✓	Incorrectly pronounced spelt phrases: 1. Show me my tronsoctions Detected as: 1. Show me my Transactions	✓
3.	Correctly spelt phrase: 1. Convert 200 Autralian dollars to pounds	✓	Correctly pronounced spelt phrase: 1. Convert 200 Autralian dollars to pounds	✓	Correctly pronounced spelt phrase: 1. Convert 200 Australian dollars to pounds	✓
	Incorrectly spelt phrases: 1. Convert 200 pound to yoyo	*	Incorrectly pronounced spelt phrases: Convert 200 pound to yoyo Detected as: Convert 200 pond to euro	✓	Incorrectly pronounced spelt phrases: Convert 200 pound to yoyo Detected as: Convert 200 pound to euro	✓

Table 9: "Overall accuracy comparison of Devices". Table 7: "Simulated Device testing"

	Google Assistant	Google Home	Web App (HTML5 Speech API)
% Response Accuracy			
% Matched Intent	<u>57.14%</u>	<u>100%</u>	<u>85.71%</u>

Thorough testing was performed across multiple simulated devices.

During the testing across multiple simulated devices It was determined that the Google Home device could recognise the intent of an utterance even when colloquial phrases were produced. It was also identified that the Google Home device could recognise utterances where

particular words were incomplete or ~~missmispronouncedpelt~~ as shown in table 87. It was found that input text provided to the Google Assistant on android phones, follows a more rigid syntax as it expects the intent present in the given input text to match the defined intent used within the training phrases, however this only applies when the use of input text is provided by the user, as the same speech recognition and machine learning algorithms are implemented with the Google Assistant regardless of the device or surface its embedded in.

~~The Google~~

~~Assistant follows a more rigid NLP syntax, however this only applies when the use of input text is provided by the user, as the same speech recognition and machine learning algorithms are implemented with the google assistant regardless of the device its embedded in.~~ The speech recognition software in the Google ~~a~~Assistant transforms the users speech, even if incorrect, to the appropriate input text before ~~its~~it's sent to the NLU. Consequently the Google Home device This is why

~~the Google Home device~~ was able to recognise and detect incomplete or ~~missmispronouncedpelt~~ words, as interaction only occurred through voice commands. Based on these results it is clear that voice interaction is the optimal method of engaging with the chatbot due to it being able to comprehend pronunciation errors and interpret them accordingly. It was also found that the Google Assistant could only match an intent of a to a misspelt utterance when the trained intent or recognised entity was typedspelt correctly during text interaction.

It is clear from the results shown in table 9 that the Google Home device was the prime mode of interaction with the chatbot, as it was able to match the intent of each supplied utterance, achieving a perfect score of 100%. Establishing that spoken natural language conversational interfaces perform best, compared to that of text-based interactions as the Google Assistant achieved a result of 57.14%. There is a stark contrast between the two conversational interfaces, implying spoken dialog is a much better method of interacting with natural language conversational interfaces. The speech recognition on the Google Assistant has a higher recognition rate in comparison to the HTML5 Speech API as it had an average speech recognition of 85.71%, suggesting the Google platforms will perform better in real interactions with users.

Table 10: "Conversation exit rate".

Intent	(n)Turns before exit
--------	----------------------

1.	3
2.	4
3.	4

Table 8: “Conversation exit rate”

Table ~~10~~ 8-outlines the number of turns the chatbot took to try and understand the intent of an utterance corresponding to the misspelt phrase _defined in table ~~87~~ 7-before exiting the conversation. On each turn the chatbot responded with fallback intents such as: "Sorry, I didn't quite get that, could you repeat that please?" or "Sorry, I'm having difficulty understanding what you said, could you say that again?". This feature handles unexpected input or when the chatbot cannot understand a given utterance. This also outlines the use of different fallback intents used even if the user utterance was the same on each turn. The chatbot exited the conversation due to repeatedly prompting the user for an appropriate utterance and lack thereof, its inability to understand the intent of the given utterance to successfully match it to a trained intent to carry out the appropriate action and fulfil the user's goal.

5.2 Webhook Testing

To test the custom fulfilment provided through the Webhook, a tool called Postman was utilised to send HTTPS POST requests to the Webhook. This was used to test the response sent back from the Webhook to a given user utterance. The body of the request is obtained from the Dialogflow developer console which is sent with the POST request to the Webhook. The body holds the JSON object which represents the users phrase along with other parameters returned by the NLU as shown- in figure 545below.

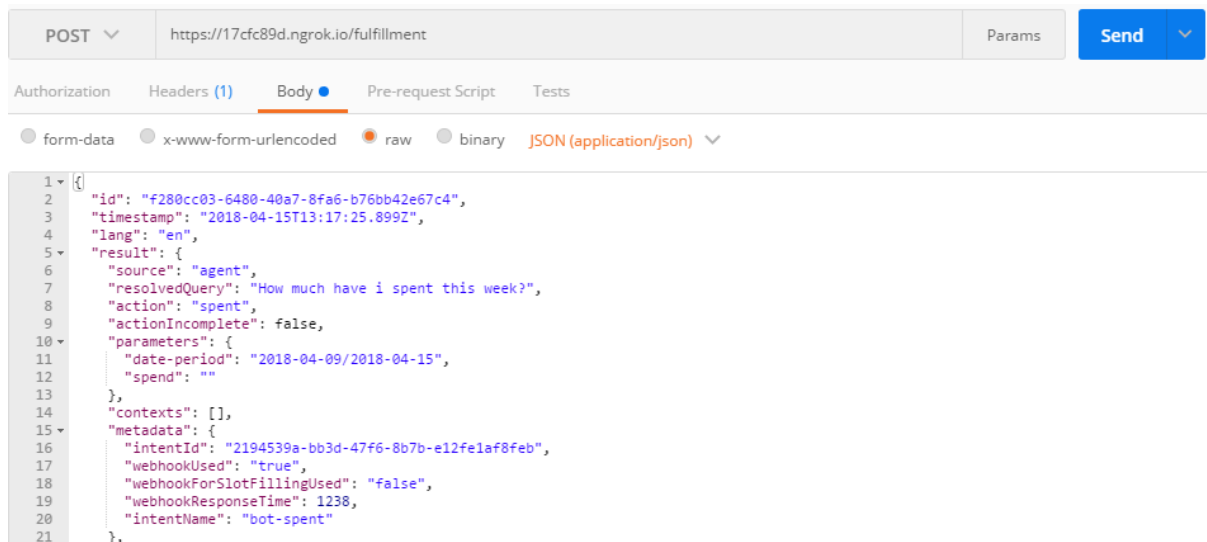


Figure 545: “Webhook POST request”.

Postman was used to test the Webhooks functionality in a separate environment, in which it was developed, meaning errors could be rectified before integrating [the agent](#) with [other](#) [the various various](#) devices such as [Google Assistant](#) [and](#) [Google Home](#) ~~or the web-based chatbot~~. This identified network and access errors as status codes are returned with each response. An example of the JSON response object returned by the Webhook is shown ~~below in figure 556XXXXXXXXXX~~:

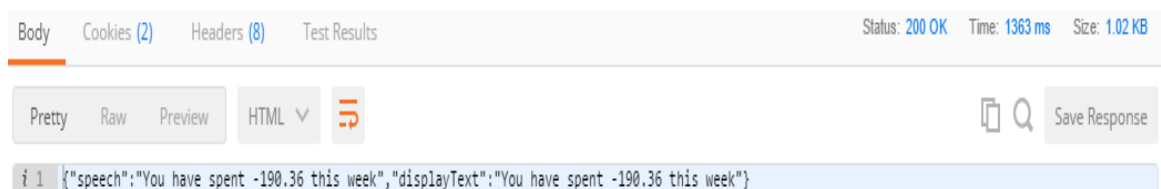


Figure 556: “Webhook response”.

The integration of other API’s implemented within the project were also tested, to determine whether it could successfully call the TrueLayer API and identify any required parameters or authentication headers that were required to be sent with each subsequent request.

5.31 User Testing:

It was decided to carry out user testing across all three communication channels of the

chatbot; [Google Assistant](#), [Home](#) and the web-based chatbot. This will give [an](#) insight into the true

quality and overall usefulness of the chatbot which is measured by the user's interaction experience. In order to conduct user testing, a set of questions were constructed to evaluate the chatbot. These questions were distributed as an online questionnaire across a group of users which can be found in the appendix section of the report.

6.5. Evaluation

The purpose of evaluating the software is to identify the quality of the chatbot by outlining the performance attributes and analysing the results, future work is purposed along with a reflection of the current work completed.

The results from the user questionnaires gave great insight into the overall response and precision rate. The questionnaires were distributed across a user group consisting of 15 individuals with varying technical knowledge. These were completed by parents and prospective students at the Ulster University Open Day as part of the school of Engineering and Computing, this user group is truly reflective of the target audience that would benefit from an application like this. The results from the user questionnaires will be compared to the simulated user interactions identifying subjective and objective metrics.

During the questionnaire the users were observed to capture free standing information regarding the interaction experience which also qualify as subjective measurements. This allowed the identification of other quality metrics which were not initially considered and led to a deeper understanding of the chatbots performance. ~~(How often the chatbot repeated itself,)~~

~~The metrics shown in table 9 were~~ They completed a user questionnaire targeted to capture the quality of the chatbot.

The
and
metrics
table 11
obtained

<i>Subjective measurement</i>	<i>(n)Users</i>	<i>Objective metrics</i>	<i>(n)Users</i>
Naturalness	44%	Speech Recognition	80%
Likeability	93.33%	Response accuracy	86.67%
Ease of use	86.66%		

subjective
objective
outlined in
were
using a

range of question types including Likert scale and numerical scales to accurately assess the chatbots performance.

Each metric outlined in table 11 was measured in relation to a specific question to gather qualitative results. These questionnaires can be found in appendices.

Table 11: “Subjective & Objective Measurements obtained from User questionnaire”.gathered from

<i>Subjective metrics</i>	<i>(n)Users</i>	<i>Objective metrics</i>	<i>(n)Users</i>
Naturalness	44.00%	Speech Recognition Speech Recognition	86.67% 80.00%
Likeability	93.33%	Response accuracy	86.67%
Ease of use	86.66%	Speed of Interaction Response rate	73.33%
Speech Recognition	80.00%		

Table 9:

“Subjective & Objective Measurements obtained from User questionnaire”

As outlined in [section 1.2](#) the majority of banks struggle to get their customers to use the technology

in place due to low user satisfaction when interacting with their services. From the results shown in [table 11](#) it can be established that integrating a chatbot into their banking services would greatly increase the rate of user satisfaction. This is evident as 73.33% of all users stated the chatbot was “extremely” quick compared~~found the chatbot to be a lot quicker to~~han the current technology offered by their

bank, for instance, online banking applications. From completing the user testing phase, it was found that 86.66% of all respondents ~~found~~stated the chatbot was

either “extremely very easy” or “moderately easy” easy to talk to and it achieved an overall likability ~~of 93.33~~of 93.33% as outlined in table 11. Reaffirming

that chatbots can successfully engage users and increase satisfaction. The objective metrics obtained from the questionnaire results identify the chatbots performance and quality in a real interaction. Overall the chatbot received a very high speech recognition rate as 80.00% of users stated they were “very strongly” well-understood by the chatbot. This measurement reflects the accuracy

of the emulated actions during testing as shown in table 7 of testing. The chatbot obtained an overall understanding score of 0.8894 (89.00%) in the range between 0.0-1 during the simulated tests as depicted in table 6 of section 5. This result is on the higher tier of the understanding scale.

Evidence from the results gathered during user testing, through measuring the response accuracy to real user interaction, that the chatbot has a high competency level as both the results from the simulated and user tests have a very low variance of 2.33% in regards to the understanding score. These measurements reflect the accuracy and thoroughness of the emulated actions performed during testing as illustrated in section 5.

Do you prefer to interact with the chatbot through voice or text-based commands?

Answered: 15 Skipped: 0

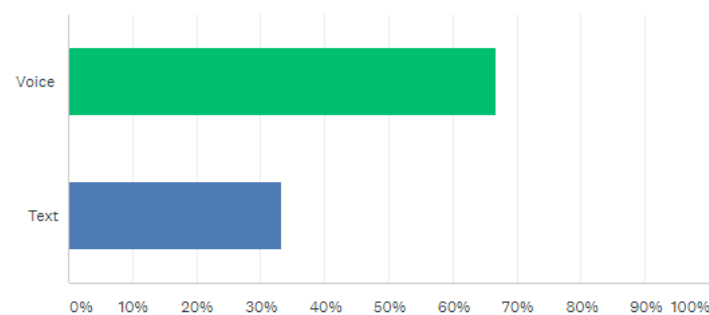
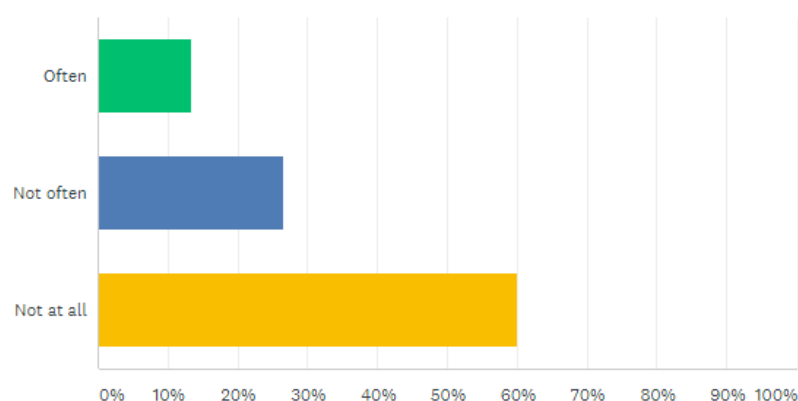


Figure 56Z: "Users preference".

It is clear from the questionnaire results that the majority of users preferred to interact with the chatbot through voice commands, with 66.67% of users preference being voice

Where you ever unsure how to carry on the conversation with the chatbot

Answered: 15 Skipped: 0



interaction, as shown in figure [567](#). This supports that more users are wishing to be able to freely

interact with applications and not be limited to traditional modes of interaction. This feature also offers users the ability to interact with chatbot without it requiring much attention on focusing on the task at hand, as users ask the chatbot a question and it will recognise the users voice negating the need for the user to type.

Where you ever unsure how to carry on the conversation with the chatbot

Answered: 15 Skipped: 0

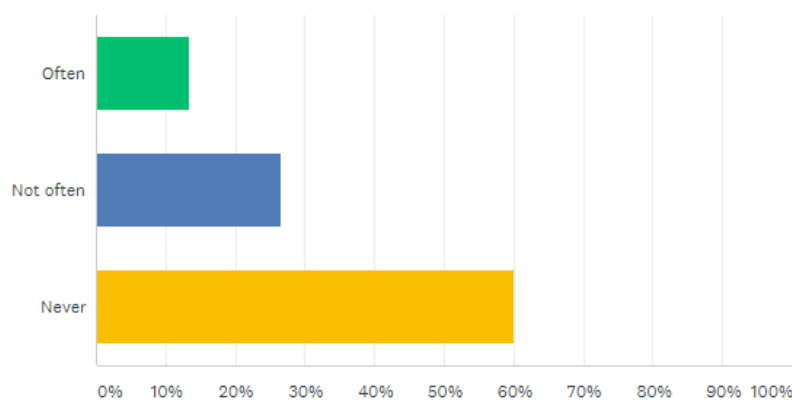


Figure 578: "Conversation flow".

The focus of any chatbot is to allow users to complete tasks with ease using natural language. It is important that the chatbot clearly understood the user and the user felt confident in interacting with the chatbot. Its outlined in figure [578](#) that this was achieved as only 13.33% of users stated they were "often" left in a state where they were unsure how to carry on the conversation. These results reflect low frustration rates among users as- 60% of user never experienced this and 26.67% stating it ~~did not~~ occurred "Not often." ~~often~~. This clearly suggests the chatbot is able to sufficiently guide users through the conversation with ease providing appropriate responses to the majority of user phrases.

Could you easily navigate through or end the conversation? For instance, could you refer back to previous conversational context and still be understood by the chatbot?

Answered: 15 Skipped: 0

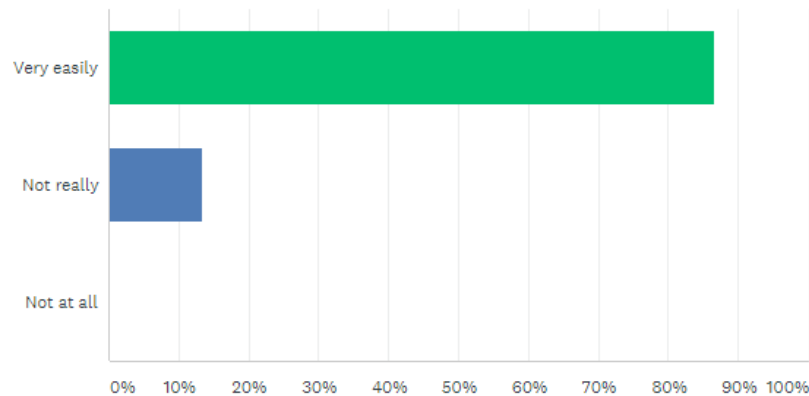


Figure 589: “Chatbot Conversational context”.

Enabling the chatbot to be context aware provides a great advantage for users as they can simply refer back to a previous conversations and still be understood by the chatbot. This is particularly useful in long conversational flows, reducing the amount in which the user has to repeat themselves thus shortening the time taken to achieve a task.

65.1 Reflection

The results gathered throughout the testing phase support the research conducted within section 1.3, which outlines the prevalence of chatbots within industries, particularly within the banking and financial sector. The findings prove that the chatbot is a very suitable method for adopting technology as a means to distribute a service. The developed chatbot allows users to interact with their bank through natural language interaction, granting users more convenient and efficient access to their banking information. [The project gradually evolved and progressed throughout its entirety and the main objectives and requirements have been met as outlined in section 2.5.1 and 2.5.2. This chatbot allows users to connect to various banks they may have an existing account with all from within the one communication channel](#) through natural language.

65.1.2. Future Advancements

[One of the integration s-streams that were initially thought of during development was to integrate the chatbot with the Facebook Messenger service, however during development it](#)

was brought to light through of recent media events such as the Cambridge Analytica discreditscandal, that which resulted in Facebook haltedpausing their process of allowing other apps to integrate with its messenger service-app review, so this integration was not developed as Facebook had stopped allowing developers to create new apps or chatbots through the service. Although Facebook have now just recently reopened their app review processs, now allowing—developersallowing developers to integrate with the Messenger service again, enabling this integration feature to be developed for future prototypes (Facebook, 2018).

Integrating Google authentication would have been an advantageous feature to implement, allowing users to link their Google account across the multiple devices the agent is distributed on, for instance, Google Assistant enhancing the cross-platform experience for users.

User can find out financial information about their account, such as how much they spent within a week. From this feature it was thought that during the later stages of development users should be able to directly query the chatbot to determine where and what they spend their money on. Although users can view this information in a graphical format onin the web app it would also be useful if they could find this information out through interaction with the chatbot.

The chatbot could also be developed to be multilingual. Supporting multiple languages would help aid banking exclusion in developing countries and enhance the overall accessibility of technology within the banking industry whilst targeting a larger user group.

From observing the users interact with the chatbot during user testing, there were numerous suggestions to integrate the chatbot across other popular platforms such as Amazon Echo or Dot, this would increase the availability of the chatbot and the integration of the DialogFlow API allow the chatbot to be easily exported Amazon Alexa.

65.2.- Conclusion

The rise and popularity of chatbots is clearly outlined in figure 1 of section 1.3. With this in mind the data gathered from testing the chatbot justifies the recent growth and demand for companies wanting to integrate a chatbot. It was determined that chatbots perform at a very

high standard and provide reliable and rapid responses to users compared to that of traditional methods. The average time spent interacting with the chatbot is very low as it provides an efficient way for users to manage their banking. The low interaction time reflects the high understanding and speech recognition rates, offered through the adoption of conversational user interfaces thus allowing users to freely interact with the chatbot to meet the demands of modern life. The chatbot has proven to fulfil the demand of users wanting instant access and availability information and services.

~~Integrating sentiment analysis to capture the user's emotions. The user's transactions could be analysed using NLP techniques to identify spending habits.~~

~~Due to the increased development of the IoT, it would be good to branch out into this area as chatbots produce a~~

~~mass amount of data which is needed for concepts such as IoT and Big Data to be successful which also go in hand with further advances in AI and Machine learning. The~~

~~IoT could be implemented through the Google Home device.~~

6. References

~~(United kingdom online banking (2002). MarketLine, a Progressive Digital Media business, 2017)~~

~~(Accessed 10/10/2017)MarketLine, a Progressive Digital Media Business. (2002). 'United Kingdom online banking'. Available at: <http://www.marketlineinfo.com> (Accessed 10/10/2017).~~

Ling, G., Fern, Y., Boon, L. and Huat, T. (2016). Understanding Customer Satisfaction of Internet Banking: A Case Study In Malacca. *'Procedia Economics and Finance'*, 37, pp.80-85.n (Accessed 11/10/2017)

~~Barty, J. and Recketts, T. (2014). 'Promoting Competetion in the UK banking Industry', *Promoting Competetion in the UK banking Industry. BBA Competition Report.*, -pp. 4. Available at: https://www.bba.org.uk/wpcontent/uploads/2014/06/BBA_Competition_Report_23.06_WEB_2.0.pdf (Accessed 10/10/2017).~~

Aburub, F., Odeh, M., & Beeson, I. (2007). *'Modelling non-functional requirements of business*

-Processes'. *Information and Software Technology*, 49(11–12), 1162-1171. Available at:

<http://dx.doi.org/10.1016/j.infsof.2006.12.002> (Accessed 10/10/2017).

~~Cardline. (2010). 'Half of UK internet users now bank online', 10(3), pp. 32-32. Availablle at:~~

~~<http://web.b.ebscohost.com/bsi/detail/detail?vid=1&sid=23ade416-2f14-4608-a8a1-3de95541175b%40sessionmgr102&bdata=JnNpdGU9YnNpLWxpdmU%3d#AN=47659678&db=bth> (Accessed 10/10/2017).~~

~~Half of UK internet users now bank online.(2010). *Cardline*, 10(3), 32-32.~~(Accessed 10/10/2017)

~~HDFC bank, niki.ai tie up for chatbot banking.(2017). *FRPT Finance Snapshot*, , 24-25.~~(Accessed 11/10/2017)

~~FRPT Research- *Finance Snapshot*. (2017). 'HDFC Bank, Niki.ai tie up chatbot for banking', pp~~

~~24-25. Available at: <http://web.b.ebscohost.com/bsi/pdfviewer/pdfviewer?vid=8&sid=c34ab15e-2256-41a5-8757-f8ef87bd08e8%40sessionmgr104>. (Accessed 11/10/2017)~~

Ling, G. M., Fern, Y. S., Boon, L. K., & Huat, T. S. (2016). *Understanding customer satisfaction of internet banking: A case study in malacca*. Available at -doi:[https://doi.org/10.1016/S2212-5671\(16\)30096-X](https://doi.org/10.1016/S2212-5671(16)30096-X)

~~Hugh J.~~ Watson, H. J. (2017). 'Preparing for the Cognitive Generation of Decision Support', *MIS Quarterly*

Executive, 16(3), pp. [Online]. Available

at: <http://www.misqe.org/ojs2/index.php/misqe/article/viewFile/705/465> (Accessed: 10/10/2017).

~~Daniëlle~~ Duijst, D. (2017). 'Can we Improve the User Experience of Chatbots with Personalisation?', *University Of Amsterdam*, ~~-0-~~ pp. 3 [Online]. Available

at: [10.13140/RG.2.2.36112.92165](https://doi.org/10.13140/RG.2.2.36112.92165) (Accessed: 11/10/2017).

~~United kingdom online banking(2002). MarketLine, a Progressive Digital Media business.~~

Watson, A. (2010). *How to succeed with NLP: Go from good to great at work using the power of neuro-linguistic programming*. Oxford: Capstone. (Accessed 11/10/2017)

~~KingING~~, W. B. (2017). Year of the chatbot: Credit unions gearing up for artificial intelligence. *Credit Union Journal*, 21(4), 18-18. (Accessed 11/10/2017).

~~Dole, A., Sansare, H., Harekar, R. and Athalye, S. (2015). [online] www.ijettcs.org. Available at:~~
[http://www.ijettca.org/Volume4Issue5\(2\)/IJETTCS-2015-10-09-16.pdf](http://www.ijettca.org/Volume4Issue5(2)/IJETTCS-2015-10-09-16.pdf) [Accessed 11 Oct. 2017].

~~Intelligent chatbot for banking system. (2017). [ebook] Available~~
~~:http://www.ijettcs.org/Volume4Issue5(2)/IJETTCS-2015-10-09-16.pdf [Accessed 11 Oct. 2017].~~

~~D.~~ Latimore, ~~D-I.~~ Watson, I. & ~~C.~~ Maver. C. (2000). The customer speaks: '3300 Internet users tell us what

they want from retail financial services'

~~Yusuf~~ Dauda, S. Y. & Lee, J (2015). 'A conjoint analysis of consumers' preference on future online banking services, Nigeria: Elsevier Ltd.

~~Genealo~~ Baptista, G. and ~~Tiago~~ Oliveira, T. (2015). 'Understanding mobile banking: The unified theory of

acceptance and use of technology combined with cultural moderators', : Elsevier Ltd.

~~Ajimon~~ George, A and ~~G.S. Gireesh~~ Kumar, G. -(2013) 'Antecedents of Customer Satisfaction In Internet Banking: Technology Acceptance Model (TAM) Redefined', *Global Business Review*, 14(4), pp. [Online]. Available at: <http://journals.sagepub.com/doi/10.1177/0972150913501602>

Available

at: https://smartech.gatech.edu/bitstream/handle/1853/58516/evaluation_of_modern_tools_for_an_omscs_advisor_chatbot%281%29.pdf?sequence=1&isAllowed (Accessed: 21/10/2017).

~~Erie~~ Leenders, E. (2018). 'Banking's digital revolution continues'. Available at: <https://www.ukfinance.org.uk/bankings-digital-revolution-continues/> (Accessed: 12/10/2017).

~~Susan~~ Etlinger, S. (2017). '~~The Conversational Business~~THE CONVERSATIONAL BUSINESS: How Chatbots will reshape Digital Experiences'. [Online]. Available at: https://andyblackassociates.co.uk/wp-content/uploads/2015/06/ConversationalBusiness_FINAL-1.pdf (Accessed: 16/10/2017).

~~YY~~ Onufreiv, YY. (2017). '~~The Rise In Chatbots Trends~~THE RISE IN CHATBOTS TRENDS. THE RISE IN CHATBOTS TRENDS'. - (-), 82-83. (Accessed 16/10/2017)

~~Erie~~ Gregori, E. (2017). 'Evaluation of Modern Tools for an OMSCS Advisor Chatbot'. [Online]. Available at: https://smartech.gatech.edu/bitstream/handle/1853/58516/evaluation_of_modern_tools_for_an_omscs_advisor_chatbot%281%29.pdf?sequence=1&isAllowed (Accessed: 21/10/2017).

~~Erik~~ Cambria, E., ~~Bebe~~ White, B. (2014) 'Jumping NLP Curves: A Review of Natural Language Processing Research', *IEEE Computational Intelligence Magazine*COMPUTATIONAL INTELLIGENCE MAGAZINE. [Online]. Available at: <http://sentit.net/jumping-nlp-curves.pdf> (Accessed: 21/10/2017).

~~Daniel~~ Jurafsky, D. and ~~James H.~~ Martin, J. H. (2017). An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition. Third Edition [Online]. Available at: <https://web.stanford.edu/~jurafsky/slp3/ed3book.pdf> (Accessed: 21/10/2017).

https://www.bba.org.uk/wp-content/uploads/2014/06/BBA_Competition_Report_23.06_WEB_2.0.pdf

~~Rehan~~-Kar, ~~R.~~ and ~~Rishin~~-Haldar, ~~R.~~-(2016).-'Applying Chatbots to the Internet of Things: Opportunities and Architectural Elements', ~~3, 0, pp.~~ [Online]. Available at: <https://arxiv.org/abs/1611.03799> (Accessed: 06/11/2017).

~~Nat~~-Landry, ~~N.~~ (2011).-'*Iterative and Agile Implementation Methodologies in Business Intelligence and Software Development*', : Lulu Enterprises.

~~A B M~~-Moniruzzaman, ~~A. B. M.~~ and ~~Dr Syed Akhter~~ Hossain, ~~S.A.~~ (2013).-'Comparative Study on Agile software development methodologies', *Global Journal of Computer Science and Technology*, 13(7), pp. [Online]. Available at: <https://arxiv.org/ftp/arxiv/papers/1307/1307.3356.pdf> (Accessed: 08/11/2017)

~~Asif Irshad~~-Khan, ~~A.I.~~, ~~Rizwan Jameel~~-Qurashi, ~~R. J~~ and ~~Usman Ali~~-Khan, ~~U. A.~~ (2011). 'A Comprehensive Study of Commonly Practiced Heavy and Light Weight Software Methodologies', *IJCSI International Journal of Computer Science Issues*, 8(4), pp. [Online]. Available at: <https://arxiv.org/ftp/arxiv/papers/1111/1111.3001.pdf> (Accessed: 08/11/2017)

~~Jonathan~~-Lazar, ~~J.~~ (2001) *User-centered Web Development*, United States of America: Jones and Bartlett Publishers .

~~Graham~~-Oakes, ~~G.~~-(2008).-'*Project Reviews, Assurance and Governance*', United Kingdom: Gower Publishing Limited.

Instructional Software Research and Development (2007) *Structured System Anal And Design Isrd*, New Delhi: Tata McGraw-Hill Publishing Company Limited.

~~Pauline M.~~-McGuirk, ~~P. M. Phillip~~ and ~~O'Neill, P.~~ (2016). 'Using questionnaires in qualitative human geography', *University of Wollongong, Faculty of Social Sciences*, ~~0~~ pp. 11 [Online]. Available at: <http://ro.uow.edu.au/cgi/viewcontent.cgi?article=3519&context=sspapers> (Accessed: 12/11/2017).


~~Melanie~~-Exner-Stöhr-, ~~M.~~-, ~~Alexander~~ Kopp-, ~~A.~~, ~~Leonhard~~ Kühne-Hellmessen-, ~~L.~~, ~~Lukas~~ Oldach-, ~~L.~~, ~~Daniela~~ Roth, ~~D. A~~ and ~~Alfred~~ Zimmermann, ~~A.~~ (2017). 'The potential of Artificial Intelligence in academic research at a Digital University', *Digital Enterprise Computing*, ~~0~~ pp. 3 [Online]. Available at: <https://dl.gi.de/bitstream/handle/20.500.12116/120/paper05.pdf?sequence=1&isAllowed=y> (Accessed: 16/11/2017).


~~Kazuo~~ Yano, K. (2017). 'How artificial intelligence will change HR', 40(3), 43-44. (Accessed 16/11/2017)

~~Stuart J.~~ Russell, S. J. -and ~~Peter~~ Norvig, P.-(1995). Artificial intelligence A modern approach.

~~Noor~~ Atikah, N., ~~Amira~~ Fauzi, A., ~~Rohayanti~~ Hassan, R., ~~Zuraini Ali~~ Shah, Z. A. and ~~Zalmiyah~~ Zakaria, Z. (2016) Extraction of Non-Functional Requirements in Reviewing Requirements Specification Document, 2, pp. 1 (Accessed 19/11/2017).

~~Frank~~ Tsui, F., ~~Orlando~~ Karam, O. -and ~~Barbara~~ Berna, B. (2016). 'Essentials of Software Engineering'. 4th Edition [Online]. Available at: <https://books.google.co.uk/books?hl=en&lr=&id=fVHDQAAQBAJ&oi=fnd&pg=PP1&dq=functional+and+nonfunctional+requirements+definition+software+development&ots=P3-Cil3TVN&sig=kWwrD0XWLzWyxft49W4XVfWxXWg#v=onepage&q&f=false> (Accessed: 19/11/2017).

~~Milan van~~ Eeuwen, V. M. (2017). 'Mobile conversational commerce: messenger chatbots as the next interface between businesses and consumers', *University of Twente*,  pp. 2 [Online]. Available at: http://essay.utwente.nl/71706/1/van%20Eeuwen_MA_BMS.pdf (Accessed: 20/11/2017)

~~Teller Vision~~ (2017). 'HGS Identifies Top 10 Customer Experience Trends', *Issues and Trends*,  pp. 4 [Online]. Available at: <http://eds.a.ebscohost.com/eds/pdfviewer/pdfviewer?vid=14&sid=1fd6a005-b85b-430e-b3c8-30a3f70d947e%40sessionmgr4007> (Accessed: 21/11/2017).

~~Bayan Abu~~ Shamar, B. A. and ~~Erie~~ Atwell, E. (2007). 'Chatbots: Are they Really Useful?', *LDV-Forum*, pp. 29, 30 [Online]. Available: <http://eprints.whiterose.ac.uk/81930/1/AComparisonBetweenAliceElizabeth.pdf> (Accessed: 22/11/2017).

~~OLGA~~ ANNENKO, O.-(2016). *When to Use Webhooks vs. APIs To Sync Data Between Applications*, Available at: <https://www.elastic.io/sync-data-between-applications-apis-webhooks/> (Accessed: 01/03/2018).

- DialogFlow.-(2015). ‘*A Speech Interface in 3 Steps*’, Available at: <https://blog.dialogflow.com/post/speech-interface-3-steps/> (Accessed: 01/03/2018).
- DZone.—(2018). ‘*A Writer’s Guide to Conversational Interfaces*’, Available at: <https://dzone.com/articles/a-writers-guide-to-conversational-interfaces?fromrel=true>(Accessed: 20/03/2018).
- Taylor Otwell, T.-(2018). *Laravel*, Available at: <https://laravel.com/docs/5.5/blade> (Accessed: 05/02/2018).
- cs.hmc.edu. (2018). ‘*Domain Specific Design Patterns: Designing For Conversational User Interfaces*’. [online] Trento: cs.hmc.edu. Available at: <https://arxiv.org/ftp/arxiv/papers/1802/1802.09055.pdf> [Accessed 14 Mar. 2018].
- Google. (2018). ‘*Design Principles and Methodology*’, Available at: <https://developers.google.com/actions/design/principles> (Accessed: 27/03/2018).
- http://www.spokenlanguagetechnology.com/docs/McTear_ESSV_2018.pdf. (Accessed d04/03/2018)
- Asbjørn Følstad, A. and, Petter Brandtzæg, P. -(August 2017). ‘*-Chatbots and the new world of HCI*’, Available at: <http://interactions.acm.org/archive/view/july-august-2017/chatbots-and-the-new-world-of-hci> (Accessed: 14/02/2018).
- Open Banking Ltd 2018 (2018) *Open Banking*, Available at: <https://www.openbanking.org.uk/about-us/> (Accessed: 18/03/ 2018)
- Facebook.(2018). ‘*Messenger Platform Changes in Development*’. Available at: <https://messenger.fb.com/newsroom/messenger-platform-changes-in-development/> (Accessed:02/05/2018).

7.Appendices

~~The appendices are an opportunity to provide secondary material in support of the description in the body of the report. In principle, the reader need not look at the appendices and no specific marks are awarded for this section.~~

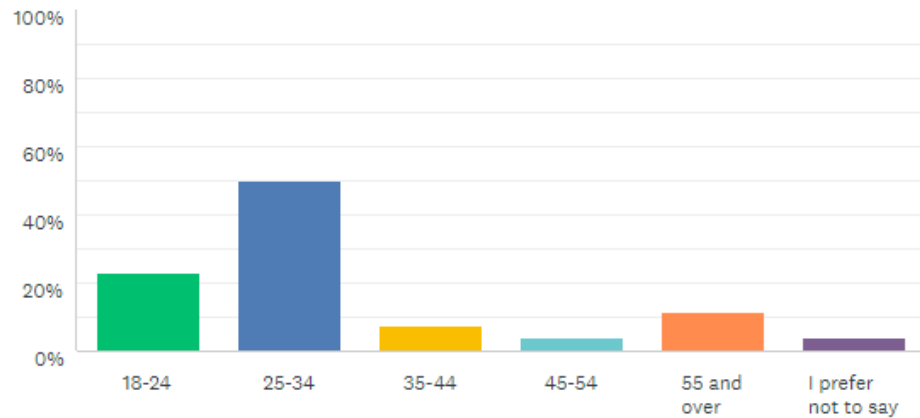
~~Sample content:~~

7.1 Appendix A Questionnaire Results User Requirements

Q1

What age group do fall under?

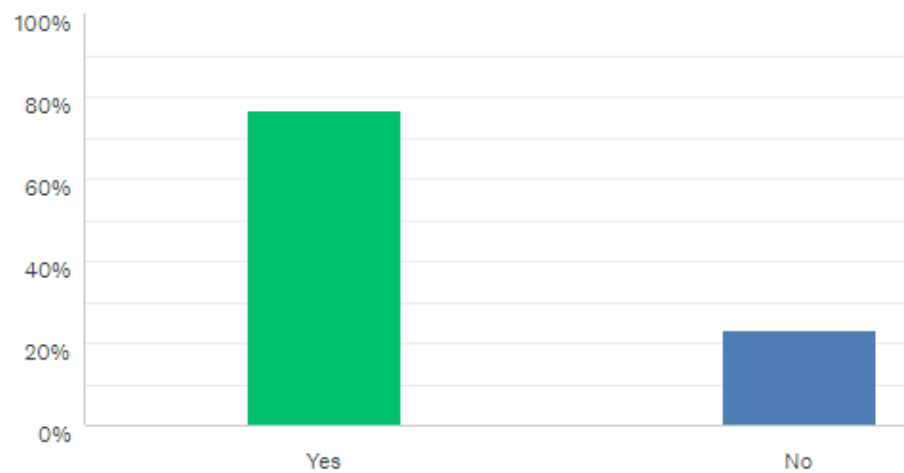
Answered: 26 Skipped: 0



Q2

Do you use online banking?

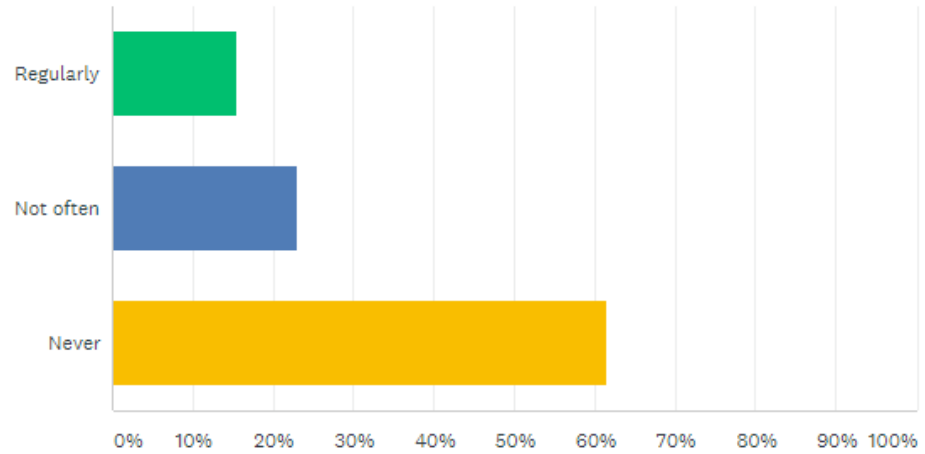
Answered: 26 Skipped: 0



Q3

How often do you visit your local branch?

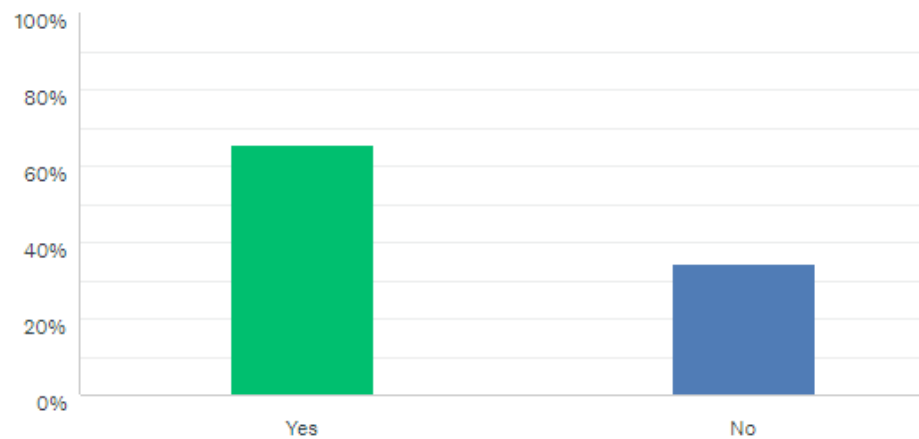
Answered: 26 Skipped: 0



Q4

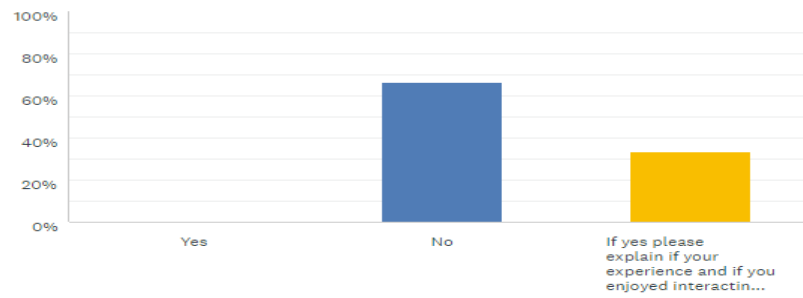
Do you feel online banking is secure?

Answered: 26 Skipped: 0



Have you used a chatbot before?

Answered: 24 Skipped: 0



ANSWER CHOICES	RESPONSES
▼ Yes	0.00%
▼ No	66.67%
▼ If yes please explain if your experience and if you enjoyed interacting with a chatbot	Responses 33.33%

RESPONSES (8)

TEXT ANALYSIS

MY CATEGORIES

Categorize as...

Filter by Category

Search responses



Showing 8 responses

When faced with a complex query it replied with simple straight forward responses asking me to repeat myself, i found this frustrating as it was a long process. I was eventually given a link to chat to an adviser to help me. It worked but could be improved.

11/19/2017 2:22 PM

[View respondent's answers](#)

I used a chat bot before for a customer service issue i had, it resolved my issues quickly. I was under the impression that i was talking with someone as it just felt like a normal conversation.

11/19/2017 2:11 PM

[View respondent's answers](#)

Very informative and helpful

11/17/2017 11:32 AM

[View respondent's answers](#)

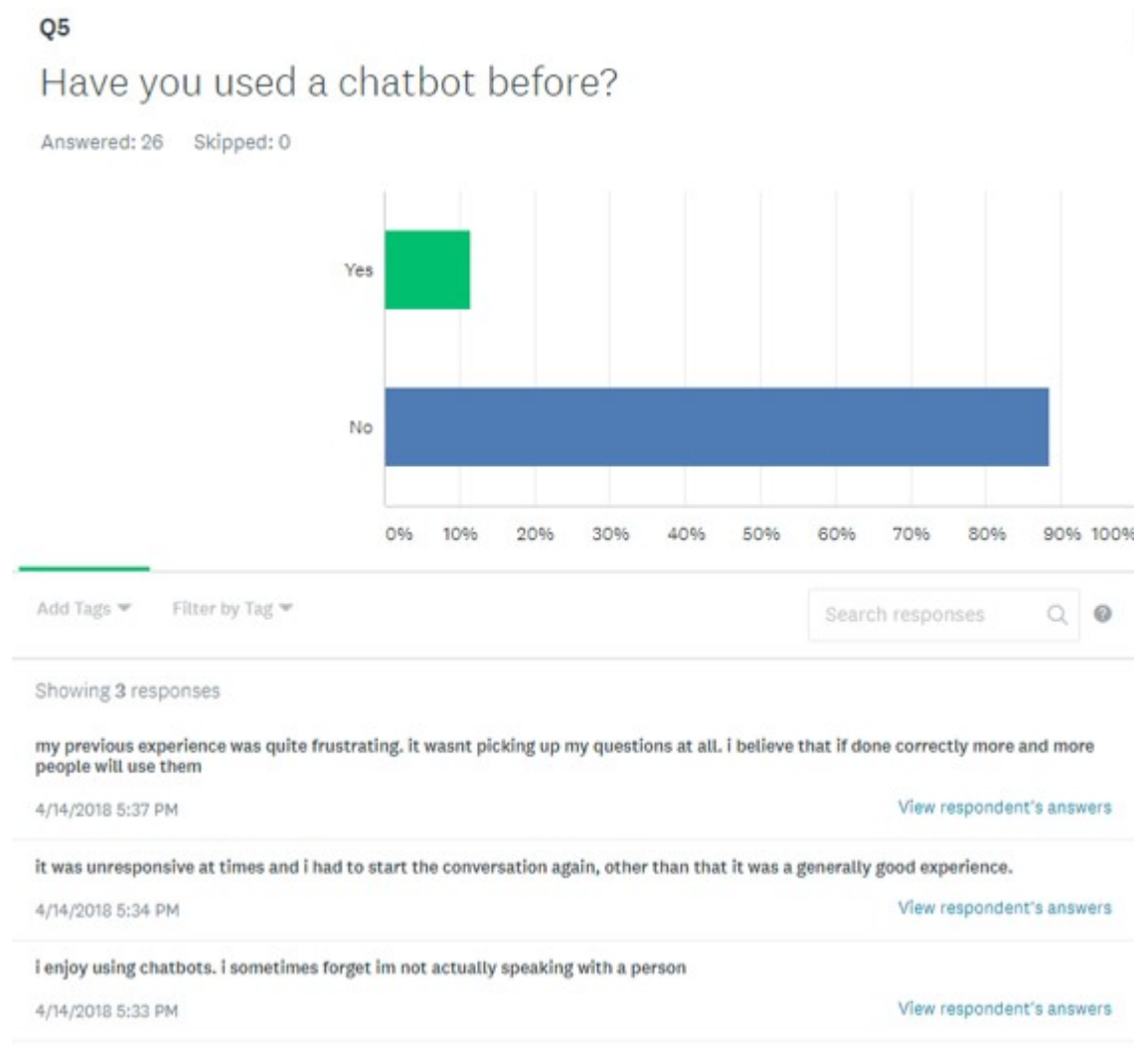
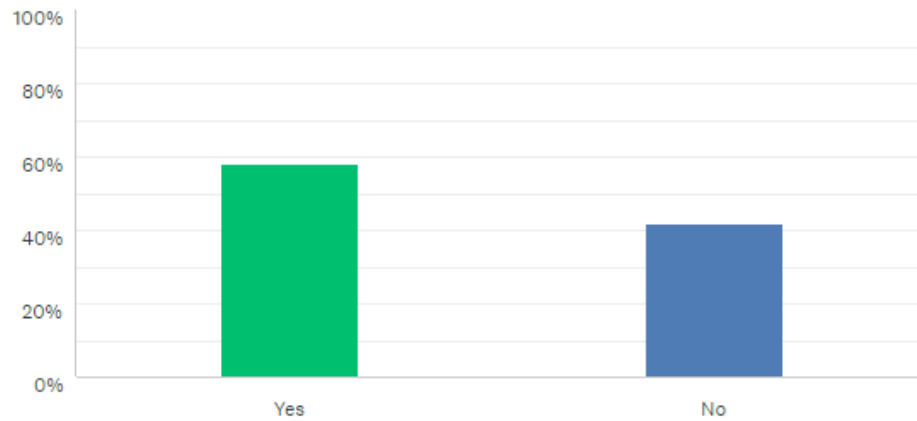


Figure 7.1.1 responses to user that have used a chat bot before

Do you feel online banking is secure?

Answered: 24 Skipped: 0



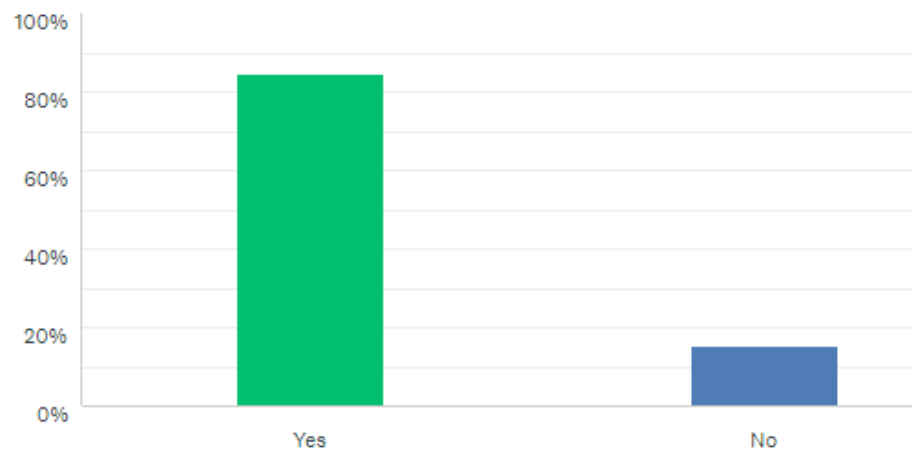
ANSWER CHOICES	RESPONSES
▼ Yes	58.33%
▼ No	41.67%
Total Respondents: 24	

[Comments \(10\)](#)

Q6

Would you use a chatbot to manage your banking online?

Answered: 26 Skipped: 0



Q7

[Customize](#)

What would be your preferred mode of interaction with the chatbot?

Answered: 26 Skipped: 0

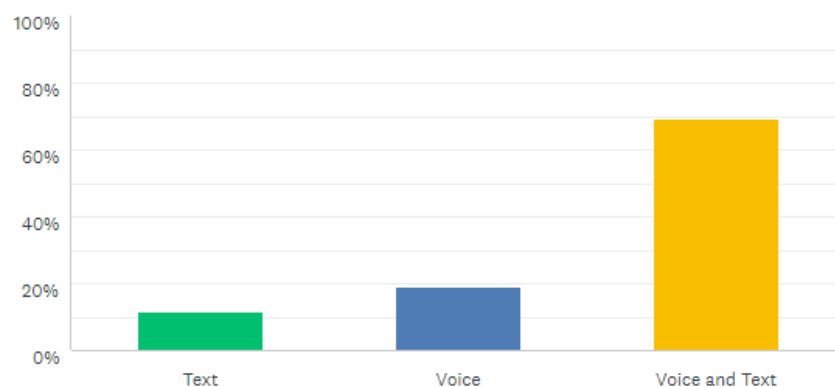


Figure 7.1.2 results displayed for how secure users feel online banking is on Survey Monkey

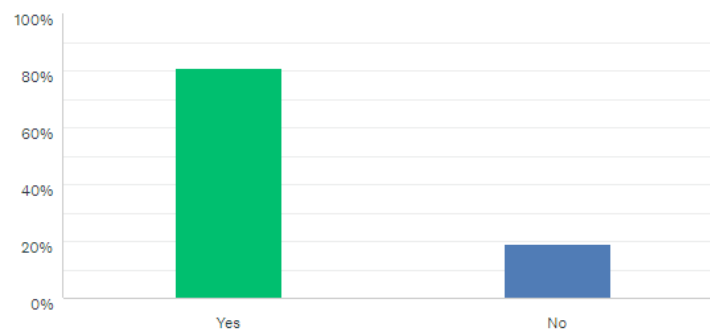
Q8

Customize

Export

Do you think the integration of a chatbot would improve the efficiency of performing banking operations compared to that of current technology in use?

Answered: 26 Skipped: 0



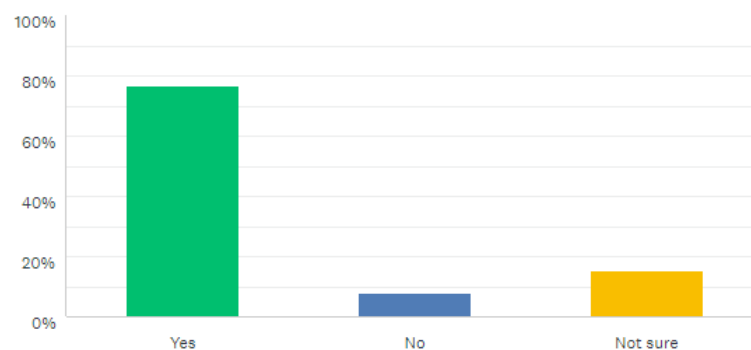
Q9

Customize

Export

If your bank had a Chatbot integrated into its services would this encourage you use online banking more?

Answered: 26 Skipped: 0



Q10

Are there any features you would find useful in a chatbot for banking?

Answered: 26 Skipped: 0

RESPONSES (26)

TEXT ANALYSIS

TAGS

Add Tags

Filter by Tag

Search responses

Showing 26 responses

If i could be guaranteed that my personal information is secure with a service like this then i would consider it.

