University of Sheffield

Building an Alexa Skill for DCS

Leon Singleton

Supervisor: Professor Rob Gaizauskas

A report submitted in fulfilment of the requirements

for the degree of BSc in Computer Science

In the

Department of Computer Science

May 1st, 2019

Declaration

All sentences or passages quoted in this report from other people's work have been specifically acknowledged by clear cross-referencing to author, work and page(s). Any illustrations that are not the work of the author of this report have been used with the explicit permission of the originator and are specifically acknowledged. I understand that failure to do this amounts to plagiarism and will be considered grounds for failure in this project and the degree examination as a whole.

Name: Leon Singleton

Acronyms

Abbreviation	Definition
DCS	Department of Computer Science
AVS	Alexa Voice Service
SDK	Software Development Kit
TTS	Text-to-speech
JSON	JavaScript Object Notation
AWS	Amazon Web Services
RSS	Rich Site Summary
RDS	Relational Database Service
SQL	Structured Query Language
GUI	Graphical User Interface
ASK	Alexa Skills Kit
SASSI	Subjective Assessment of Speech-System Interface Usability

Abstract

Speech is considered to be the most natural form of human communication and in recent years has led to the rise of many forms of speech processing technologies. Spoken language dialogue systems, more commonly known as virtual assistants, allow humans to interact with devices in a more intuitive way compared to regular typing, to access information, complete tasks or control devices.

This aim of this project was to build an Alexa Skill for the Amazon Echo that will let visitors or members of the Department of Computer Science at the University of Sheffield converse with Amazon's Alexa to gain departmental related information.

This report provides details concerning the entire implementation of this project, including an extensive literature survey and requirements specification, the design and implementation choices along with an evaluation of the current performance.

Contents

Declaration i

Acronyms ii

Abstract iii

Introduction 1

1.1 Background 1
1.2 Project Aims and Objectives 2
1.3 Constraints 2
1.4 Overview of the Report 2

Literature Survey 3

2.1 History of Speech Enabled Systems 3
2.2 Types of Dialogue System 4
2.2.1 Rule Based Systems 4
2.2.2 Frame Based Systems 4
2.2.3 Reinforcement Learning for Dialogue Systems 5
2.2.4 Information State Approach for Dialogue Systems 6
2.3 Smart Speaker Comparison 6
2.3.1 Amazon Echo 6
2.3.2 Google Home 7
2.3.3 Apple HomePod 7
2.4 SDK Comparison 7
2.4.1 Alexa Skills Kit 7
2.4.2 Google Assistant Actions 9
2.4.3 SDK Conclusion 9
2.5 Evaluating Spoken Dialogue Systems 9
2.6 Related Skills 10
2.6.1 RSS Based Skills 10
2.6.2 Fact Based Skills 11

Requirements and Analysis 12

3.1 Requirements 12
3.2 Implementation Tools 13
3.2.1 Programming Language 13
3.2.2 Database 14

3.3 Testing 15
3.4 Evaluation 15
3.5 Data Collection and Processing 15
3.6 Ethics 16
3.7 Risk Management 16

Design **18**
4.1 System Architecture 18
4.2 Interaction Model 18
4.2.1 Skill Invocation 18
4.2.2 User Interaction 19
4.2.3 Conversational Design 21
4.3 Database Design 23
4.4 Error Handling 23

Implementation and Testing **24**
5.1 The Interaction Model 24
5.1.1 Slots 24
5.1.2 Intents 25
5.1.3 Utterances 27
5.2 The AWS Lambda Function 28
5.2.1 Sessions 28
5.2.2 Data Persistence 29
5.2.3 FAQ Processing 29
5.2.4 Help Options 29
5.2.5 Error Handling 30
5.3 Database 30
5.3.1 Data Collection 31
5.3.2 Full-Text Searching 31
5.4 Testing 32
5.4.1 End-to-end Testing 32
5.4.2 Unit Testing 33

Results and Discussion **35**
6.1 Requirements Analysis 35
6.2 User Evaluation Framework 36
6.2.1 Results 37
6.3 Further Development 38

Summary and Conclusions **40**

Bibliography **42**

Appendices **47**

Appendix A: System Dialogue Flows **48**

Appendix B: SQL Statement Code Examples **51**

Appendix C: User Evaluation Forms **53**

Appendix D: User Evaluation Results **61**

List of Figures

Figure 2.1: A finite-state automaton architecture describing a simple frame-based dialogue 5
Figure 2.2: Expected Return Function 5
Figure 2.3: Alexa Custom Skill System Architecture 8

Figure 4.1: Dialogue flow for obtaining course information 22
Figure 4.2: Database Schema 23

Figure 5.1: Screenshot from the Developer Console illustrating the slot values for a given slot type . 25
Figure 5.2: Screenshot showing the validation rule for a slot type used by an intent 27
Figure 5.3: Screenshot showing the variation of sample utterances implemented for each intent 27
Figure 5.4: Screen capture of the AWS Management console, depicting the Skills triggers and links 28
Figure 5.5: Screenshot showing the retrieval of module related information 31
Figure 5.6: Screenshot depicting the Alexa developer console test simulator 32

Figure A.1: Dialogue flow showing how the system can handle course related requests and ask follow-up questions 48
Figure A.2: Dialogue flow showing how the system can handle lecturer related requests and ask follow-up questions 49
Figure A.3: Dialogue flow showing how a user can ask about a lecturer's modules and receive information concerning them by answering system follow-up questions 50

Figure B.1: SQL Showing how web scraped data was inserted into the database 51
Figure B.2: SQL Showing how manually defined data was inserted into the database 51
Figure B.3: SQL Showing how MySQL Full-Text Search was implemented 52

Figure C.1: Page 1 of the Information Sheet used in the User Evaluation 53
Figure C.2: Page 2 of the Information Sheet used in the User Evaluation 54
Figure C.3: Page 3 of the Information Sheet used in the User Evaluation 55
Figure C.4: Task sheet provided to User Evaluation candidates 56
Figure C.5: Scan of an anonymised Consent Form used by candidates to confirm their participation in the Evaluation 57
Figure C.6: Scan of the Background Questionnaire results completed by candidate 5 58
Figure C.7: Scan of page 1 of the Evaluation Questionnaire completed by candidate 5 59
Figure C.8: Scan of page 2 of the Evaluation Questionnaire completed by candidate 5 60

Figure D.1: Table showing the mean results from the user evaluation attitude statements and their weighting factors 62

List of Tables

Table 3.1: Table containing the project’s functional requirements 13
Table 3.2: Table containing the project’s non-functional requirements 13
Table 3.3: A weighted decision matrix evaluating the strengths and weaknesses of each programming language 14
Table 3.4: A weighted decision matrix evaluating the strengths and weaknesses of each database service 15
Table 3.5: Risk severity ranking from 1 to 5 16
Table 3.6: Risk likelihood rankings from 1 to 5 16
Table 3.7: Risk Assessment 17

Table 4.1: Table covering the design of the principle intents, including descriptions and example invocation utterances 20

Table 5.1: Table covering the built-in intents frequently used during Skill interaction 26
Table 5.2: End-to-end testing example conducted for a dialogue flow concerning courses 33
Table 5.3: Example of unit-testing involving the {LecturerGroupIntent} 34

Table 6.1: Table displaying the completion status of functional requirements 36
Table 6.2: Table displaying the completion status of non-functional requirements 36
Table 6.3: Mean subjective satisfaction ratings 37

Chapter 1

Introduction

This chapter provides background information concerning the project, describing what virtual assistants are and what the Amazon Echo is. In addition, the project's aims and objectives are set out and constraints summarised. Lastly, an overview of this report is included.

1.1 Background

A virtual assistant is a software agent that can either provide a service to a user or perform some type of task. Virtual assistants make use of natural language processing to match user text or voice input to executable tasks. Common services offered by virtual assistants include question answering, playing music and providing conversational commerce. Such technologies are continually improving and developing new capabilities due to machine learning. Virtual assistants are active within many modern devices such as mobile phones, computers and other independent devices such as smart speakers. The latter of which have achieved tens of millions of sales in the United States alone [1]. In recent years, the popularity of virtual assistants has increased dramatically, largely due to the market competition between developers such as Amazon, Google and Microsoft, each striving for the top spot [2].

The Amazon Echo is a series of smart speaker devices offered by Amazon that allow users to interact with the Alexa voice service (AVS). This service encompasses a virtual assistant that is continuously listening to speech whilst awaiting the wake word, 'Alexa'. Once the wake word is detected the device will anticipate a user request before handling it in real time and providing a response. In the words of Amazon, the devices do this "by handling complex speech recognition and natural language understanding in the cloud" [3]. The response provided is in the form of a natural lifelike voice output resulting from Speech-unit technology. The Amazon Echo is designed for domestic purposes and as such, it requires an active power source and a wireless internet connection in order to function.

The Amazon Alexa platform supports third party development of a concept known as a 'Skill', allowing independent developers to add new capabilities to Amazon's existing Alexa speech assistant. The Alexa Skills Kit is a place where users can browse thousands of third party-built Skills before choosing which to enable on their devices. An Alexa Skill typically provides a new feature to a user's associated Amazon Echo device, allowing users to interact in new ways that seem a-typical to other virtual assistants such as playing a custom trivia game.

1.2 Project Aims and Objectives

The aim of this project was to design, develop and evaluate an Alexa Skill. The Skill will let visitors or members of the Department of Computer Science at the University of Sheffield converse with Amazon's Alexa to gain departmental related information. For example, students might want to find out more information about courses offered; visitors might want to ask the location of academic staff offices or know how to contact a specific academic.

This project was a joint project with two other members focusing on other areas, one on the extensive analysis of the User Requirements and Evaluation (**P1**) and the other focusing on Data collection and Processing (**P3**). The primary focus of this project (**P2**) was to create a seamless user interaction in order to provide a viable alternative to providing information over manually searching the departmental related webpages. The Skill needed to ensure the correct information relating to a request was portrayed accurately in a formal but simple manner. The overall goal for the project was to build a Skill worthy of Amazon certification so that the Department of Computer Science could have a new channel to get information about itself across to its "customers".

1.3 Constraints

The most significant constraint was being able to test the Skill at a large enough scale in order to make significant interaction improvements. This was because of the time constraints imposed on the project. Regardless of these time constraints though, Amazon web services imposes restrictions on usage limits within its 'Free Tier Service' [4] which meant that testing could not have been conducted with more than a handful of users anyway.

Further constraints on the project related to the user interaction of the Skill as a result realising of the limitations of using the Alexa Skills during development. One of these was having to launch the Skill before interacting in a manner that requires a series of dialogue turns to process a user's request. An example scenario would be a user interacting with the intention to learn about courses that can be undertaken at a master's level but asking their question in such a manner where further clarification is required (see Figure 4.1). Another constraint was having to provide carrier phrases such as "search" during interactions where the input was less predictable.

1.4 Overview of the Report

This report contains six further chapters, which outline the various stages that were involved in the project. Chapter 2 includes all material relating to the literature survey carried out in preparation for this project. This includes a brief history of spoken dialogue systems, comparisons between smart speaker devices, evaluation methods and a review of related Skills. Chapter 3 introduces the requirements outline, an exploration of the possible implementation technologies, testing, evaluation and a detailed risk analysis. Chapter 4 provides an overview of the system design and justifications of the design choices made. The implementation and testing of the project are covered in chapter 5, in particular the challenges and constraints of the project are highlighted. In chapter 6 the main findings of the project and the potential future developments are discussed. Finally, chapter 7 concludes with a summary of the project-related achievements and discoveries.

Chapter 2

Literature Survey

This chapter provides a brief overview of the milestones involved with the development of virtual assistants along with a look at their designs. It also provides an in-depth look at the latest commercial smart speaker devices and the development kits on offer. To conclude, common evaluation techniques are covered alongside a review of some related university-based Skills currently available.

2.1 History of Speech Enabled Systems

The first influential tool towards modern day virtual assistants was the IBM Shoebox [5] launched in 1961. This device was capable of recognising 16 spoken words and the digits 0-9 to perform arithmetic calculations, laying the foundations of speech recognition. The next revolutionary tool, the “DRAGON” [6] system developed in 1975 utilized “a probabilistic method for temporal pattern recognition” involving a Markov network that could analyse entire word sequences rather than being limited to individual words. Using inspiration found from the success of the “DRAGON” system, ‘HARPY’ [7] was created at Carnegie Mellon University, Pittsburgh, Pennsylvania which also used a Hidden Markov Model involving a finite state network. It improved upon the “DRAGON” system by introducing “a mathematically tractable model” and “speech-dependent heuristics”, which in turn led to verification of recognized word sequences against reference patterns. These new techniques reduced computation times concerning the matching of strings as well providing increased accuracy of word sequences [8]. However, a limitation due to hardware at the time not being powerful enough to address the problem of word segmentation meant that users were forced to enunciate words individually with clear pauses in between for successful recognition [9]. Leading on from these, more practical uses of digital speech recognition became commercially available and packaged as a feature of the personal computer (PC), as offered by Microsoft, IBM, Philips and Lernout & Hauspie in the 1990’s. However, the next big advancement came with the Sphinx-II in 1992 which was the first system able to handle “speaker independent, continuous speech recognition”. By this time, commercial speech recognition software was said to have a larger vocabulary than the average human [10]. The late 2000’s witnessed a transition from using the traditional approaches such as Hidden Markov Models combined with feedforward artificial neural networks to deep learning methods, which can be found in modern day virtual assistants. The first such revolutionary device to implement deep learning was Siri [11], a smartphone assistant introduced by Apple in 2011, which ushered in the age of the voice-enabled virtual assistant, as we know it today.

2.2 Types of Dialogue System

A dialogue system [12] describes the process of conversing with a computer system using the human voice. A typical dialogue between humans consists of many turns, with each human providing a contribution to the overall dialogue experience in the form of one or more sentences or words. Dialogue systems of varying complexities aim to simulate this turn based dialogue experience in order to respond to user requests.

2.2.1 Rule Based Systems

Rule based dialogue systems involve the use of a rule base, an inference engine and a temporary data store [12]. A well-known rule-based system is ELIZA, first created by Weizenbaum in 1966 [13]. This system attempts to match a user's conversational input to a matching rule contained from within its knowledge base. The aim is to form a match with a pattern/transform rule that will allow the system to output a response that is relevant to the current domain. It does this by using a ranking system to rank keywords contained within the input, before matching against a rule relevant to the input. For example, the user might say "my boyfriend made me come here" and ELIZA would respond "Your boyfriend made you come here", demonstrating the use of a transform rule based on the keywords "my" and "me". If there are no matching rules obtained Eliza will respond with one of its pre-programmed non-committal responses contained within its data store. This approach enabled ELIZA to provide the illusion of containing some degree of "conversational intelligence". However, when conducting lengthier conversations, it becomes apparent this is not the case. Regarding this project, a rule-based system could provide relevant responses to user questions, but it lacks a solid control structure concerning the flow of dialogue.

2.2.2 Frame Based Systems

Frame based dialogue systems make the use of a domain ontology (a knowledge framework of common concepts relating to a specific domain), specifying the kinds of intentions/actions that can be handled by a user utterance [12]. Frame based dialogue consists of one or more frames and a series of slots that define the values accepted by a frame. Frames form the control structure of the dialogue and slots allow for variation in the intent. The aim is to fill the necessary slots related to a given frame so that the system can perform the desired response or proceed to the next frame within the dialogue. If the required slots have not been filled, a frame-based system will typically try to obtain the missing slot information by asking questions that explicitly ask for them. In this type of system, it is therefore possible to build upon an intention by specifying the required slot information sequentially. This is represented in Figure 2.1 which recommends a dog breed based on the features specified (assuming the slot values are valid). Furthermore, this is a typical example of a single-initiative finite-state dialogue architecture which has the advantage of always knowing what question the user is currently answering. This means that the system can build a language model tuned to the possible answers that the user provides which makes language understanding easier. Such finite-state systems will also accept universal commands that can be said anytime throughout the dialogue. Common universal commands include "help" which provides a help message, "repeat" which repeats the previous message and "start over" which returns the user to the main start state.

Modern task-based dialogue systems like Apple's Siri, Amazon's Alexa and the Google Assistant implement a frame-based approach [12]. These systems either ask questions of the user to restrict them to a pre-determined frame based-dialogue path or obtain a question from a user before leading them down a relevant frame-based dialogue path. They work upon the same principles as described above providing a clear dialogue structure to control the flow of conversation.

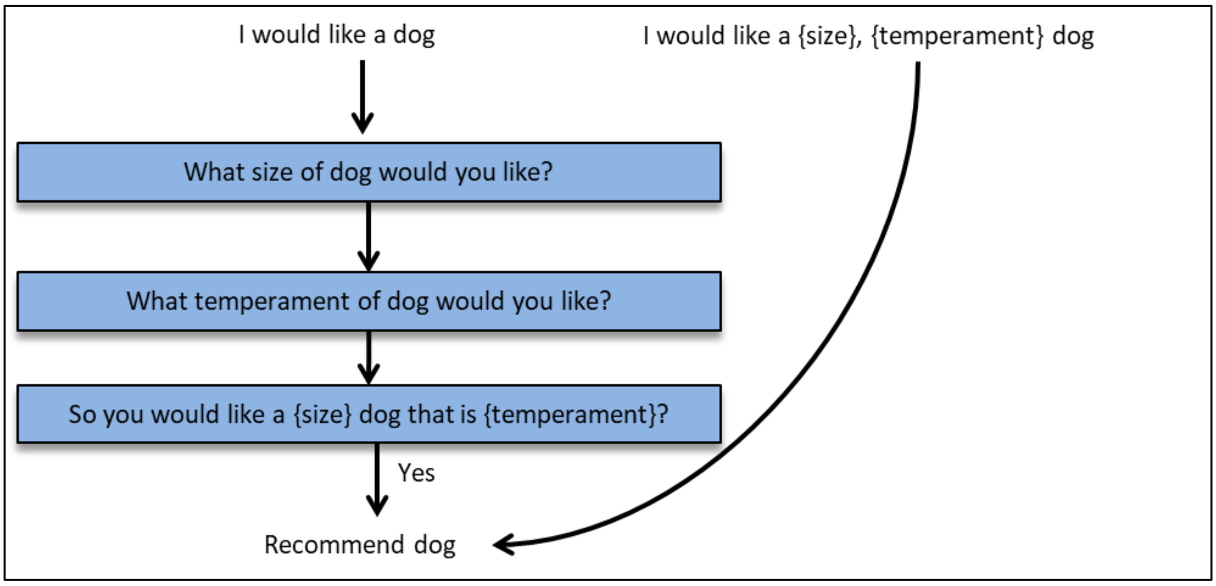

I would like a dog
I would like a {size}, {temperament} dog
What size of dog would you like?
What temperament of dog would you like?
So you would like a {size} dog that is {temperament}?
Yes
Recommend dog

Figure 2.1: A finite-state automaton architecture describing a simple frame-based dialogue

2.2.3 Reinforcement Learning for Dialogue Systems

The objective of Reinforcement-Learning is to produce the optimal actions and behaviours by choosing an action from an action space that maximises the expected total reward. An action space contains the dialogue acts that the system can give as a response. The reward values that are assigned to each action are a result of analysing previous user feedback/interaction. Reinforcement learning in multi-turn dialogue systems commonly use a policy-based approach [14] that search for the optimal policy(action) based on a cumulative value return for every possible state, this can be modelled using the following expected return function (fig 2.2).

$$Q_\pi(b_t, a_t) = E_\pi\ (R_t | b_t, a_t)$$

Figure 2.2: Expected Return Function

This function consists of a continuous markov decision process that outputs the optimal policy π by finding the maximum expected return in each belief state vector b_t from the belief state space B, for each time step. It does this by choosing an action a_t from the action space A and observing a reward r_t that is produced by the environment. The Q-value function is defined as the expected return R of the state-action pair (b_t, a_t) under the policy π. In value-based systems the optimal policy can be reached by greedily taking the action which maximises the Q-value function. Here, the belief state can be defined as “a vector representing a probability distribution over the different goals, intents and histories that occur in the dialogue” [14]. In a typical slot filling spoken dialogue system this is defined by the ontology; the structured database of entities that can be retrieved by talking to the system where each entity consists of one or more slots.

Reinforcement learning is typically the more favoured dialogue system approach due to its ability to output the optimal response at every input and learn from previous data. The drawback though is that reinforcement learning requires a large amount of data and previous user interactions to work successfully, which is why for the scope of this project it was considered unsuitable.

2.2.4 Information State Approach for Dialogue Systems

The Information State Approach to building dialogue systems is a method that helps to formalize the theories of dialogue modelling [15]. Doing so makes the process of implementing a dialogue system more straightforward through the use of a clear dialogue structure. The dialogue theory itself is built using five different aspects, which can be categorized as either being static or dynamic. The static aspects consist of the informational components and the formal representations. The informational components correspond to the domain knowledge and conventions of the dialogue. The formal representations describe these components, e.g. as lists, logic structures, records etc. The dynamic aspects consist of the dialogue moves, update rules and the update strategy. The dialogue moves are abstractions of the possible information that can be exchanged between the system and user. Triggering of the dialogue moves results in the update of the information state, which stores the dialog history, including the previous interactions and the question under discussion. A set of update rules has to be defined which control the updating of the information state. These update rules depend on conditions being fulfilled that relate to the current information state and the previous dialogue moves. The update strategy simply defines which of the applicable update rules should be applied at a given stage in the dialogue, e.g. "pick the first rule that applies". The informational state approach addresses the deficiencies found in dialogue systems that conceive the dialogue structure as a finite state machine by allowing more flexible interaction and improved scaling to larger tasks.

2.3 Smart Speaker Comparison

This section describes the technical aspects of the most popular smart speakers currently available on the market, whilst providing comparisons and contrasts between them. Furthermore, views are expressed as to why particular speakers are more suitable than others for development.

2.3.1 Amazon Echo

The Amazon Echo is a series of smart speakers offered by Amazon that first became available in the United States on June 23, 2015 [16] and on 28 September 2016 in the United Kingdom. They make use of the cloud-based Alexa Voice Service to process human commands and requests. Each Amazon Echo share the common functionality of being able to stream music, control smart home devices, connect external speaker devices and manage audio calls between Echo devices and the Alexa app.

The Echo Dot (3rd Gen) comes available at a retail price of £49.99 [17], which provides an affordable option for consumers to add Alexa to any room within their home. The Echo (2nd Gen) contains the same internal hardware as the Echo dot but improves upon its sound quality using a dual 2.5" woofer and 0.6" tweeter compared to the Dot's 1.6" single speaker. These improvements in sound quality come at the higher cost of £89.99 [18]. The Echo plus (2nd Gen) yet again improves upon the sound quality with a dual speaker setup containing a 3" woofer and 0.8" tweeter. However, the most notable addition included within the Echo plus is a built in Zigbee smart home hub, it comes priced at £139.99 [19].

The Echo also comes in the display variants of the Echo Show and Echo Spot first released June 28, 2017 and 27 September 2017 respectively [16]. These devices bring the additional functionality of being able to display pictures and stream video through a series of new API's available to developers. They both also contain front facing cameras allowing for video calls between Echo devices. The Echo Spot comes with a 2.5" screen and 1.4" speaker and is priced at £119.99 [20]. The Echo Show (2nd Gen) boasts a much larger screen of 10.1" and dual 2" drivers with a passive bass radiator, coming priced at £219.99 [21].

2.3.2 Google Home

The Google Home are a series of smart speakers that incorporate the Google Assistant which is arguably better at answering questions in comparison to Echo devices due to its "extensive searching Skills" [22]. It's two devices the Google Home Hub (November 4, 2016) and Google Mini (October 19, 2017) come priced at £129 and £49 respectively, providing viable competition to the closely priced Echo devices. However, the Google Home Hub devices are lacking when it comes to integrations with smart home technology and Google's own services such as Google Calendar [23].

2.3.3 Apple HomePod

The Apple HomePod released February 9th, 2018 comes priced at £319, a staggering amount compared to other smart speaker devices. This large price tag is due to its focus on sound quality as it aims "to deliver precision sound that fills the room, coming equipped with 7 tweeters and a 4" subwoofer [24]. The HomePod utilises Siri as an intelligent assistant that has good integration with IOS devices, but it too is lacking in integration with the control of smart home devices [25]. Furthermore, Apple do not provide tools allowing developers to create their own voice experiences, making this a non-viable product for this project.

2.4 SDK Comparison

Both Amazon and Google offer sophisticated software development kits (SDK) allowing anyone to build custom voice interaction with a frame-based dialogue system approach. This section provides a comparison between their two respective approaches.

2.4.1 Alexa Skills Kit

The Alexa Skills kit allows developers to create their own custom Skills for Amazon Echo devices. Skills can be built to provide users with many different types of abilities and can generally be categorized as one of the following [26]:

- Smart: Enable hands-free control of smart devices
- Flash Briefing: Delivers pre-recorded audio clips or text-to-speech (TTS) updates
- Video: Allows video content to be consumed on other devices
- Music: Enables consumers to stream and control music
- List Skill: Enables customers to manage lists for any purpose or occasion
- Custom: Allows a custom interaction model not covered by the above API's

A custom Skill model allows developers to design and build tailored user interaction through the use of an interaction model [27]. A developer first begins by defining an invocation name and the intents, which represent the possible actions a user can do with a Skill. An intent then needs to be mapped to one or more utterances in order to invoke it. For example, the utterance "tell me about the Speech Processing module" would be mapped to the getModuleDescription intent. An utterance can be accompanied with a slot, allowing developers to parse parameters from the user input for endpoint logic. A developer may also want to provide a visual component alongside their Skill to enhance the user's voice interactions in the form of an information card or through pictures/video. Finally, the

Skill must be connected to an endpoint (cloud-based service) capable of handling the user intents before sending the requested data back.

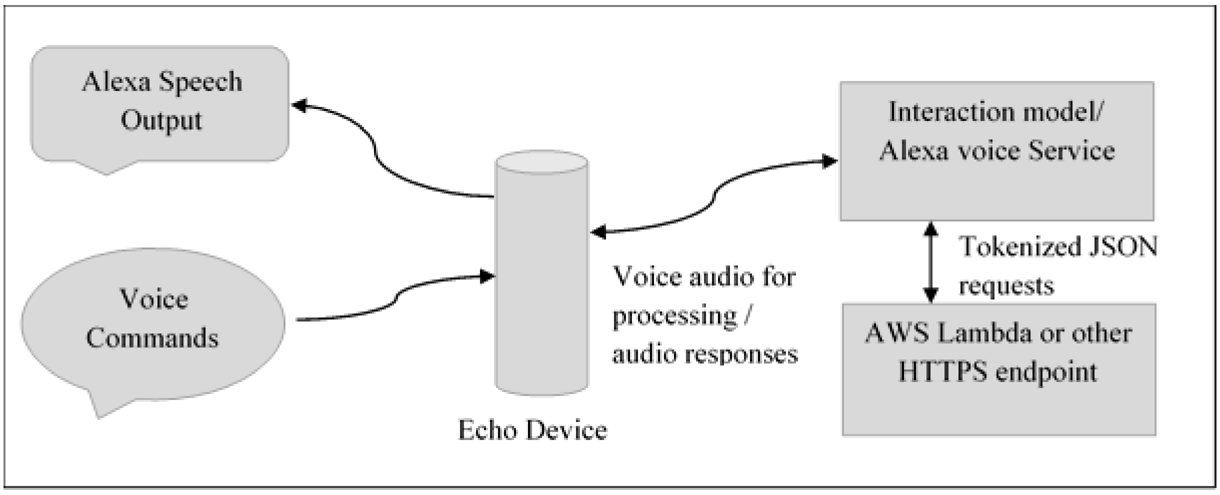

Alexa Speech Output
Voice Commands
Echo Device
Voice audio for processing / audio responses
Interaction model/ Alexa voice Service
Tokenized JSON requests
AWS Lambda or other HTTPS endpoint

Figure 2.3: Alexa Custom Skill System Architecture

Figure 2.3 demonstrates the relationships between different processes concerned with the Alexa Custom Skill System Architecture [27]. Firstly, a user speaks to their Echo device, using trigger words so that their device is aware that it is being addressed alongside a request. This request is then sent to the Alexa Voice Service, which handles speech recognition, turning the speech into a JSON (JavaScript Object Notation) encoded text document where the intent and any associated contextual parameters into have been converted into tokens [28]. The tokenised JSON is then forwarded to the Amazon Web Services (AWS) lambda function to perform any associated processing (e.g. database lookup or API requests) before returning a response JSON back to the Alexa Voice Service containing the text that should be output back to the user. Upon receiving the response JSON, the Alexa Voice Service uses text-to-speech technology to speak the response to the user.

Amazon provides a number of helpful guides [29] that allow developers to learn how to develop the different Skill types as mentioned in section 2.4.1. These tutorials range from beginner to advanced allowing people of little programming knowledge to slowly ease into Alexa Skill development. Other methods of learning that Amazon endorse include webinars and training courses offered by third party education organizers [29]. These resources allow developers to go through the different stages of developing and publishing an Alexa Skill using a template. This consists of:

1. Creating the interaction model using the Alexa Developer console by specifying the intents, slots and utterances.
2. Setting up a lambda function using Amazon Web Services and configuring its triggering to be “Alexa Skills Kit”.
3. Writing the program logic to the lambda function to handle the intents in the interaction model.
4. Connecting the interaction model to the Amazon Web Services lambda function using an endpoint.
5. Testing the Skill using the methods described in section 3.3.
6. Submitting a Skill for certification by Amazon before publication.

These templates offered by Amazon are a good way of learning the different Alexa capabilities supported by the Alexa Skill Kit SDK because each one demonstrates the use of a different Skill capability. As an example, Amazon has a template for a high low game [30] that focuses on the use of persistence attributes and the persistence adapter in the Alexa Skill Kit SDK. These templates also provide a good way for developers to build Skills quickly as they can simply customize the existing code logic to suit their requirements.

2.4.2 Google Assistant Actions

Google Assistant Actions allows developers to extend the functionality of the existing Google Assistant application offered by Google [31]. Similarly, to the Alexa Skills Kit, Google's service uses intents that are mapped to corresponding user utterances. These intents are processed by something known as a fulfilment (a service, app, feed, conversation) that carries out a corresponding action. An action specifies the interaction that a developer has created for the intent. Unlike Alexa Skills, users do not need to install anything to invoke developer created actions on the Assistant. Once certified an action becomes an integrated part of Google Assistant. This means that anyone using the Google Assistant could discover your action if they ask a question that relates to an action that your application can handle [32].

To begin creating a custom Actions SDK project a developer must create a project using the Actions Console. Upon creating a project, a developer can choose to use one of the pre-built templates to get started quickly or begin with a fresh Action package, the more favoured choice for developers who want more control over their actions' language processing. The Actions that a developer defines act as the entry points into the application that are used for Skill invocation and discovery. They are declared in a JSON file format, which are eventually uploaded to the developer's project for testing or for submitting an Actions Project for approval. In order to define Actions within an Actions project a developer first creates the intents that describe how the Action is invoked. These intents require what Google call "training phrases" [33] (utterances) in order to be triggered, variations of the provided phrases are handled by the DialogueFlow Agent, the conversational platform allowing developers to design and build conversational interfaces for Actions. Alongside, the intents a developer must define the fulfilments that receive the user speech input and associated parameters to process intent requests and return their corresponding responses. Fulfilments are written using the Actions SDK for either the Node.js or Java client library and must be hosted by a cloud-based platform that can support HTTPS requests and responses. Upon defining of the fulfilments, a webhook between the cloud-based platform and the Actions interface needs to be created using the cloud function URL to associate intents with fulfilments [34].

2.4.3 SDK Conclusion

The two SDK's both work in a similar manner that draws upon a frame-based dialogue approach that maps user intents to actions with the additional capability of slot elicitation. They both also operate as cloud-based services that manage the intent requests and associated data retrieval. As such either one of these SDK's could have been chosen for this project, allowing the scope for further future development on an additional platform. However, for this project the Alexa SDK has been chosen as a result of Amazon's more extensive development documentation and examples.

2.5 Evaluating Spoken Dialogue Systems

Spoken Dialogue systems are usually evaluated using performance measures of factors such as recognition accuracy, error recovery, response time and situational awareness [35]. These methods are all involved in objective evaluation of a system. Recognition accuracy is based on word error rate and is expressed as a percentage. Error recovery in a system is assessed on how easy it is for a user to undo actions triggered by previous spoken commands. The response time of a system is important in order to assess the usability of a system when compared to alternate means. This is defined as the time it takes from the end of a spoken utterance to the beginning of the triggered action. The situational awareness of a system is an important factor in concluding whether the system has good knowledge of the speech domain. Users of systems have expectations of the different commands that they can provide, and a situationally aware system should be able to handle many commands to reach the same state. Furthermore, the situational awareness of a system can be evaluated based on its current state. This is done by counting the number of times a user utters a command in a context which is not accepted by the current state.

Subjective evaluation methods of spoken dialogue systems involve the use of human test subjects interacting with the system [35]. Subjective measures are based on factors such as intelligibility of speech output, user-friendliness, intuitiveness, level of difficulty and impression of response time. The most important factor though in a subjective evaluation is whether the task can be completed. This alongside the time taken to complete a task are major factors in a test subject’s general impression. These impressions help to give an indication of how successful a given system performs.

A general framework for evaluating spoken dialogue systems is the PARADISE framework [36]. This framework addresses the limitations associated with calculating the performance of a system’s sub dialogues as well as entire dialogues, correlating performance with external validation criteria and normalizing the performance of task complexity. Briefly put the PARADISE framework derives a performance function for a task domain within a dialogue system, which is based on estimations of the relative contribution factors leading to user satisfaction. From here, the performance function of one or more other dialogue agents capable of operating in the same task domain are compared with the calculated performance function. This determines the best performing dialogue system operating in a specified task domain.

2.6 Related Skills

There are several university-based Skills available from the Amazon Alexa Skill Store. The ones that are available are predominantly targeted at universities in the United States. This is not alarming considering the market penetration of smart-speaker devices across consumers in the United States (41%) [37], thus it seems logical that universities would recognise this as a popular new means to share information.

2.6.1 RSS Based Skills

Several universities currently offer Skills that allow a user to obtain news relating to their university as a part of their flash briefing. A flash briefing provides an overview of the news headlines and in Amazon’s words “a flash briefing typically becomes a part of a customer’s daily routine” [38]. As such a flash briefing Skill is a good way for universities to incorporate their news into an interested person’s Alexa news feed. A university-based flash briefing Skill is very easy to create because most universities commonly incorporate an RSS (Rich Site Summary) feed into their website content to distribute news, hence the feeds are regularly updated. This means that a developer wanting to create a flash briefing Skill would only have to link an RSS feed URL to their Skill for it to work. Most universities make these feeds readily available on their websites and even allow the feeds to focus on specific subject areas, which can be seen in the University of Sheffield’s case [39].

2.6.2 Fact Based Skills

Fact based Skills are also offered by some Universities allowing users to ask University based questions and retrieve an answer. There are two types of fact-based Skills that are offered. The first simply extends upon Amazon’s fact Skill tutorial [40] where a user can give an utterance such as “tell me a fact” to obtain a random fact about a University that is selected from a wider list of facts. The second type of University fact-based Skills on offer allows more specific questions in order to obtain tailored information. The most advanced of these Skills is offered by Leeds Beckett University [41], which allows users to get information about courses currently offered at the University and takes them through the UCAS application process of applying for a course. In this Skill users can obtain course information in multiple ways:

- They can either tell the Skill how many UCAS points they have, in which case the Skill will recommend courses that either match or are below the given amount.
- They can specifically ask about the course via name i.e. “Tell me more about Computing”
- They can specifically ask about the course via code i.e. “Tell me about N420”

This approach demonstrates a good consideration of the various way’s users might interact with the Skill, thus providing a larger scope for interactivity.

The capabilities of the Skills offered by other universities is very limited, which are mostly confined to providing broad non-specific information. They also lack the common attributes associated with good dialogue systems such as multi-turn interaction, universal commands and error handling. All of these were problems that this project aimed to address.

Chapter 3

Requirements and Analysis

In this chapter the project requirements are defined in terms of functional and non-functional requirements. An analysis of the potential implementation tools that can be used to develop the system is provided including justifications of the tools chosen. A discussion of testing and evaluating the software is discussed as well as a brief statement regarding data collection and processing. Finally, the ethical concerns related to the project are considered and potential risks relating to the project are described with accompanying prevention measures.

3.1 Requirements

The final Skill needed to cater to both current students and prospective students that each have different interests regarding interaction. To this effect requirements were formed as a result of discussion between **P1** and **P2** in the initial stages of the project, identifying the interactions and scenarios that seemed most plausible. They were then expanded upon as work progressed and the full scope of the project was understood. A weighting code was used to assign weights to requirements in terms of their desirability, as witnessed in Table 3.1 and Table 3.2 using the MoSCoW method. These weighting codes and their corresponding description are as follows:

Must Have (M): requirements that are critical to the delivery of the system and its success within the delivery time frame.

Should Have (S): requirements that are important but are not necessary for successful delivery and not as time critical as “must have” requirements.

Could Have (C): requirements that are desirable and could improve the user experience but are not necessary, only to be included if time permits.

When acknowledging the different requirements associated with the project, separate categories were created in the form of functional and non-functional requirements.

Description	Weighting
The Skill must be able to be invoked easily	M
The Skill must have help options for each section of user interaction	M
The Skill must allow a range of utterances to be used to invoke intents	M
The Skill must not have long-winded responses	M
The Skill must elicit slot information when required slots have not been filled in the dialogue management	M
The Skill must provide course and module related information	M
The Skill must provide information relating to placements and University life	M
The Skill must include information relating to lecturers	M
The Skill must be able to handle non-system requests gracefully	M
The Skill should be able to repeat information if requested	S
The Skill should be able to maintain necessary attributes across sessions	S
The Skill could allow users to ask directions to buildings around Campus	C
The Skill could allow users to get lecture timetable information	C
The Skill could display information visually for display enabled devices	C
The Skill could allow users to manage a list of their favourite modules	C

Table 3.1: Table containing the project's functional requirements

Description	Weighting
The Skill must be easy to use and contain contextual awareness	M
The Skill must ensure that data from responses is not misrepresentative	M
The Skill must be efficient in its response times	M
The Skill should be easy to manage in order to allow future additions	S
The language of the Skill should be formal in its responses	S

Table 3.2: Table containing the project's non-functional requirements

3.2 Implementation Tools

The Alexa Skills Kit was chosen as the development option of choice for the project because of the findings from the Literature Survey (Section 2.4). When it comes to hosting a Skill, the logical choice was to use Amazon Web Services lambda function which is "a compute service that lets you run code without provisioning or managing servers" [42]. As a cloud-based web service it is preferred over other independent web services because of its extensive documentation as well as its compatibility with other AWS technologies such as its Relational Database Service (RDS) [43].

3.2.1 Programming Language

An AWS lambda function is capable of supporting Node.js, Java, Go and Python, but for this project the Alexa Skills Kit only supports Node.js, Python and Java. The weighted decision matrix in Table 3.3 shows how a suitable programming language was chosen by assessing strengths and weaknesses using five crucial factors. Online documentation was given the highest weightings as this would have an impact on the overall development time as less time would be spent solving problems. Library support and ease of use were given the next highest rating since a major aspect of the project will be integrating a database. Finally, performance and prior knowledge were ranked as the lowest priorities. Performance is not really a factor when it comes to the programming language because the AWS lambda function handles compute instances in real time [42]. Prior knowledge was given a low weighting because of previous experience with all three programming languages.

Python was chosen as the choice of programming language because it scored highest in the decision matrix, scoring high amongst all the factors. Although, Node.js scored similarly to python in terms of its documentation and performance, it became apparent during experimentation with the two languages that integrating a database into a Python solution was easier. Furthermore, Python offers more extensive libraries when it comes to data mining and data analysis libraries that **P3** will make use of, so using a common programming language is important for potential future integration.

		Programming Language					
Criteria	**Weighting**	**Node.js**		**Python**		**Java**	
		Score	**Total**	**Score**	**Total**	**Score**	**Total**
Online Documentation	4	5	20	5	20	3	12
Ease of Use	3	4	12	5	15	4	12
Performance	2	5	10	5	10	5	10
Library Support	3	4	12	5	15	4	12
Prior Knowledge	2	3	6	4	8	3	6
	TOTAL:		**60**		**68**		**52**

Table 3.3: A weighted decision matrix evaluating the strengths and weaknesses of each programming language

3.2.2 Database

Amazon web services offers support for many different database services but when deciding upon the best option the following three were considered: MySQL, DynamoDB and PostgreSQL. DynamoDB is a NoSQL based database endorsed by Amazon as the best choice when it comes to developing Alexa Skills as many of its guides are centred around it [44]. MySQL and PostgreSQL were considered because they are both supported by Amazon Relational Database Services “free tier” [45].

The querying method was given the most importance when evaluating each service as SQL (Structured Query Language) databases allow data to be stored in easily managed tabular structures that support linking of data using table relations. Prior knowledge and online documentation were both important considerations due to time involved with learning how to use an unfamiliar NoSQL based database service. Administrative tools were an important factor because they would concern the time taken to set up the initial database. Data security and reliability was not deemed to be a huge concern because AWS supports automated back-ups and point-in-time recovery [46].

The result of the decision matrix (Table 3.4) shows that MySQL was the best database service and hence it was chosen. The main factor leading to this choice was the desire for a SQL based database structure. The factor differentiating MySQL from PostgreSQL is its dedicated administrative tool MySQL Workbench that provides visual tools to “design, model, generate and manage databases” [47].

		Database Service					
Criteria	**Weighting**	**MySQL**		**DynamoDB**		**PostgreSQL**	
		Score	**Total**	**Score**	**Total**	**Score**	**Total**
Online Documentation	4	4	16	5	20	3	12
Administrative Tools	3	5	15	2	6	3	9
Data Security/Reliability	2	5	10	4	8	5	10
Querying Method	5	5	25	2	10	5	25
Prior Knowledge	4	4	16	2	8	4	16
	TOTAL:		**82**		**52**		**72**

Table 3.4: A weighted decision matrix evaluating the strengths and weaknesses of each database service

3.3 Testing

Testing of the finished Skill involved manual unit testing and end to end testing using the Alexa Developer Console and AWS lambda console [48]. Manual unit testing made use of either the Developer Console Alexa Simulator or an Amazon echo device associated with the Skill. These tests were conducted to ensure dialogue flow Skill sessions were maintained and utterances were correctly mapped to intents. Unit testing took the form of inputting JSON requests either directly into the Developer console or by configuring a custom test event within the AWS lambda console, to confirm the resulting JSON output was as it should be.

3.4 Evaluation

Evaluation of the system involved candidate user's interacting with an Alexa enabled device where the Skill had been installed, in order to simulate a natural interaction environment. Collecting feedback took the form of a survey/questionnaire which candidates were instructed to complete according to certain criteria. Furthermore, the survey allowed candidates to rate different aspects of the system using an appropriate scale and provide more general written feedback opportunities. These evaluations helped to determine whether the system acted responsively as would be expected and whether the initial requirements/project aims had been met. Analysis of the typical user interactions with the system is carried out in more extensive detail in **P1**.

3.5 Data Collection and Processing

Much of the information that this project is concerned with is readily available within the DCS intranet webpages. Thus, one option of data collection was to use web scraping to extract the information relating to a user's intent at run-time. This is considered the optimal approach as it would allow for large amounts of data to be accessible by the application. However, this method is not achievable within the scope of this project due to the aforementioned time constraints and the primary focus of this project (**P2**). Project **P3** examined the possible methods of data collection in detail and provided a solution that was able to web-scrape large amounts of data for Skill interaction. Where it was not feasible to use web scraping, data was collected manually and stored within the pre-defined database. This method allowed for more control when it came to the data collection process as the data was optimised for use in speech communication.

3.6 Ethics

Since the evaluation of the system required human participants, an ethics application was submitted as per the University of Sheffield Guidelines [49]. This application had to take into consideration any sensitive personal data that may be in use during the project so that there were no breaches in the universities General Data Protection Regulations. To this extent candidates had to sign a consent form. In addition, candidates had to accept that the data used by the system may not be wholly correct, or answers provided may not be relevant to the context of a question and could result in inaccurate representations of the department and its people.

3.7 Risk Management

Risk management is an important aspect of projects. It identifies, analyses and mitigates associated risks through planned preventative measures. If risk management does not take place, unwanted events or situations may occur that can lead to missing features or in the worst case, failure of the project. In this section risk management of the project has been carried out in the form of a risk assessment.

Table 3.5 and Table 3.6 are incorporated within the risk assessment to help analyse each potential risk in terms of its likelihood and severity. To do this a likelihood and severity rank between one and five is assigned to each risk, these rankings then correspond to a description.

Level	Likelihood
1	Improbable
2	Remote
3	Possible
4	Probable
5	Almost certain

Table 3.6: Risk likelihood rankings from 1 to 5

Level	Severity
1	Negligible
2	Minor
3	Moderate
4	Major
5	Critical

Table 3.5: Risk severity ranking from 1 to 5

Table 3.7 provides an evaluation of the risks identified with developing this project. The likelihood rankings in this assessment indicate that the chance of these risks occurring is not likely as no risk has been assigned a value greater than three. On the other hand, the severity rankings indicate that the potential hinderance these risks could have on the project is significant as the lowest severity ranking is a three. Therefore, it is of great importance that the provided prevention measures are followed up on closely during software development.

Description	**Likelihood**	**Severity**	**Prevention Measure**
Lack of Development Knowledge	2	3	Ensure that a good knowledge of the implementation technologies is obtained in advance.
Inadequate Completion Time	3	3	Ensure that there is a clear and detailed plan providing a list of tasks that must be completed on a weekly basis.
Inadequate Performance/Quality	2	4	Ensure regular testing of the system is done throughout the implementation stages.
Alteration/Addition of requirements	3	3	Ensure that the project requirements have been clearly defined prior to implementation.
Unsuccessful Ethics Application	1	5	Liaise closely with those responsible for ethics approval and prepare an ethics application with plenty of notice.
Poor Code Quality	2	3	Ensure the code is well written and documented so that future additions are easy to implement for those unfamiliar with the project.
Code Corruption	2	5	Ensure code is backed up regularly using a cloud-based version control.
Data Corruption	2	5	Ensure the data is regularly backed up and stored using a cloud-based solution.

Table 3.7: Risk Assessment

Chapter 4

Design

This section covers the design considerations that were made leading up to the implementation. This includes the design of the interaction model including related methods of invocation and conversational dialogue flows. Also provided is the design of the database schema and a discussion of error handling.

4.1 System Architecture

The Skill system architecture follows that associated with a custom Skill as outline in section 2.4.1. Once a user has provided a valid intent to the interaction model, a JSON request containing the user's speech is sent to the AWS lambda function which handles the processing of the Skill's intents. If the information requested is obtained in the database, then the AWS lambda function conducts a query request to form a JSON response containing both the speech response and the visual component. This JSON is then returned to the Alexa voice service where the speech response is spoken to the user and the visual component displayed.

4.2 Interaction Model

The Skill interaction model [50] consists of the intents, sample utterances, slots and dialogue models that map the user's spoken input to the intents that the Skill's cloud-based logic can handle. The sample utterances are the phrases that users can say to invoke the intents, hence each intent is mapped to several different utterances to allow variations in the user interaction. The slots are the optional arguments that can be supplied in the utterances to pass values to the intent from the spoken speech. An example of this would be the user asking "Tell me about the course Computer Science" where the words "Computer Science" would be an acceptable slot value contained within the custom slot-type {CourseName}. The dialogue models are the structures that define the multi-turn conversational properties of the Skill. These dialogues allow Alexa to ask a series of follow-up questions and prompts to fill the slot values required to fulfil an intent.

4.2.1 Skill Invocation

The consideration of the Skill Invocation was an important part of the design process in relation to how users would perform interactions. In order to use a custom Alexa Skill an invocation name must be provided by the user so that an intent can be associated with it. The invocation name must contain a minimum of two words and meet a stringent set of requirements outline by Amazon [51]. This name must be easy to remember since it is the phrase that the user will say each time they interact with the Skill. In addition, after the Skill is certified and published its invocation name cannot be changed which places more emphasis on the importance of getting the name right.

There are three different ways that users can use an invocation name in order to start using a custom Skill [51]. The first method, known as the “one-shot” method involves the user invoking the Skill with a particular request such as “Alexa, Ask Sheffield Computer Science for a summary of the module Web Technologies”. By combining the invocation name with a request, this approach allow users an easy way to ask a quick question without having to launce the Skill. The second method involves the user invoking the Skill without a particular request using a phrase such as “Alexa, open” or “Alexa, launch” followed by the invocation name. This method allows all subsequent user interactions with the Skill to be performed without the need for the prefix “Alexa” to be said at the beginning of an utterance. Furthermore, this method provides the scope for interesting dialogue with the Skill as it can handle turn based dialogue, a constraining factor of the first method. Lastly, the third method involves invoking the Skill using just the invocation name, for example “Alexa, launch Sheffield Computer Science”. This is effectively the same as the second method in that it performs a launch request to begin an instance of the Skill.

4.2.2 User Interaction

The intents that were identified during the design process were developed as a result of the Skill’s elicited requirements. These intents can be considered as the principle intents that cover the essential Skill functionality; they were however extended upon in the implementation in order to optimise the user interaction. Table 4.1 lists the names of the 14 identified principle intents alongside a description and an example invocation. Where the phrase “Alexa, ask Sheffield Computer Science” has been omitted it implies that the given intent can only be invoked after a Skill Launch Request due to the requirement of a turn-based dialogue flow.

Intent	Description	Invocation
LaunchRequest	Sends a LanuchRequest to the Skill service in order to invoke the Skill so that all subsequent interaction is handled by the Skill.	Alexa, launch Sheffield Computer Science
GetModuleDescription	Starts a dialogue flow that allows the user to obtain a full overview of a specific module.	Can you tell me about {ModuleCode}/{ModuleName}
GetModuleInfo	Returns the information requested about a specific module.	Alexa, ask Sheffield Computer Science for the {ModuleInfo} of {ModuleCode}/{ModuleName}
GetCourseInfo	Returns the information requested about a specific course.	Alexa, ask Sheffield Computer Science for the {CourseInfo} of {CourseName}
GetCoursesOffered	Begins a dialogue flow that allows a user to ask what courses are offered and obtain further course related information.	Tell me what {CourseName} is like
GetDCSInfo	Returns the information requested regarding the Department of computer Science.	Alexa, ask Sheffield Computer Science for the {DCSInfo} of the {DepartmentName}
GetLecturerInfo	Returns the information requested regarding the specified lecturer.	Alexa, ask Sheffield Computer Science for the {LecturerInfo} of {LecturerFname}/{Title} {LecturerSname}
GetLecturerTeaches	Starts a dialogue flow where a user can get a list of the modules that a given lecturer teaches and obtain further information about these modules.	Can you tell me about the courses that are on offer
GetTimetableInfo	Returns the lectures/classes for a given module.	Alexa, ask Sheffield Computer Science for the lectures of {TimetableUnitCode}/ {TimetableUnitTitle}
GetFAQ	Returns the closest matching answer that satisfies the user’s question.	Alexa, ask Sheffield Computer Science {FAQ}
GetFavouritesList	Returns the contents of the device owners favourite list.	Alexa, ask Sheffield Computer Science for the contents of my favourites list
AddToFavouritesList	Inserts the specified module into the device owners favourite list.	Alexa, ask Sheffield Computer Science to add {ModuleCode}/{ModuleName} to my favourites list
RemoveFromFavouritesList	Removes the specified module into the device owners favourite list.	Alexa, ask Sheffield Computer Science to remove {ModuleCode}/{ModuleName} from my favourites list
ClearFavouritesList	Clears the contents of the device owners favourite list.	Alexa, ask Sheffield Computer Science to clear my favourites list

Table 4.1: Table covering the design of the principle intents, including descriptions and example invocation utterances

4.2.3 Conversational Design

An important aspect of the design phase consisted of identifying the ways that the Skill can fulfil the requests made by the user. The Alexa Skills Kit makes use of a frame based- dialogue design and as such can prompt the user where necessary for the required slot values required to produce an answer. Eliciting slot information allows the user to retrieve an answer to their request in the cases where they are unfamiliar with the capabilities of the system or are unaware of what they want to achieve through their interaction.

A typical approach to modelling these scenarios is through the use of flow-charts representing how a user’s choice of interaction can lead to a provided answer. Figure 4.1 provides an example of such a scenario. In this scenario a user is attempting to retrieve the entry requirements for a given course. There are two ways in which this can be achieved. The first approach would be the user providing valid slot values for the {query} and {courseName} slots in a single utterance to retrieve the answer, which is better known as a “one-shot” interaction. The alternative approach demonstrates a user not having much awareness of how the Skill operates and asking a more general question regarding courses. In this approach the Skill guides the user through a dialogue process which will lead them to information that they have expressed interest in. It is important to note that at any point during this conversational process the “one-shot” interaction can be used to override the process and receive the informational response.

A further consideration that has been taken regarding the conversational design is the ability for the Skill to provide further information related to the topic of the user’s question. This is done through a follow-up question such as “would you like information about…” after output of an answer. Examples of these cases can be seen in the appendices.

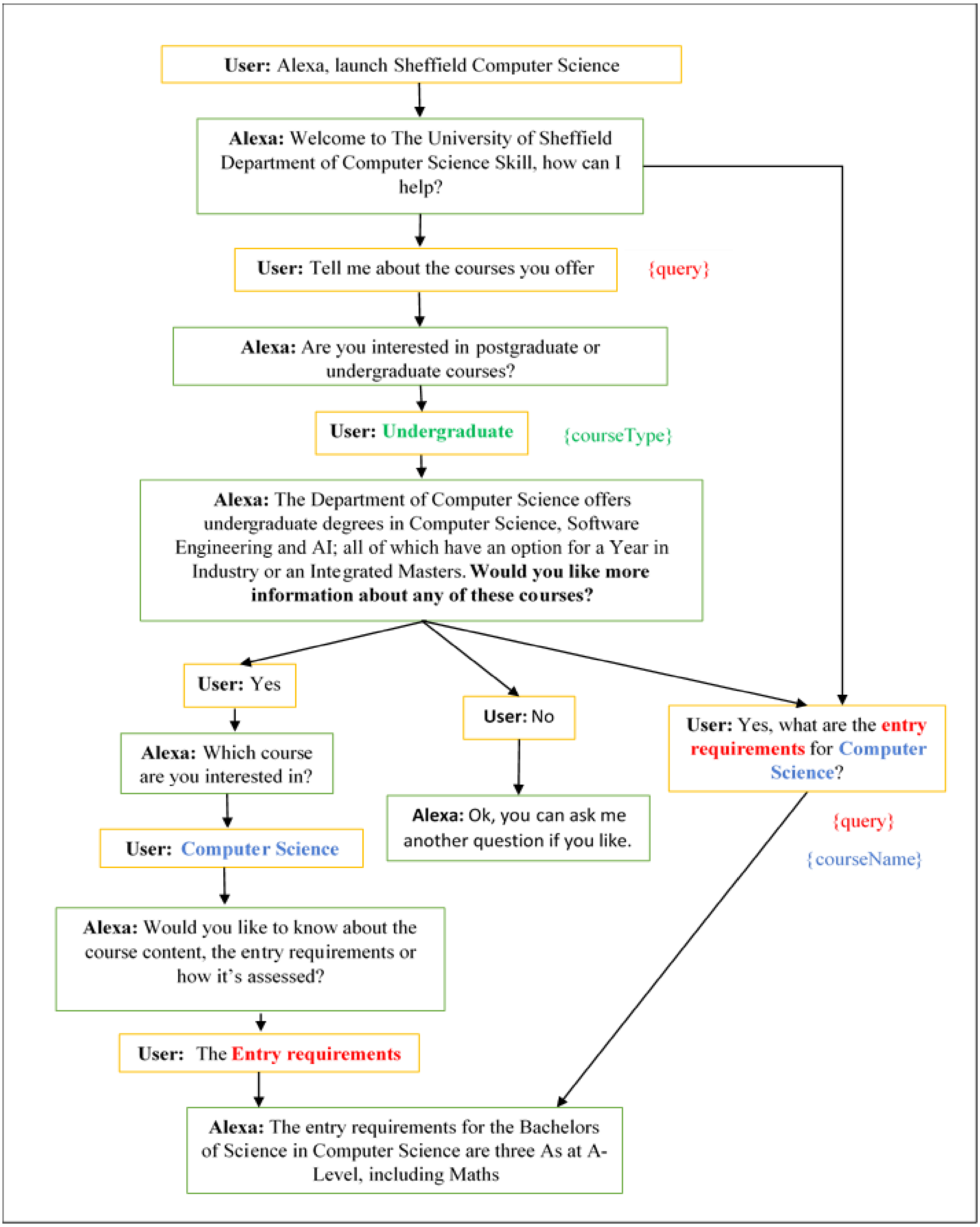
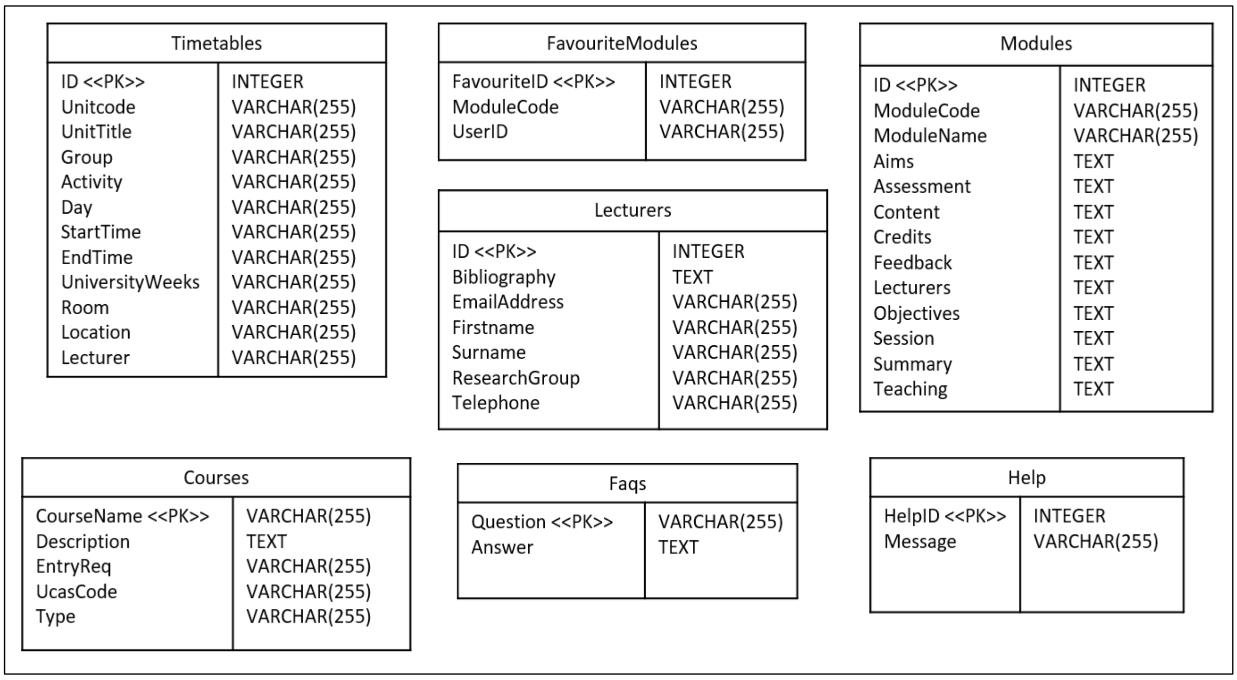

User: Alexa, launch Sheffield Computer Science
Alexa: Welcome to The University of Sheffield Department of Computer Science Skill, how can I help?
User: Tell me about the courses you offer
{query}
Alexa: Are you interested in postgraduate or undergraduate courses?
User: Undergraduate
{courseType}
Alexa: The Department of Computer Science offers undergraduate degrees in Computer Science, Software Engineering and AI; all of which have an option for a Year in Industry or an Integrated Masters. Would you like more information about any of these courses?
User: Yes
User: No
User: Yes, what are the entry requirements for Computer Science?
Alexa: Which course are you interested in?
Alexa: Ok, you can ask me another question if you like.
{query}
{courseName}
User: Computer Science
Alexa: Would you like to know about the course content, the entry requirements or how it's assessed?
User: The Entry requirements
Alexa: The entry requirements for the Bachelors of Science in Computer Science are three As at A-Level, including Maths

Figure 4.1: Dialogue flow for obtaining course information

4.3 Database Design

The design of the database was an important aspect of the design process since it would contain the majority of the data that the Skill would use to form its output responses. The database schema has been structured using a third-normal form approach to ensure there is no duplication of data and that the referential integrity of the data is ensured. In addition, the data types and lengths of the database columns have been considered to ensure that the correct types of data are stored within each column and that non-conforming values are prohibited.



Timetables	
ID <<PK>>	INTEGER
Unitcode	VARCHAR(255)
UnitTitle	VARCHAR(255)
Group	VARCHAR(255)
Activity	VARCHAR(255)
Day	VARCHAR(255)
StartTime	VARCHAR(255)
EndTime	VARCHAR(255)
UniversityWeeks	VARCHAR(255)
Room	VARCHAR(255)
Location	VARCHAR(255)
Lecturer	VARCHAR(255)

FavouriteModules	
FavouriteID <<PK>>	INTEGER
ModuleCode	VARCHAR(255)
UserID	VARCHAR(255)

Modules	
ID <<PK>>	INTEGER
ModuleCode	VARCHAR(255)
ModuleName	VARCHAR(255)
Aims	TEXT
Assessment	TEXT
Content	TEXT
Credits	TEXT
Feedback	TEXT
Lecturers	TEXT
Objectives	TEXT
Session	TEXT
Summary	TEXT
Teaching	TEXT

Lecturers	
ID <<PK>>	INTEGER
Bibliography	TEXT
EmailAddress	VARCHAR(255)
Firstname	VARCHAR(255)
Surname	VARCHAR(255)
ResearchGroup	VARCHAR(255)
Telephone	VARCHAR(255)

Courses	
CourseName <<PK>>	VARCHAR(255)
Description	TEXT
EntryReq	VARCHAR(255)
UcasCode	VARCHAR(255)
Type	VARCHAR(255)

Faqs	
Question <<PK>>	VARCHAR(255)
Answer	TEXT

Help	
HelpID <<PK>>	INTEGER
Message	VARCHAR(255)

Figure 4.2: Database Schema

4.4 Error Handling

The handling of errors during use of the Skill was an important factor to be considered as part of the design process. This is because it is impossible to envision all of the different interactions that users might intend to have with the Skill. As such when user speech input has not been recognised by the Skill lambda function an appropriate error response is returned prompting the user to ask their question in a different manner. Furthermore, if a user provides information that is not valid in the context of the question, they will be prompted to provide a valid slot value within their speech input. If a user finds that they are unable to supply a correct response to a question asked by the Skill during the course of a conversational dialogue flow they will be re-prompted with the same question until a valid input is obtained. Alternatively, a user can choose to restart or escape the current dialogue by saying a phrase such as “start over”. This in turn will reset all the Skill session attributes associated with the given dialogue flow.

Chapter 5

Implementation and Testing

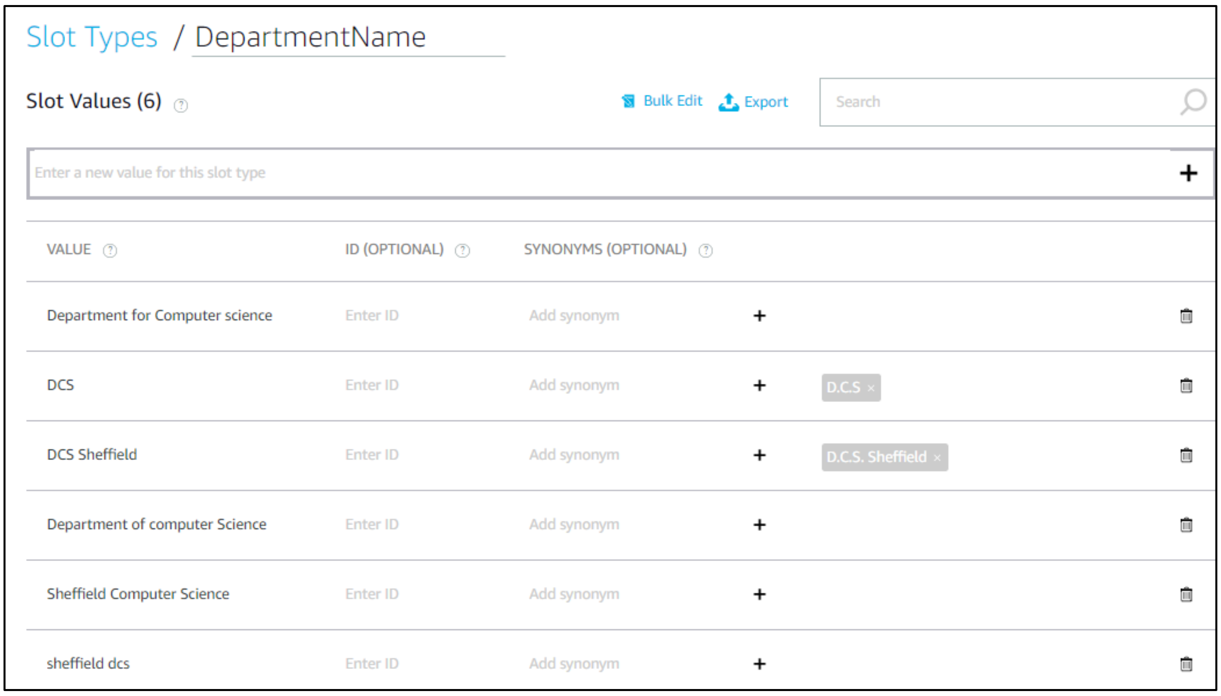
This chapter extends upon the design considerations made in chapter 4 showing how the different elements were implemented into the final system. These include the Interaction Model, the AWS lambda function and the database. This chapter concludes with a description of the testing processes carried out along with example tests and their results.

5.1 The Interaction Model

The interaction model consists of a singular JSON file that contains the Skill intents, their sample utterances and the slot definitions. Writing the JSON file directly is not a trivial task, so the Alexa Developer Console was used instead to build the interaction model. The Alexa Developer Console is an online tool that "provides a streamlined experience to help you create and manage Skills" [52] via a user-friendly GUI, that allows developers to build and manage all aspects of the interaction model. Furthermore, it simplifies the process of building the dialogue model by handling the ID references between dialogue delegations and validation rules. These rules must be defined in order for Alexa to determine the next step in the conversation based on the prompts and utterances collected.

5.1.1 Slots

There is a total of 15 custom slot types used within the interaction model. Each of these slot types contains a list of possible slot values. The slot values are the phrases that a user must say in order to fill the slot type and hence satisfy an intent utterance. Slot types are a good way of handling cases where phrases can be expressed in multiple different ways. This is the case as shown in Figure 5.1, where each slot value is a variation of the phrase "Department of computer Science". To help with the speech recognition of slot values containing abbreviations, alternate synonyms have been provided that make use of punctuation between the letters. The alternative use of slot types is to specify a list of unique values that they can accept. This is the case for the {ModuleCode} and {ModuleName} slot types. When these slots are filled, their values are passed to the AWS lambda function and used to retrieve the information from the database relevant to their request.


Slot Types / DepartmentName

Slot Values (6)

VALUE	ID (OPTIONAL)	SYNONYMS (OPTIONAL)
Department for Computer science	Enter ID	Add synonym
DCS	Enter ID	D.C.S
DCS Sheffield	Enter ID	D.C.S. Sheffield
Department of computer Science	Enter ID	Add synonym
Sheffield Computer Science	Enter ID	Add synonym
sheffield dcs	Enter ID	Add synonym

Figure 5.1: Screenshot from the Developer Console illustrating the slot values for a given slot type

5.1.2 Intents

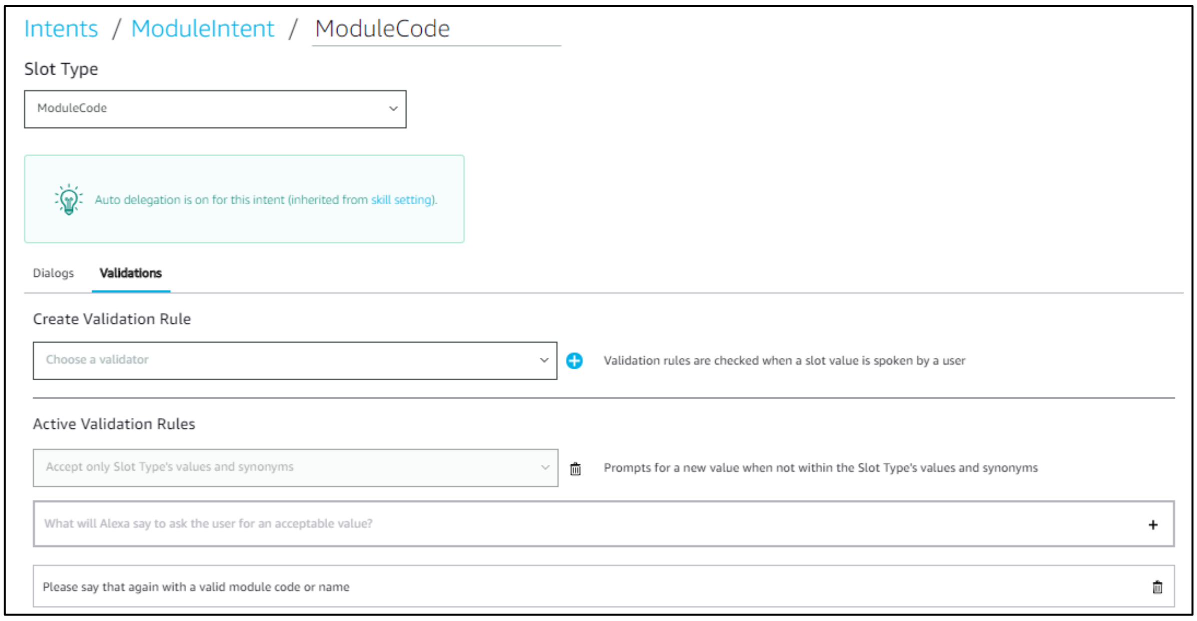
The final Skill uses a total of 53 Intents, a number which far exceeds the number identified during the design stage. This is because many of the intents were split up into several intents in order to more clearly define the utterances that should be capable of invoking an intent. This is the case when asking about Module Information. The original plan was to handle this with a singular intent that used a slot named {ModuleInfo} that would have contained values such as credits, assessment, objectives and feedback. This approach though would have resulted in confusing voice interactions. A scenario that demonstrates this would be the user expecting to be able to ask, "When is {ModuleName} taught". The utterance for a singular intent handler would have to be "when is {ModuleName} {ModuleInfo}", where {ModuleInfo} includes the slot value "taught". However, this slot would also have to include other values such as "feedback". Hence, it does not make sense that a user would be able to say, "when is {ModuleName} feedback". Accordingly, the other intents that were identified in the design that would have experienced this issue were also divided up into several intents.

Included within the total number of intents are 19 built-in Intents. Each of these can be invoked when a user says a phrase that Amazon has included within their definitions. The names of the 8 frequently used built-in intents and their uses are listed in Table 5.1. The remaining 11 built-in intents are used to navigate the display of a visual Echo device e.g. AMAZON.ScrollRightIntent.

Intent	Common Utterances	Purpose
AMAZON.HelpIntent	• Help • Help me • Can you help me	Provides a customised help message that provides information on how to use the Skill
AMAZON.CancelIntent	• Cancel • Never Mind • Forget It	Allows the user to completely exit the Skill
AMAZON.FallbackIntent	NA: Uses an out-of-domain model that is generated based on the interaction model.	Provides a fallback for user utterances that do not match any of the Skill’s intents
AMAZON.StopIntent	• Stop • Off • Shut up	Allows the user to completely exit the Skill
AMAZON.RepeatIntent	• Repeat • Say that Again • Repeat that	Allows the user to hear a repeat of the last message that was output
AMAZON.StartOverIntent	• Start over • Restart • Start again	Allows the user to effectively restart the Skill, resetting all of the Skill session attributes
AMAZON.YesIntent	• Yes • Yes please • Sure	Allows the user to provide a positive response to yes/no question
AMAZON.NoIntent	• No • No thanks	Allows the user to provide a negative response to yes/no question

Table 5.1: Table covering the built-in intents frequently used during Skill interaction

Each of the custom intents in the interaction model includes one or more custom slot types. For each of these Intents slots validation rules have been implemented. These validation rules prompt the user for a value that would comply with the list of values contained within their slot definitions. These prompts occur when the Skill recognises that a user is attempting to invoke a given intent but has not provided a valid slot value that is required to fulfil the intent. The active validation rule that has been used is: “Accept only Slot Type’s values and synonyms”. Figure 5.2 demonstrates the use of a validation rule and prompt message for the {ModuleCode} slot type within the ModuleIntent.


Intents / ModuleIntent / ModuleCode

Slot Type

ModuleCode

Auto delegation is on for this intent (inherited from skill setting).

Dialogs Validations

Create Validation Rule

Choose a validator

Validation rules are checked when a slot value is spoken by a user

Active Validation Rules

Accept only Slot Type's values and synonyms

Prompts for a new value when not within the Slot Type's values and synonyms

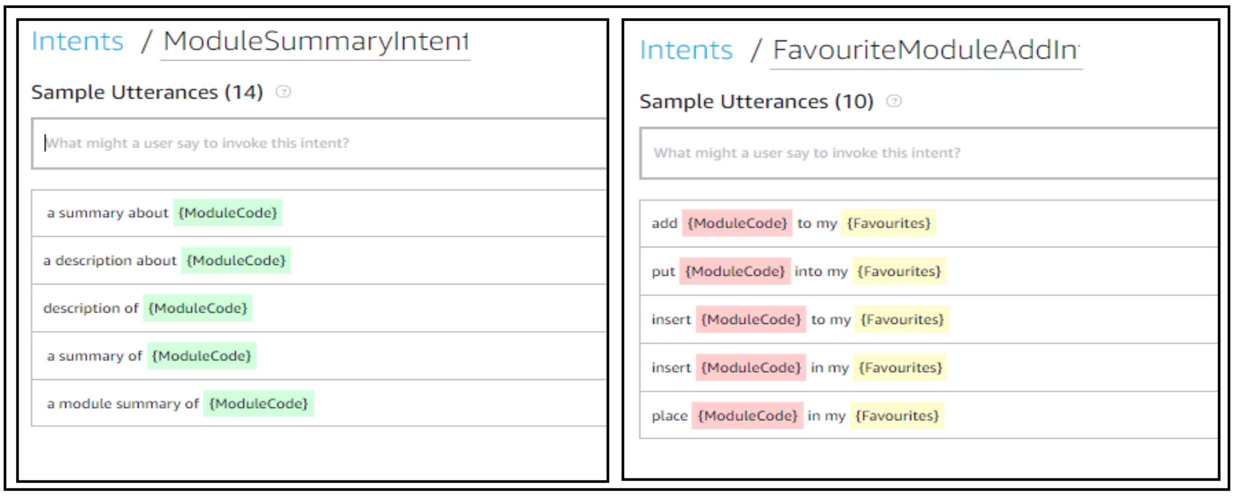
What will Alexa say to ask the user for an acceptable value?

Please say that again with a valid module code or name

Figure 5.2: Screenshot showing the validation rule for a slot type used by an intent

5.1.3 Utterances

The sample Utterances of an intent are the phrases that when included within a user's speech input invoke the intent. It is important to provide as many utterances as possible for each intent since a user can provide many variations in their input that mean the same thing. Amazon provides a guide on how best to define variation of utterances for intents [53]. As a result of following this guide closely, the utterances that have been used throughout the Skill have been designed in a manner that allows them to be mapped to as many potential interactions as possible. Figure 5.3 provides examples of this in the solution, where user input such as “tell me a summary of …”, “give me a summary of …” and “can I have a summary of …” will all map to the single utterance “a summary of …”. This is because Alexa attempts to generalize based on the utterances provided to interpret spoken phrases that differ in minor ways from the specified utterances.


Intents / ModuleSummaryIntent

Sample Utterances (14)

What might a user say to invoke this intent?

a summary about {ModuleCode}

a description about {ModuleCode}

description of {ModuleCode}

a summary of {ModuleCode}

a module summary of {ModuleCode}

Intents / FavouriteModuleAddIn

Sample Utterances (10)

What might a user say to invoke this intent?

add {ModuleCode} to my {Favourites}

put {ModuleCode} into my {Favourites}

insert {ModuleCode} to my {Favourites}

insert {ModuleCode} in my {Favourites}

place {ModuleCode} in my {Favourites}

Figure 5.3: Screenshot showing the variation of sample utterances implemented for each intent

5.2 The AWS Lambda Function

The AWS lambda function was built using Python 3.6 and was initially set up using the Python Hello World Skill example [54]. Initially the code for the project was written using the inline code editing feature of the AWS management console. However, as development progressed and packages required for the Skill code to work were added, the size of the lambda function exceeded the size restrictions required for inline editing. As such the contents of the Skill had to be packaged as a zip file each time changes and additions were implemented and then uploaded. This slowed down the development process considerably as the upload time could take up to five minutes depending on the internet's upload speed. Configuring the lambda function within the AWS management console required setting the lambda function to be triggered by the ASK, linking to other required AWS resources and setting the code entry point.

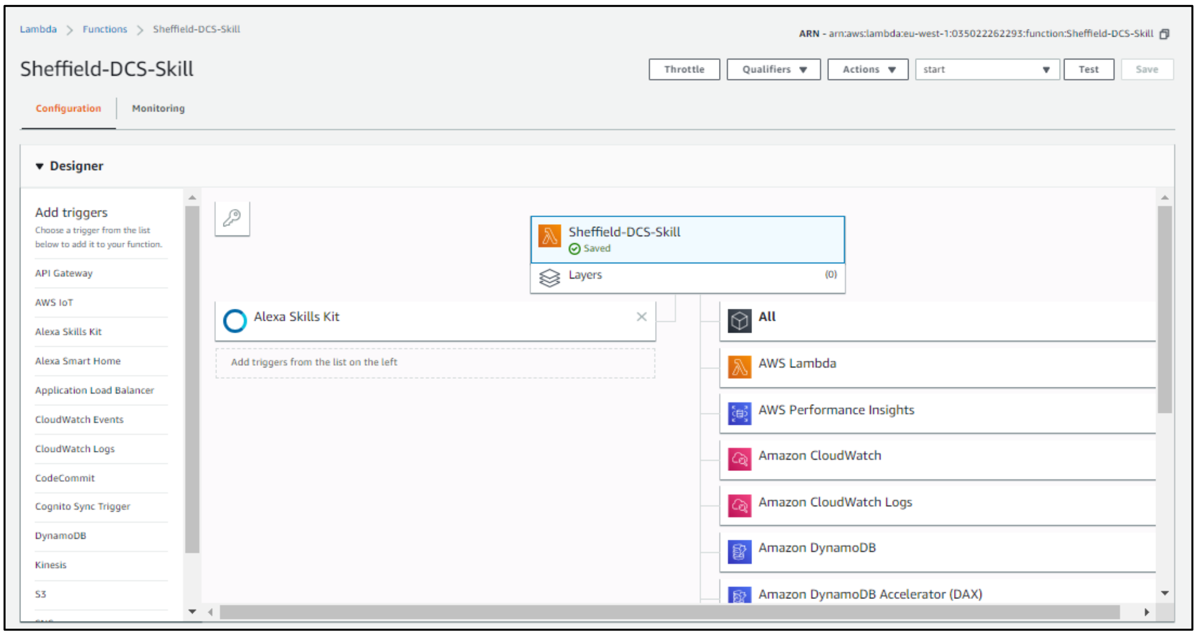


Figure 5.4: Screen capture of the AWS Management console, depicting the Skills triggers and links

The code within the Lambda function processes incoming requests sent by the Alexa Service using request handlers [55]. These request handlers specify the incoming requests (intents) that they can handle and then output a response. Each request handler implements two methods, "can_handle" and "handle". The "can_handle" method returns a Boolean value that indicates whether the given request handler can process the request. The "handle" method of the request handler is called if its corresponding "can_handle" method returns true. It is responsible for generating and returning the response using ResponseBuilder. ResponseBuilder is an interface which includes helper methods for constructing the response such as setting the visual response, the reprompt message and whether the Skill session should end.

5.2.1 Sessions

Where Intents used within the Skill require a multi-turn interaction for them to be fulfilled, session attributes have been used within their response handlers. Session attributes are contained within the JSON object that is sent back-and-forth during the conversational turns. They allow the current state of a dialogue flow to be stored between user interactions, essentially acting as global variables. In the implementation these have been implemented as a series of turns. The turn value and the user's response, e.g. yes/no, then map to a subsequent action in the dialogue flow. At any time, the user can invoke the AMAZON.StartOverIntent to escape the dialogue flow and reset all the session attributes. Using session attributes gives developers greater flexibility when it comes to implementing interesting user interaction, but they do come with constraints on their use. Any dialogue that makes use of session attributes requires the Skill to be opened first so that they can be initialized. Furthermore, session attributes only persist during the duration of a single Skill session which means that any progress made within a dialogue flow will not be kept if the user exits the Skill and then re-opens it.

5.2.2 Data Persistence

A feature of the Skill requiring persistence was that relating to the management of a user's unique favourites list. To maintain persistence of the contents of a user's favourites list, they are stored within the database. This ensures that a favourites list can be managed across different session instances. When interacting with an Alexa enabled device, a unique deviceID and userID are sent in the request JSON object to the Alexa Service. The userID was used to associate the contents of a favourites list with a unique Amazon user. This ensures that a user will only ever have access to their own favourites list during any Skill interactions with devices that are associated with their Amazon account.

5.2.3 FAQ Processing

Processing of non-departmental related FAQ's was done using a search engine created by **P3** which made use of the university AskUs webpages [56]. The AskUs webpages are a place where both prospective and current students can type a question and retrieve a list of matching answers most closely related to their question. The webpages cover a large range of more general university related topics such as accommodation and student wellbeing. This functionality was integrated into the project by extracting the user's question from a slot and performing a search over a directory of indexed documents. The directory of indexed documents in this case is the HTML source code files of all the AskUs question/answer pages. When the search concludes, a tuple containing the top three most relevant answers is returned. The top most relevant answer is then returned as the speech output. There are two constraints with the way that this functionality has been implemented. The first is that the directory of indexed files does not account for updates made to the AskUs webpages and so currently must be updated manually. The second constraint relates to the use of the AMAZON.SearchQuery [57] slot type which was used to accommodate this functionality. The AMAZON.SearchQuery slot type is used to capture less-predictable input that makes up a search query. However, it must include a carrier phrase that precedes the search query such as “search” or “find out”. A full example would be “search, how do I register?”, where the slot value is “How do I register”. This type of interaction is not intuitive for users of the Skill, because of this, efforts have been made to make this clear via the help messages.

5.2.4 Help Options

In order to guide users during different stages of their skill experience, in-Skill contextual help messages have been implemented that provide details on the possible phrases a user can say. There are two types of help message that have been included within the Skill, master help messages and contextual help messages. Master help messages are those that the user hears when explicitly asking for help. The first time the user invokes the master help message they will hear a general summary of what they can do with the Skill. On subsequent invocations of the master help message a random utterance example is provided, demonstrating more specifically how users can interact with the Skill (see Appendix B.2). A contextual help message is a help message heard when an error is caught or when the system detects that a user is struggling with a command. These have been implemented in the form of reprompt messages. Another useful feature of adding reprompt messages throughout the Skill is that they extend the duration the Skill will remain open. If the user does not respond within 8 seconds of the Skill finishing its last speech output the Skill will close. Providing reprompt messages ensure the Skill remains open for a further 8 seconds whilst also clarifying the current dialogue state.

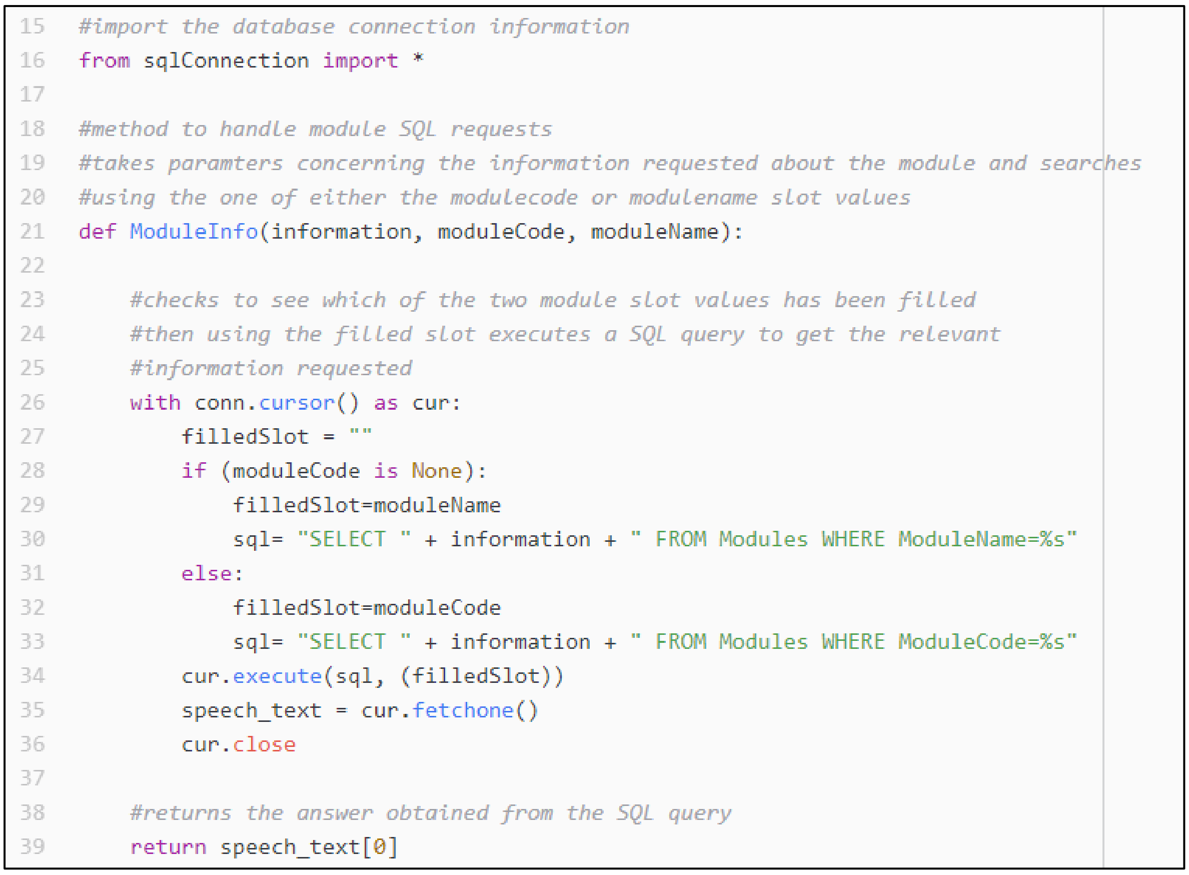
5.2.5 Error Handling

The handling of errors has been implemented in accordance with the practices outlined in the design of the Skill and through custom error responses built within the AWS lambda function. Without error handing the Skill will simply exit if the speech is out-of-domain of the Skill's interaction model. The problem with this is that it will lose the contents of the session attributes and so any progress made within dialogue flows will be lost. The AMAZON.FallbackIntent prevents this from happening and instead provides a message prompting the user to ask their question again but in a different manner. When slot values are required for intents to be fulfilled, validation has been implemented along with Reprompt messages in order to gain a valid value (see section 5.1.2).

Several custom error handling features have also been included within the AWS lambda function. This includes validation that handles instances where users request valid information, but no data relating to the request exists in the database, e.g. missing a lecturer's research group. For these occurrences, the system informs the user that there is no information available relating to their request. Error handling has also been implemented for the features concerning the management of a favourites list. These handle cases that do not make sense such as adding a module to their list when it is already in the list or requesting to remove a module from their list while it is empty.

5.3 Database

The database was implemented using a MySQL database that was hosted using Amazon's Relational Database Service. To set up the database, SQL table creation queries (Appendix B) were defined based on the schema outlined in the project's design. These were then executed alongside queries responsible for inserting the relevant data into the tables. Database interaction was implemented using the python library PyMySQL. The first stage of building this interaction was to define the database connection information for the remote MySQL database. The following stage consisted of building the SQL query required to fulfil the needs of an intent. The final stage to building this interaction was to open a connection to the remote database using the previously defined connection information and SQL query in order to conduct a database operation. Figure 5.5 provides an example of how information relating to user's request was retrieved from the database using PyMySQL.


```python
#import the database connection information
from sqlConnection import *

#method to handle module SQL requests
#takes paramters concerning the information requested about the module and searches
#using the one of either the modulecode or modulename slot values
def ModuleInfo(information, moduleCode, moduleName):

    #checks to see which of the two module slot values has been filled
    #then using the filled slot executes a SQL query to get the relevant
    #information requested
    with conn.cursor() as cur:
        filledSlot = ""
        if (moduleCode is None):
            filledSlot=moduleName
            sql= "SELECT " + information + " FROM Modules WHERE ModuleName=%s"
        else:
            filledSlot=moduleCode
            sql= "SELECT " + information + " FROM Modules WHERE ModuleCode=%s"
        cur.execute(sql, (filledSlot))
        speech_text = cur.fetchone()
        cur.close

    #returns the answer obtained from the SQL query
    return speech_text[0]
```

Figure 5.5: Screenshot showing the retrieval of module related information

5.3.1 Data Collection

One of the biggest constraints that was identified in the initial stages of this project was being able to process the vast amount of departmental information. This problem was largely overcome by the web-scraping techniques used within project **P3**. The techniques implemented allowed for all the data regarding lecturers, modules and timetabling to be gathered from the DCS Intranet pages. Once collected the data was dumped as three separate SQL files where the data collected conformed to the database schema definitions. This was done so that the collected data could then be integrated within this project (**P2**). The remaining data used within the Skill was collected manually. In the cases where data was collected manually it was deemed unnecessary to implement an alternative web-scraping solution due to the limited amount of data required.

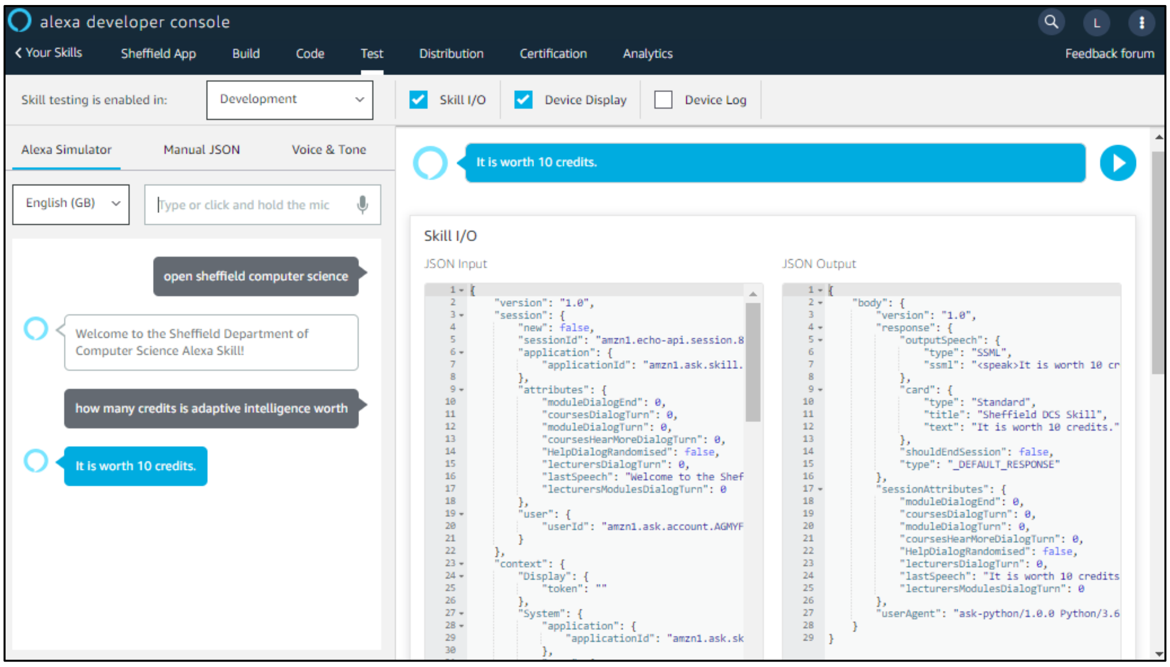
The biggest constraint relating to the data used within this project is that it is all dynamic. Currently management of the Skill would require manually updating the data stored within the database at regular intervals. A more optimal approach could be to access the data directly from the relevant webpages at runtime. The reason this method was not implemented was that it allowed for less control in the speech responses.

5.3.2 Full-Text Searching

A useful feature that MySQL provided was the ability to use full-text indexing and searching of data [58]. This was done for the departmental related FAQ data that project **P1** had gathered from interaction with students at open days. The data provided consisted of a potential question a student would ask and its matching answer. Since a user interacting with the Skill could ask these questions in various ways full-text search was implemented (Appendix Figure B.3). This matches the user's interaction with the closest matching question stored within the "Faqs" database table and subsequently finds the correct answer using its index position.

5.4 Testing

Testing of the Skill was carried out to verify the correctness of the software by pointing out the defects and errors that were made during development. There are two ways that testing of the Skill could have been conducted. The first is to speak directly to an Echo device and the second is to type the input into the Alexa Development console. The latter option was chosen as the preferred testing method for multiple reasons. It ensured that there were no errors relating to the speech recognition of the input and provided a visual display of the input and output JSON's to see if the request had been handled correctly (Figure 5.6). In order to debug any errors that were found, test events using the JSON input were configured within the AWS lambda console, these then returned the cause of the error and line number of the code responsible.



open sheffield computer science

Welcome to the Sheffield Department of Computer Science Alexa Skill!

how many credits is adaptive intelligence worth

It is worth 10 credits.

```
{
    "version": "1.0",
    "session": {
        "new": false,
        "sessionId": "amzn1.echo-api.session.8
        "application": {
            "applicationId": "amzn1.ask.skill.
        },
        "attributes": {
            "moduleDialogEnd": 0,
            "coursesDialogTurn": 0,
            "moduleDialogTurn": 0,
            "coursesHearMoreDialogTurn": 0,
            "HelpDialogRandomised": false,
            "lecturersDialogTurn": 0,
            "lastSpeech": "Welcome to the Shef
            "lecturersModulesDialogTurn": 0
        },
        "user": {
            "userId": "amzn1.ask.account.AGMYF
        }
    },
    "context": {
        "Display": {
            "token": ""
        },
        "System": {
            "application": {
                "applicationId": "amzn1.ask.sk
            },
```

```
{
    "body": {
        "version": "1.0",
        "response": {
            "outputSpeech": {
                "type": "SSML",
                "ssml": "<speak>It is worth 10 cr
            },
            "card": {
                "type": "Standard",
                "title": "Sheffield DCS Skill",
                "text": "It is worth 10 credits."
            },
            "shouldEndSession": false,
            "type": "_DEFAULT_RESPONSE"
        },
        "sessionAttributes": {
            "moduleDialogEnd": 0,
            "coursesDialogTurn": 0,
            "moduleDialogTurn": 0,
            "coursesHearMoreDialogTurn": 0,
            "HelpDialogRandomised": false,
            "lecturersDialogTurn": 0,
            "lastSpeech": "It is worth 10 credits
            "lecturersModulesDialogTurn": 0
        },
        "userAgent": "ask-python/1.0.0 Python/3.6
    }
}
```

Figure 5.6: Screenshot depicting the Alexa developer console test simulator

5.4.1 End-to-end Testing

End-to-end testing was carried out for intents that made use of a dialogue flow. These tests were done to ensure that at each stage of the dialogue flow a user could successfully navigate to one of the succeeding stages. The tests conducted in Table 5.2 illustrate how the stages of dialogue flows were robustly tested to ensure that the session states update correctly. In particular it demonstrates how user input is handled correctly depending on the current dialogue state.

Stage in Dialogue	User Input	Speech output	Session State
Launch Skill	Alexa, launch The Department of Computer Science Skill	Welcome to The University of Sheffield Department of Computer Science Skill, how can I help?	0
Invoke dialogue flow	Tell me about the courses you offer	Are you interested in postgraduate or undergraduate courses?	1
Stage 1: Specify the course type (requires session state: 1)	Undergraduate	The Department of Computer Science offers… Would you like more information about any of these courses?	2
	Postgraduate	The Department of Computer Science offers… Would you like more information about any of these courses?	2
Stage 2: Specify intent to hear more about a course (requires session state: 2)	Yes	Which course are you interested in?	3
	No	Ok, you can ask me another question if you like.	0
	Yes, what are the entry requirements for Computer Science?	The entry requirements for the Bachelor of Science in Computer Science are three As at A-Level, including Maths	0
	Undergraduate	Would you like more information about any of the courses?	2
Stage 3: Specify the name of the course to hear more (requires session state: 3)	Computer Science	In computer science you'll learn to understand… The UCAS code for Computer Science is … and the entry requirements are …	0
	Start Over	Ok, you can ask me another question if you like.	0
	yes	Which course are you interested in?	3

Table 5.2: End-to-end testing example conducted for a dialogue flow concerning courses

5.4.2 Unit Testing

To ensure that individual intents satisfied their requirements, unit-testing was conducted which tested the intents using several inputs to make sure the results were as expected. A useful tool which helped with this testing was the development consoles "utterance profiler" [59]. This tool allows developers to submit utterances and view the intent that would be sent to the lambda function alongside the slots and their values. As a result, the utterance profiler was used to check that the Skill would maintain results across different variations of utterance. Table 5.3 shows how unit testing was conducted using the {LecturerGroupIntent} as an example. It also demonstrates the handing of validation by the lambda function for cases where the intent utterance is mapped correctly to an intent, but no data exists relating to the request.

Test Condition	**User Input**	**Speech output**
A valid utterance is provided using the {LecturerFname} and {LecturerSname} slots	Can you tell me the research group for Tony Simons?	Tony Simons is head of the Testing research group
A valid utterance is provided using the {Title} and {LecturerSname} slots	What research group is Dr Simons in?	Tony Simons is head of the Testing research group
A valid utterance is provided using the {LecturerFname} and {LecturerSname} slots but no data exists	Would you give me the research group for Achim Brucker?	No information exists regarding the research group of Achim Brucker
A valid utterance is provided but one or more slots values is not valid	What is the research group for Dean Taylor?	Please say that again with a valid lecturer name
A non-valid utterance is provided	What are the research interests of Tony Simons?	Sorry, I didn't quite get that. Can you please say it again?

Table 5.3: Example of unit-testing involving the {LecturerGroupIntent}

Chapter 6

Results and Discussion

This chapter provides a summary of the requirements that have been met and to what extent they have been achieved. The framework used for testing of the Skill with human participants is also outlined alongside the results collected and their significance. Finally, a summary of further project-related work is provided.

6.1 Requirements Analysis

The requirements that were originally outlined in section 3.1 have largely been met as evident in Table 6.1 and Table 6.2. They show that all the requirements that were considered mandatory to the success of the project have been implemented. This allowed for attention to be placed on the less desirable project requirements, where the majority were also completed.

The requirement concerning long-winded responses was only partially completed. For data that was collected manually, efforts were made to ensure that this requirement was met. However, for data that was web-scraped, it was not possible to optimise it an efficient manner to satisfy this requirement. The limitations imposed by web-scraping were also evident with the providing of timetable information. This was because the format of the data was not optimised for searching over using SQL queries and as such only a limited amount of timetable information could be provided in the final Skill.

The requirement that was left non-implemented was the ability to retrieve directions to the different campus buildings across the university. This was mainly due to time constraints but also because doubts were raised during the course of the project about the usefulness of this feature, since students are more likely to use other more readily available directional methods, e.g. Google Maps via a smartphone device.

To help in assessing whether some of the more subjective the non-functional requirements had been completed, the results of the user evaluations were considered. These results also helped to show how successful the requirements were satisfied and are discussed in section 6.2.

Description	**Weighting**	**Completion**
The Skill must be able to be invoked easily	M	Completed
The Skill must have help options for each section of user interaction	M	Completed
The Skill must allow a range of utterances to be used to invoke intents	M	Completed
The Skill must not have long-winded responses	M	Partial
The Skill must elicit slot information when required slots have not been filled in the dialogue management	M	Completed
The Skill must provide course and module related information	M	Completed
The Skill must provide information relating to placements and University life	M	Completed
The Skill must include information relating to lecturers	M	Completed
The Skill must be able to handle non-system requests gracefully	M	Completed
The Skill should be able to repeat information if requested	S	Completed
The Skill should be able to maintain necessary attributes across sessions	S	Completed
The Skill could allow users to ask directions to buildings around Campus	C	Incomplete
The Skill could allow users to get lecture timetable information	C	Partial
The Skill could display information visually for display enabled devices	C	Completed
The Skill could allow users to manage a list of their favourite modules	C	Completed

Table 6.1: Table displaying the completion status of functional requirements

Description	**Weighting**	**Completion**
The Skill must be easy to use and contain contextual awareness	M	Completed
The Skill must ensure that data from responses is not misrepresentative	M	Completed
The Skill must be efficient in its response times	M	Completed
The Skill should be easy to manage in order to allow future additions	S	Completed
The language of the Skill should be formal in its responses	S	Completed

Table 6.2: Table displaying the completion status of non-functional requirements

6.2 User Evaluation Framework

The user evaluations that were conducted were done collaboratively between both **P1 and P2**. Although the evaluations of the Skill were a major focus of **P1,** they were of an integral importance to the success of the entire project in order to determine where improvements could be made. As such a joint user-evaluation testing framework was produced:

Testing was carried out upon seven randomly selected human participants. The random selection process ensured as much variation as possible between the previous experiences people had using an Alexa-enabled device. In order to limit the external noise that could influence the evaluations, the test environment was maintained each time using a private room consisting of only the participant and the test supervisor. Each participant was provided with an information sheet, consent form and background questionnaire before the evaluations began (Appendix C). The purpose of the background questionnaire was mainly to deduce how familiar the participant was with using an Echo device. In the cases where the participant was not familiar with an Alexa device, the test supervisor provided training that was out-of-context of the actual evaluation.

The evaluation itself made use of SASSI [60]; an evaluation method designed for speech systems that measures subjective satisfaction. This involved creating a questionnaire consisting of attitude statements that cover the entire domain of the system and the six factors considered important in user attitude to speech systems. These factors were system response accuracy, likeability, cognitive demand, annoyance, habitability and speed. When completing the questionnaire (Appendix C.7), the user was required to rate each attitude statement according to a seven-point scale labelled strongly agree, agree, slightly agree, neutral, slightly disagree, disagree and strongly disagree. Moreover, participants were instructed to write down more general feedback about their experience.

Questionnaire responses were formed as a result of the user attempting to complete a list of 13 tasks which covered the main functionalities of the Skill (Appendix C.4). Care was taken when writing the tasks themselves so as to not influence the way a participant might say their utterance. The reason for this was to see whether the Skill would handle the different interaction expectations that the participants might have.

6.2.1 Results

Following the conclusion of the user evaluations a mean satisfaction rating was calculated for each of the six assessment factors. This rating is based on the seven-point scale, where a score greater than 4 is above average and hence considered to be positive, and vice versa. In order to calculate these ratings, a positive/negative weighting factor had to be calculated from the mean rating of the participant responses for each attitude statement, see Appendix D. This was because some statements were worded in a manner where a lower score was actually a positive outcome. Finally, these weighting factors, were used to form the mean satisfaction ratings for each assessment factor as witnessed in Table 6.3.

	System Response Accuracy	**Likeability**	**Cognitive Demand**	**Annoyance**	**Habitability**	**Speed**
Mean Satisfaction Rating (1-7)	4.94	5.92	6.056	5.086	4.75	5.925

Table 6.3: Mean subjective satisfaction ratings

System Response Accuracy

Generally, the Skill proved to be responsive during the user evaluations, providing accurate answers to all the questions that it was able to recognize. Furthermore, participants noted that the Skill was consistent throughout and generally did what they expected it to do, whilst doing so in a formal manner. However, there were occasional times during the evaluations where the speech input was not recognized correctly. This is a drawback of the AVS rather than the Skill itself which subsequently impacted upon some of the ratings provided for this factor.

Likeability

It is clear to see from the high likeability rating that the participants found the system to be useful, providing features that they would commonly use. In addition, the participants thought that it was easy to recover from errors at any point during their interaction and concluded that the overall experience was pleasant.

Cognitive Demand

The results of the cognitive demand ratings determined that participants thought the system was intuitive and easy to use. This was further evidenced by the fact that many participants agreed strongly that a high level of concentration is not required to use the Skill successfully.

Annoyance

As would be expected based on the likeability rating and cognitive demand ratings, the annoyance rating was also positive. One of the candidates though who had difficulty getting the Skill to recognize their speech inputs did find the Skill to be frustrating, but later noted that this could have been a result of their non-native accent. Although the participants found the Skill to not be uninteresting, they generally agreed that the interaction was repetitive and thus expressed a desire for more variation in the follow-up questions during dialogue flows.

Habitability

Most of the participants found that the system did contain good contextual awareness, by asking prompt questions where appropriate which helped them to understand where they were within conversation flows and the possible interactions that could be made. The evaluations also made clear that users were aware of how they could interact with the Skill but sometimes found it difficult to think of how they could phrase their utterances for them to be recognizable.

Speed

Participants found that the system responded to their interactions almost instantaneously throughout their experiences. Many positive comments were provided regarding this aspect with one participant stating that “I would rather use this Skill to find information than using the internet” and another that they “like that it provides a hands-free option” for information retrieval.

Conclusions

Overall, the user evaluations can be considered as a success, proving that the final Skill can provide a pleasant experience that can meet the interaction expectations of its potential users. The results also help to demonstrate how many of the projects requirements have been met, in particular the non-functional ones. However, the success of the evaluations cannot be considered in isolation as the need for further work across some areas was made evident.

6.3 Further Development

Further work would firstly involve expanding upon the features currently implemented and meeting the optional requirements that were not fully realised due to time constraints. The evaluations highlighted the need to improve upon current aspects of the interaction model. One of the solutions would involve adding more permutations of synonyms to help invoke each of the intents. Doing so would ensure that the Skill would be more accessible to users wanting to retrieve information by providing a broader scope of interactivity. Another alteration to the interaction model would include adding more varied prompt/validation messages to make the system seem less rigid. The optional requirement of being able to ask for directions around campus would then be prioritised as the next major expansion to the Skill in order to fulfil the original vision that that was laid out. Following these additions and more robust error handling the Skill would then be ready for Skill certification by Amazon.

Additional work would then concern the data used by the Skill, which is considered the next largest priority. Efforts would be placed into ensuring that all data handled by the Skill is collected dynamically at run-time in order to form up-to date responses to questions. This would solve the current time-consuming issue of having to manually update the data used by the Skill. Furthermore, a constraint of the current Skill regarding data that has been web scraped is that it has not been optimised for speech communication and thus in some cases the responses are rather verbose. To this extent it would also be useful to program a solution capable of converting relevant collected data into a form suitable for speech communication.

Another further expansion prompted by the research taken place into current Skills offered by other universities, would include implementing an RSS feature into the Skill. This RSS feature would allow users to ask the Skill for news briefings that would consist of the latest departmental stories. Personalisation could also be added here allowing users to specify which departments of the university they would like to hear news about.

Finally, if the Skill is proved to be successful within DCS, work could commence on increasing the availability of the concepts and principles used within the Skill, implementing a solution for other virtual assistants, e.g. Google Assistant.

Chapter 7

Summary and Conclusions

The aim of this project was to build an Alexa Skill that would allow interested persons to converse with Amazon's Alexa to ask questions and get answers about the Department of Computer Science, University of Sheffield. This was achieved using a hosted cloud-based service to handle Alexa interactions alongside a search engine facility for finding relevant answers.

To identify the feasibility of implementing a solution that addressed the problem, an extensive literature survey was carried out that explored potential types of dialogue system and software development kits that could be used. Included within this was a review of the current skills on offer, which concluded that they incapable of providing any more than single one-step interactions, making this skill unique with its ability to provide multi-turn conversation.

Following on from the research that chapter 2 provided, chapter 3 determined a set of plausible requirements. These requirements were given an order of preference based on their desirability and importance in terms of making the project a success. Chapter 3 also provided comparisons between the potential implementation tools concluding that the Skill would make use of an AWS lambda function programmed in Python alongside a MySQL database for data storage and retrieval.

The designs for the system were then created which took into consideration the ways in which users would expect to be able to interact with the Skill. This resulted in the designing of the intents which considered the slot's that would be required to fulfil them and the utterances that should cause them to trigger. In addition, flow diagrams were created to show how the system would achieve conversational dialog flow to handle interactions similar to those experienced in human conversation to improve the user experience: a major priority of this project.

The Skill itself was then built which brought to fruition the previous system designs. Core Alexa frameworks such as dialog prompts and validations were utilised throughout, to ensure that information could be retrieved and actions completed through varied conversational input, whilst ensuring a degree of contextual awareness. The maintainability of the final solution was also prioritised to allow for additional features not achieved within the project scope and for the easy management of the data used. All the while extensive use case testing took place to secure the stability and reliability of the Skill.

The requirements analysis in chapter 6 demonstrated that all of the mandatory and most of the less desirable requirements had been completed. Furthermore, encompassed within this chapter were the results of the user evaluations. The evaluations themselves simulated real scenarios that potential students would have with the Skill. These evaluations were particularly useful in identifying where the Skill could be improved. Further potential work was also established based on the limitations of the implemented solution due to time constraints, such as returning up-to-date information.

The overall success of the project can be illustrated by the degree to which the requirements were completed, the positive satisfaction of the users that took part in the evaluations and the identifying of a potential for further work.

Bibliography

[1] S. Perez, “39 million Americans now own a smart speaker, report claims,” 17 January 2018. [Online]. Available: https://techcrunch.com/2018/01/12/39-million-americans-now-own-a-smart-speaker-report-claims/. [Accessed 04 October 2018].

[2] Nae, “Global trends for the virtual assistant market,” 27 March 2018. [Online]. Available: https://nae.global/en/global-trends-for-the-virtual-assistant-market/. [Accessed 15 October 2018].

[3] Amazon, “Alexa Voice Service,” [Online]. Available: https://developer.amazon.com/alexa-voice-service. [Accessed 10 October 2018].

[4] Amazon, “AWS Free Tier,” [Online]. Available: https://aws.amazon.com/free/. [Accessed 10 October 2018].

[5] IBM, “IBM Shoebox,” [Online]. Available: http://www-03.ibm.com/ibm/history/exhibits/specialprod1/specialprod1_7.html. [Accessed 16 October 2018].

[6] J. K. Baker, “The DRAGON system - An overview,” *IEEE Transactions on Acoustics, Speech, and Signal Processing,* vol. 23, no. 1, pp. 24-29, February 1975.

[7] B. Lowerre, “The HARPY Speech Recogntion System,” Carnegie Mellon University Pittsburgh, PA, USA ©1976, Pittsburgh, April, 1976.

[8] B. H. Juang and L. R. Rabiner, Automatic Speech Recognition - A Brief History of the Technology Development, Georgia Institute of Technology, Atlanta: Rutgers University and the University of Califronia, Santa Barbara, 10/08/2004.

[9] WikiPedia, “Dragon Naturally Speaking,” [Online]. Available: https://en.wikipedia.org/wiki/Dragon_NaturallySpeaking. [Accessed 26 February 2019].

[10] X. Huang, J. Baker and R. Reddy, “A Historical Perspective of Speech Recognition,” *Communications of the ACM,* vol. 57, no. 1, pp. 94-103, January 2014.

[11] J. Bellegarda, “Natural Language Technology in Mobile Devices: Two Grounding Frameworks,” in *Mobile Speech and Advanced Natural Language Solutions,* Springer, 2013, pp. 185-196.

[12] J. Martin and D. Jurafsky, “Dialog Systems and Chatbots,” in *Speech and Language Processing (3rd ed. draft),* Stanford University, September, 2018.

[13] J. Weizenbaum, “Joseph Weizenbaum: Creator of the 'Eliza' program,” 18 March 2008. [Online]. Available: https://www.independent.co.uk/news/obituaries/professor-joseph-weizenbaum-creator-of-the-eliza-program-797162.html. [Accessed 12 October 2018].

[14] G. Weisz, P. Budzianowski, P. Su and M. Gasic, “Sample Efficient Deep Reinforcement Learning for Dialogue Systems with Large Action Spaces,” 11 February 2018. [Online]. Available: https://arxiv.org/pdf/1802.03753.pdf. [Accessed 24 November 2018].

[15] D. R. Traum and S. Larsson, “The Infromation State Approach To Dialogue Management,” [Online]. Available: http://ict.usc.edu/pubs/The%20Information%20State%20Approach%20to%20Dialogue%20Management.pdf. [Accessed 25 April 2019].

[16] Wikipedia, “Amazon Echo,” [Online]. Available: https://en.wikipedia.org/wiki/Amazon_Echo#Echo. [Accessed 16 October 2018].

[17] “Amazon Echo Dot (3rd Generation),” [Online]. Available: https://www.amazon.co.uk/dp/B0792KWK57/ref=fs_dn. [Accessed 2018 October 16].

[18] “Amazon Echo (2nd Gen),” [Online]. Available: https://www.amazon.co.uk/dp/B06Y5ZW72J/ref=ods_mccc_rd. [Accessed 16 October 2018].

[19] “Amazon Echo Plus (2nd Gen),” [Online]. Available: https://www.amazon.co.uk/dp/B07H3NY1H6/ref=fs_ld. [Accessed 16 October 2018].

[20] “Amazon Echo Spot,” [Online]. Available: https://www.amazon.co.uk/dp/B01J2BK6CO/ref=fs_roo. [Accessed 16 October 2018].

[21] “Amazon Echo Show (2nd Gen),” [Online]. Available: https://www.amazon.co.uk/dp/B07H3ZRTT5/ref=fs_bs. [Accessed 16 October 2018].

[22] Google, “Google Home,” [Online]. Available: https://store.google.com/gb/product/google_home. [Accessed 15 October 2018].

[23] N. Pino and J. Porter, “Google Home review,” [Online]. Available: https://www.techradar.com/uk/reviews/google-home. [Accessed 15 October 2018].

[24] Apple, “Homepod,” [Online]. Available: https://www.apple.com/uk/homepod/. [Accessed 15 October 2018].

[25] G. Beavis, “Apple HomePod review,” [Online]. Available: https://www.techradar.com/uk/reviews/apple-homepod-review. [Accessed 15 October 2018].

[26] Amazon, “Understand the Different Skill Models,” [Online]. Available: https://developer.amazon.com/docs/ask-overviews/understanding-the-different-types-of-skills.html. [Accessed 02 October 2018].

[27] Amazon, “Understand Custom Skills,” [Online]. Available: https://developer.amazon.com/docs/custom-skills/understanding-custom-skills.html. [Accessed 12 October 2018].

[28] Amazon, “Request and Response JSON Reference,” [Online]. Available: https://developer.amazon.com/docs/custom-skills/request-and-response-json-reference.html. [Accessed 17 February 2019].

[29] Amazon, “Learn How to Develop an Alexa Skill,” [Online]. Available: https://developer.amazon.com/alexa-skills-kit/learn. [Accessed 03 November 2018].

[30] Amazon, “skill-sample-python-highlowgame,” [Online]. Available: https://github.com/alexa/skill-sample-python-highlowgame. [Accessed 2 February 2019].

[31] Google, “Actions on Google,” [Online]. Available: https://developers.google.com/actions/extending-the-assistant. [Accessed 12 October 2018].

[32] Google, “How Action Distribution Works,” [Online]. Available: https://developers.google.com/actions/distributing-your-actions. [Accessed 12 October 2018].

[33] Google, “Build Fulfillment,” [Online]. Available: https://developers.google.com/actions/dialogflow/fulfillment. [Accessed 06 March 2019].

[34] Google, “Deploy Fulfillment,” [Online]. Available: https://developers.google.com/actions/dialogflow/deploy-fulfillment. [Accessed 6 March 2019].

[35] L. Dybkjaer, H. hemsen and W. Minker, Evaluation of Text and Speech Systems, Springer, 2007.

[36] M. Walker, D. Litman, C. Kamm and A. Abella, “PARADISE: A Framework for Evaluating Spoken Dialogue Agents,” in *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics,* Madrid, 1997.

[37] S. perez, “Smart speakers hit critical mass in 2018,” TechCrunch, December 2019. [Online]. Available: https://techcrunch.com/2018/12/28/smart-speakers-hit-critical-mass-in-2018/?guccounter=1&guce_referrer_us=aHR0cHM6Ly93d3cuZ29vZ2xlLmNvbS8&guce_referrer_cs=Hq55pGH8g73C0nAU5HqULQ. [Accessed 29 April 2019].

[38] Amazon, “Understand the Flash Briefing Skill API,” [Online]. Available: https://developer.amazon.com/docs/flashbriefing/understand-the-flash-briefing-skill-api.html. [Accessed 14 October 2018].

[39] Sheffield, The University of, “Media Team RSS feeds,” [Online]. Available: https://www.sheffield.ac.uk/news/rss-feeds. [Accessed 14 October 2018].

[40] Amazon, “Fact Skill Tutorial,” [Online]. Available: https://developer.amazon.com/alexa-skills-kit/tutorials/fact-skill-1. [Accessed 14 October 2018].

[41] Amazon, “Leeds Beckett Univeristy Courses Skill,” [Online]. Available: https://alexa.amazon.co.uk/spa/index.html#skills/dp/B07G9GD18W/?ref=skill_dsk_skb_sr_3&qid=1542547426. [Accessed 14 October 2018].

[42] Amazon, “What is AWS Lambda?,” [Online]. Available: https://docs.aws.amazon.com/lambda/latest/dg/welcome.html. [Accessed October 25 2018].

[43] Amazon, “Amazon Relational Database Service (RDS),” [Online]. Available: https://aws.amazon.com/rds/. [Accessed 25 October 2018].

[44] Amazon, “Persistence,” [Online]. Available: https://developer.amazon.com/alexa-skills-kit/big-nerd-ranch/alexa-implementing-persistence. [Accessed 25 October 2018].

[45] Amazon, “Amazon RDS Free Tier,” [Online]. Available: https://aws.amazon.com/rds/free/. [Accessed 25 October 2018].

[46] Amazon, “Amazon RDS Backup and Restore,” [Online]. Available: https://aws.amazon.com/rds/details/backup/. [Accessed 5 February 2019].

[47] MySQL, “MySQL Workbench,” [Online]. Available: https://www.mysql.com/products/workbench/. [Accessed 25 October 2018].

[48] Amazon, “Test Your Skill,” [Online]. Available: https://developer.amazon.com/docs/devconsole/test-your-skill.html. [Accessed 16 November 2018].

[49] University of Sheffield, “Applying for Ethics Approval,” [Online]. Available: https://www.sheffield.ac.uk/rs/ethicsandintegrity/ethicspolicy/educationresources/onlinesystem. [Accessed 12 November 2018].

[50] Amazon, “Create the Interaction Model for Your Skill,” [Online]. Available: https://developer.amazon.com/docs/custom-skills/create-the-interaction-model-for-your-skill.html. [Accessed 18 March 209].

[51] Amazon, “Choose the Invocation Name for a Custom Skill,” [Online]. Available: https://developer.amazon.com/docs/custom-skills/choose-the-invocation-name-for-a-custom-skill.html. [Accessed 18 March 2019].

[52] Amazon, “Manage Skills in the Developer Console,” [Online]. Available: https://developer.amazon.com/docs/devconsole/about-the-developer-console.html. [Accessed 9 April 2019].

[53] Amazon, “Best Practices for Sample Utterances and Custom Slot Type Values,” [Online]. Available: https://developer.amazon.com/docs/custom-skills/best-practices-for-sample-utterances-and-custom-slot-type-values.html. [Accessed 9 April 2019].

[54] Amazon, “Developing Your First Skill,” [Online]. Available: https://alexa-skills-kit-python-sdk.readthedocs.io/en/latest/DEVELOPING_YOUR_FIRST_SKILL.html. [Accessed 11 April 2019].

[55] Amazon, “Request Processing,” [Online]. Available: https://alexa-skills-kit-python-sdk.readthedocs.io/en/latest/REQUEST_PROCESSING.html#standard-request. [Accessed 22 April 2019].

[56] University of Sheffield, “AskUs,” [Online]. Available: https://askus.sheffield.ac.uk/faqs/GuestFaq.aspx?CCTC=PROSTU. [Accessed 10 April 2019].

[57] Amazon, “Slot Type Reference,” [Online]. Available: https://developer.amazon.com/docs/custom-skills/slot-type-reference.html#amazonsearchquery. [Accessed 10 April 2019].

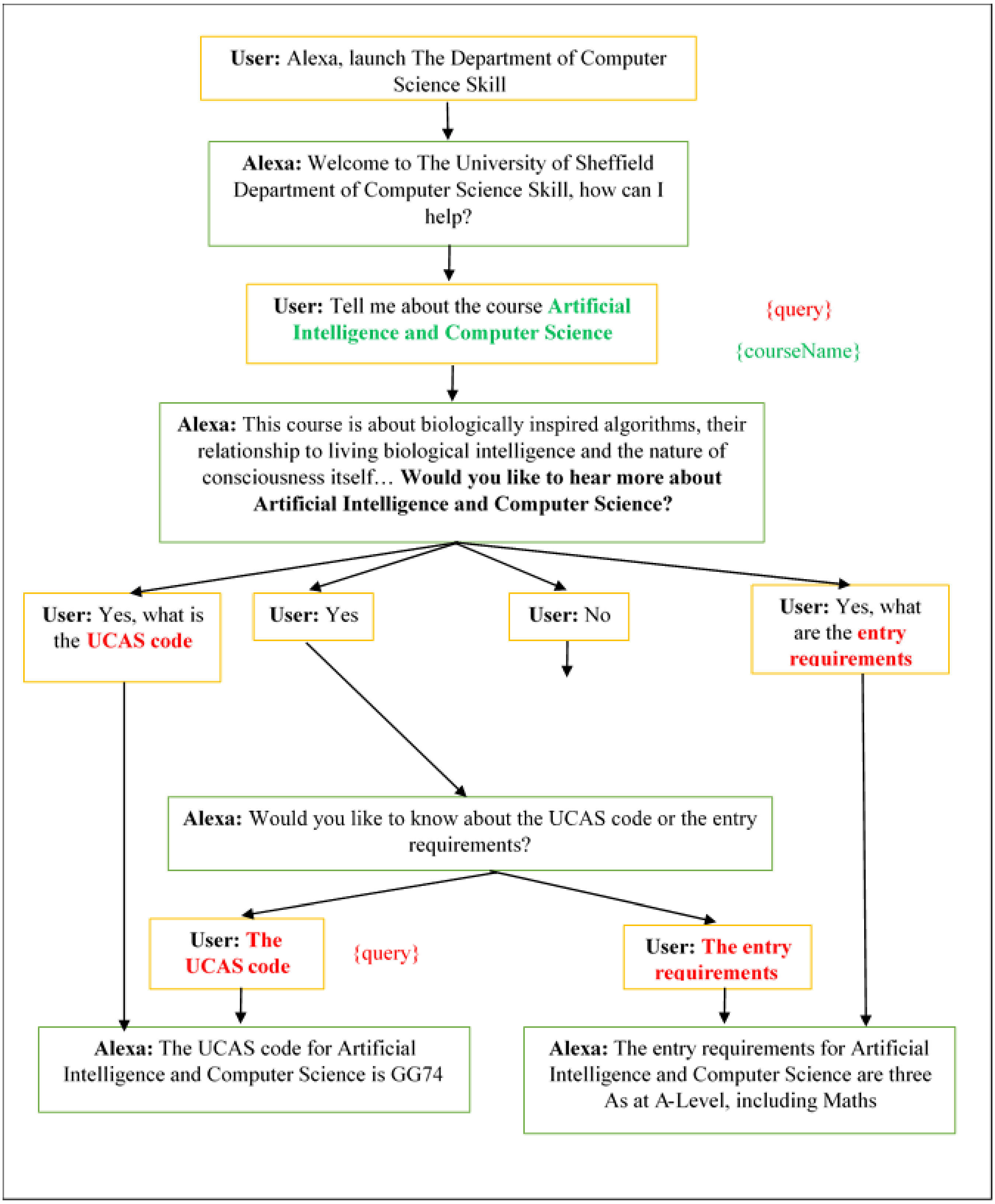
[58] MySQL, “Full-Text Search Functions,” [Online]. Available: https://dev.mysql.com/doc/refman/8.0/en/fulltext-search.html. [Accessed 9 April 2019].

[59] Amazon, “Test Your Utterances and Improve Your Interaction Model,” [Online]. Available: https://developer.amazon.com/docs/custom-skills/test-utterances-and-improve-your-interaction-model.html. [Accessed 14 April 2019].

[60] K. S. Hone and G. Robert, “Subjective Assessment of Speech-System Interface Usability,” January 2001. [Online]. Available: https://www.researchgate.net/publication/221489714_Subjective_assessment_of_speech-system_interface_usability. [Accessed 22 April 2019].

Appendices

Appendix A: System Dialogue Flows



User: Alexa, launch The Department of Computer Science Skill
Alexa: Welcome to The University of Sheffield Department of Computer Science Skill, how can I help?
User: Tell me about the course Artificial Intelligence and Computer Science
{query}
{courseName}
Alexa: This course is about biologically inspired algorithms, their relationship to living biological intelligence and the nature of consciousness itself… Would you like to hear more about Artificial Intelligence and Computer Science?
User: Yes, what is the UCAS code
User: Yes
User: No
User: Yes, what are the entry requirements
Alexa: Would you like to know about the UCAS code or the entry requirements?
User: The UCAS code
{query}
User: The entry requirements
Alexa: The UCAS code for Artificial Intelligence and Computer Science is GG74
Alexa: The entry requirements for Artificial Intelligence and Computer Science are three As at A-Level, including Maths

Figure A.1: Dialogue flow showing how the system can handle course related requests and ask follow-up questions

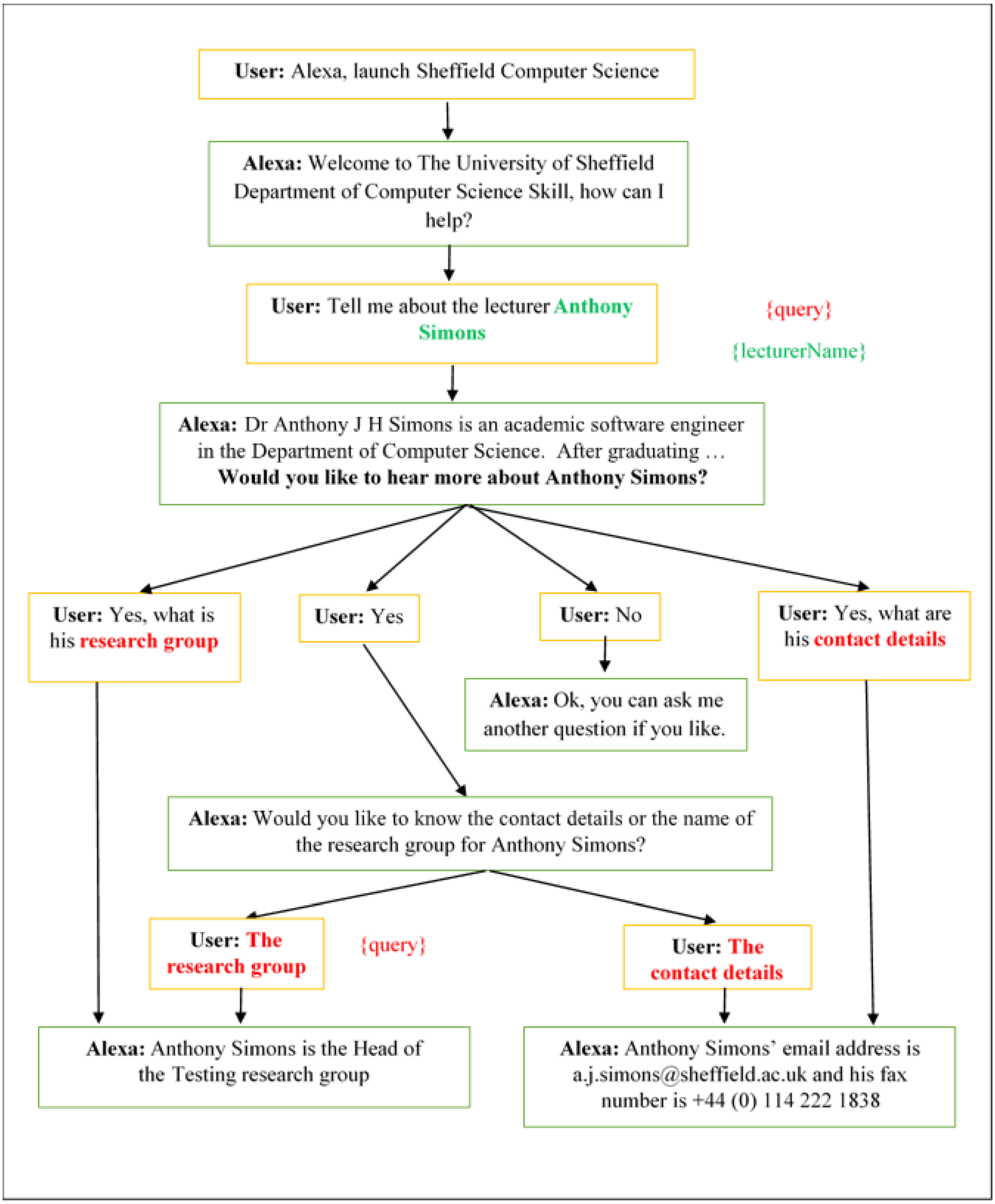

User: Alexa, launch Sheffield Computer Science
Alexa: Welcome to The University of Sheffield Department of Computer Science Skill, how can I help?
User: Tell me about the lecturer Anthony Simons
{query}
{lecturerName}
Alexa: Dr Anthony J H Simons is an academic software engineer in the Department of Computer Science. After graduating … Would you like to hear more about Anthony Simons?
User: Yes, what is his research group
User: Yes
User: No
User: Yes, what are his contact details
Alexa: Ok, you can ask me another question if you like.
Alexa: Would you like to know the contact details or the name of the research group for Anthony Simons?
User: The research group
{query}
User: The contact details
Alexa: Anthony Simons is the Head of the Testing research group
Alexa: Anthony Simons' email address is a.j.simons@sheffield.ac.uk and his fax number is +44 (0) 114 222 1838

Figure A.2: Dialogue flow showing how the system can handle lecturer related requests and ask follow-up questions

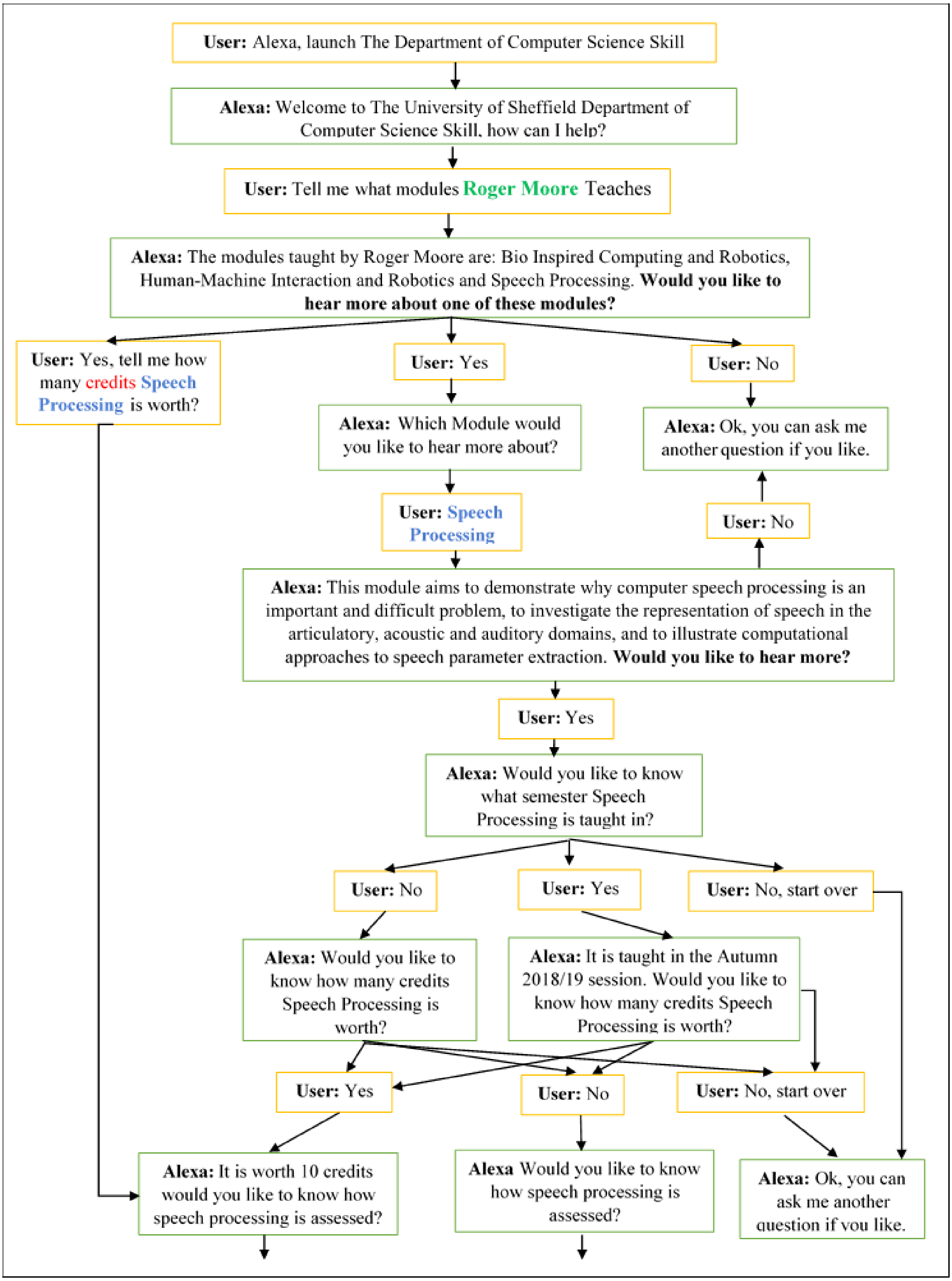
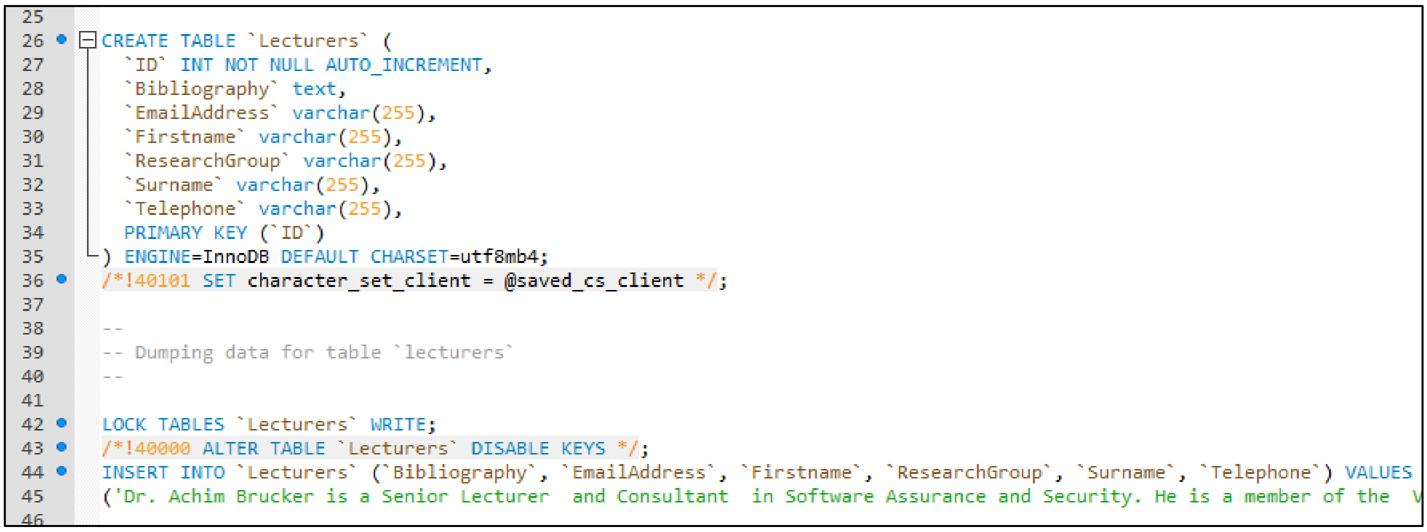

User: Alexa, launch The Department of Computer Science Skill
Alexa: Welcome to The University of Sheffield Department of Computer Science Skill, how can I help?
User: Tell me what modules Roger Moore Teaches
Alexa: The modules taught by Roger Moore are: Bio Inspired Computing and Robotics, Human-Machine Interaction and Robotics and Speech Processing. Would you like to hear more about one of these modules?
User: Yes, tell me how many credits Speech Processing is worth?
User: Yes
User: No
Alexa: Which Module would you like to hear more about?
Alexa: Ok, you can ask me another question if you like.
User: Speech Processing
User: No
Alexa: This module aims to demonstrate why computer speech processing is an important and difficult problem, to investigate the representation of speech in the articulatory, acoustic and auditory domains, and to illustrate computational approaches to speech parameter extraction. Would you like to hear more?
User: Yes
Alexa: Would you like to know what semester Speech Processing is taught in?
User: No
User: Yes
User: No, start over
Alexa: Would you like to know how many credits Speech Processing is worth?
Alexa: It is taught in the Autumn 2018/19 session. Would you like to know how many credits Speech Processing is worth?
User: Yes
User: No
User: No, start over
Alexa: It is worth 10 credits would you like to know how speech processing is assessed?
Alexa Would you like to know how speech processing is assessed?
Alexa: Ok, you can ask me another question if you like.

Figure A.3: Dialogue flow showing how a user can ask about a lecturer's modules and receive information concerning them by answering system follow-up questions

Appendix B: SQL Statement Code Examples

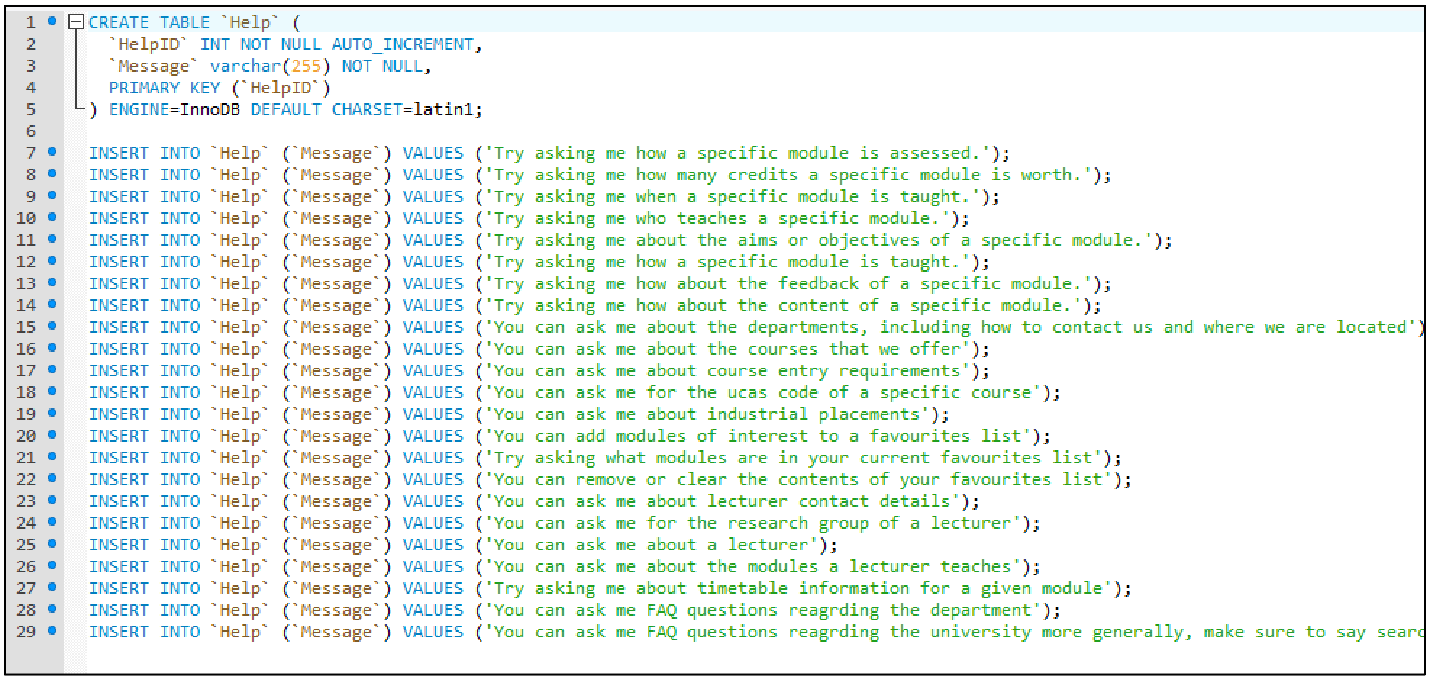


```sql
CREATE TABLE `Lecturers` (
  `ID` INT NOT NULL AUTO_INCREMENT,
  `Bibliography` text,
  `EmailAddress` varchar(255),
  `Firstname` varchar(255),
  `ResearchGroup` varchar(255),
  `Surname` varchar(255),
  `Telephone` varchar(255),
  PRIMARY KEY (`ID`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Dumping data for table `lecturers`
--

LOCK TABLES `Lecturers` WRITE;
/*!40000 ALTER TABLE `Lecturers` DISABLE KEYS */;
INSERT INTO `Lecturers` (`Bibliography`, `EmailAddress`, `Firstname`, `ResearchGroup`, `Surname`, `Telephone`) VALUES
('Dr. Achim Brucker is a Senior Lecturer  and Consultant  in Software Assurance and Security. He is a member of the  V
```

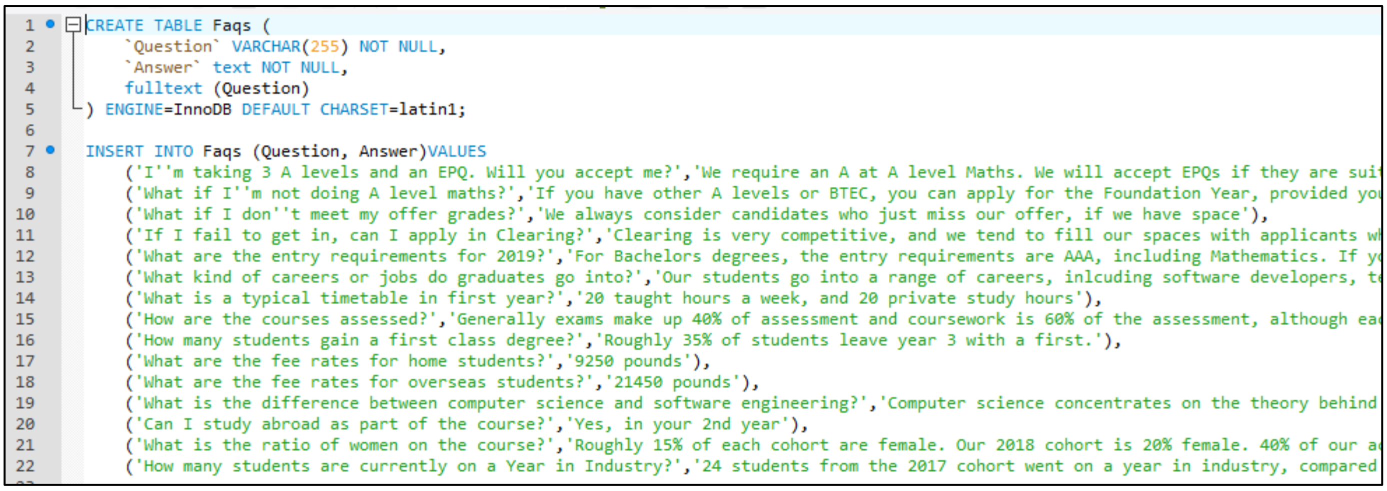
Figure B.1: SQL Showing how web scraped data was inserted into the database



```sql
CREATE TABLE `Help` (
  `HelpID` INT NOT NULL AUTO_INCREMENT,
  `Message` varchar(255) NOT NULL,
  PRIMARY KEY (`HelpID`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

INSERT INTO `Help` (`Message`) VALUES ('Try asking me how a specific module is assessed.');
INSERT INTO `Help` (`Message`) VALUES ('Try asking me how many credits a specific module is worth.');
INSERT INTO `Help` (`Message`) VALUES ('Try asking me when a specific module is taught.');
INSERT INTO `Help` (`Message`) VALUES ('Try asking me who teaches a specific module.');
INSERT INTO `Help` (`Message`) VALUES ('Try asking me about the aims or objectives of a specific module.');
INSERT INTO `Help` (`Message`) VALUES ('Try asking me how a specific module is taught.');
INSERT INTO `Help` (`Message`) VALUES ('Try asking me how about the feedback of a specific module.');
INSERT INTO `Help` (`Message`) VALUES ('Try asking me how about the content of a specific module.');
INSERT INTO `Help` (`Message`) VALUES ('You can ask me about the departments, including how to contact us and where we are located')
INSERT INTO `Help` (`Message`) VALUES ('You can ask me about the courses that we offer');
INSERT INTO `Help` (`Message`) VALUES ('You can ask me about course entry requirements');
INSERT INTO `Help` (`Message`) VALUES ('You can ask me for the ucas code of a specific course');
INSERT INTO `Help` (`Message`) VALUES ('You can ask me about industrial placements');
INSERT INTO `Help` (`Message`) VALUES ('You can add modules of interest to a favourites list');
INSERT INTO `Help` (`Message`) VALUES ('Try asking what modules are in your current favourites list');
INSERT INTO `Help` (`Message`) VALUES ('You can remove or clear the contents of your favourites list');
INSERT INTO `Help` (`Message`) VALUES ('You can ask me about lecturer contact details');
INSERT INTO `Help` (`Message`) VALUES ('You can ask me for the research group of a lecturer');
INSERT INTO `Help` (`Message`) VALUES ('You can ask me about a lecturer');
INSERT INTO `Help` (`Message`) VALUES ('You can ask me about the modules a lecturer teaches');
INSERT INTO `Help` (`Message`) VALUES ('Try asking me about timetable information for a given module');
INSERT INTO `Help` (`Message`) VALUES ('You can ask me FAQ questions reagrding the department');
INSERT INTO `Help` (`Message`) VALUES ('You can ask me FAQ questions reagrding the university more generally, make sure to say searc
```

Figure B.2: SQL Showing how manually defined data was inserted into the database

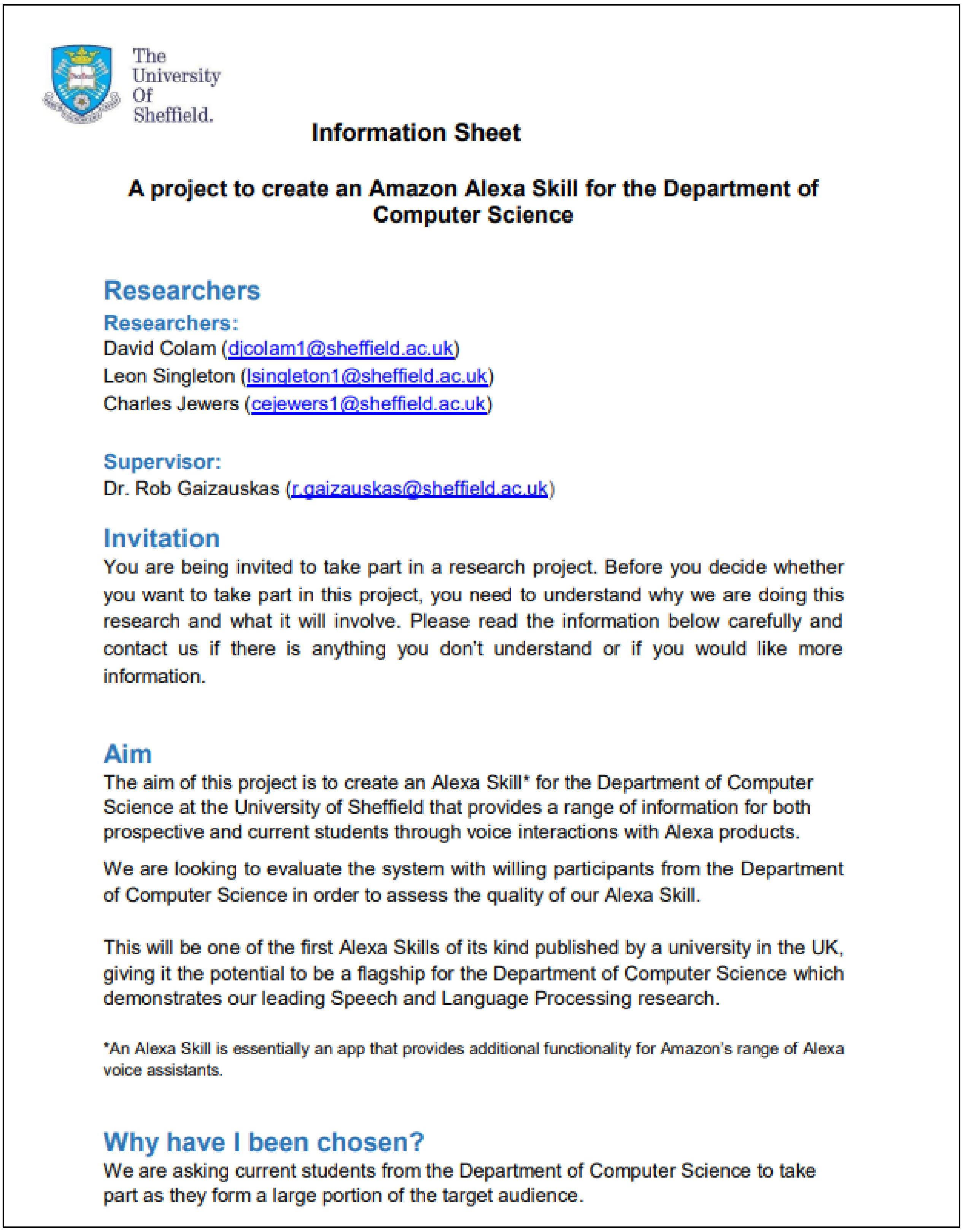


```sql
CREATE TABLE Faqs (
    `Question` VARCHAR(255) NOT NULL,
    `Answer` text NOT NULL,
    fulltext (Question)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

INSERT INTO Faqs (Question, Answer)VALUES
    ('I''m taking 3 A levels and an EPQ. Will you accept me?','We require an A at A level Maths. We will accept EPQs if they are sui
    ('What if I''m not doing A level maths?','If you have other A levels or BTEC, you can apply for the Foundation Year, provided you
    ('What if I don''t meet my offer grades?','We always consider candidates who just miss our offer, if we have space'),
    ('If I fail to get in, can I apply in Clearing?','Clearing is very competitive, and we tend to fill our spaces with applicants wh
    ('What are the entry requirements for 2019?','For Bachelors degrees, the entry requirements are AAA, including Mathematics. If yo
    ('What kind of careers or jobs do graduates go into?','Our students go into a range of careers, inlcuding software developers, te
    ('What is a typical timetable in first year?','20 taught hours a week, and 20 private study hours'),
    ('How are the courses assessed?','Generally exams make up 40% of assessment and coursework is 60% of the assessment, although eac
    ('How many students gain a first class degree?','Roughly 35% of students leave year 3 with a first.'),
    ('What are the fee rates for home students?','9250 pounds'),
    ('What are the fee rates for overseas students?','21450 pounds'),
    ('What is the difference between computer science and software engineering?','Computer science concentrates on the theory behind
    ('Can I study abroad as part of the course?','Yes, in your 2nd year'),
    ('What is the ratio of women on the course?','Roughly 15% of each cohort are female. Our 2018 cohort is 20% female. 40% of our ac
    ('How many students are currently on a Year in Industry?','24 students from the 2017 cohort went on a year in industry, compared
```

Figure B.3: SQL Showing how MySQL Full-Text Search was implemented

Appendix C: User Evaluation Forms



The University Of Sheffield.

Information Sheet

A project to create an Amazon Alexa Skill for the Department of Computer Science

Researchers

Researchers:
David Colam (djcolam1@sheffield.ac.uk)
Leon Singleton (lsingleton1@sheffield.ac.uk)
Charles Jewers (cejewers1@sheffield.ac.uk)

Supervisor:
Dr. Rob Gaizauskas (r.gaizauskas@sheffield.ac.uk)

Invitation

You are being invited to take part in a research project. Before you decide whether you want to take part in this project, you need to understand why we are doing this research and what it will involve. Please read the information below carefully and contact us if there is anything you don't understand or if you would like more information.

Aim

The aim of this project is to create an Alexa Skill* for the Department of Computer Science at the University of Sheffield that provides a range of information for both prospective and current students through voice interactions with Alexa products.

We are looking to evaluate the system with willing participants from the Department of Computer Science in order to assess the quality of our Alexa Skill.

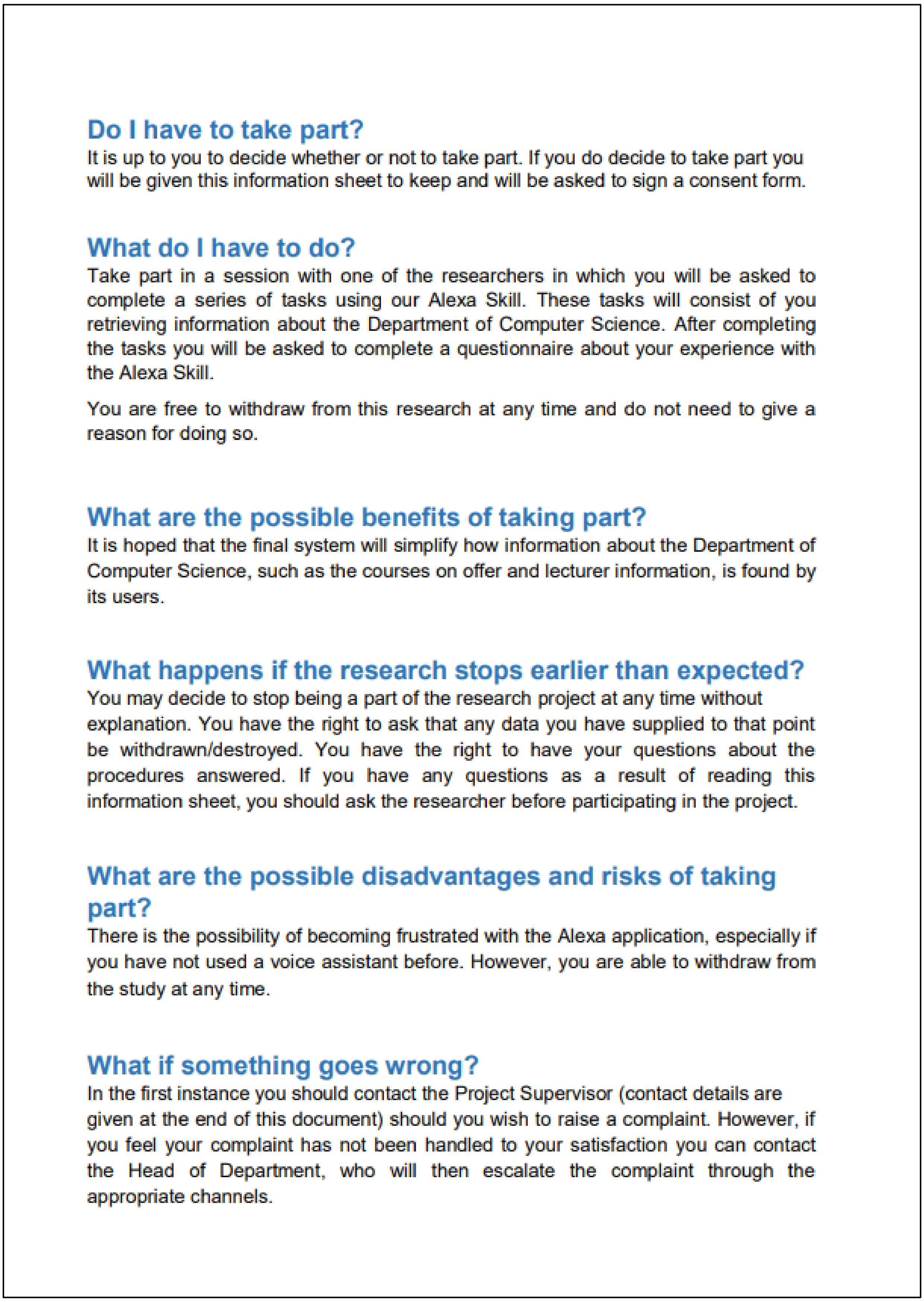
This will be one of the first Alexa Skills of its kind published by a university in the UK, giving it the potential to be a flagship for the Department of Computer Science which demonstrates our leading Speech and Language Processing research.

*An Alexa Skill is essentially an app that provides additional functionality for Amazon's range of Alexa voice assistants.

Why have I been chosen?

We are asking current students from the Department of Computer Science to take part as they form a large portion of the target audience.

Figure C.1: Page 1 of the Information Sheet used in the User Evaluation



Do I have to take part?

It is up to you to decide whether or not to take part. If you do decide to take part you will be given this information sheet to keep and will be asked to sign a consent form.

What do I have to do?

Take part in a session with one of the researchers in which you will be asked to complete a series of tasks using our Alexa Skill. These tasks will consist of you retrieving information about the Department of Computer Science. After completing the tasks you will be asked to complete a questionnaire about your experience with the Alexa Skill.

You are free to withdraw from this research at any time and do not need to give a reason for doing so.

What are the possible benefits of taking part?

It is hoped that the final system will simplify how information about the Department of Computer Science, such as the courses on offer and lecturer information, is found by its users.

What happens if the research stops earlier than expected?

You may decide to stop being a part of the research project at any time without explanation. You have the right to ask that any data you have supplied to that point be withdrawn/destroyed. You have the right to have your questions about the procedures answered. If you have any questions as a result of reading this information sheet, you should ask the researcher before participating in the project.

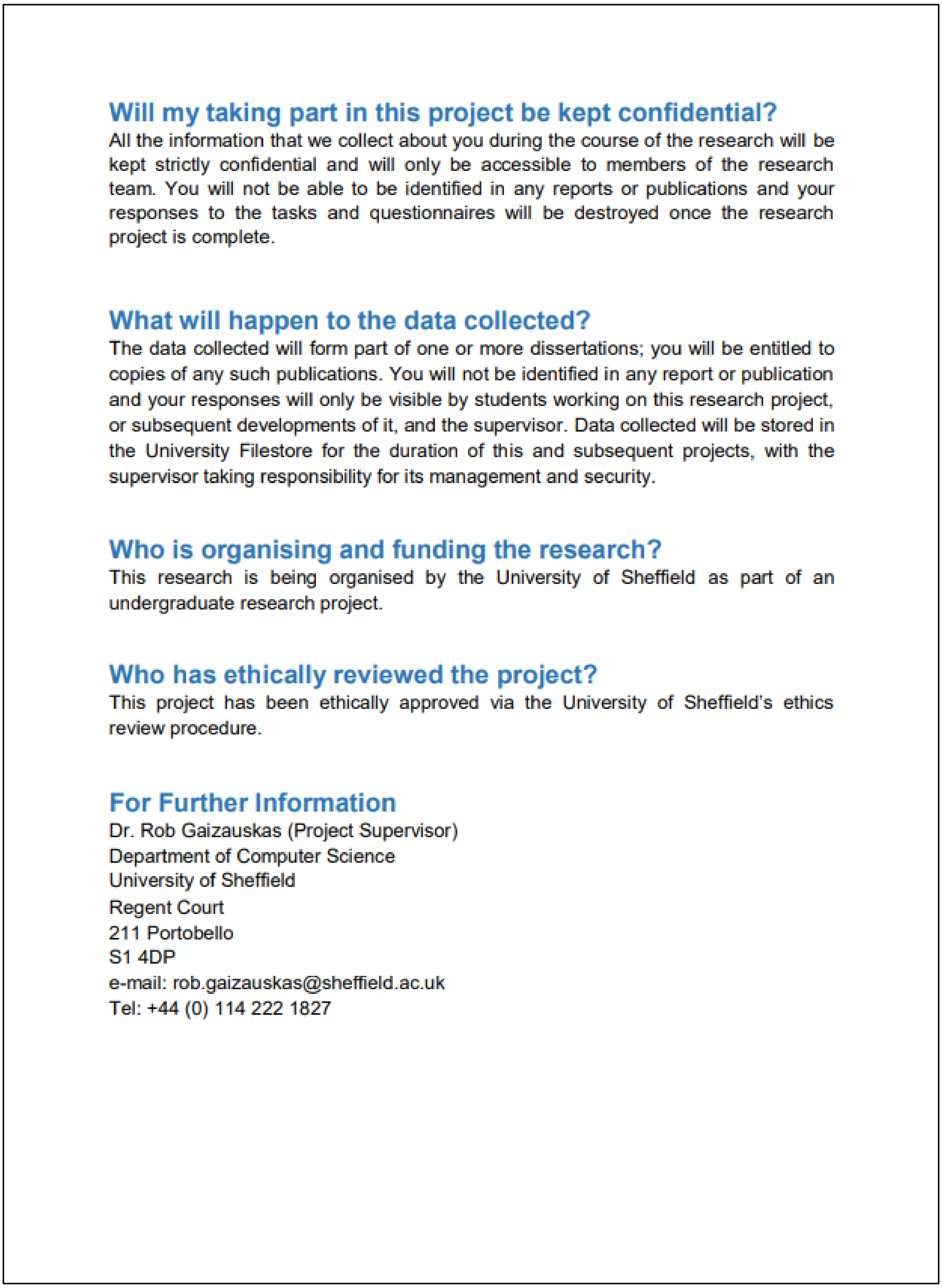
What are the possible disadvantages and risks of taking part?

There is the possibility of becoming frustrated with the Alexa application, especially if you have not used a voice assistant before. However, you are able to withdraw from the study at any time.

What if something goes wrong?

In the first instance you should contact the Project Supervisor (contact details are given at the end of this document) should you wish to raise a complaint. However, if you feel your complaint has not been handled to your satisfaction you can contact the Head of Department, who will then escalate the complaint through the appropriate channels.

Figure C.2: Page 2 of the Information Sheet used in the User Evaluation



Will my taking part in this project be kept confidential?

All the information that we collect about you during the course of the research will be kept strictly confidential and will only be accessible to members of the research team. You will not be able to be identified in any reports or publications and your responses to the tasks and questionnaires will be destroyed once the research project is complete.

What will happen to the data collected?

The data collected will form part of one or more dissertations; you will be entitled to copies of any such publications. You will not be identified in any report or publication and your responses will only be visible by students working on this research project, or subsequent developments of it, and the supervisor. Data collected will be stored in the University Filestore for the duration of this and subsequent projects, with the supervisor taking responsibility for its management and security.

Who is organising and funding the research?

This research is being organised by the University of Sheffield as part of an undergraduate research project.

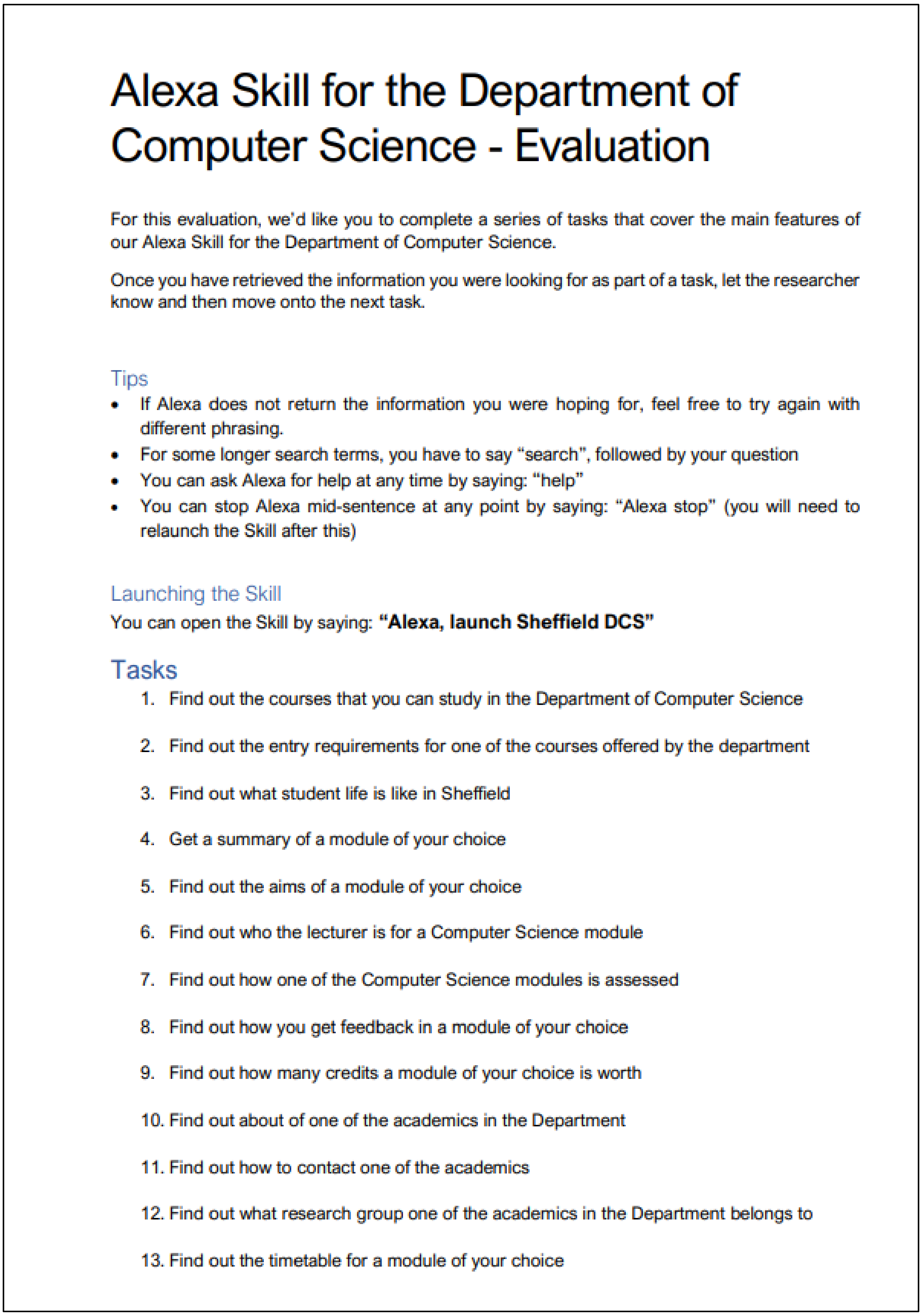
Who has ethically reviewed the project?

This project has been ethically approved via the University of Sheffield's ethics review procedure.

For Further Information

Dr. Rob Gaizauskas (Project Supervisor)
Department of Computer Science
University of Sheffield
Regent Court
211 Portobello
S1 4DP
e-mail: rob.gaizauskas@sheffield.ac.uk
Tel: +44 (0) 114 222 1827

Figure C.3: Page 3 of the Information Sheet used in the User Evaluation



Alexa Skill for the Department of Computer Science - Evaluation

For this evaluation, we'd like you to complete a series of tasks that cover the main features of our Alexa Skill for the Department of Computer Science.

Once you have retrieved the information you were looking for as part of a task, let the researcher know and then move onto the next task.

Tips

- If Alexa does not return the information you were hoping for, feel free to try again with different phrasing.
- For some longer search terms, you have to say "search", followed by your question
- You can ask Alexa for help at any time by saying: "help"
- You can stop Alexa mid-sentence at any point by saying: "Alexa stop" (you will need to relaunch the Skill after this)

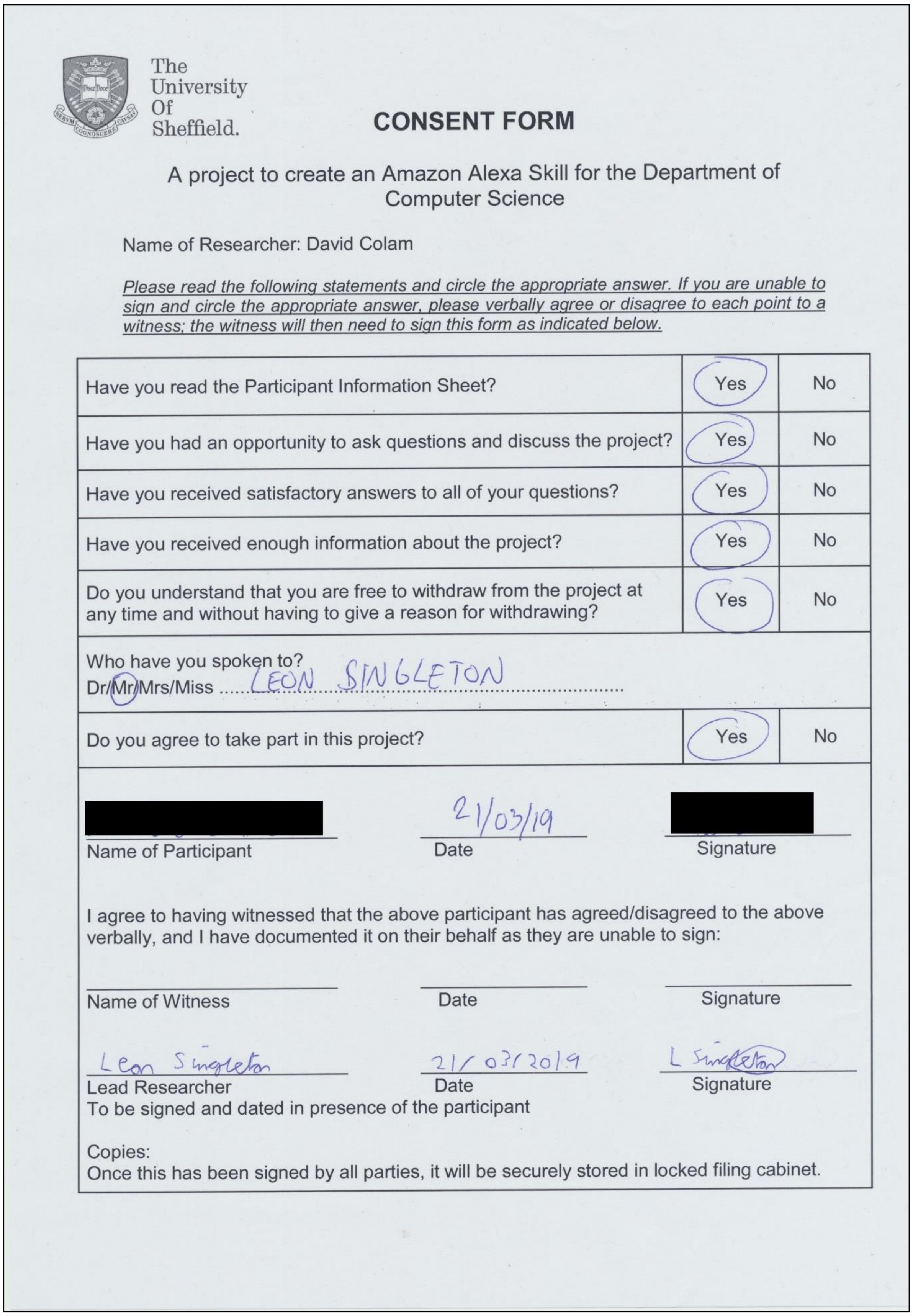
Launching the Skill

You can open the Skill by saying: **"Alexa, launch Sheffield DCS"**

Tasks

1. Find out the courses that you can study in the Department of Computer Science
2. Find out the entry requirements for one of the courses offered by the department
3. Find out what student life is like in Sheffield
4. Get a summary of a module of your choice
5. Find out the aims of a module of your choice
6. Find out who the lecturer is for a Computer Science module
7. Find out how one of the Computer Science modules is assessed
8. Find out how you get feedback in a module of your choice
9. Find out how many credits a module of your choice is worth
10. Find out about of one of the academics in the Department
11. Find out how to contact one of the academics
12. Find out what research group one of the academics in the Department belongs to
13. Find out the timetable for a module of your choice

Figure C.4: Task sheet provided to User Evaluation candidates



The University Of Sheffield.

CONSENT FORM

A project to create an Amazon Alexa Skill for the Department of Computer Science

Name of Researcher: David Colam

Please read the following statements and circle the appropriate answer. If you are unable to sign and circle the appropriate answer, please verbally agree or disagree to each point to a witness; the witness will then need to sign this form as indicated below.

Have you read the Participant Information Sheet?	[✓ Yes]	No
Have you had an opportunity to ask questions and discuss the project?	[✓ Yes]	No
Have you received satisfactory answers to all of your questions?	[✓ Yes]	No
Have you received enough information about the project?	[✓ Yes]	No
Do you understand that you are free to withdraw from the project at any time and without having to give a reason for withdrawing?	[✓ Yes]	No
Who have you spoken to? Dr/[✓ Mr]/Mrs/Miss ....LEON SINGLETON....		
Do you agree to take part in this project?	[✓ Yes]	No

[redacted]	21/03/19	[redacted]
Name of Participant	Date	Signature

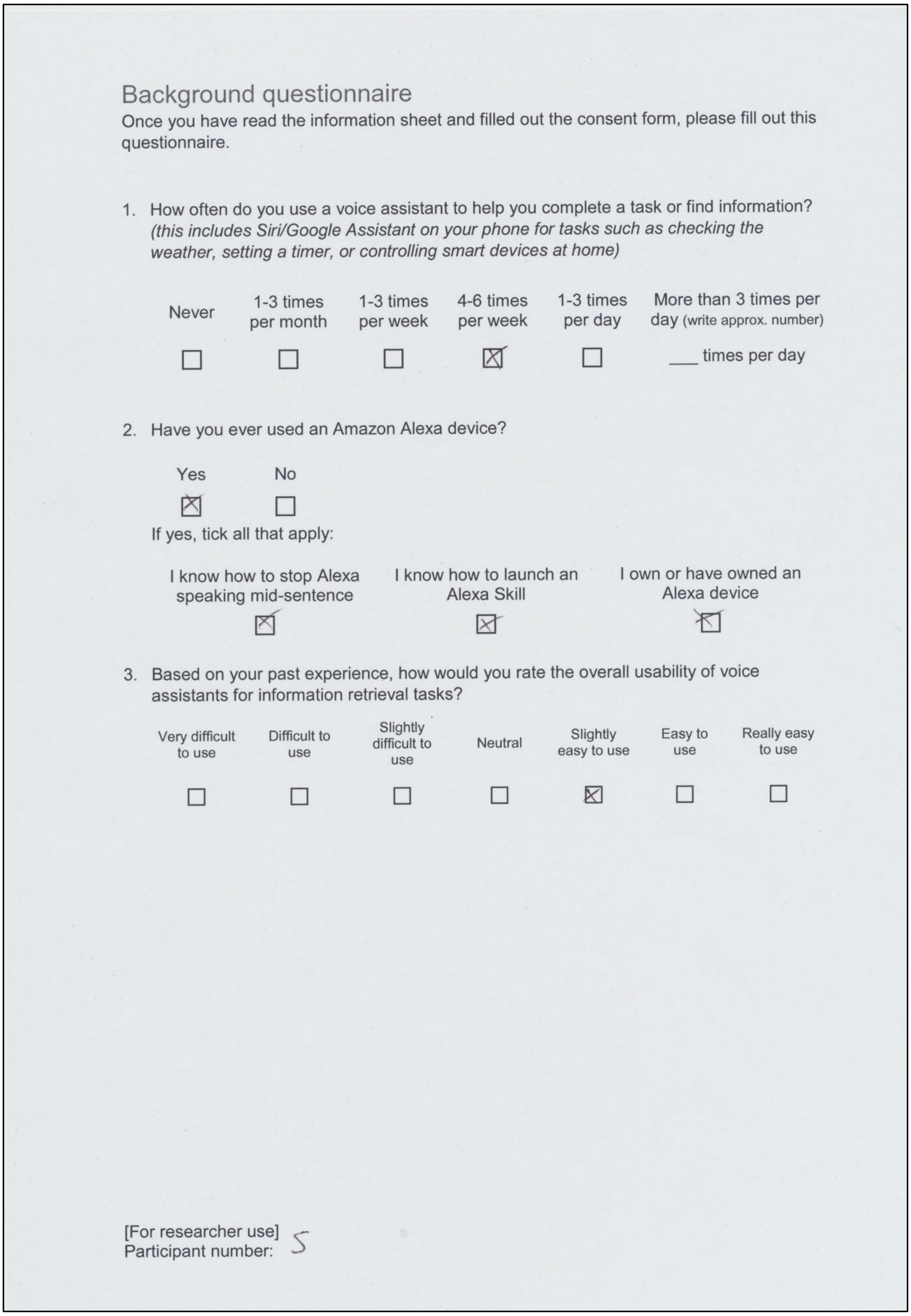
I agree to having witnessed that the above participant has agreed/disagreed to the above verbally, and I have documented it on their behalf as they are unable to sign:

Name of Witness	Date	Signature
Leon Singleton	21/03/2019	L Singleton
Lead Researcher	Date	Signature

To be signed and dated in presence of the participant

Copies:
Once this has been signed by all parties, it will be securely stored in locked filing cabinet.

Figure C.5: Scan of an anonymised Consent Form used by candidates to confirm their participation in the Evaluation



Background questionnaire

Once you have read the information sheet and filled out the consent form, please fill out this questionnaire.

1. How often do you use a voice assistant to help you complete a task or find information? *(this includes Siri/Google Assistant on your phone for tasks such as checking the weather, setting a timer, or controlling smart devices at home)*

Never	1-3 times per month	1-3 times per week	4-6 times per week	1-3 times per day	More than 3 times per day (write approx. number)
☐	☐	☐	☒	☐	___ times per day

2. Have you ever used an Amazon Alexa device?

Yes	No
☒	☐

If yes, tick all that apply:

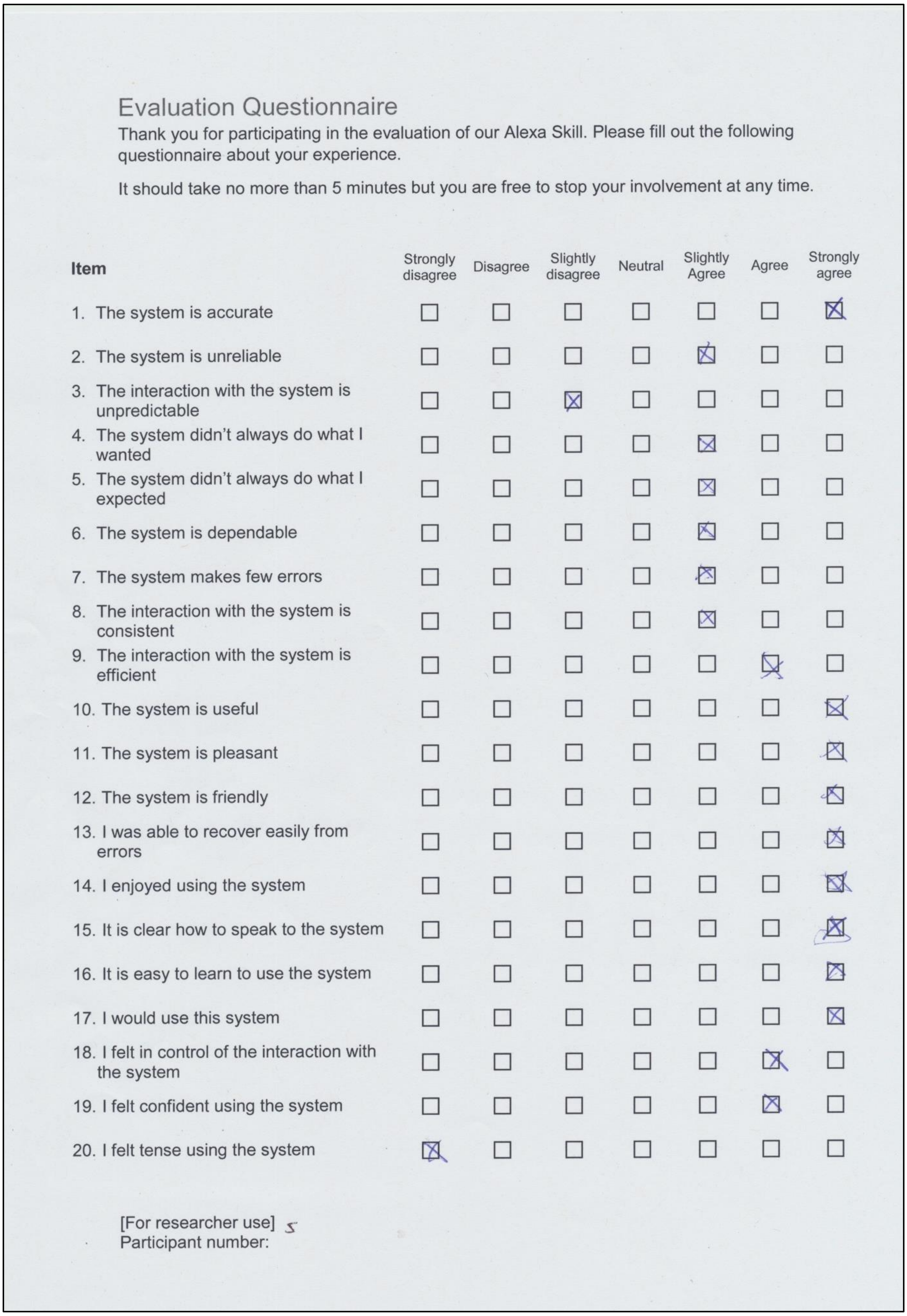
I know how to stop Alexa speaking mid-sentence	I know how to launch an Alexa Skill	I own or have owned an Alexa device
☒	☒	☒

3. Based on your past experience, how would you rate the overall usability of voice assistants for information retrieval tasks?

Very difficult to use	Difficult to use	Slightly difficult to use	Neutral	Slightly easy to use	Easy to use	Really easy to use
☐	☐	☐	☐	☒	☐	☐

[For researcher use]
Participant number: 5

Figure C.6: Scan of the Background Questionnaire results completed by candidate 5



Evaluation Questionnaire

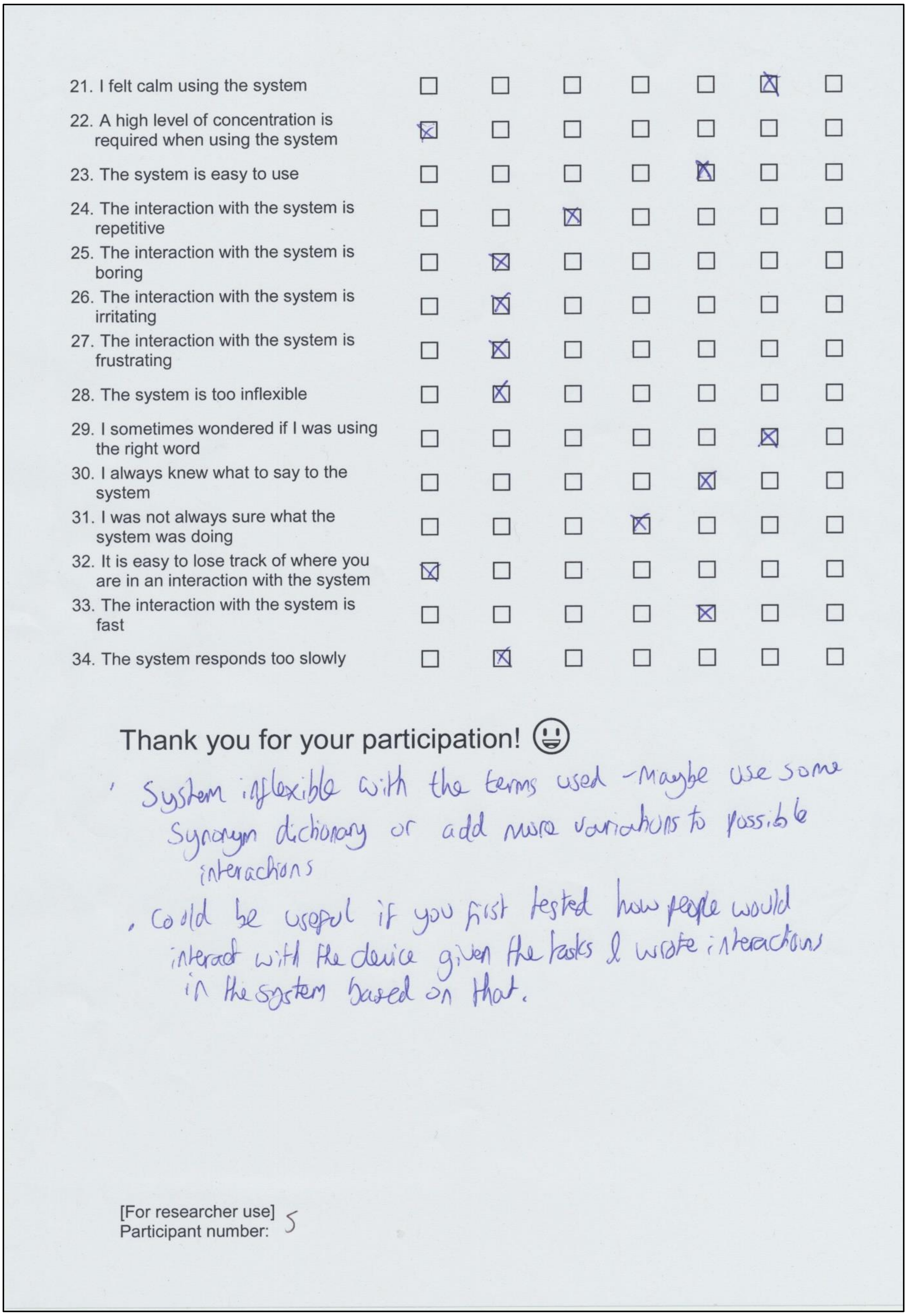
Thank you for participating in the evaluation of our Alexa Skill. Please fill out the following questionnaire about your experience.

It should take no more than 5 minutes but you are free to stop your involvement at any time.

Item	Strongly disagree	Disagree	Slightly disagree	Neutral	Slightly Agree	Agree	Strongly agree
1. The system is accurate	☐	☐	☐	☐	☐	☐	☒
2. The system is unreliable	☐	☐	☐	☐	☒	☐	☐
3. The interaction with the system is unpredictable	☐	☐	☒	☐	☐	☐	☐
4. The system didn't always do what I wanted	☐	☐	☐	☐	☒	☐	☐
5. The system didn't always do what I expected	☐	☐	☐	☐	☒	☐	☐
6. The system is dependable	☐	☐	☐	☐	☒	☐	☐
7. The system makes few errors	☐	☐	☐	☐	☒	☐	☐
8. The interaction with the system is consistent	☐	☐	☐	☐	☒	☐	☐
9. The interaction with the system is efficient	☐	☐	☐	☐	☐	☒	☐
10. The system is useful	☐	☐	☐	☐	☐	☐	☒
11. The system is pleasant	☐	☐	☐	☐	☐	☐	☒
12. The system is friendly	☐	☐	☐	☐	☐	☐	☒
13. I was able to recover easily from errors	☐	☐	☐	☐	☐	☐	☒
14. I enjoyed using the system	☐	☐	☐	☐	☐	☐	☒
15. It is clear how to speak to the system	☐	☐	☐	☐	☐	☐	☒
16. It is easy to learn to use the system	☐	☐	☐	☐	☐	☐	☒
17. I would use this system	☐	☐	☐	☐	☐	☐	☒
18. I felt in control of the interaction with the system	☐	☐	☐	☐	☐	☒	☐
19. I felt confident using the system	☐	☐	☐	☐	☐	☒	☐
20. I felt tense using the system	☒	☐	☐	☐	☐	☐	☐

[For researcher use] 5
Participant number:

Figure C.7: Scan of page 1 of the Evaluation Questionnaire completed by candidate 5



21. I felt calm using the system	☐	☐	☐	☐	☐	☒	☐
22. A high level of concentration is required when using the system	☒	☐	☐	☐	☐	☐	☐
23. The system is easy to use	☐	☐	☐	☐	☒	☐	☐
24. The interaction with the system is repetitive	☐	☐	☒	☐	☐	☐	☐
25. The interaction with the system is boring	☐	☒	☐	☐	☐	☐	☐
26. The interaction with the system is irritating	☐	☒	☐	☐	☐	☐	☐
27. The interaction with the system is frustrating	☐	☒	☐	☐	☐	☐	☐
28. The system is too inflexible	☐	☒	☐	☐	☐	☐	☐
29. I sometimes wondered if I was using the right word	☐	☐	☐	☐	☐	☒	☐
30. I always knew what to say to the system	☐	☐	☐	☐	☒	☐	☐
31. I was not always sure what the system was doing	☐	☐	☐	☒	☐	☐	☐
32. It is easy to lose track of where you are in an interaction with the system	☒	☐	☐	☐	☐	☐	☐
33. The interaction with the system is fast	☐	☐	☐	☐	☒	☐	☐
34. The system responds too slowly	☐	☒	☐	☐	☐	☐	☐

Thank you for your participation! 😀

, System inflexible with the terms used - Maybe use some Synonym dictionary or add more variations to possible interactions

. Could be useful if you first tested how people would interact with the device given the tasks I wrote interactions in the system based on that.

[For researcher use]
Participant number: 5

Figure C.8: Scan of page 2 of the Evaluation Questionnaire completed by candidate 5

Appendix D: User Evaluation Results

		Mean	Weighting factor (+/-)
System Response accuracy	1. The system is accurate	5.43	1.43
	2. The system is unreliable	3.14	0.86
	3. The interaction with the system is unpredictable	3.00	1.00
	4. The system didn’t always do what I wanted	4.00	0.00
	5. The system didn’t always do what I expected	3.71	0.29
	6. The system is dependable	5.14	1.14
	7. The system makes few errors	5.14	1.14
	8. The interaction with the system is consistent	5.29	1.29
	9. The interaction with the system is efficient	5.29	1.29
Likeability	10. The system is useful	6.29	2.29
	11. The system is pleasant	5.57	1.57
	12. The system is friendly	6.43	2.43
	13. I was able to recover easily from errors	6.00	2.00
	14. I enjoyed using the system	6.29	2.29
	15. It is clear how to speak to the system	5.86	1.86
	16. It is easy to learn to use the system	5.71	1.71
	17. I would use this system	5.86	1.86
	18. I felt in control of the interaction with the system	5.29	1.29
Cognitive demand	19. I felt confident using the system	6.00	2.00
	20. I felt tense using the system	1.29	2.71
	21. I felt calm using the system	6.14	2.14
	22. A high level of concentration is required when using the system	2.00	2.00
	23. The system is easy to use	5.43	1.43

Annoyance	24. The interaction with the system is repetitive	4.00	0.00
	25. The interaction with the system is boring	2.43	1.57
	26. The interaction with the system is irritating	3.00	1.00
	27. The interaction with the system is frustrating	2.57	1.43
	28. The system is too inflexible	2.57	1.43
Habitability	29. I sometimes wondered if I was using the right word	4.29	-0.29
	30. I always knew what to say to the system	5.14	1.14
	31. I was not always sure what the system was doing	2.86	1.14
	32. It is easy to lose track of where you are in an interaction with the system	3.00	1.00
Speed	33. The interaction with the system is fast	5.71	1.71
	34. The system responds too slowly	1.86	2.14

Figure D.1: Table showing the mean results from the user evaluation attitude statements and their weighting factors