

Augmented Conversation System



MSC Computing

Jianshu Wu

C2007603

Supervised: Ian M Cooper

Moderated: David J Humphreys

Cardiff University

December, 2021

Abstract

The aim of the Augmented Conversation(AC) System is to be able to analyse the conversations that participants are having and to present and update information relevant to the topic in a timely manner. Human communication joined by AC will enable us to access information from the web in real time so that we can better express, share and explore new ideas with others in conversation. This will make chatting more interesting. The system provides a 'live update' - what you see is what you are discussing, and we can do a better job of communicating through the information-rich web. Whenever you bring up a new topic, AC, the 'third party' to the conversation, immediately gives the user the ability to search for information. The Augmented Conversation is a shift in the traditional way of chatting, changing the monotony of communication between users and allowing two strangers to have a conversation that is not 'silent'.

Table of Contents

| | |
|---|-----|
| Abstract..... | I |
| Table of Figures..... | III |
| Table of Tables..... | IV |
| 1. Introduction..... | 1 |
| 1.1 Problem Statement..... | 1 |
| 1.2 Approach..... | 1 |
| 1.3 Project Aim..... | 2 |
| 2. Background..... | 3 |
| 3. Product..... | 4 |
| 3.1 Software Development Lifecycle..... | 4 |
| 3.2 Software requirements..... | 4 |
| 3.3 Use case diagram..... | 5 |
| 3.4 Component diagram..... | 6 |
| 3.5 Project Management..... | 6 |
| 3.5.1 Development tools and methods used..... | 6 |
| 3.6 Implementation of the system..... | 8 |
| 3.6.1 Development of the Back-end | 8 |
| 3.6.2 Development of the Front-end..... | 9 |
| 3.6.3 Building a questionnaire..... | 9 |
| 3.6.4 Creation of Tfidf_tree..... | 10 |
| 3.6.5 The flow of data..... | 10 |
| 3.6.6 Final product..... | 12 |
| 4. Results and Analysis..... | 15 |
| 4.1 Selection of test method..... | 15 |
| 4.2 Test Cases Development..... | 17 |
| 4.2.1 Writing of test scenarios..... | 17 |
| 4.3 User Feedback Summary..... | 28 |
| 5.Future versions..... | 30 |
| 5.1 Features..... | 30 |
| 5.2 Improvement of data..... | 30 |
| 5.3 Maintenance..... | 31 |
| 6. Conclusion..... | 32 |
| 7. Reflection..... | 34 |
| 7.1 Useful decisions..... | 34 |
| 7.2 Incorrect decisions..... | 35 |
| 7.3 Acquisition of new skills..... | 35 |
| 7.4 Acquisition of soft skills..... | 35 |
| 8.Bibliography..... | 36 |

Table of Figures

| | |
|--------------------------------------|----|
| Figure 1: Use case diagram..... | 6 |
| Figure 2: Component diagram..... | 6 |
| Figure 3: Project Management..... | 6 |
| Figure 4: The flow of data..... | 10 |
| Figure 5: Cosine..... | 12 |
| Figure 6: Flowchart..... | 13 |
| Figure 7: Usability..... | 25 |
| Figure 8: Evidence 1..... | 25 |
| Figure 9: Accuracy..... | 26 |
| Figure 10: Compatibility..... | 26 |
| Figure 11: Reliability..... | 27 |
| Figure 12: Data fault tolerance..... | 27 |
| Figure 13: Performance..... | 28 |

Table of Tables

| | |
|--|----|
| Table 1: Stop word list..... | 8 |
| Table 2: The test case scenario and their states for 'When the system is started, it will open the speech recognition module'..... | 18 |
| Table 3: The test case scenario and their states for 'When the user clicks on the 'start1' or 'start2' button, the system will open the information display window'..... | 18 |
| Table 4: The test case scenario and their states for 'When the user clicks on the 'start1' button and the 'Listen' mode of the speech recognition module is enabled, the speech recognition module is able to receive the user's voice input'..... | 19 |
| Table 5: The test case scenario and their states for 'The voice recognition module can convert the voice message into text message'..... | 19 |
| Table 6: The test case scenario and their states for 'The user can enter the text message in the text input field'..... | 19 |
| Table 7: The test case scenario and their states for 'The system can remove common stop words from the text message according to the Stop Word List'..... | 20 |
| Table 8: The test case scenario and their states for 'The system can perform TF-IDF calculations on those messages which have been recognized'..... | 20 |
| Table 9: The test case scenario and their states for 'The system can display a list of keywords from largest to smallest according to their weight'..... | 21 |
| Table 10: The test case scenario and their states for 'The system is able to use Selenium framework to simulate a person performing keyword search in Google Chrome'..... | 21 |
| Table 11: The test case scenario and their states for 'The system is able to open, read and parse the URL of the searched web page via Urllib'..... | 22 |
| Table 12: The test case scenario and their states for 'The system is able to obtain text and image information from the web pages'..... | 22 |
| Table 13: The test case scenario and their states for 'The information display window is able to display the data that has been obtained'..... | 22 |
| Table 14: The test case scenario and their states for 'When the user updates the input information, the keywords analyzed by the system will be updated at the same time'..... | 23 |
| Table 15: The test case scenario and their states for 'When the topic of the conversation has changed, the system can analyse the new keywords as soon as possible'..... | 23 |
| Table 16: The test case scenario and their states for 'When the centre of the conversation returns to the old topic, the system can quickly 'recall' the old keywords'..... | 24 |

1. Introduction

With the rapid development of Internet communication technology, various chatting tools have become an important communication medium in people's daily lives (Özyurt & Köse 2010). However, in the process of having conversations with others, we inevitably encounter situations where we do not know much about the topic the other person is talking about. This often leads to the conversation ending in an awkward atmosphere. Therefore, analyzing the content of conversations and presenting the information that participants want to know can effectively improve the quality of people's conversations.

The main objective of this project is to enable the AC system to provide the user with additional information deemed thematically relevant to the participant's conversation in a timely manner. In this way the quality of user conversations is improved. The AC system first listens to the participant's session, extracts the keywords in the disaggregated chunks through topic detection, and then converts the text conversation into search terms through the keyword set. Finally a query is made in Google to provide the user with the content that can be displayed.

1.1 Problem Statement

When two people are chatting they may encounter many problems such as not understanding the jokes the other person is making, not knowing the news the other person is talking about and not having knowledge of what someone is saying. When chatting online, the inability to engage in a discussion with another person on the same topic is undoubtedly a major reason for blocking communication. Therefore, we can observe that communication barriers are something that needs to be taken seriously.

Today there are many ways to improve the quality of conversations between people. For example, by finding out about others' hobbies and any knowledge they may have before chatting and checking out some relevant information in advance. This will always give both parties a topic to talk about together when chatting. People can also Google information they don't know as soon as they come across it.

However, these methods can waste a lot of time, meaning that people need to think about the topic they are trying to find and find useful information on the web. a study by Lead Response Management.org found that responding within five minutes was more effective in connecting with others (LeanData 2021). Therefore, responding as quickly as possible tends to yield better results (Baker & Oswald 2010). In addition, people often prefer call chat as their form of communication. However, chat platforms do not provide additional services to participants' conversations, as the messages are transmitted via voice. Therefore, it does not matter what the user is communicating or whether the conversation can continue.

1.2 Approach

In the last decade, Internet technology has been developing at a faster pace than people

could have imagined. Four years ago, for example, ordering food online was just a concept. Now, however, we can see it everywhere. People are always coming up with ways to solve their problems. And for how to improve the quality of a conversation, a timely search for information is undoubtedly the best. However, it is clearly not feasible to carry out an uninterrupted search for information while conversing with others. After all, few people can have two minds and still grasp the subject of a conversation. Therefore, we need a software to solve this problem.

This project addresses this problem by providing a real-time information analysis and retrieval service for people's conversations. After the user runs the system, it continuously provides and updates information related to the subject of the conversation. The system runs in the following four steps: firstly, converting the user's speech into text during a conversation through the computer's speech recognition function; secondly, extracting the keywords in the user's conversation using the TF - IDF algorithm and calculating their weights, storing all the data in a list and sorting them; thirdly, selecting a few keywords with higher weights to form a search term and searching in Google; fourthly, displaying the searched results are displayed in a display window, which contains two text output windows and two image output windows.

It is worth mentioning that a large number of real-time voice tests are not easy to carry out. Therefore, a text input function has been included in the project to simplify the testing process by entering text directly during the testing phase. The main disadvantage of this test method is that it does not fully simulate a real conversation. Therefore, it can only be used as part of the test content and the data obtained can be used for functional analysis and comparison.

1.3 Project Aim

The objectives of the project are fourfold:

- To find a way to reasonably handle the keyword weighting of the text, which will handle all input, both voice input and manual input.
- Select a free speech-to-text method that is available.
- Create an easy to use GUI through which the user can perform voice and text input, select their preferred method and view the results.
- Display information about changes in keyword weights and update them in real time based on input.

2. Background

This section discusses the reasons for developing an enhanced conversation system for this project. Firstly, it is important to know what created the opportunity to change the way people chat. With advances in technology, users can text chat, voice chat and video chat on their computers and smart-phones through various software. This has led to people not having to chat face-to-face in many situations. Some studies have shown that online socializing gives shy people a better environment to chat (Baker & Oswald 2010). In 2021, many parts of life will begin to transform digitally because of the new crown epidemic (Rospigliosi 2020). Online social networking is even more widely used in people's work and school learning. Therefore, grasping the focus of the discussion is something that will be very helpful for all those who need to work and study through online social networking.

3. Product

In this section the methodology and design ideas used to build the entire software, such as the application used to analyse the text message section, are shown. The software development lifecycle section describes the software development modules used in this project. This is followed by a discussion of the requirements of the product and the tools used in the development process. The next section describes the development tools and the methods used in detail. Finally, the development process of the application is described in detail in chronological order.

3.1 Software Development Lifecycle

SDLC is an essential process to produce high quality software in the shortest possible time when designing, coding and testing software features. The model used in this project is the waterfall model (ALTVATER 2020). The advantage of the waterfall model is that it is easy to manage and is suitable for small projects with clear requirements (Kramer 2018). In the waterfall model, the inputs and outputs of each phase are well defined. The whole system is completed in a linear way. This method completes the development process in a structured way according to the requirements. In addition, most other models tend to be team-based development projects and are therefore not chosen.

In summary, the development steps for this project were as follows: firstly, analyse all possible requirements for the system; secondly, design the architecture of the system; thirdly, divide the requirements into small units and develop them, such as voice input, text input, etc.; fourthly, integrate and test the units developed in the previous phase.

3.2 Software requirements

Software requirements are mainly divided into functional and non-functional requirements. Functional requirements are the functions of the product, i.e. what the system is supposed to do. The focus is on the needs of the user. Non-functional requirements are what the product should do; it does not affect the basic functionality of the system, but it does affect the user experience (QRA 2020). The focus is on the expectations of the user.

Functional requirements:

- When the user starts the system, the system will open the speech recognition system that comes with the computer.
- When the user clicks on the 'start1' or 'start2' button, the system will open the information display window.
- When the user clicks on the 'start1' button and the 'Listen' mode of the speech recognition module is enabled, the speech recognition module is able to receive the user's voice input.
- The voice recognition module can convert the voice message into text message.
- The user can enter the text message in the text input field.
- The system can remove common stop words from the text message according to the Stop Word List.

- The system can perform TF-IDF calculations on those messages which have been recognized.
- After the user enters the text information and clicks the 'start1' button to confirm, the system can perform TF-IDF calculation on the text information.
- The system can display a list of keywords from largest to smallest according to their weight.
- The system is able to use Selenium framework to simulate a person performing keyword search in Google Chrome.
- The system is able to open, read and parse the URL of the searched web page via Urllib.
- The system is able to obtain text and image information from the web pages.
- The information display window is able to display the data that has been obtained.
- When the user updates the input information, the keywords analyzed by the system will be updated at the same time.
- When the topic of the conversation has changed, the system can analyse the new keywords as soon as possible.
- When the centre of the conversation returns to the old topic, the system can quickly 'recall' the old keywords.

Non-functional requirements:

- Usability: The system should be easy to use for all users.
- Accuracy: The keywords analyzed by the system should be accurate and the keyword list should be updated as soon as possible.
- Compatibility: The system should be compatible with all computer systems.
- Reliability: The system should be able to handle more than one input continuously.
- Data Tolerance: The system should be able to analyse common stop words such as eh, ohh, etc. which are spoken by users and eliminate them.
- Performance: The system should be able to provide information assistance for a complete conversation.

3.3 Use case diagram

After the basic requirements analysis has been completed, the next step to be completed is the construction of a use case diagram, UML involves many different diagrams that provide many different perspectives on the system. One such diagram is the use case diagram, which shows the functional requirements of the system from the point of view of use cases. It serves to enable the developer to relate the system requirements and the ways in which they can be met (Sengupta & Bhattacharya 2006). The use case diagram for the enhanced dialogue system is as follow:

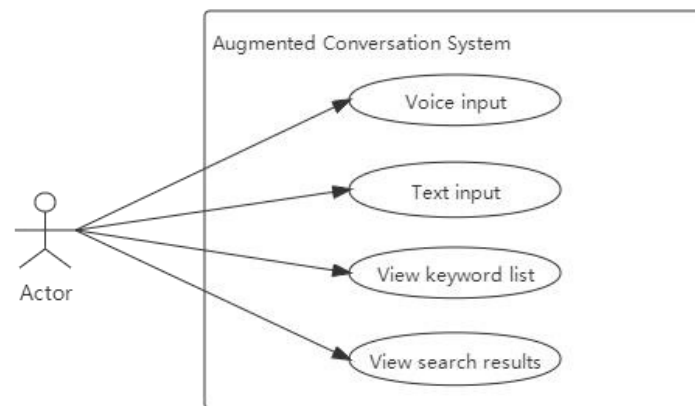


Figure 1: Use case diagram

3.4 Component diagram

The component diagram shows how components are connected into a system. It shows the architecture of the system components and their dependencies (Bell 2004). The image below shows the overall architecture of the system.

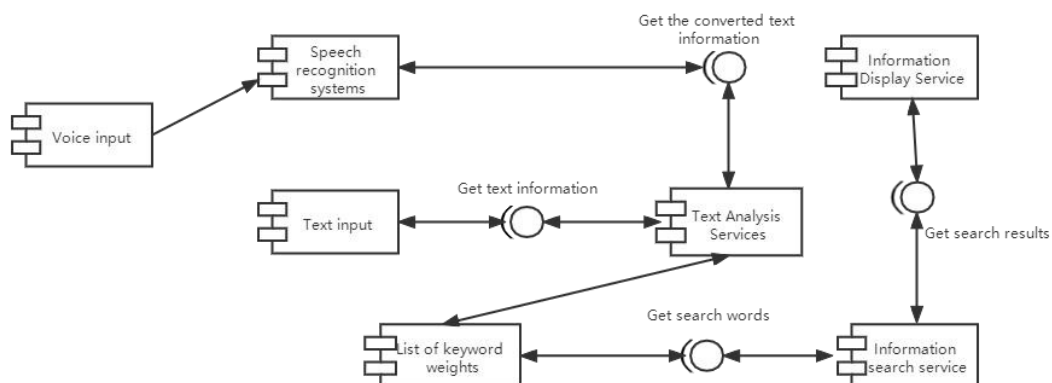


Figure 2: Component diagram

3.5 Project Management

For project management, a table in Excel is used to keep track of tasks to be done, tasks in progress and tasks that have been completed. The image below shows the status of the task list, which will be updated every Monday.

| Augmented Conversation System | | | | | | | | | | | | | | | | |
|-------------------------------|----------|------------|------------|------|--------|--------|--------|--------|--------|--------|--------|--------|--------|---------|---------|---------|
| Project start: Fri, 17/9/21 | | | | | | | | | | | | | | | | |
| Task | PROGRESS | START | END | DAYS | 1 week | 2 week | 3 week | 4 week | 5 week | 6 week | 7 week | 8 week | 9 week | 10 week | 11 week | 12 week |
| Requirements | 100% | 13/09/2021 | 17/09/2021 | 5 | | | | | | | | | | | | |
| Outline design | 100% | 18/09/2021 | 24/09/2021 | 6 | | | | | | | | | | | | |
| Detailed design | 100% | 25/09/2021 | 01/10/2021 | 6 | | | | | | | | | | | | |
| Coding | 100% | 02/10/2021 | 02/11/2021 | 32 | | | | | | | | | | | | |
| Test | | | | | | | | | | | | | | | | |
| Unit Test | 100% | 22/10/2021 | 02/11/2021 | 12 | | | | | | | | | | | | |
| Performance test | 100% | 03/11/2021 | 06/11/2021 | 4 | | | | | | | | | | | | |
| Iteration test | 100% | 07/11/2021 | 15/11/2021 | 9 | | | | | | | | | | | | |

Figure 3: Project Management

3.5.1 Development tools and methods used

The next step is to identify the tools and methods used to implement the enhanced dialogue

system. A number of tools and libraries have been used to develop this system in this project and the reasons for their use in the development process are discussed in this section:

- Python

Python is one of the most popular programming languages today, and it has many advantages. Firstly, Python is easy for many beginners to get started with and the syntax is simple enough for people who are new to it to make small programs with a small amount of code. This can boost the confidence of beginners in programming and stimulate their interest in learning. Python also has a rich library of standard libraries designed for scientific computing, resource provisioning, text analysis and more. This greatly facilitates the development of enhanced dialogue systems and reduces the development cycle. Finally, writing code in Python is highly efficient. It has many common functions built in, and all a developer needs to do to program is import the package and call a function (Tulchak & Marchuk 2016).

- Tkinter

Most of the current popular software interfaces are GUI, which use the mouse to trigger commands to graphical elements such as menus and buttons, and to obtain human-computer dialogue information from graphical display containers such as tabs and dialog boxes. Tkinter is a standard object-oriented GUI library that comes with Python and allows developers to create GUI applications quickly. Simple GUI can be easily implemented with Tkinter. And it runs well on most platforms (Chaudhary 2013).

- TF - IDF

TF-IDF is a very simple and effective keyword extraction method that involves little to no advanced mathematical theory. The calculation of TF-IDF is very simple: calculate the TF, then calculate the IDF and multiply the two together (Qaiser & Ali 2018).

The TF is calculated as follows:

$$TF_{ij} = \frac{n_{ij}}{\sum_k n_{kj}}$$

The IDF is calculated as follows:

$$IDF_i = \log\left(\frac{N}{N_i + 1}\right)$$

In the calculated result, the higher the TF, the more frequently the word appears in the document and the more important it is to the article. However, there are often words that occur in many documents. Even if the analysis filters for common stop words, there will still be such words. The higher the IDF, the less frequently the word appears in all documents, and the more likely it is to be a keyword in that document. the TF and IDF calculate the importance of a word to a document using frequency from two different perspectives: the individual document and the document set, and the product of them is the TF-IDF.

- Urllib

The Urllib library is used to manipulate the URL of a web page and to crawl the content of the page. Once the augmented dialogue system has processed the information entered by

the user and extracted the keywords, the system needs to search for these keywords on Google. The system needs to open, read and parse the URLs of the web pages in order to retrieve the search data (Mei & Liu 2019). All of this is done using the Urllib library.

· Selenium

Selenium is an interesting framework, I call it the 'unstoppable crawler spider', the advantage of Selenium is that it simulates a browser opening the website that the system needs to crawl. The only downside to Selenium is that it is too slow, as it simulates opening a web page (Gojare etc. 2015). However, for an online conversation, it is capable of getting the job done.

· Stop Word List

Removing stop words is an important step in text analysis. People often pause in conversation and there are many words in each sentence such as you, me and the that are not valid information. The large number of stop words in a document can easily interfere with valid information, so the system removes the interference from the input information before analysis. By reducing the frequency of stop words in the text being analyzed, the keyword density can be increased and the system can filter out more representative keywords (Kaur & Buttar 2018). The following is a list of some free stop words:

| Stop word list | URL |
|-------------------------|---|
| Snowball Stop word list | http://snowball.tartarus.org/ |
| Terrier Stop word list | http://bitbucket.org/kganes2/text-mining-resources/downloads/terrier-stop.txt |
| Minimal Stop word list | https://bitbucket.org/kganes2/text-mining-resources/downloads/minimal-stop.txt |

Table 1: Stop word list

3.6 Implementation of the system

The Enhanced Dialogue System uses a number of tools and libraries in its development. This section will describe how the system was developed. Firstly, the project was divided into modules based on the previously listed requirements. Each module was then divided into smaller parts to make it easier to save development time and manage the project schedule. Before each part was developed, test scenarios were created to test it. These test scenarios are created in Pycharm. For ease of observation, specific analysis data and detailed explanations are placed later in the analysis section. The development of the entire project followed the previous project management task list. The following section will show the development of the system in chronological order.

3.6.1 Development of the Back-end

In the first part of the development phase, the back-end structure is set up. The first step was to set up the system to receive input from the user. When the user opens the system, the system automatically switches on the speech recognition system that comes with the computer. When the user says 'Listening', the system will start speech recognition and store

the converted text information in a list. The second step is to build a list of deactivated words. Once the system has received the user's input, all the information stored in the list is deactivated to reduce the impact of deactivated words on the text analysis process. This helps to improve the accuracy of the keywords analyzed by the system. The third step is to obtain the keywords for the text. The enhanced dialogue system obtains the keywords and their weights from the user input and stores them in the list by calling the keyword extraction algorithm, TF-IDF. The fourth step is information retrieval. The system will initialize the browser driver without a data directory. After confirming the use of the Selenium framework, the system will simulate the manual use of the browser. It means that the system will enter keywords into the browser and perform a search. The final step is the collection of information. The augmented dialogue system will open, read and parse the URL of the web page through the Urllib library. The website is then opened according to the obtained URL. Next, the xpath syntax is used to retrieve the text and image parts of the first two search items on the page and to find their tags (Gunawan etc. 2019). Finally, all the data is populated into the system's display window.

3.6.2 Development of the Front-end

After the development of the back-end, the basic navigation of the front-end of the application is created using Tkinter. During the front-end development phase, the system displays the following screens: the user interface, the image display window and the text display window. The user interface is the basic screen of the application. The user has to enter information via the 'start' button on the operator interface. Once the user has started a conversation and entered information, the keywords and their weighting are displayed on the operator interface. The user can then view the information searched by the system on the image display window and the text display window. At this stage the Enhanced Dialogue System listens to the user's conversation via the speech recognition on the computer and it runs automatically whenever the user makes a voice input.

3.6.3 Building a questionnaire

After completing the front-end and back-end development, the system needed to be used to identify some hidden issues through user feedback. The user experience plan was to create an online questionnaire for each user depicting the functional design of the enhanced dialogue system and asking the user to choose whether the feature was implemented or not based on the description. The questions were created using a free online questionnaire creation site called All Counted. All questions can be placed on a website and the survey can then be distributed to the experience via a shared link or email. The project was lucky enough to find a number of students to volunteer for the experience survey. A total of 13 users responded. Some of the feedback received included:

1. If the subject of the conversation changes, the system cannot accurately locate the changed topic.
2. The keywords analyzed by the system are blurred when long conversations are entered.
3. The performance of the system also needs to be optimized
4. There are some stop words that have not been removed.
5. When the conversation takes too long it can cause the system to get stuck.

6. Maybe you can improve the beauty of the interface?

3.6.4 Creation of Tfidf_tree

Most of the suggestions made in the user feedback were about the system not being up to date with new topics when the conversation topic changes. As a simple example, the user starts with a conversation about 'dogs', so the system displays keywords related to dogs and gives it a high weighting. Then, after 20 sentences on the topic of dogs, the topic of discussion becomes 'cats'. The discussion about cats has about 5 sentences. According to the purpose for which the system was created, 'cats' should be at the top of the list of keywords displayed at this point. However this was not the case and 'cats' was not given very high weight. After some investigation the reason for this emerged. The system calculates the weighting each time for all the information entered. In this way, the proportion of keywords in all messages for the first topic decreases with the number of calculations, but it is still high and it is difficult for the second conversation to overtake it in a short time.

The main purpose of the Tfidf_tree is to allow the system to calculate weights for each input rather than for all inputs. The calculated keywords and their weights are stored in a list, and if the same keyword is obtained again on the second input, the weights are added together; if the same keyword is not obtained on the second input, then the weights of the keywords that do not recur are changed to half the original weights.

With the Tfidf_tree, whenever the topic of discussion changes, the new keyword can quickly appear at the top of the display and the old keyword will gradually sink. If the old topic is brought back up, it is quickly brought back to the top by adding up the weights.

3.6.5 The flow of data

Once the basic functions of an augmented dialogue system have been completed, the next concern is the flow of data. The flow of data determines whether the user gets the desired information, for example:

- whether the textual content displayed in the text window is relevant to the topic being talked about
- whether the content displayed in the image window is relevant to the topic
- The system's assessment of the similarity of the input and output content.

The functions involved in the flow of data are built in sequence and implemented in the system. First, we build the function that receives the user input. For this function, the UML sequence diagram can be used to show its underlying design as it shows the flow of messages between two objects (Bell 2004). The image below shows the flow of messages from the speech recognition system to the text analysis system when the user makes a voice input.

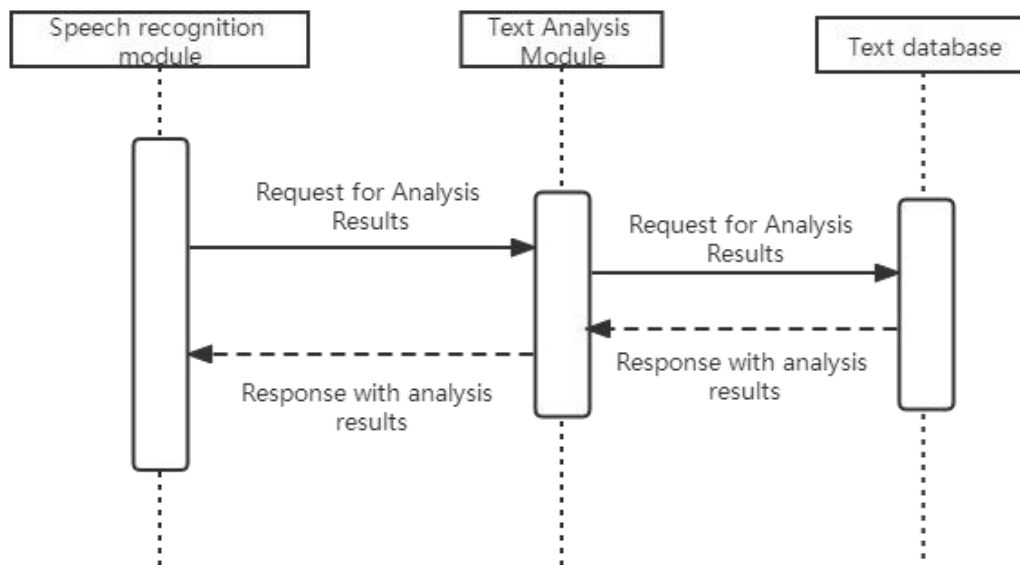


Figure 4: The flow of data

When the user opens the enhanced dialogue system, a simple interactive interface appears. When the user clicks the 'start' button, the message display window is loaded and the `doit` function is called. In this function, a request is made to the voice recognition system for the content of the voice recognition via the `get_voice()` method. The response is returned from the speech recognition system in the form of a list. The information in the list is the content of the user's conversation. Before the messages are stored, they are deactivated by the system and calculated by the TF-IDF algorithm. The data obtained is then displayed in a list of keywords.

In voice input mode, the way the user refreshes the keyword list is to continue the discussion and each time new content is obtained the system automatically analyses and refreshes the display list. 'start' button simply sends a `get_voice()` request to the speech recognition system and gets all the converted content. In addition, as the voice input is sometimes not conducive to the detection of a long speech, a text input box is included in the interactive interface in order to make it easier for the user to get the keywords they want.

Following the implementation of speech recognition, a feature was added to the system to allow users to analyse keywords by entering text directly. Initially, the function was developed to allow the user to quickly access the topic of the previous conversation. Once the user has entered the text they wish to analyse, they are navigated to the information display window. Once the display window is loaded, a back-end program is called to retrieve the information from the web page. The user can view the keywords of the entered text and the data stored locally after the system search.

The system compares the similarity of the searched information with the information entered by the user and uses the result as an automatic evaluation value for the system. In

the first step, the system builds a bag-of-words model and then identifies the two searched texts, i.e. divides the text into small chunks, for example in one English word or one Chinese character. The text is identified by splitting the unstructured data or natural language text into different discrete chunks of information that can be counted. The reason for this is that we can represent a document by counting the number of times each token appears in the document to form a frequency vector group, as computers cannot directly process text information that humans can distinguish, so we use statistics to convert pure text information into digital information. The main purpose of this operation is to facilitate a more focused analysis of the content of the text and the meaning it is intended to convey. In the second step, all the words in the two texts are stored in a dictionary and numbered. The number of occurrences of each word in the text is counted according to the bag-of-words model and a vector is formed. In the third step, the cosine similarity is calculated. Cosine similarity is the application of the cosine function to the field of natural language processing to do text similarity analysis. The process of calculation is to combine all the sub-words of the two texts to obtain a total set of words. This total set is then used as a reference to represent the two texts with a 1/0 vector. It means that each word in the total set is traversed, and the sentence is represented by a 1 if it is present, and a 0 if it is not, so that the two sentences can be represented by a string of 1,0 strings similar to a bitstream. Finally, the formula for the cosine function is applied. This is shown in the following diagram.

$$\cos(\theta) = \frac{a^2 + b^2 - c^2}{2ab}$$

Cosine similarity uses the cosine of the angle between two vectors in a vector space as a measure of the magnitude of the difference between two individuals. The closer the cosine value is to 1, the closer the angle is to 0°. This means that the more similar the two vectors are, which is called ‘cosine similarity’ (Li & Han 2013). This can be easily understood by looking at the picture.

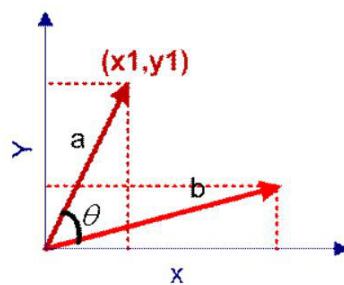


Figure 5: Cosine

We know that for two vectors, if the angle between them is smaller, then we consider that the two vectors are more similar. Cosine similarity makes use of this theoretical idea. It measures the value of similarity between two vectors by calculating the cosine of the angle between them. There is also a variant of the cosine formula (Proofwiki 2021).

$$\cos \theta = \frac{a \bullet b}{||a|| ||b||}$$

This formula is applied when a and b are both vectors. This means that the string of 0 and 1 that we got above can be calculated perfectly by this formula for the cosine.

3.6.6 Final product

The Augmented Conversation system has been tested and modified to complete its development. The flowchart below depicts how the final product will be used. The flowchart shows that when the user opens the application, the initial screen of the system is opened and the speech recognition system that comes with the computer is activated. After the user has selected an input method, two picture display windows and two text display windows will open. The user can then start the text analysis, either by selecting voice input or text input. After obtaining the keywords, the browser driver will be initialised by the system. Next, the system simulates a human search using the browser input. Finally the system parses the web page to retrieve the required information and stores all data locally.

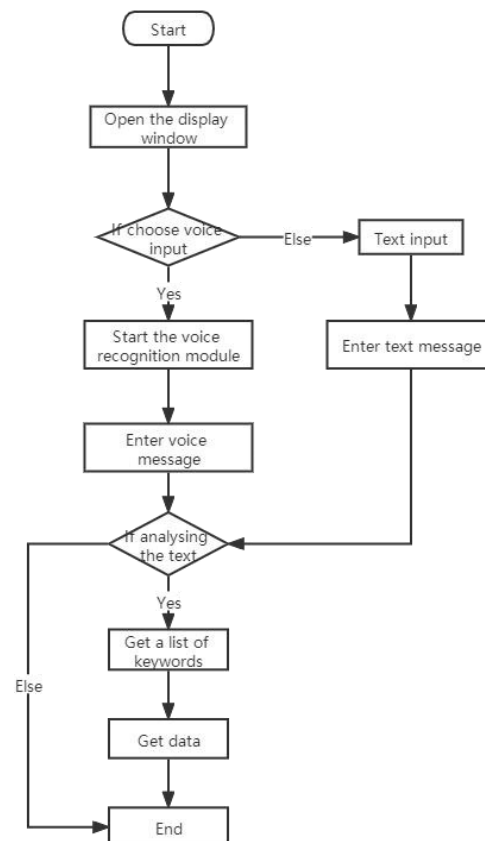


Figure 6: Flowchart

After the back-end functionality was completed, the system's GUI was developed and the feedback from the questionnaire study provided useful input for changes to the Augmented Conversation system. There are two of the most significant pieces of feedback: One was to add a text input field to the user action page; the other was to create a Tfidf_tree to ensure that the keywords analyzed by the system kept up to date with changes in the topics discussed.

The Augmented Conversation system's GUI was designed with the principle of clean, easy-to-use graphics in mind. It is mainly built using components from Tkinter. Tkinter was chosen for three reasons: firstly, because it is the simplest; secondly, because it comes with its own libraries, which do not need to be downloaded and installed and can be used at any

time; and thirdly, because Python, as a scripting language, is an advantage when developing simple programs. In this case, Tkinter is more than adequate!

The design of the program is consistent throughout, maintaining the width and height of similar items. The simplicity of all interfaces is maintained throughout the application. This allows users to easily access the information they want, such as a list of keywords and weights. To the right of this list is the list of evaluations used to display the results. Separate the image insertion part of the GUI interface into a separate function in the image information display window and define a class for tkinter directly. The advantage of this is that the main window interface created can be directly replaced by self. Finally there will be a space to display thumbnails of the crawled photos. The most important thing is that the system assigns the selected images `img1` and `img 2` to the property `set_image` of window 1 and window 2. Without this line of code, the images would not be displayed properly. The reason for this is that Python automatically recycles objects that are not in use, so the system needs to declare the objects using properties.

The content of the information display window will change as the keyword list is refreshed. It is worth mentioning that the Selenium framework was chosen for development in order to ensure that the Augmented Conversation system could obtain the majority of information. It is based on the principle of mimicking a manual search, which requires a process of typing, clicking on the search and selecting to obtain results. However, manual searches are generally slow. As a result, there may be a delay in updating the data information.

The previous section described the software development model used to create the system and described each of the methods used to project manage the Augmented Conversation system. Then, the tools and methods used for the development of the program are explained in detail. Finally, the development steps of the program are explained in chronological order. Pictures of the final product are also shown to illustrate the functionality of the system. In the next section the product is analyzed and the results of the tests are shown.

4. Results and Analysis

The content of this section will consist of two parts. The first part will describe the reasons for choosing the Behaviour Driven Development (BDD) testing approach. The second part is to verify the functional integrity of the Augmented Conversation system using various test cases.

4.1 Selection of test method

Comprehensible code is important when developing applications, and so is test code. One important thing that good test code must do is to keep the test logic clear. The following will describe the reasons for the choice of testing methods in this project.

A complete test case usually consists of three parts: 1. Set Up 2. Exercise 3. Verify.

If a test case can clearly distinguish between these three parts, it is already halfway there. However, if the test case only completes this step, it does not become the final 'understandable' test code.

If you want to make your test code comprehensible, the first thing you need to do is to express the test procedure for each test case in a clear and concise way (Steve 2016). We don't want to see a long method in a test case, because we need to understand it. If the tester calls many methods, the other participants will not know what he is doing. Of course, we also don't want to see the other extreme: a single function call in a test case. This behaviour achieves the apparent simplicity of the test case, but the opportunity to show the internal details is lost. What we need to represent is the testing process and methodology. For example, what the tester does, what data is inserted, what result is obtained, how it compares to the expected result, etc. If you encapsulate all this in a method, others will only understand what you might do, but it is difficult to understand how you will do it.

Secondly, when a test case fails, the tester needs to be able to clearly and accurately pinpoint the problem. In the extreme case, there should only be one assertion, or checkpoint, in a test case. Reducing the granularity of the test can speed up the location of the problem. It is worth noting that when a test case fails, we also need to find out whether the problem is with the test code or the code under test. This can be a tedious process. The person who locates the problem may not be the person who wrote the test code. Even if it is the person who wrote the test code, it will take time to understand what the test case did. At the time, testers will recognize the importance of articulating the testing process and methodology. Some people choose to use documentation to articulate the testing process and methods, but this is not a good method. There is an idea in agile development that "an outdated document is worse than no document at all". Why do you need documentation to articulate your testing process? It just means that your test code is bad enough. Others can't understand it. Even if there is a detailed documentation for bad code, as time passes, people change and the code is modified and maintained, the complex documentation and the code become increasingly out of sync. The result is that the code gets worse and more complex and the documentation gets more and more outdated. The best documentation is actually

code.

BDD is the preferred method for this project when building test cases. The reason is that BDD describes the behaviour of the software in an almost natural language (DAN & ASSOCIATES 2021). Therefore, it can be used as a requirements document for software, or it can be applied to testing as a standard document for testing. When doing unit tests, testers often test a function, or a class. However, the function or class which is being tested is likely to change frequently and our test cases need to change frequently with it. Then, BDD describes the whole system behaviour of the software, almost like a requirements document, and the variability is significantly reduced. As a result, the test cases do not have to change significantly. At the same time, these test cases are the closest to the requirements and to the actual system behaviour.

The behaviors described by BDD are like stories. Developers and testers work together to analyse the requirements of the software. And then write these requirements into individual stories. The developers are responsible for filling in the content of these stories and the testers are responsible for verifying the results of these stories. A story template is usually used to describe the stories (VIDEO 2017):

As a [X]

I want [Y]

so that [Z]

The same story may have different scenes. Having described the story through the template above, then the different scenes are described through the following template:

Given some initial context (the givens),

When an event occurs,

then ensure some outcomes.

A classic example is that of the ATM cash machine. The story is described as:

Title: Customer withdraws cash

As a customer,

I want to withdraw cash from an ATM,

so that I don't have to wait in line at the bank.

As a customer, I don't need to be in queues when I go to an ATM to withdraw money. The same story would have happened in a different scenario:

Scenario 1: Account is in credit

Given the account is in credit

And the card is valid

And the dispenser contains cash

When the customer requests cash

Then ensure the account is debited

And ensure cash is dispensed

And ensure the card is returned

If you withdraw more money than you have on deposit, it will be the following scenario:

Scenario 2: Account is overdrawn past the overdraft limit

Given the account is overdrawn

And the card is valid
When the customer requests cash
Then ensure a rejection message is displayed
And ensure cash is not dispensed
And ensure the card is returned

With such a story, a description of the scenario, testers can turn the above story into test code with some BDD testing framework. Developers implement the product code and ensure that the test code passes.

In summary, there are three advantages to using the BDD testing approach: Firstly, by using BDD, we can make the logic of our case descriptions clearer and reduce most of the time spent by maintenance staff in understanding the code; Secondly, by using BDD, we can build a bridge between automated testing and manual system testing. Perhaps a model could be formed: manual testers are responsible for designing stories and scenarios, and automated testers are responsible for implementing them. Through such a connection, manual testers can have better understanding of what the automated tests are doing, thus eliminating unnecessary duplication of effort; Thirdly, using BDD, the testing process can be made closer to actual user behaviour, thus finding out what is wrong with the system.

4.2 Test Cases Development

The first step in test case development is to determine how to design test cases that match each functional requirement. The second step is to test each of the proposed functional requirements. In the far right column of each table the completion of each test case in the corresponding scenario is marked (Raharjana & Justitia 2020). After the functional requirements have been tested, the non-functional requirements will be tested in the next section. Finally, in the concluding section, users' comments on the overall program are discussed, as well as difficulties encountered during the development process and how they were resolved.

4.2.1 Writing of test scenarios

This section shows some detailed information about those tests that have been completed. And describe how they show the completion status of the functional requirements. Each test scenario was finished with manual testing and screen shots were taken as test data. In this section, each functional requirement is delineated in detail and explanatory notes are provided for the test cases within each requirement. The test cases are matched to the test scenarios which describe the test behaviour. A list containing all test scenarios is presented in a table in the functional requirements. In addition to the scenarios, the table also shows the completion status of the tests. Furthermore, screen shots of some test scenarios are provided in the table as evidence of test completion.

Functional Requirements:

- **When the user starts the system, the system will open the speech recognition system that comes with the computer.**

This feature was developed to allow the user to perform voice input operations. As shown in the table below, this test scenario tests the loading of the speech recognition module. There is no need for negative testing of this feature.

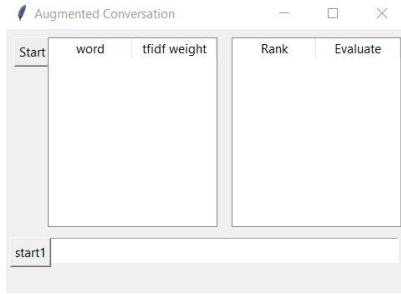
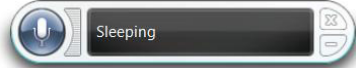
| SL No | TEST SCENARIO | STATUS |
|-------|--|--------|
| 1 | The user starts the AC system | PASS |
| | <p>EVIDENCE</p>  | |
| 2 | System turns on the voice recognition module | PASS |
| | <p>EVIDENCE</p>  | |

Table 2: The test case scenario and their states for 'When the system is started, it will open the speech recognition module'.

• When the user clicks on the 'start1' or 'start2' button, the system will open the information display window.

This feature is designed to ensure that the system can give feedback after the user has entered information. There is no need for negative testing of this feature.

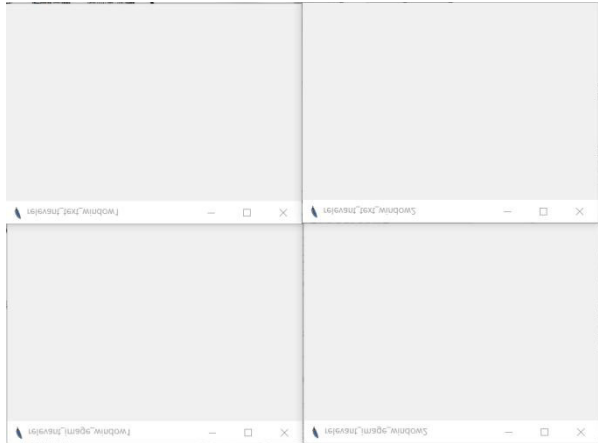
| SL No | TEST SCENARIO | STATUS |
|-------|--|--------|
| 1 | The user clicks on the 'start1' button, the system open the information display window | PASS |
| 2 | The user clicks on the 'start2' button, the system open the information display window | PASS |
| | <p>EVIDENCE</p>  | |

Table 3: The test case scenario and their states for 'When the user clicks on the 'start1' or 'start2' button, the system will open the information display window'.

- When the user clicks on the 'start1' button and the 'Listen' mode of the speech recognition module is enabled, the speech recognition module is able to receive the user's voice input.

Table 4 provides test scenarios to test whether the 'listening' mode of the speech recognition module is able to receive the user's voice input.

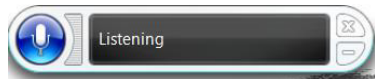
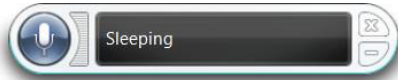
| SL No | TEST SCENARIO | STATUS |
|-------|---|--------|
| 1 | The user turns on the 'Listen' mode of the voice recognition module, the voice recognition module can receive the voice input from the user | PASS |
| | EVIDENCE  | |
| 2 | The user does not turn on the "Listen" mode of the voice recognition module. The voice recognition module cannot receive the user's voice input | PASS |
| | EVIDENCE  | |

Table 4: The test case scenario and their states for 'When the user clicks on the 'start1' button and the 'Listen' mode of the speech recognition module is enabled, the speech recognition module is able to receive the user's voice input'.

- The voice recognition module can convert the voice message into text message.

In order to test the completion of this requirement, only one test scenario was created, which is shown in Table 5. This function is used to ensure that text information can be obtained by the voice input module as well. This feature does not require negative testing.

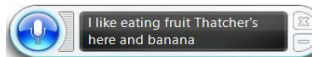
| SL No | TEST SCENARIO | STATUS |
|-------|---|--------|
| 1 | Voice messages are converted into text messages | PASS |
| | EVIDENCE  | |

Table 5: The test case scenario and their states for 'The voice recognition module can convert the voice message into text message'.

- The user can enter the text message in the text input field.

Table 6 provides test scenarios for testing the addition of information into the text input box.

| SL No | TEST SCENARIO | STATUS |
|-------|--|--------|
| 1 | The user enters a normal text message in the text input field. | PASS |
| | EVIDENCE | |

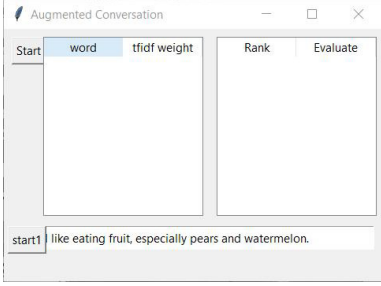
| | | |
|---|--|------|
| |  | |
| 2 | The user enters an empty text message in the text input field. | PASS |

Table 6: The test case scenario and their states for 'The user can enter the text message in the text input field'.

• **The system can remove common stop words from the text message according to the Stop Word List.**

There was only one test scenario created to test this feature, as shown in Table 7. This feature is to ensure that the system is not disturbed by useless stop words when analyzing keywords from text information. This feature does not require negative testing.

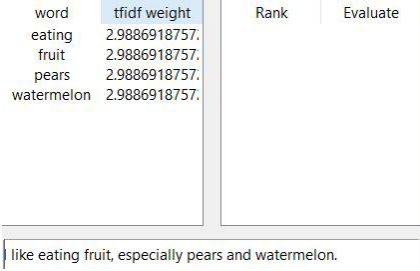
| SL No | TEST SCENARIO | STATUS |
|-------|---|--------|
| 1 | Common stop words in text messages are removed | PASS |
| | <p style="text-align: center;">EVIDENCE</p>  | |

Table 7: The test case scenario and their states for 'The system can remove common stop words from the text message according to the Stop Word List'.

• **The system can perform TF-IDF calculations on those messages which have been recognized.**

Table 8 shows the test scenario which is used to test the system's calculation of TF-IDF for text messages. The TF-IDF values are used to rank keywords in the next section.

| SL No | TEST SCENARIO | STATUS |
|-------|--|--------|
| 1 | Calculation of all keywords screened by using the TF-IDF algorithm | PASS |
| | EVIDENCE | |

| | |
|--|---|
| | <div> <div> wordtfidf weight eating2.9886918757, fruit2.9886918757, pears2.9886918757, watermelon2.9886918757, </div> <div>RankEvaluate</div> </div> <div>like eating fruit, especially pears and watermelon.</div> |
|--|---|

Table 8: The test case scenario and their states for 'The system can perform TF-IDF calculations on those messages which have been recognized'.

- The system can display a list of keywords from largest to smallest according to their weight.

As shown in Table 9, the system sorts the keywords according to their weighting. The purpose of this feature is that the user can always observe the topic of the current conversation.

| SL No | TEST SCENARIO | STATUS |
|-------|--|--------|
| 1 | The list of keywords is sorted with descending order of weight | PASS |
| | <div>EVIDENCE</div> <div> <div>wordtfidf weight</div> <div> pear5.9773837514, apple5.9773837514, eating1.4943459378, fruit1.4943459378, pears1.4943459378, watermelon1.4943459378, </div> </div> | |

Table 9: The test case scenario and their states for 'The system can display a list of keywords from largest to smallest according to their weight'.

- The system is able to use Selenium framework to simulate a person performing keyword search in Google Chrome.

The test scenarios used to test the feature are shown in Table 10. These scenarios tested the Selenium framework which simulates almost all the processes of a person performing a keyword search in Google Chrome.

| SL No | TEST SCENARIO | STATUS |
|-------|--|--------|
| 1 | The system opens Google Chrome | PASS |
| 2 | The system enters a keyword in the search box | PASS |
| 3 | The system performs search action | PASS |
| 4 | The system opens the first and second search results | PASS |

Table 10: The test case scenario and their states for 'The system is able to use Selenium framework to simulate a person performing keyword search in Google Chrome'.

- The system is able to open, read and parse the URL of the searched web page via Urllib.

The test scenarios used to test this feature are shown in Table 11. These scenarios test the full functionality of Urllib for opening, reading and parsing the URLs of the searched web

pages.

| SL No | TEST SCENARIO | STATUS |
|-------|--|--------|
| 1 | The system can open the URL of a web page | PASS |
| 2 | The system can read the URL of a web page | PASS |
| 3 | The system can parse the URL of a web page | PASS |

Table 11: The test case scenario and their states for 'The system is able to open, read and parse the URL of the searched web page via Urllib'.

- **The system is able to obtain text and image information from the web pages.**

Only one test scenario has been created in Table 12 to test this feature. This feature is designed to ensure that the system is able to retrieve text and image information from the web page. The feature does not require negative testing.

| SL No | TEST SCENARIO | STATUS |
|-------|--|--------|
| 1 | Text and image information on web pages can be crawled by the system | PASS |

Table 12: The test case scenario and their states for 'The system is able to obtain text and image information from the web pages'.

- **The information display window is able to display the data that has been obtained.**

Table 13 shows the test scenario used to test whether the information display window is able to show the data that has been obtained.

| SL No | TEST SCENARIO | STATUS |
|-------|--|--------|
| 1 | The information display window can show the data that has been obtained from the website | PASS |

EVIDENCE

Table 13: The test case scenario and their states for 'The information display window is able to display the data that has been obtained'.

- When the user updates the input information, the keywords analyzed by the system will be updated at the same time.

The feature is used to ensure that the system will analyse the keywords based on the information which is entered by the user, rather than giving vague results without any basis.

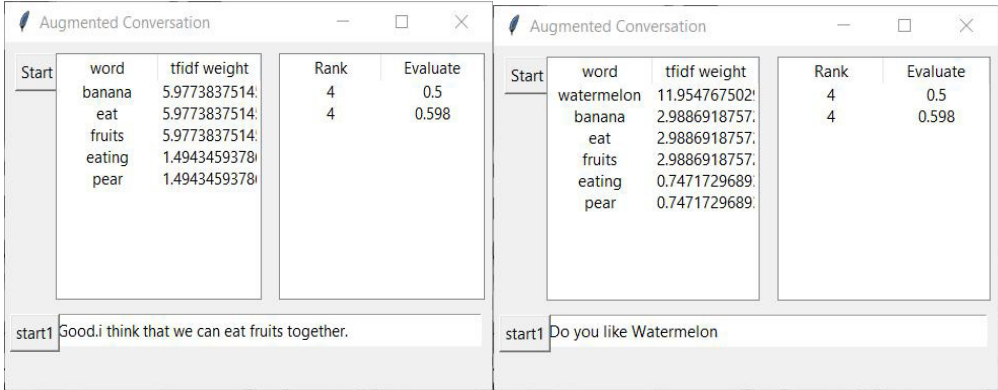
| SL No | TEST SCENARIO | STATUS |
|---|--|--------|
| 1 | User input is synchronized with the update of the keyword list | PASS |
| <p style="text-align: center;">EVIDENCE</p>  | | |

Table 14: The test case scenario and their states for 'When the user updates the input information, the keywords analyzed by the system will be updated at the same time'.

- When the topic of the conversation has changed, the system can analyse the new keywords as soon as possible.

This feature is used to ensure that the keywords which are analyzed by the system give real-time feedback on the centre of the current conversation and do not stay on the old topic.

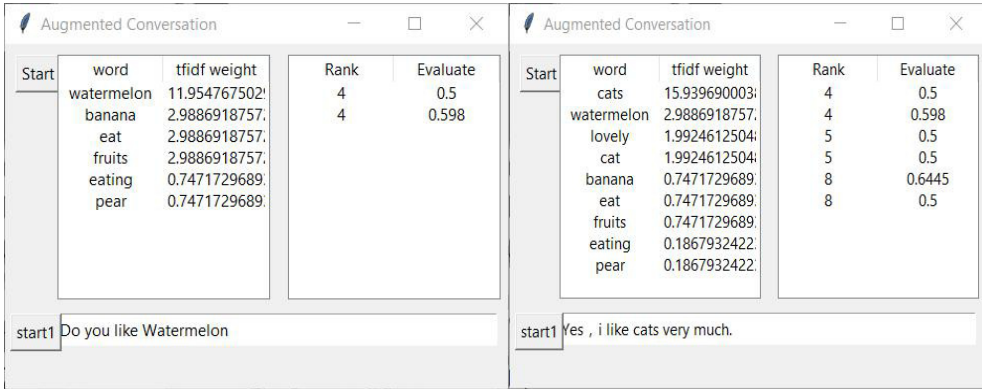
| SL No | TEST SCENARIO | STATUS |
|---|---|--------|
| 1 | The list of keywords can keep pace with the topic of the conversation | PASS |
| <p style="text-align: center;">EVIDENCE</p>  | | |

Table 15: The test case scenario and their states for 'When the topic of the conversation has changed, the system can analyse the new keywords as soon as possible'.

- When the centre of the conversation returns to the old topic, the system can quickly

'recall' the old keywords.

The purpose of this feature is designed to make it possible for the system to quickly correct the change of keywords when a brief shift in the topic of a conversation has been encountered.

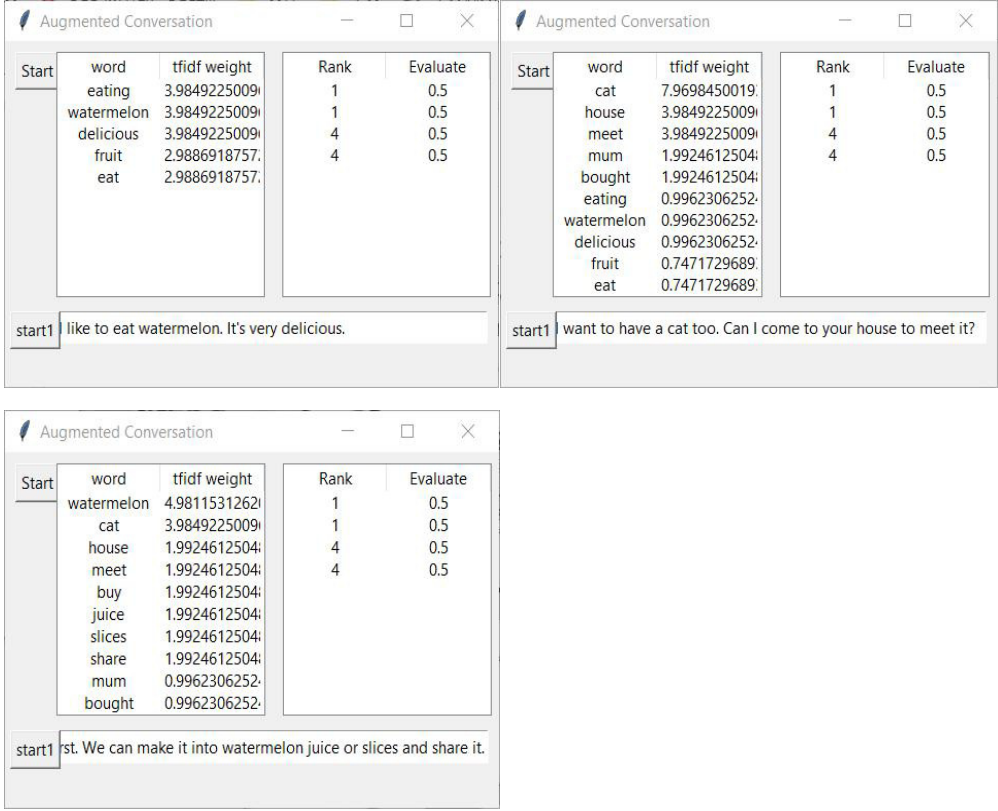
| SL No | TEST SCENARIO | STATUS |
|--|--|--------|
| 1 | The system can quickly 'recall' the old keywords | PASS |
| <div style="text-align: center;">EVIDENCE</div>  | | |

Table 16: The test case scenario and their states for 'When the centre of the conversation returns to the old topic, the system can quickly 'recall' the old keywords'.

Non-functional requirements:

The non-functional requirements for this project identified during the design phase included ease of use, accuracy, compatibility, reliability, data fault tolerance and performance. A detailed description of how these requirements were achieved follows. All non-functional requirements can be tested in a number of ways. The following section discusses usability requirements, then accuracy requirements, then compatibility, reliability requirements, then data fault tolerance requirements and finally about performance.

During the development of the system UI was designed using Tkinter in order to meet the requirements for ease of use of the application. This provides the user with a simple and easy to operate interactive interface. All the required tests were completed by users filling in a feedback form on the All Counted website. In the questionnaire there was a question about the ease of use of the system. The user can choose from numbers 1 - 5 to indicate how easy the system is to use. These numbers indicate "very uncomplicated", "not too easy",

"average", "slightly easy" and "very easy". All users answered between "slightly easy" and "very easy", with 18.18% answering "slightly easy" and 81.82% answering "Very easy". This is shown in Figure 7. Thus, it can be seen that the user response to ease of use is fairly consistent.

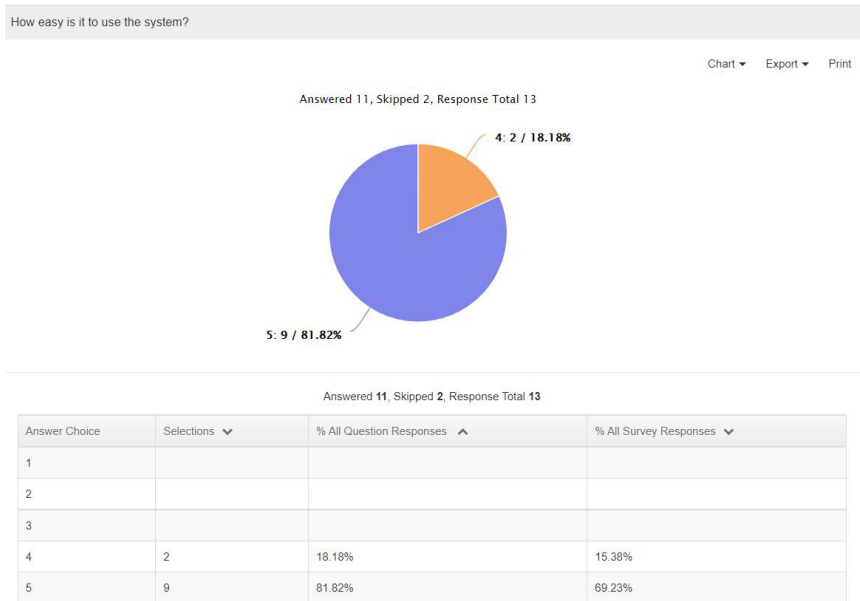


Figure 7: Usability

The requirement for accuracy of text analysis in the AC system was also completed during the development stage. Three texts with different themes were used to test this requirement. The system successfully analyzed the topic of each text in all three stages and the keyword list was updated in time when new text was entered. The screen shot in Figure 8 demonstrates the completion of this requirement.

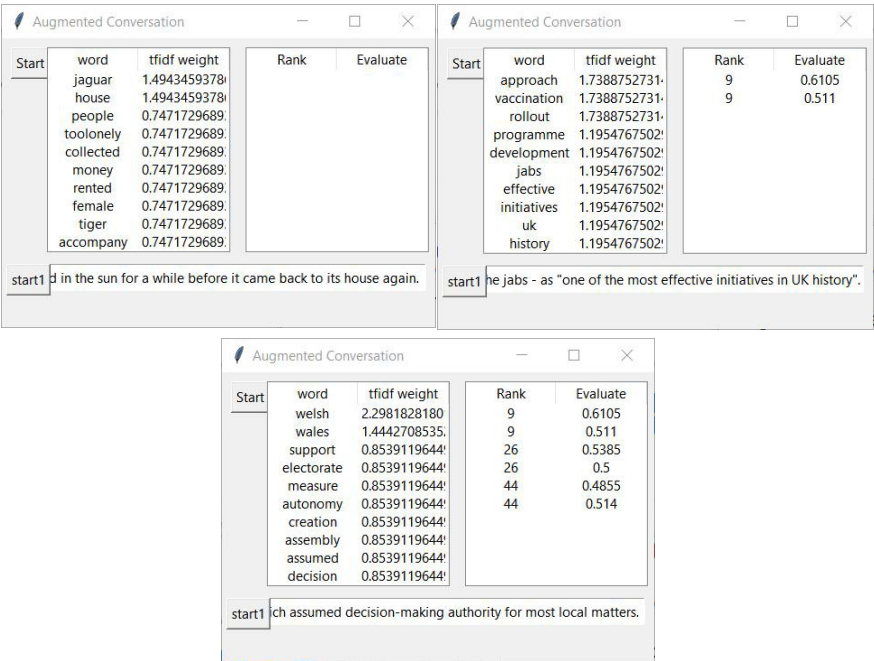


Figure 8: Evidence 1

The accuracy of the system is also evidenced by the feedback of some users. 53.85% of users considered the keywords to be "very accurate", 30.77% answered "somewhat accurate", 7.69% said "average" and 7.69% said "not very accurate". Fair" and 7.69% "Not very accurate".

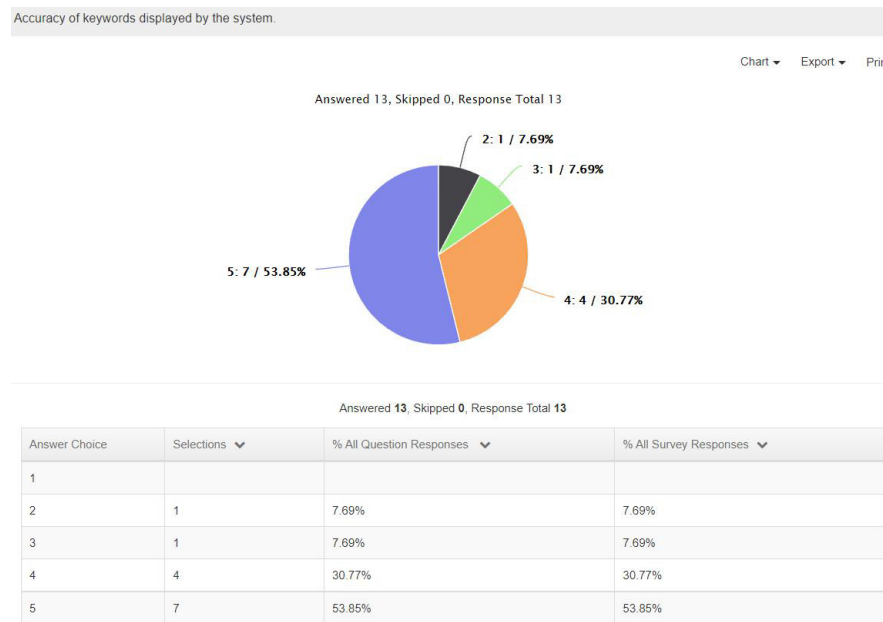


Figure 9: Accuracy

The application's compatibility requirements were also completed. To test this requirement, the application was used by several users with different systems. The successful operation of the system in all simulators proved that the requirement was fulfilled.

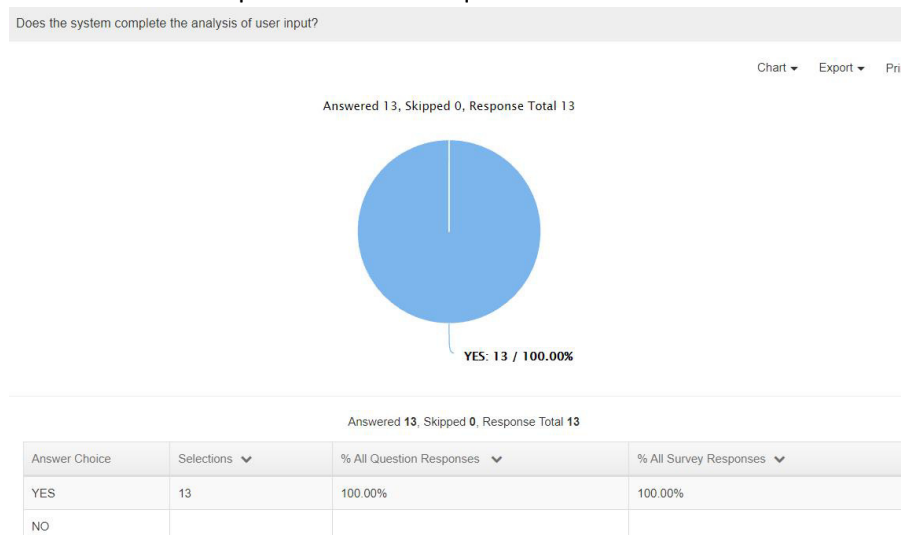


Figure 10: Compatibility

Reliability requirements are also met in the development phase. When users continuously input information, the system is able to process it in a timely manner and provide appropriate feedback to the user. The system tests the reliability requirements with BDD scenarios, which are created by the developers before development begins. Therefore, it is ensured that each feature is developed to meet the requirements. A number of screen shots

are provided in Figure 11 as proof of requirement completion.

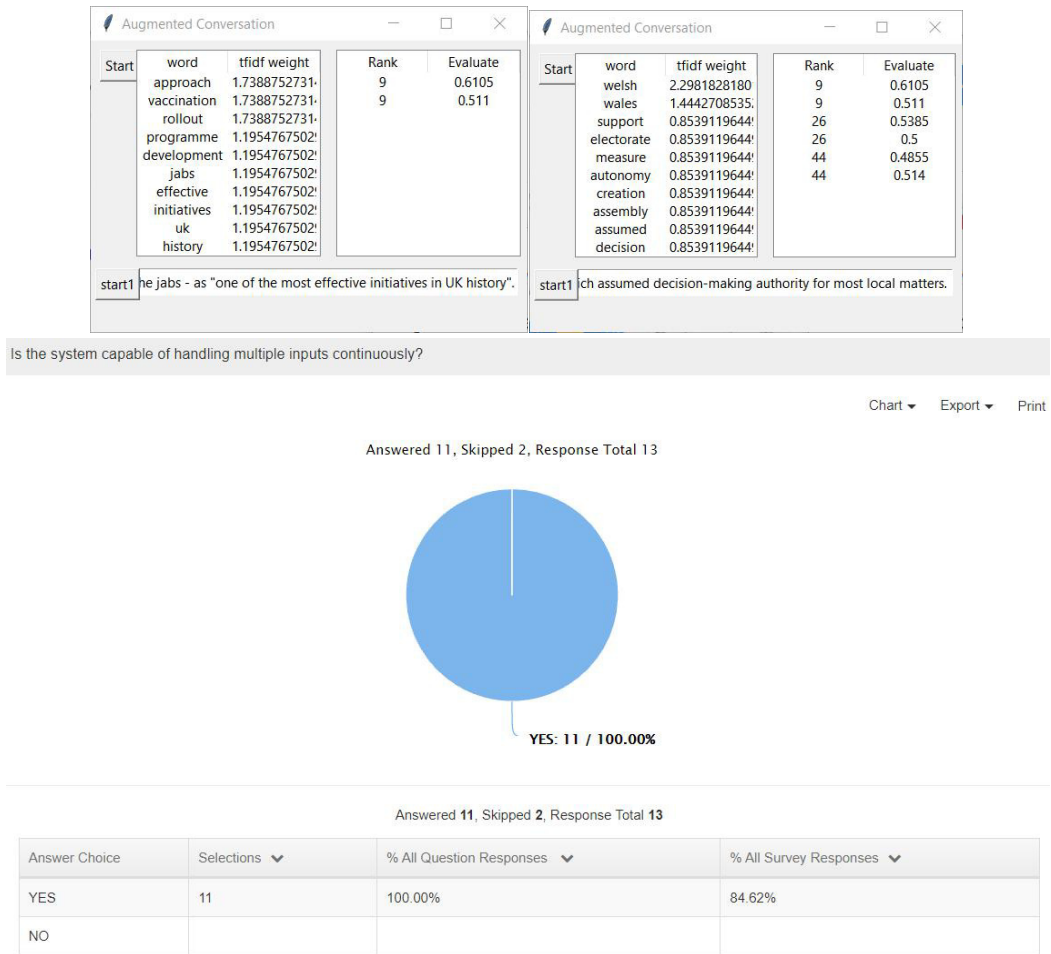


Figure 11: Reliability

Next, on the completion of the data fault tolerance. A piece of text with some common stop words was used to test the requirement. As can be seen from the results, all deactivated words were excluded from the keyword list. The screenshot in Figure 12 demonstrates the completion of this requirement.

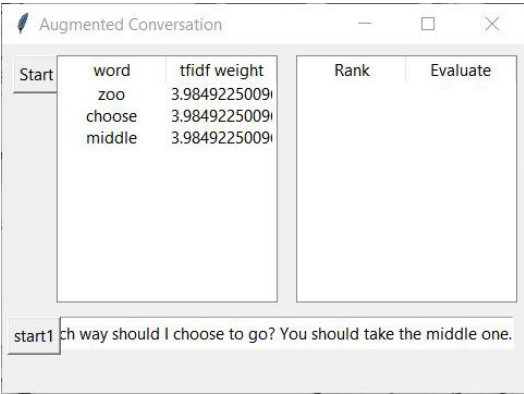


Figure 12: Data fault tolerance

In the end, similarly, the performance requirements are completed during the development phase, and the AC system needs to provide a complete information support service to the

users. As shown in Figure 12, 81.82% of users answered "YES" and 18.18% answered "NO" in the user survey. Therefore, it can be seen that the AC system is able to provide a complete service to the user in most cases.

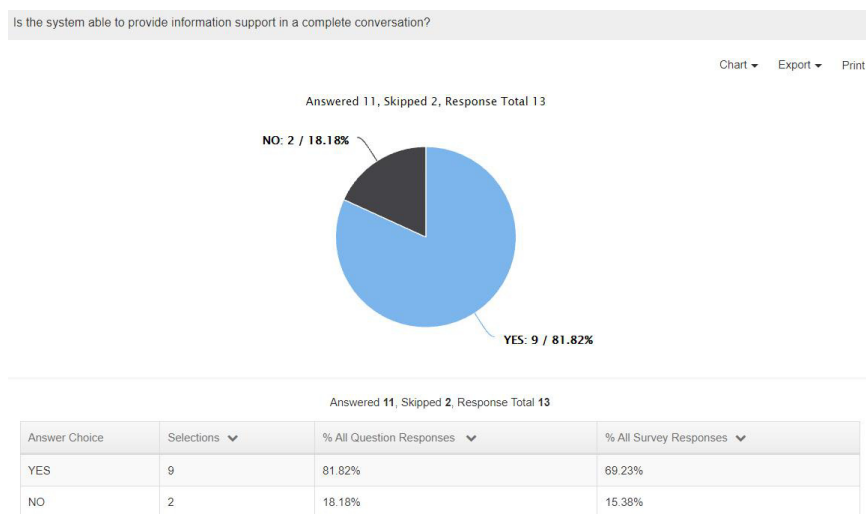


Figure 13: Performance

4.3 User Feedback Summary

In this section the feedback provided by users from the questionnaire is presented. The user feedback is focused on two main areas, one of them is about the keyword algorithm and the other is about the UI. The feedback on the keyword algorithm will be discussed first, and then on the issue of some pages failed to crawl information.

The questionnaire was created based on the initial functionality of the AC system. Therefore the initial functions and requirements are included in these questions. The questionnaire is sent to volunteers in the form of a link. They could also invite others to participate in the study by sharing the link. As a result, a total of 13 users participated in the activity.

The suggestions from the volunteers have been very helpful in improving the system. Since the main function of the program is to analyse text, users are concerned about the results. Some volunteers identified problems with the system not being able to update keywords in time when conversation topics were changed.

Because some websites have anti-crawl functions, the system is unable to collect information from the website. The problem reduced the efficiency of the system in collecting data significantly. Other feedback which has been received is about the voice input mode sometimes not being convenient for volunteers to do some tests, so the feedback has also received some suggestions about adding a text input field. We analyzed this feedback during development and added features such as a new keyword weighting decay algorithm, the Selenium framework and a text input field to the final product.

When the volunteers used the AC system again, they were asked to fill in an online questionnaire about their satisfaction. This time, the feedback received from the users was

consistent. They all found the system easy to use and helpful for them when socializing online. In particular it also solved the problem of some students not being able to get to the point during online classes.

In this chapter, the advantages of the chosen test methods and all the test cases designed are described in detail. Firstly, it introduces behaviour-driven development techniques. Then test cases are listed for functional as well as non-functional requirements. The feedback from users collected in the questionnaire is summarized and the comments from volunteers are analyzed in order to improve the functionality of the system. The next chapter describes the features that could be added to the product in the future.

5. Future versions

This chapter discusses the future prospects of the project. Due to time constraints, some features were not able to be implemented. As the AC system was developed using a modular architecture, these features can be added with minimal changes in future releases. This chapter then describes these unimplemented features, followed by improvements regarding data information, and then improved maintenance.

5.1 Features

This section will discuss the features that can be added in future version updates. The first two features to be described are those that could be implemented in the first version of the system before release. Then a list will show some of the features that can be presented in future versions of the system after they have been developed, tested and implemented.

The main function of the AC system will be to analyse the keywords in the text. The information in the keyword list will be constantly updated as the user enters new information. Currently, when the user uses the program, they can only view the topic of the current conversation and cannot review old topic keywords. Old messages are refreshed and cannot be found on their own. Adding the ability to find old messages will provide users with a better information service. They can select any message they want to review and view it themselves, taking note of the highlights.

Another feature that can be added is the ability to export the information searched for in text form. Currently, users can only view information in real time while using the system. Adding this feature would make it easier for users to keep track of messages in meetings and gather the information they want in a timely manner. As the number of users grows, the app will not only be used for web chat, it could potentially be used in a variety of scenarios. For example, office meetings, online classes and business communication. The ability to export information will help users to solve these problems.

The functions that can be added in the new version are shown below:

- Get the source site for the information displayed by keyword.
- Delete unwanted keywords.
- Create new sessions.
- Delete history.
- Share images and documents.

5.2 Improvement of data

This project was developed to help users get to grips with topics in web conversations, so only some basic steps have been taken to process the data. The next section discusses what is still lacking in the functionality that has been implemented and discusses some solutions.

Both the keyword list and the information display are important components of the AC system. They are used to provide users with information support that can be browsed, i.e. if,

whenever user A or user B says a sentence, the system analyses the information and displays the results. However, the information is displayed and updated in real time. This can result in users not being able to view the full message in a timely manner. It is therefore important to add the ability to store and find historical information. This will increase user satisfaction with the information service.

In addition to the ability to store and find historical information, the program needs to be able to record conversations and export them for some users' work or study needs. This function should, of course, be able to be turned on and off by the user on their own. At present, the project is not able to provide a better service to the users, however, before the program is released, a suitable group of users needs to be planned. And add the appropriate features to the system.

5.3 Maintenance

This section will discuss how to maintain the system and the tools that can be used to implement the maintenance steps.

As this project has not yet been released, there are no resources available to maintain the system at this time. Long term maintenance is very important for any product. It helps to retain a large number of customers, to fix bugs in a timely manner, to compete with products in the industry and to improve user satisfaction. The easiest way to maintain a system is to add new features to it that meet the needs of the users. In addition to this, ensuring that the product is compatible with the latest hardware and operating system releases on the market, improving performance, fixing bugs and updating the user interface in response to customer feedback are all maintenance tasks that need to be completed (Lenarduzzi & Taibi 2018).

Bugzilla is a common tool used for "bug tracking". Used by thousands of software companies, Bugzilla is primarily used to track bugs that occur in software products. In addition, Bugzilla allows software engineers to manage quality assurance, track bugs and submit bug fix patches for review, and is a secure program that can scan and correct database inconsistencies.

HP Quality Center is a web-based application that helps developers test for errors in their programs. It helps developers to test for errors in their programs. HP Quality Center has features such as planning and scheduling tests, analyzing results and managing problems and defects (Cohen 2021).

The above describes what needs to be done in the future with this product. It discusses these tasks in terms of Features, Improvement of data and Maintenance. It also provides some simple solutions and tools for these problems. This is followed by a summary of the project.

6. Conclusion

In this section a brief description of the project will be given. It demonstrates the main advantages of the AC system. First describe the functions accomplished by the program and then consider its shortcomings. Finally it discusses how the program has helped the client.

With the increasing development of Internet technology. The way people talk to each other has been changing from voice calls more than a decade ago, to text conversations and now to video conversations. Similarly, most of the voice calls have been used for years to connect with each other. And with the needs of people, especially with the new crown epidemic that occurred in 2020 has led to the way people work and study having to change dramatically. The epidemic has forced people to communicate less offline. As a result, online conversations are being used in more places, such as education, work or shopping. On the one hand this has helped people to be able to continue their studies and work, but on the other hand it has also caused many people to experience a decrease in their work and study efficiency. Online communication creates a disconnection between people and distracts them easily. There is no tool to help people get to the point in a conversation.

The main objective of this project is to create an application that effectively helps the user to get to the point in a conversation and receive relevant information services. It allows the user to enter the data to be analyzed by voice or text and to give useful feedback in a simple form. The results can be found in the user feedback from the questionnaires and the completed test cases. The project has successfully achieved all of the initial objectives identified prior to its development.

A list of commonly used stop words obtained from some official websites can be very useful to help the system to remove those stop words that can affect the analysis results. After the initial processing, the information is analyzed by the TF-IDF algorithm. The keywords obtained through the analysis are displayed in the keyword list and presented to the user. In addition, the system will simulate a person performing a keyword search in Google Chrome through the Selenium framework. The URL of the searched web page is opened, read and parsed using Urllib. The text and image information on the web page is then crawled and the results obtained are fed back in the display window.

The application UI was created using Tkinter. It can be easily done with a simple graphical interface and it runs well on most platforms. However some users have mentioned in their feedback on the questionnaire that some optimization of the system interface could be done. This should be improved in a subsequent release.

One of the main limitations of the program at the moment is that all the information it displays is updated in real time. This means that users may miss out on some important and useful data. The best solution proposed now is to add a search history to the program, displaying historical data and exporting data. This would help the user to keep track of the key points in the conversation.

Overall, the AC system is very helpful for people from all backgrounds. It can be used by people who are socializing online, by students who need to take classes online or by professionals who need to conduct teleconferences. It will help users to get the information they need without them having to perform tedious operations. It can also help two strangers to have a smoother communication. Its biggest advantage is that it improves the quality of the user's conversation, so that the subject of the conversation between two people does not deviate significantly.

7. Reflection

Developing the Augmented Conversation system has been a great boost to my research skills, technical skills and project management skills. It has helped me in my future academic research and personal development. The next section discusses the skills that I have learned while researching and developing this project. The first part discusses the decisions that were helpful in developing this system. The second part will describe the causes of the errors and the lessons I have learned. The final point to be made is the new computer knowledge and other abilities I gained from the project.

7.1 Useful decisions

When I joined Cardiff University in 2020 and started taking classes online, I noticed that international students sometimes didn't catch the whole point of the class because of the language aspect. It was then that I had the idea to create a program that could analyse voice messages in meetings. I didn't have time to complete my idea because of the required reports and projects I had to complete during my studies. Fortunately I found this project when choosing my thesis topic. I think it coincides with my project. The main objectives of the project were identified prior to the development of the application. This provided a good direction for the background research for the project, narrowed down the scope of finding information and saved a lot of time.

During the development of this project, I researched many online articles. Knowledge that may be used in the project was listed and collected. For speech-to-text, we looked at many available solutions and analyzed their advantages and disadvantages. These preparations prepared the ground for the successful completion of this project.

In addition, this project investigates knowledge about natural language algorithms. This involved reviewing a large number of papers and online articles. This helped in understanding the process of analyzing information in the system and provided help in fixing flaws later. In addition, the development process required some exploratory research into the technologies to be used, such as Tkinter, Urllib and the Selenium framework. This was mastered by reading online documentation on the relevant technologies and by reading a lot of sample code. I'm sure I'll be using these technologies again in future projects.

It's worth mentioning that I use GitHub as version control. During my studies in Cardiff I learned to use it to back up the AC system at every point in the development process. This has been a great help to me. For example when I have changed the code to fix some bugs or have corrupted the local version of the project by installing some libraries and need to restart the project from its last working state. I can get the previous version directly from GitHub. I therefore feel that I will continue to use it even after I have worked on it.

This project also provided me with an opportunity to use the knowledge and skills I had gained during my year-long course of study. Skills such as a clean coding style and BDD testing were very helpful in the development of this project. This has helped me to fix bugs

and add new features. It also enabled me to complete the project within the time limit. These skills will definitely help me in any future projects.

I have acquired this skill of project management during my course work and used it in this project. I listed all the tasks to be done and tracked their completion in a table. It has helped me to manage my time better in software development. I think I can use this method to manage future individual projects and team projects as well.

7.2 Incorrect decisions

When designing the text analysis function, I thought that it would be sufficient to extract keywords and calculate their weights. With limited research experience in natural language analysis, I didn't realize that this crude approach was problematic. It was only during testing that I realized that if the first input was much longer than the second, the keyword weighting of the first part would be greater than that of the second part for a long time. This could result in the subject of the conversation changing without the system analyzing it. This is a fatal flaw for the whole project.

I chose to add a new weight decay algorithm to help the system detect changes to the conversation topics in a timely manner. This mistake was serious, but it could have been avoided if I had tested each feature rigorously as it was completed. Therefore, I will certainly keep this lesson in mind when working on future projects.

7.3 Acquisition of new skills

During my university years I did an internship in a private company where I learned some software development. Examples include JavaScript, MySQL and Oracle databases, Python and some automated testing techniques. Although I have some experience. However, I made a mistake when working on this personal project.

Developing the AC system has exposed me to new skills. Based on the knowledge I gained at school and in my internship company, I started to learn Tkinter, the Selenium framework and natural language algorithms on my own. I looked up online videos of instruction on the internet. I eventually learnt these techniques and used them in my project. In addition, this project helped me to improve my skills in research topics and project management, which will help me with my future work.

7.4 Acquisition of soft skills

It is worth mentioning that developing the AC system has also helped me to improve some of my personal skills. At the beginning of the project it was difficult for me to keep working on the same thing for many days and to complete multiple tasks in a limited amount of time. But during the development process I organized all my work, completed each requirement in an organized manner and achieved all my goals step by step. I have learned how to set goals for myself and manage my time so that I can handle multiple tasks at once. These soft skills will be very helpful in all aspects of life.

8. Bibliography

- [1] Smith, D. A. (2020, October). The Augmented Conversation and the Amplified World. In Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology (pp. 4-4). [Accessed 20 October 2021].
- [2] Krieger, M. (2020, March). Chatting with glue: cognitive tools for augmented conversation. In Conference Companion of the 4th International Conference on Art, Science, and Engineering of Programming (pp. 208-208). [Accessed 20 October 2021].
- [3] Özyurt, Ö., & Köse, C. (2010). Chat mining: Automatic determination of chat conversations' topic in Turkish text based chat mediums. Expert Systems with Applications, 37(12), 8705-8710. [Accessed 20 October 2021].
- [4] LeanData. (2021). The Modern Rules of Lead Response Time. [Online] Available at: <https://www.leandata.com/blog/the-modern-rules-of-lead-response-time/> [Accessed 20 October 2021].
- [5] Baker, L. R., & Oswald, D. L. (2010). Shyness and online social networking services. Journal of Social and Personal Relationships, 27(7), 873-889. [Accessed 20 October 2021].
- [6] Rospigliosi, P. A. (2020). Digital transformation of education: can an online university function fully?. [Accessed 20 October 2021].
- [7] ALTVATER, A. (2020). What Is SDLC? Understand the Software Development Life Cycle. [Online] Available at: <https://stackify.com/what-is-sdlc/> [Accessed 20 October 2021].
- [8] Kramer, M. (2018). Best practices in systems development lifecycle: An analyses based on the waterfall model. Review of Business & Finance Studies, 9(1), 77-84. [Accessed 20 October 2021]
- [9] QRA. (2020). Functional vs Non-Functional Requirements: The Definitive Guide. [Online] Available at: <https://qracorp.com/functional-vs-non-functional-requirements/> [Accessed 20 October 2021].
- [10] Sengupta, S., & Bhattacharya, S. (2006, June). Formalization of UML use case diagram-a Z notation based approach. In 2006 International Conference on Computing & Informatics (pp. 1-6). IEEE. [Accessed 20 October 2021].
- [11] Bell, D. (2004). Uml basics: The component diagram. IBM Global Services. [Accessed 28 October 2021].
- [12] Tulchak, L. V., & Marchuk, A. O. (2016). History of python (Doctoral dissertation, BHTY). [Accessed 28 October 2021].

[13] Chaudhary, B. (2013). Tkinter GUI application development hotshot. Packt Publishing. [Accessed 28 October 2021].

[14] Qaiser, S., & Ali, R. (2018). Text mining: use of TF-IDF to examine the relevance of words to documents. *International Journal of Computer Applications*, 181(1), 25-29. [Accessed 01 November 2021].

[15] Mei, H., & Liu, D. (2019). Design and Implementation of Network Data Reptile. *International Core Journal of Engineering*, 5(9), 8-19. [Accessed 01 November 2021].

[12] Adams, P. H., & Martell, C. H. (2008, August). Topic detection and extraction in chat. In *2008 IEEE international conference on Semantic computing* (pp. 581-588). IEEE. [Accessed 01 November 2021].

[13] Gojare, S., Joshi, R., & Gaigaware, D. (2015). Analysis and design of selenium webdriver automation testing framework. *Procedia Computer Science*, 50, 341-346. [Accessed 01 November 2021].

[14] Kaur, J., & Buttar, P. K. (2018). A systematic review on stopword removal algorithms. *International Journal on Future Revolution in Computer Science & Communication Engineering*, 4(4), 207-210. [Accessed 01 November 2021].

[15] Gunawan, R., Rahmatulloh, A., Darmawan, I., & Firdaus, F. (2019, March). Comparison of web scraping techniques: regular expression, HTML DOM and Xpath. In *International Conference on Industrial Enterprise and System Engineering (IcoIESE 2018) Comparison* (Vol. 2, pp. 283-287). [Accessed 08 November 2021].

[16] All Counted. (2021). [Online] Available at: https://www.allcounted.com/?lang=en_US [Accessed 08 November 2021].

[17] Bell, D. (2004). UML basics: The sequence diagram. Retrieved July, 17, 2015. [Accessed 08 November 2021].

[18] Li, B., & Han, L. (2013, October). Distance weighted cosine similarity measure for text classification. In *International conference on intelligent data engineering and automated learning* (pp. 611-618). Springer, Berlin, Heidelberg. [Accessed 08 November 2021].

[19] Proofwiki. (2021). Cosine Formula for Dot Product. Available at: https://proofwiki.org/wiki/Cosine_Formula_for_Dot_Product [Accessed 15 November 2021].

[20] Steve Fox. (2016). All You Need to Know About Behaviour-Driven Software [Online] Available at: <http://behaviour-driven.org/> [Accessed 20 November 2021].

- [21] DAN NORTH & ASSOCIATES LTD. (2021). Introducing BDD [Online] Available at: https://www.allcounted.com/?lang=en_US [Accessed 20 November 2021].
- [22] Raharjana, I. K., Harris, F., & Justitia, A. (2020). Tool for generating behavior-driven development test-cases. *Journal of Information Systems Engineering and Business Intelligence*, 6(1), 27-36. [Accessed 20 November 2021].
- [23] Lenarduzzi, V., Sillitti, A., & Taibi, D. (2018, June). A survey on code analysis tools for software maintenance prediction. In *International Conference in Software Engineering for Defence Applications* (pp. 165-175). Springer, Cham. [Accessed 06 December 2021]
- [24] Cohen B. (2021). List of Software Maintenance Tools [Online] Available at: <https://www.techwalla.com/articles/list-of-software-maintenance-tools> [Accessed 06 December 2021].
- [25] VIDEO. (2017). Bad Behavior (Driven Development) + FREE CHEAT SHEET [Online] Available at: <https://www.developmentthatpays.com/posts/98-bdd-erratum> [Accessed 06 December 2021].