ALEXA SKILL VOICE INTERFACE FOR THE MOODLE LEARNING MANAGEMENT SYSTEM

A Thesis
by
MICHELLE LAYNE MELTON

Submitted to the Graduate School
Appalachian State University
in partial fulfillment of the requirements for the degree of
MASTER OF SCIENCE

May 2019
Department of Computer Science

ALEXA SKILL VOICE INTERFACE FOR THE
MOODLE LEARNING MANAGEMENT SYSTEM

A Thesis
by
MICHELLE LAYNE MELTON
May 2019

APPROVED BY:

Dr. James B. Fenwick Jr.
Chairperson, Thesis Committee

Dr. Cindy Norris
Member, Thesis Committee

Dr. Rahman Tashakkori
Member, Thesis Committee

Dr. Rahman Tashakkori
Chairperson, Department of Computer Science

Michael J. McKenzie, Ph.D.
Dean, Cratis D. Williams School of Graduate Studies

Abstract

ALEXA SKILL VOICE INTERFACE FOR THE
MOODLE LEARNING MANAGEMENT SYSTEM

Michelle Layne Melton
B.S., University of Baltimore
M.S., Appalachian State University

Chairperson: Dr. James B. Fenwick Jr.

Most educational and training organizations today use some type of learning management system (LMS) to make course material available online to participants. An LMS can be used for face-to-face, fully online, or hybrid courses incorporating versions of both. Learning management system users want easy and fast access to learning materials. LMS access is typically provided through an online interface or a mobile application, both of which require the use of touch and sight on a computer or device. With the rapid growth of technology advancements and user knowledge, LMS users expect faster and more convenient access.

The last decade has brought considerable progress in voice technology. Significant improvement in the accuracy of speech to text translation has made the use of voice-enabled devices more common. Since both technology and usage are continuing to grow, voice interfaces will become even more important for modern applications.

Two of the top three LMS frameworks on the market today have voice interfaces. Both Blackboard Learn and Canvas by Instructure have Amazon Alexa skill integrations that provide basic course information such as announcements, assignments, and grades.

Presently, there is no distributed voice integration for Moodle, the second-ranked LMS provider.

The purpose of this thesis is to develop a voice user interface for a learning management system: specifically, an Amazon Alexa skill for Moodle. The research thoroughly outlines the process of developing an Alexa skill for Moodle, including:

- Researching and making informed decisions about the interface design;
- Developing effective prototypes for early feedback on the design;
- Learning the Alexa Skills Kit for the front-end development of the skill;
- Implementing the Moodle API for the development of the back-end web service for the skill; and
- Planning and conducting effective usability testing sessions and evaluating results.

An Alexa skill integration with Moodle will allow users to more quickly and conveniently access information from the LMS. Immediate benefits of the project include providing site announcements to all users, course announcements to students and teachers, and overall course grades and upcoming due dates to students. In the future, the application may be expanded to implement instructor capabilities like getting a list of assignments that need grading and the ability to create voice activities for students. Future development may also include providing additional course content for students, such as attendance, missing assignments, and instructor contact information.

Acknowledgements

I was extremely lucky to be supported by many incredible mentors, advisors, team members, friends, and family throughout my thesis work. First and foremost, I would like to thank Dr. Fenwick, since it was he who motivated me to embark on the graduate school journey in the first place. He has also been an inspiring teacher and mentor, in addition to providing guidance during the many months of work on my thesis.

I would like to thank Dr. Parry for his direction during the initial stages of my project. With no prior experience in academic research and writing of this caliber, the tools he provided during the Seminar class were of immense help to me. I would also like to express my sincere gratitude to my committee for their feedback during the final stages of my thesis. Their comments and support were extremely helpful in pushing the project to be the best it could be.

The amazing team I have the honor of working with in the Center for Academic Excellence at Appalachian provided invaluable support in helping me prototype my application, as well as reaching out to the entire campus of faculty and students for recruiting survey and usability test participants.

My friends and family are always the greatest sources of support, motivation, and strength, and their help throughout the development of my thesis as well as through my entire graduate academic career is something for which I will be forever grateful. Their love and encouragement are what keep me moving forward.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Today, most educational and training organizations make at least some of their course content available on a learning management system (LMS). Whether the courses are online, face-to-face, or hybrid, an LMS makes course announcements, discussions, assignments, files, quizzes, and grades available online. In addition to assisting with instructor course administration, this platform increases the availability of course material and can help facilitate communication between course members.

Students rank easy access and fast access to learning materials as second and third in importance for LMS needs [6]. Most learning management systems attempt to meet these needs with an online interface as well as some type of mobile access, whether it be the responsive design of their web interface or an integrated mobile application. With both innovations in technology and user savviness growing rapidly, LMS users want even faster and more convenient access to course material than the online and mobile interfaces can provide.

Historically, the biggest challenge for voice user interfaces (VUIs) was the accurate translation of speech to text [10]. Modern voice technology has improved significantly in the past decade; the speech recognition error rate is now only about 8% [25]. With such a dramatic improvement in the usability of voice-enabled devices, they are becoming more commonplace. In fact, 20% of Google searches are now performed by voice [23].

Many popular applications have already started integrating voice interfaces, including some LMS frameworks. Blackboard Learn and Canvas by Instructure, two of the top three learning management systems, implemented voice interfaces in 2017. Each application has an Amazon Alexa skill that provides standard course content like announcements, assignments, and grades. There is currently no distributed voice integration for Moodle, the second-ranked LMS provider.

This thesis describes the development of an Amazon Alexa skill that enhances the speed and convenience of accessing information in the Moodle LMS. Current features include providing all users access to public site announcements and enabling student access to course announcements, grades, and upcoming due dates. Future development may expand functionality to include instructor actions, such as accessing assignments that need grading and possibly even creating voice interactive assignments, as well as expanding the content available to students.

Chapter 2 provides background on voice interface design, voice assistant and smart speaker technology, and learning management systems. Chapter 3 discusses existing voice interfaces for the top three learning management systems. Chapter 4 details the complete process for the development of the Alexa skill, which includes planning the voice interface design, creating the front-end, and building and implementing the back-end web service. Chapter 5 outlines the technical contributions provided by the voice application, as well as the planning and execution of usability tests and the evaluation of results. Chapter 6 summarizes the overall project and describes enhancements of the application's functionality, as well as other areas of research for the future.

Chapter 2

Background

The related areas of interest for this thesis that require some background explanation include voice interface design, prototyping, and usability testing; the Amazon Alexa voice assistant and smart speakers; and learning management systems, specifically the Moodle open source LMS.

2.1 Voice Interface Design

General interface design principles can and should be applied to creating voice applications, but a few characteristics of voice user interfaces require special consideration in their design. The auditory modality, rather than visual or tactile, and the use of spoken language for the interaction are two such characteristics [7].

Auditory interactions differ from visual ones in that they are sequential, dynamic, and transient. A graphic interface can present multiple pieces of information all at once, the information typically stays the same, and the user can refer back to the information as needed. Conversely, voice interfaces present information one word at a time, the information is constantly changing, and there is no permanent record of what was said [10]. The unique characteristics of voice applications can place cognitive demands on users by requiring them to use short-term memory and to move at a predetermined pace [7]. It is important to take these cognitive issues into account during the design of the voice interface.

In addition to the cognitive considerations, there are also important social aspects to recognize in the design of speech interfaces. Designing effective voice experiences requires learning and applying the principles of spoken language from the start, not just converting a graphic user interface to an auditory one [9]. Even though people don't talk to computers the same way they talk to other people [10], they tend to apply similar rules of conversation [21]. These rules are not explicit; people are typically unaware of the protocol until they encounter a response that seems uncooperative. The question "Do you have the time?" is a good example. If a person responds only with "Yes," they are likely to be perceived as rude because the question implies a request for the actual time. Since these rules are not explicit, users who experience uncooperative responses typically describe them in terms of politeness or personality, not as violating this conversational protocol [10]. Even though the protocol is inconspicuous, it should be taken into account during the design of a voice interface, as it can have a substantial impact on a user's experience with the system.

Due to the differences between visual and voice interfaces, standard prototyping for user feedback early in the design process has to be modified for voice interactions. Graphic interfaces use wireframes or paper prototypes to test basic interaction independent of the presentation, or look and feel, of the application. The interaction layer (the dialog and responses of the system) and the presentation layer (the voice, word choice, and speaking rate of the system) are more connected in a voice application. Since these components are so coupled in a voice interface, both should be included in prototypes. Prompts should be fully scripted for the interaction layer, so the user's ability to complete a task is not impaired. The production voice should be used for the presentation layer because pitch and pace (the personality of the system) can affect a user's evaluation of the interface [10].

Most common usability testing techniques are still applicable for voice user interfaces; however, some need to be adjusted to account for the nuances of speech interactions. Traditional think-aloud techniques, where users narrate their experience while using the application, interrupt the voice interaction. Pausing the interaction to allow spoken user feedback also influences the experience and may make it difficult for users to pick up the interaction where they left off. Retrospective think-aloud is the preferred alternative for voice interfaces. This technique involves the researcher taking notes and/or videotaping the interaction and using these records to interview the user after the interaction is complete [10].

Leaders in the user interface design industry have outlined effectiveness, efficiency, and satisfaction as the three primary areas of usability measurement. During usability testing, effectiveness can be measured as the rate of successful completion of tasks and efficiency can be measured as the time to successfully complete tasks. The assessment of satisfaction is a bit more complicated because it is based on subjective aspects of the interface. Standardized questionnaires provide a simplified way to obtain this subjective evaluation from users during testing. The maximally reduced version of the Speech User Interface Service Quality questionnaire (SUISQ-MR) is a condensed version of the SUISQ developed to assess the usability of interactive voice response applications with an element of focus on satisfaction with customer service. The SUISQ-MR questionnaire covers four factors of usability: user goal orientation, customer service behavior, speech characteristics, and verbosity. Scores for each factor are calculated as the mean of each item's scores, and the overall score is the mean of the factor scores after reversing the verbosity mean [15].

2.2 Amazon Alexa Voice Assistant and Smart Speakers

Similar to Google Home and Apple's Siri voice assistant, Alexa is Amazon's cloud-based voice service available on Alexa devices like the Echo and Echo Dot, as well as Alexa companion devices like the Fire tablet and Fire TV. Alexa skills are apps that enable voice-activated capabilities for connected smart devices and online services. The subsections that follow provide definitions for several key terms, an overview of the architecture behind Alexa devices and skills, and more in-depth descriptions of the front-end and back-end that comprise that architecture.

2.2.1 Terminology

This thesis uses a number of terms that may have different meanings in different contexts or are used interchangeably, particularly in mass media. To help clarify the discussions in this thesis, some key terms are defined here.

Alexa is Amazon's digital voice assistant cloud service, available on enabled devices.

An **Alexa account** is a user account with Alexa and is managed by Amazon.

The **Alexa app** is a website and app built by Amazon for managing a user's Alexa account, devices, music, lists, skills, and Alexa-related settings.

Alexa devices are smart speakers and devices with the ability to use Alexa, like the Amazon Echo and Echo Dot and Alexa companion devices like the Fire tablet and Fire TV.

Alexa skills are third-party apps that provide voice-activated capabilities for connected smart devices and online services. These are often simply referred to as skills.

The **Alexa Skills Kit (ASK)** is a set of tools, including documentation and code samples, provided by Amazon that help developers to build skills.

The **back-end** of an Alexa skill is the data access layer, typically connecting the front-end interface with the data or service the user needs to access.

A **cloud service** is a service made available through resources provided over the internet. Internet-enabled devices typically connect to a companion cloud service.

The **dialog model** defines the properties to be included in skill requests and responses for conversation between the skill and a user so the information necessary to fulfill an intent can be collected.

The **front-end** of an Alexa skill is the presentation layer, typically involving user interaction. For a voice user interface, this includes the dialog model and speech processing.

An **intent** is a core functionality of a skill. In other contexts, these are sometimes referred to as features, use cases, or user stories.

The **interaction model** defines the voice interface and mapping of users' spoken input to intents within the skill.

The **invocation name** is the name of a skill, spoken by users to begin an interaction with the skill. It is the voice equivalent to locating the name and icon of a mobile app.

An **invocation phrase** is the combination of the invocation name of a skill and an utterance, used to invoke a skill with a specific request. An example is "Ask Daily Horoscopes for the horoscope for Gemini."

A **slot** is a list of possible arguments passed to a skill through an utterance and defined in an intent. A slot in an Alexa skill is comparable to a method parameter or a command line argument. Slots allow users to provide spoken inputs to a skill. In the invocation phrase example above, "Gemini" is user data spoken to fill a slot in the intent utterance.

Supported connecting words are words within the invocation phrase that are defined by Alexa (e.g. ask, open, using, from, to, about) and allow for more natural conversation in the interaction.

An **utterance** is a word or phrase users speak to invoke an intent.

A **voice user interface (VUI)** is a way for users to interact with an application or device using voice input.

The **wake word** is a word spoken to start an interaction with an Alexa device, set to "Alexa" by default.

A **web service** is a web-based application that uses a standardized message format for communications. A web service can be hosted on a server in the cloud or on premises.

2.2.2 Overall Architecture

Skills can be enabled on Alexa devices for a variety of voice-enabled services. Users interact with Alexa by saying a wake word, such as “Alexa,” to wake the device and then speaking an invocation phrase that consists of an utterance, the invocation name of the skill, and supported connecting words. For example, “Alexa, ask Daily Horoscopes for the horoscope for Gemini” consists of the “Alexa” wake word, the “Daily Horoscopes” invocation name, “the horoscope for Gemini” utterance, and the “ask” and “for” connecting words. This command wakes the device, opens the Daily Horoscopes skill, and sends the request for the horoscope for Gemini utterance from the front-end of the skill to the back-end web service. The web service processes the request and returns a response with the requested information to the front-end which sends it on to the Alexa cloud service to speak to the user [4].

The Amazon Echo and Alexa family of devices were selected for this project due, in part, to the availability of developer tools. The publicly available API (application programming interface) for Alexa, known as the Alexa Skills Kit (ASK), has contributed to the growth of an active third-party developer community that provides users with these skills [24].

Another justification for the use of Alexa for this thesis is Amazon’s voice assistant device superiority with regard to voice interaction [27]. In 1990, Jakob Nielsen developed ten usability heuristics, which are industry-accepted principles for general user interface design today [22].

- Visibility of system status: Users are kept informed about what is happening with the system.
- Error prevention: The system prevents errors from occurring when possible.
- Flexibility and efficiency of use: Users of all experience levels are able to interact with the system; more familiarity with the system allows for more efficient interactions.
- Match between system and the real world: The system speaks words, phrases, and concepts familiar to the user.
- User control and freedom: The ability to undo or redo an action is available.
- Consistency and standards: Users are able to assume words or actions in this system mean the same as in other systems.
- Recognition rather than recall: Instructions are easily retrievable.
- Aesthetic and minimalist design: Dialogues do not contain extraneous information.
- Help users recognize, diagnose, and recover from errors: The system provides plain and precise error messages and solutions.
- Help and documentation: The system provides any additional instructions if necessary.

While all ten heuristics can be applied to the design and implementation of a voice user interface, the first three can also be specifically addressed by smart speaker hardware. Alexa devices successfully apply all three of these principles.

Helping users recognize, diagnose, and recover from errors is an important facet of user interface design; however, applying an error prevention heuristic is preferable. Smartphone voice assistants tend to fail on detecting the activation phrase if the device is at a distance, inside a purse or pocket, or if there is background noise like music playing. With seven microphones, Alexa devices excel at recognizing activation phrases and voice commands from a distance and with a variety of ambient noise. They also place a priority on semantic processing. Voice commands are initially associated with an activated or enabled skill, which is the most likely intent of the user. Conversely, smartphone voice assistants tend to perform web searches first [27].

Alexa devices are less mobile than their smartphone counterparts because they need a constant power source, but their lack of mobility does not make them inferior at the flexibility and efficiency of use heuristic. Their accuracy in detecting and interpreting voice commands from far away and when obscured by background noise makes them more flexible and efficient than mobile devices. Users can access Alexa without having to carry the device around. They can initiate voice commands from a variety of locations in the vicinity, when they are in the middle of another activity, if they have a disability, or if they just don't want to move near the device [27].

The animated light ring on the Echo devices demonstrates the visibility of the system status heuristic. Different colors and motions of the light ring indicate different system statuses. The device documentation provides potential light ring statuses and their respective meanings [27].

A front-end and a back-end make up the voice user interface (VUI) of a skill. The front-end is comprised of two components. The Alexa VUI handles speech recognition (speech to text and text to speech translation) and also handles natural language understanding (the context of the text). The front-end also includes the skill configuration in the developer console. The logic that controls the application makes up the back-end [8]. Figure 2.1 shows the user interaction flow for a custom Alexa skill [4]. The diagram outlines the events that happen during the use of an Alexa skill, including the actions taken by the customer or user (1), the passing of those events to the Alexa cloud service (2), and the sending of the requests on to the specific Alexa skill front-end (3). While the forwarding of requests from the Alexa skill front-end to the skill back-end web service is not visually depicted in the diagram, it is implied in the description of block (3) as the web service is used to process the request and return a response to the skill front-end, which then continues the interaction flow back to Alexa and ultimately the customer.

In terms of data flow and storage, the Alexa cloud service and devices do not store any data. The device connects to the cloud service to handle speech recognition and natural language processing for understanding of the user spoken command. If the Amazon Alexa account is linked with an account at the back-end web service, an access token is stored in the user's Alexa account so it can be included with any requests. The Alexa account also stores the user interaction history, which can be viewed or deleted from the Alexa app [4].

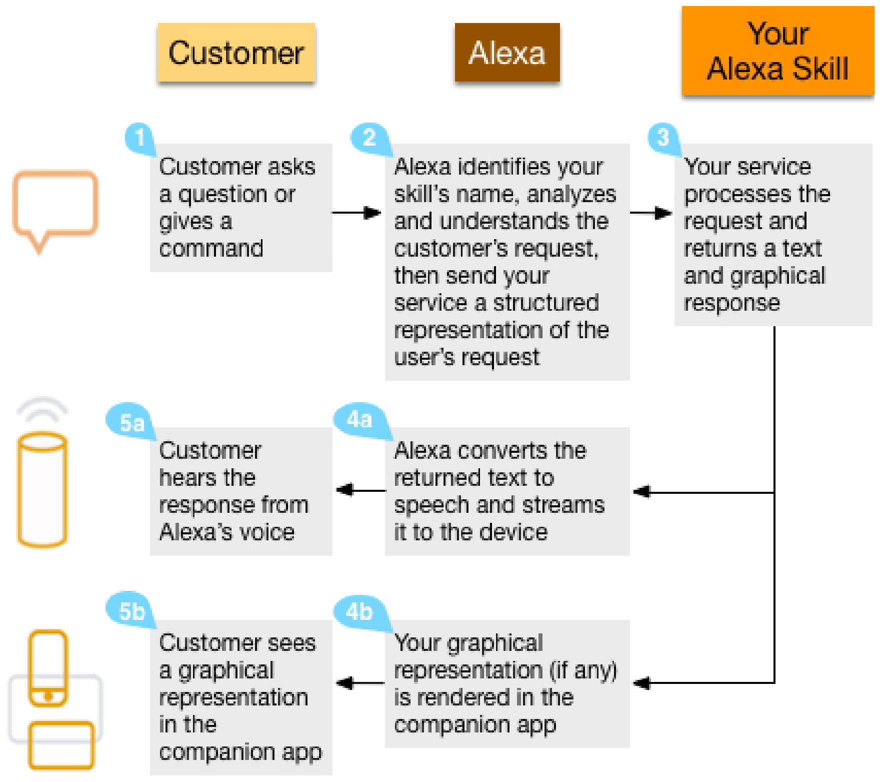

Customer
Alexa
Your Alexa Skill
1
Customer asks a question or gives a command
2
Alexa identifies your skill's name, analyzes and understands the customer's request, then send your service a structured representation of the user's request
3
Your service processes the request and returns a text and graphical response
4a
Alexa converts the returned text to speech and streams it to the device
5a
Customer hears the response from Alexa's voice
4b
Your graphical representation (if any) is rendered in the companion app
5b
Customer sees a graphical representation in the companion app

Figure 2.1: User interaction flow. Amazon image used with permission [4].

2.2.3 Front-end

The front-end of the voice user interface for a custom Alexa skill is created in the Alexa Skills Kit (ASK) developer console. Building the interaction model involves configuring several components.

- The *invocation name* is a phrase users say to open a skill.
- *Intents* are the tasks the skill can perform.
- *Sample utterances* are phrases users will speak to trigger each intent.
- *Slot types* define information that can vary within an utterance and are used to facilitate dialog with the user.

In addition to the interaction model, the ASK developer console provides configuration for the web service endpoint that serves as the back-end for the skill and for the account linking that allows users to connect their Alexa accounts with accounts in the back-end service [4].

Three core request types can be sent from an Alexa front-end to its back-end web service. The LaunchRequest is sent when a user opens a skill. The IntentRequest is sent when a user invokes a skill with a command for a specific intent. The SessionEndedRequest is sent when the user gives the exit command, does not respond within a certain amount of time, gives a command that is not mapped to any intent, or if an error occurs. Three built-in intents are included in the interaction model by default, and Amazon recommends these be implemented in the web service: AMAZON.CancelIntent, AMAZON.HelpIntent, and AMAZON.StopIntent. Additional built-in intents that provide pre-configured options for skills are also available. The AMAZON.FallbackIntent is initiated when the user's request does not map to any of the other intents for the skill. This intent covers a very large range of utterances and allows skill developers to benefit from spoken requests Alexa can recognize but that are not already included in the interaction model for the skill. With the FallbackIntent implemented in the back-end web service, a more logical and meaningful response can be sent to the user instead of generating a generic error. FallbackIntents are typically implemented to provide responses about intents that are planned for future development such as, "I can't do that yet, but here's what I can do..." Alexa analytics also include FallbackIntent statistics for unmapped utterances invoked in the skill. This data informs the skill developer about unimplemented intents users tried to invoke, which can guide future development based on actual skill use [4].

The Dialog interface in the ASK makes dialog between a user and Alexa possible. A Dialog directive returned with a skill response lets Alexa know that a response is needed from the user to complete the processing of a request. The responses are then stored in slots in subsequent requests to Alexa. For example, a weather skill needs to know the city for which the user would like a forecast. If the user does not include the name of the city in the initial command, the web service will include a Dialog directive in its response to Alexa. This directive lets Alexa know that the user will be prompted for a city name that needs to be processed from the response. The user response is then sent back to the web service as a value for the city slot so the request can be completed.

The ASK provides pre-defined slot types such as AMAZON.FOUR_DIGIT_NUMBER for a spoken number. However, slots can also be customized with utterances and values completely designed by the skill developer. Identifiers and synonyms for each custom value can be added to help the Alexa web service properly handle a variety of spoken values. It is important for a custom slot to have a comprehensive set of possible values to ensure accurate processing of user responses to Dialog directives. If a user responds to a Dialog directive prompt with a value that has been mapped to a custom slot, Alexa's request will contain that precise value, which can be more easily parsed by the web service. Other built-in slots provided in the developer console allow skill developers to make use of common pre-defined utterances, such as numbers, dates, and times.

Amazon recommends that intents implementing slots for dialog include one-shot invocations in their sample utterances. A one-shot invocation allows a user to invoke the intent and provide the slot value in one phrase, rather than requiring the dialog prompt to elicit the slot value after the initial intent invocation. "Ask Weather Bot for the weather in Los Angeles" is an example of a one-shot invocation that provides all the necessary information to fulfill the user's request [4].

Speech Synthesis Markup Language (SSML) provides a way to mark up the text of a response so the speech generated conforms to certain spoken characteristics, such as increasing the volume or speaking rate. Certain SSML tags are available as part of the ASK for the enhancement of the responses from the web service as spoken by Alexa [4]. One disadvantage is that SSML tags can only be applied to static content since the tags chosen are based on the specific text and how it should be read or pronounced. An exaggerated pause can be added between the list of capabilities Alexa speaks for a skill as part of a help response because this text does not change from one interaction to the next. However, much of the content in a learning management system is dynamic, as it is information specific to an individual user that may change over time. In these instances, the way Alexa speaks the text cannot be adjusted for context, such as reading the roman numeral IV as four instead of [ahy] [vee].

2.2.4 Back-end

The back-end of the voice user interface for a custom Alexa skill is a web service. Representational State Transfer (REST) is one of the web service protocols supported by Alexa. The REST protocol allows communication between computers on the internet using a standard set of operations. The web service endpoint must meet certain Amazon requirements, including internet accessibility, support for hypertext transfer protocol secure (HTTPS), and adherence to the ASK interface. HTTPS secures the sending of data between a client browser and a web server and has become standard in modern web development. Alexa uses JavaScript Object Notation (JSON) requests and responses to communicate with the back-end web service. JSON is a way to structure data that is heavily used in web services because of its readability for humans and its ability to be parsed easily by computers. The ASK provides detailed specifications for the formatting of these JSON documents [4].

To implement account linking on the back-end, the web service must provide an account token to identify the user in its system. Authorization code grant and implicit grant are the OAuth 2.0 authorization grant types supported by the Alexa Skills Kit for obtaining this access token; however, the Moodle framework is only able to act as an authorization server for the implicit grant type. OAuth 2.0 is an open authorization framework that enables an application to access a resource from another application through a web service, and the implicit grant type is used to retrieve the user access token [12]. To facilitate account linking with access token authorization, the web service must have a custom login that allows a user to sign in to the web service system from the Alexa account and generate a token that is then stored in the Alexa account to verify the user's authorization [4]. If a linked account is required but has not been set up when an intent is invoked, the user should receive a spoken prompt and LinkAccount card to complete the process. A card is a graphic that is displayed in the Alexa app to describe or support the spoken response [4].

Figure 2.2 illustrates the process of account linking when a user enables a skill in the Alexa app [4]. Once the Alexa app user initiates enabling the skill, Alexa starts the OAuth process by redirecting the request to the authorization URI, which is configured in the Alexa developer console. For this thesis, authorization is directed to a specially configured demo AsULearn/Moodle site. The user is presented with the custom login, where the credentials for the Moodle site are entered. The Moodle site acts as the authorization server by authenticating the user and creating or obtaining a web service token for the account. The access token is sent back to Alexa via the redirect URI provided in the initial request. The access token is stored in the user's Alexa account and is included by Alexa with future requests of this skill from devices linked to the account.

Figure 2.3 outlines the steps that are performed when a request is made and the skill sends the user's access token to retrieve protected information from Moodle [4]. When a user with a linked account invokes the skill, the access token is included in the

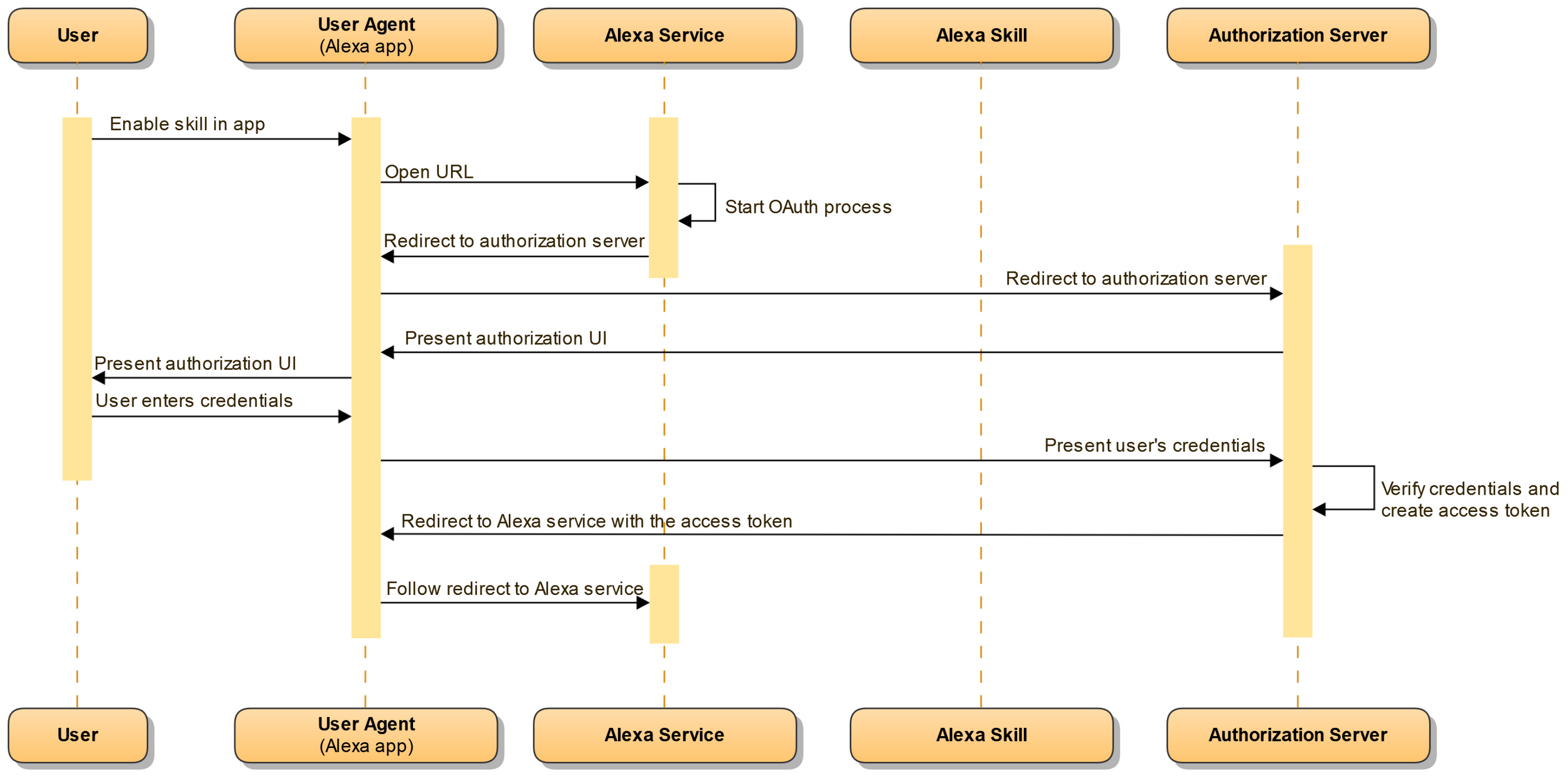

User
User Agent (Alexa app)
Alexa Service
Alexa Skill
Authorization Server
Enable skill in app
Open URL
Start OAuth process
Redirect to authorization server
Redirect to authorization server
Present authorization UI
Present authorization UI
User enters credentials
Present user's credentials
Verify credentials and create access token
Redirect to Alexa service with the access token
Follow redirect to Alexa service

Figure 2.2: Implicit grant flow. Amazon image used with permission [4].

request sent to the skill and passed on to the web service endpoint. The web service uses the access token to authenticate the user and verify access to protected resources within the Moodle site. The requested resources can then be returned in the response to the skill front-end which is passed on to the Alexa cloud service to be spoken to the user.

2.3 Learning Management Systems

A learning management system (LMS) is an application that makes course content accessible online. Instructors can share activities like assignments, forums, and quizzes as well as resources, such as files and links, with their students through a web interface. An LMS also supports online communication between course members, such as email, chat, and web conferences. It is a technology used in distance education for completely online courses, as a supplement to face-to-face courses, and in hybrid courses that incorporate elements of both.

Moodle is an open source learning management system and is the second-ranked LMS provider in terms of market share in the U.S. and Canada. Moodle was selected for this project because it is the only framework of the top three LMS providers without a distributed voice integration. Blackboard Learn and Canvas by Instructure, the first and third providers respectively [13], released Alexa skill integrations in 2017 [1]. Blackboard is a fully pay-for-service technology platform, and while Instructure does have an open source implementation of Canvas, it is a significantly inferior version of the paid platform with no service integration capabilities. A very simple Alexa skill for the Moodle community site Moodle.org exists, but it is specific to this hosted instance and is not

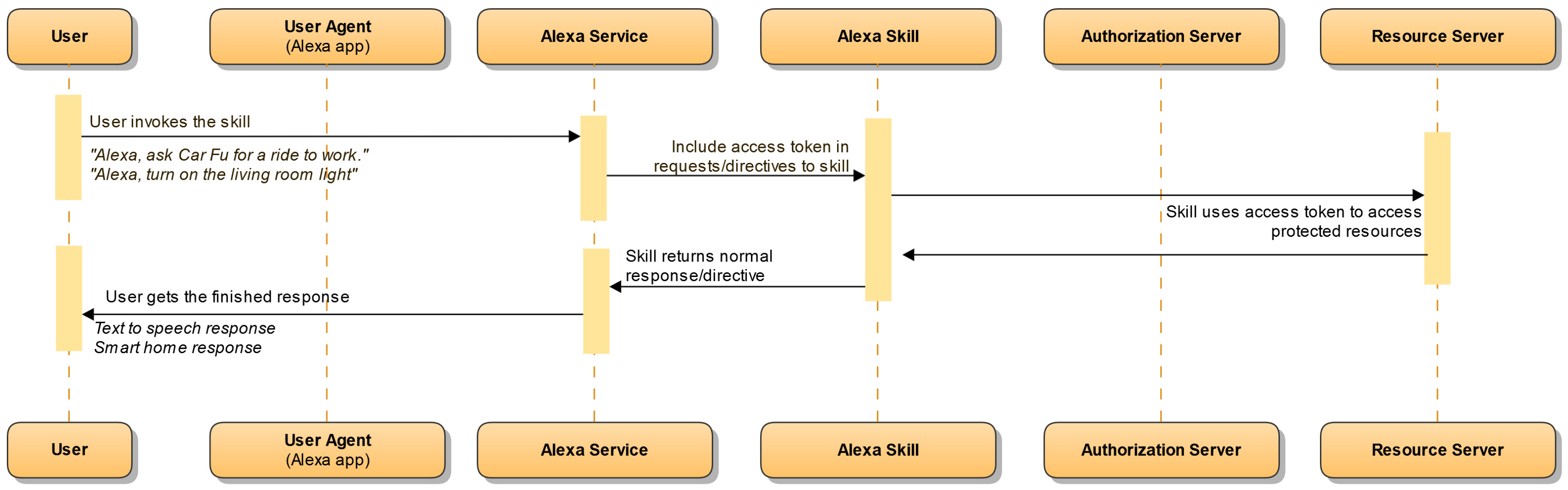

User
User Agent (Alexa app)
Alexa Service
Alexa Skill
Authorization Server
Resource Server
User invokes the skill
"Alexa, ask Car Fu for a ride to work."
"Alexa, turn on the living room light"
Include access token in requests/directives to skill
Skill uses access token to access protected resources
Skill returns normal response/directive
User gets the finished response
Text to speech response
Smart home response

Figure 2.3: Skill interaction flow. Amazon image used with permission [4].

available for integration with other installations. Moodle is also the framework behind AsULearn, the Appalachian State University branded LMS, and the development of a voice interface could more immediately benefit campus users.

As an open source platform, Moodle has a large community of developers who contribute to core Moodle development, as well as create custom plugins to add new functionality to the LMS. Moodle has provided several APIs for different types of plugins, including forms, web services, and external functions.

Moodle offers two interfaces for accessing the application, which are both visual and tactile in modality. The web interface allows users to access a Moodle site through desktop and mobile web browsers. The default theme for the web interface is responsive, so the layout adjusts based on the display size making it more user-friendly on a variety of devices. The mobile app interface, Moodle Mobile, is available for iOS and Android devices. Moodle Mobile has just started supporting almost all core activities in Moodle in the past year; however, it still has some limitations. Little to no support exists for third-party plugins or blocks on the mobile app and, while development of the app is ongoing, this means course content is not yet fully available through the mobile interface [19].

Moodle provides several methods for authenticating a user by default, as well as the ability to integrate with external authentication systems via third-party plugins. Moodle core provides manual user authentication, as well as OAuth 2 integration with Google, Microsoft, and Facebook accounts. External authentication systems, such as the Shibboleth single sign-on framework used by AsULearn, are integrated by installing and configuring compatible plugins [19].

Roles and capabilities in Moodle control the content and actions individual users have in specific areas of the site. Roles, such as teacher and student, are defined by enabling capabilities, such as creating a web service token or retrieving an overall course grade, within specific contexts of the site, such as sitewide or just within a course [19].

The Privacy API is a relatively new implementation in Moodle that was developed to enable compliance with the General Data Protection Regulation (GDPR). The GDPR is a regulation that serves to protect the privacy of individuals within the European Union (EU) [26]. The policy has worldwide implications, however, because websites and applications used by EU residents must also comply with GDPR or face serious fines and penalties. The Privacy API in Moodle gives plugin developers a way to comply with GDPR by implementing methods that provide a description of user data the plugin stores or sends to external services, a way for user data stored in the plugin to be exported by the user, and a way for user data stored in the plugin to be deleted by the user [18].

Chapter 3

Related Work

Voice user interfaces have been around since the 1950s when Bell Labs developed a digit recognition system [23]. It wasn't until the 2000s that interactive voice response (IVR) systems, which were able to translate human speech over the telephone, became widespread for services like getting stock quotes, booking flights, and transferring money. Users were happy with the ability to complete these tasks at their convenience, after business hours or when agents were not available, but they were consistently disappointed with the systems' usability [23].

It has only been since Apple's introduction of the Siri voice assistant in 2011 that speech recognition technology has improved enough for voice-enabled devices to become more common [10]. Amazon was the first to offer a smart speaker with the Echo in 2015, and they are currently the leader in the voice assistant market in terms of both hardware (71.3% market share) and software (25,000 skills) [25].

Even with all these skills available on the Alexa Skills Store, only two of the top three learning management systems currently have distributed Alexa integrations.

3.1 Instructure Canvas

In 2017, Instructure announced an Alexa skill for Canvas, its learning management system. The integration enables students, teachers, advisors, and parents to get information and complete simple tasks in Canvas through Alexa devices [11].

Canvas users can get information about their courses from Alexa. Currently, students and parents can get announcements, grades, missing submissions, and due dates; instructors can get announcements and assignments that need grading. Canvas has plans to enable more complex tasks, like helping users create new announcements, providing students with feedback, or checking in on at-risk students [2].

Users can enable the Alexa skill for Canvas on their Amazon Alexa accounts and then link to their Canvas accounts. After this process, the skill is accessible on all their Alexa devices. The Alexa skill provides Canvas users an easier way to access important course information, without having to open a visual application like a mobile app or browser.

3.2 Blackboard Learn

At their annual conference in 2017, Blackboard presented versions of two Alexa skills they have in development for their learning management system, Blackboard Learn [1]. The first skill, The Game for Blackboard Learn, allows users to discover new features in Blackboard Ultra and to quiz their knowledge of Blackboard Learn. The second skill is My Blackboard for Blackboard Learn and it provides announcements and grade information to users, similar to the Instructure Canvas skill. Users can enable the Blackboard

Learn skill on their Amazon Alexa accounts and then link their accounts from their school's Blackboard system. The learning management system administrator needs to enable the LTI (learning tool interoperability) link in the Blackboard system before this option is available to students and instructors [16].

3.3 Moodle.org

There are currently no distributed plugins for integrating an Alexa skill with the Moodle learning management system; however, a Flash Briefing skill is available for the Moodle.org Community site specifically. This specialized type of skill provides a quick summary of content from each enabled skill on an Alexa device. The skill developer configures a feed that populates the content for the Flash Briefing.

The Moodle.org Alexa skill provides announcements, forum posts, and a list of available resources from the site as part of the Flash Briefing. Users can enable the Moodle.org Alexa skill on their Alexa accounts and it is then available on all their enabled devices anytime they request the news or their Flash Briefing.

There are two distinct disadvantages to the Moodle.org skill. Authentication is not integrated with the skill, so user-specific content is not available. The Flash Briefing only provides public content. Also, the web service integration for the Moodle endpoint is not currently available as a distributed plugin, which is surprising given the open source nature of the Moodle community. Other installations of Moodle would need a custom web service plugin to make Alexa skill integration possible. Without sufficient knowledge and resources, this development may not be possible for some maintainers.

Chapter 4

Methodology

There were three overall stages to the project: learning, designing, and building [4]. The learning stage encompassed researching voice interaction design and voice user interface prototyping and usability testing. It included learning how to build a custom Amazon Alexa skill and the Moodle web service API.

The design stage involved using the voice interaction design research to design the voice experience for the skill and its integration with Moodle. Prototypes for early user feedback were also implemented during this stage to refine the design.

During the build stage, the Alexa skill was developed and coded, integrating both the Amazon Alexa skill (front-end) and the Moodle web service endpoint (back-end). Usability testing was conducted on the development implementation before completing the skill and preparing it for submission to the Appalachian State University Center for Academic Excellence Learning Technology Services team for approval and eventual submission for launch in the Alexa Skills Store.

4.1 Voice Interface Design

Design planning for the voice user interface was conducted to establish a seamless user experience. The Amazon Alexa Voice Design Guide provided the basis for the design process [3]. This process ensured careful consideration of the purpose and capabilities of the skill, what users would say when interacting with the skill, and planning for how Alexa would respond to build a voice interface that provides value and is easy to use. Each step is described in the following subsections.

4.1.1 Identification of Desired User Intents

The design process began by identifying user stories for the skill. To determine the capabilities users would find most beneficial, reports from Google Analytics for AsULearn were examined to verify the most viewed pages in the Appalachian State University instance of Moodle. Figure 4.1 shows pageviews for the top nine pages for the academic year from fall 2017 through spring 2018 (based on almost 62 million total pageviews for the time period). The chart illustrates that the course view is the most accessed page in the learning management system (LMS). The course view provides an overview of all the course materials, including the latest news (course announcements) and upcoming events (due dates). The third most viewed page in the LMS is the site home, which displays the site announcements as well as being the main portal for accessing the LMS. The fourth and fifth most viewed pages are the individual assignment and quiz pages. In addition to providing details about and access to these course activities, the view pages also list the due dates for completion. The next most viewed page is the grades overview, which

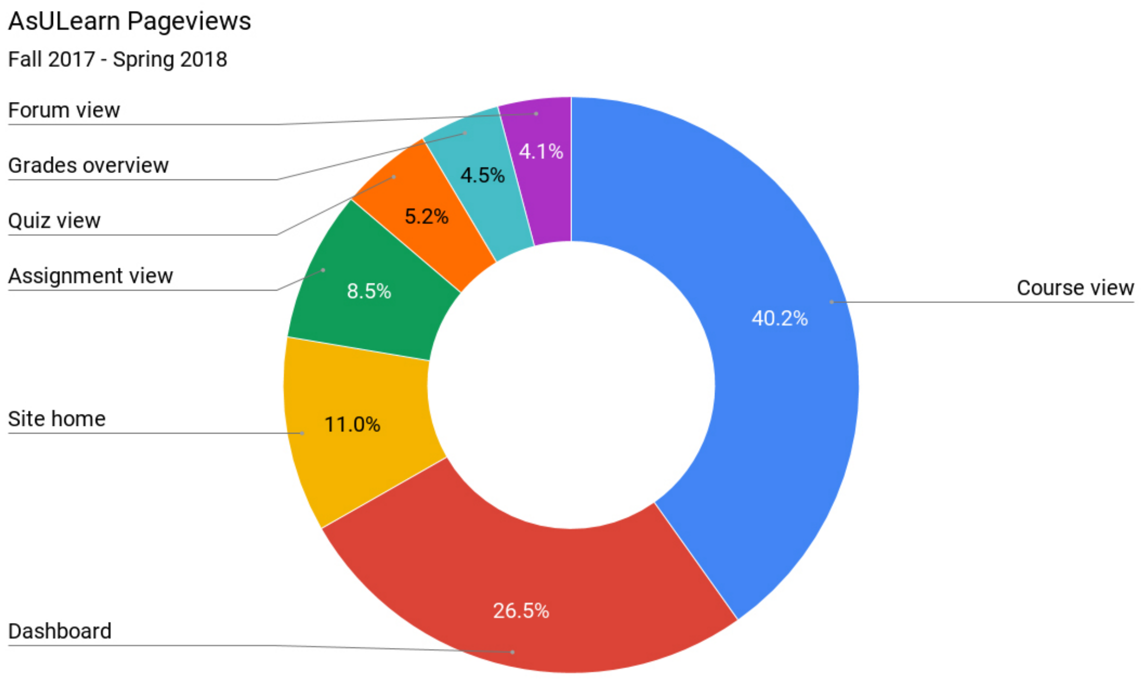


Figure 4.1: AsULearn pageviews for fall 2017 through spring 2018.

provides the user with all overall course grades to date. The forum view is the ninth most viewed page, which provides a listing of all the course announcements the instructor has emailed to course members. The pageviews analytics reports helped inform the decisions of the initial intents for the skill:

- GetSiteAnnouncementsIntent,
- GetCourseAnnouncementsIntent,
- GetDueDatesIntent, and
- GetGradesIntent.

4.1.2 Design of User Intent Utterances

With the user story intents established, the way users will speak their intentions needed to be considered. The next step in the design process involved outlining the utterances for each intent. Skill invocation phrases were designed using the supported phrases documentation in the Alexa design guide [4]. The guide recommends incorporating three grammatical forms of utterances in a skill; noun utterances, like “the horoscope for ...”; question utterances, like “what is the horoscope for ...”; and verb utterances, like “give me the horoscope for ...”. Identifying and implementing a large variety of utterances makes for a better user experience; however, Amazon does not support an unlimited number of phrase formats.

To ensure that the invocation phrases considered actually match the words students might use, 185 undergraduate and graduate students from a variety of majors completed a basic survey about their preferences for the phrasing of the application name, courses, announcements, grades, and due dates. Figure 4.2 shows the majority of students refer to the application by its branded name, pronounced “as you learn.” Most students prefer to refer to a course by its catalog name instead of the course number or the combination of course number and name, as evidenced by Figure 4.3. Figures 4.4, 4.5, and 4.6 show the primary phrases surveyed students use to refer to announcements, grades, and due dates, respectively, within their courses. These results helped guide the design of the Alexa skill in terms of the invocation name, the way courses were spoken to the user and how the user would speak them to the skill, and the implementation of additional utterances for each intent.

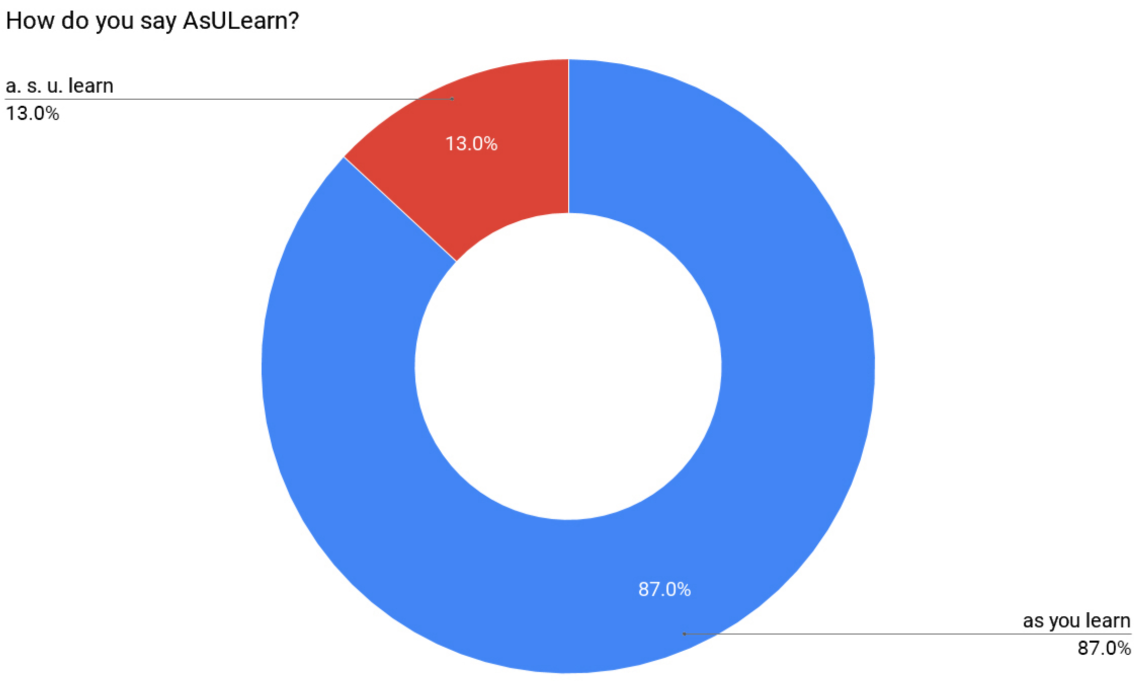


Figure 4.2: Student survey results for invocation name.

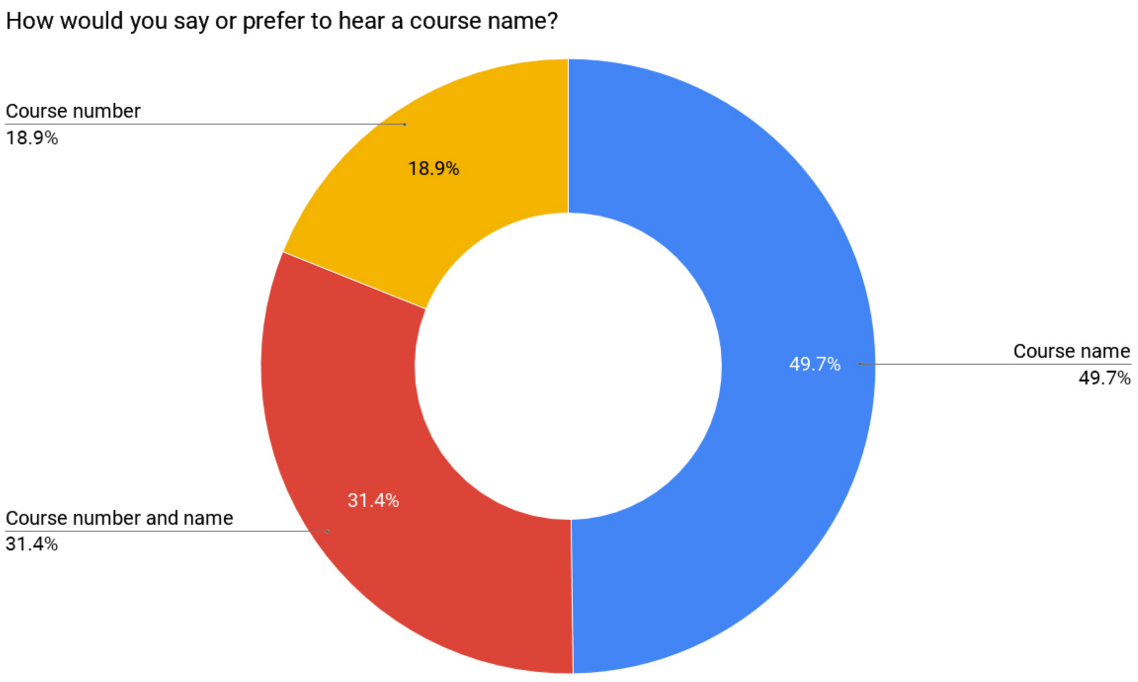


Figure 4.3: Student survey results for course names.

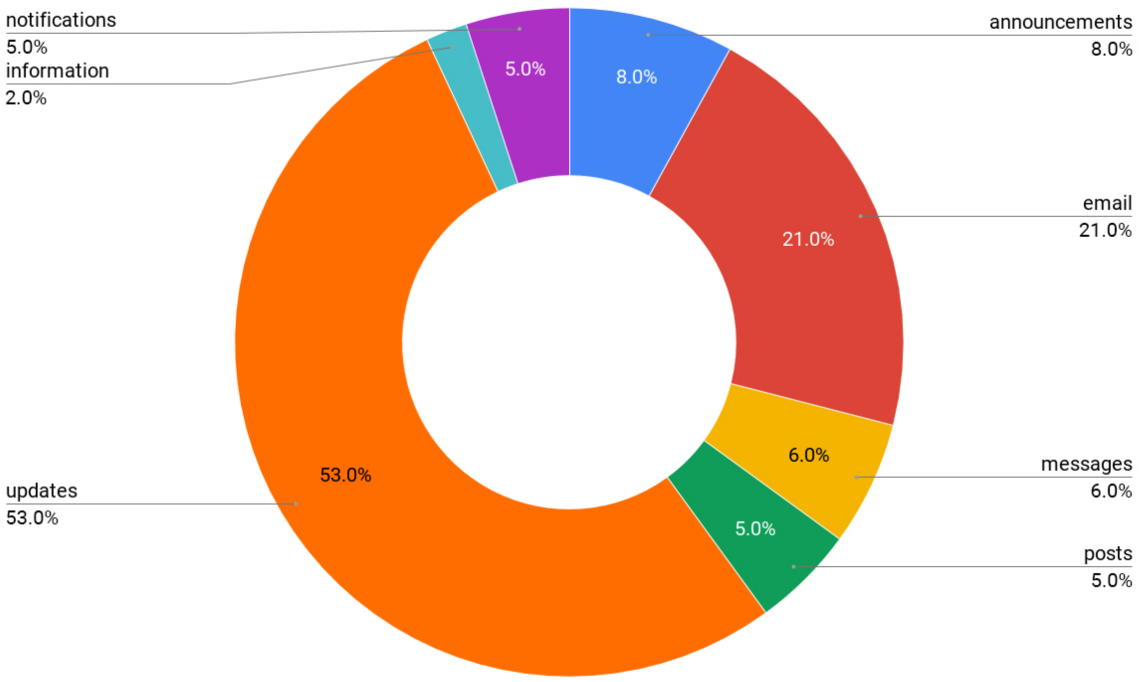


Figure 4.4: Survey results for the ways students refer to "announcements."

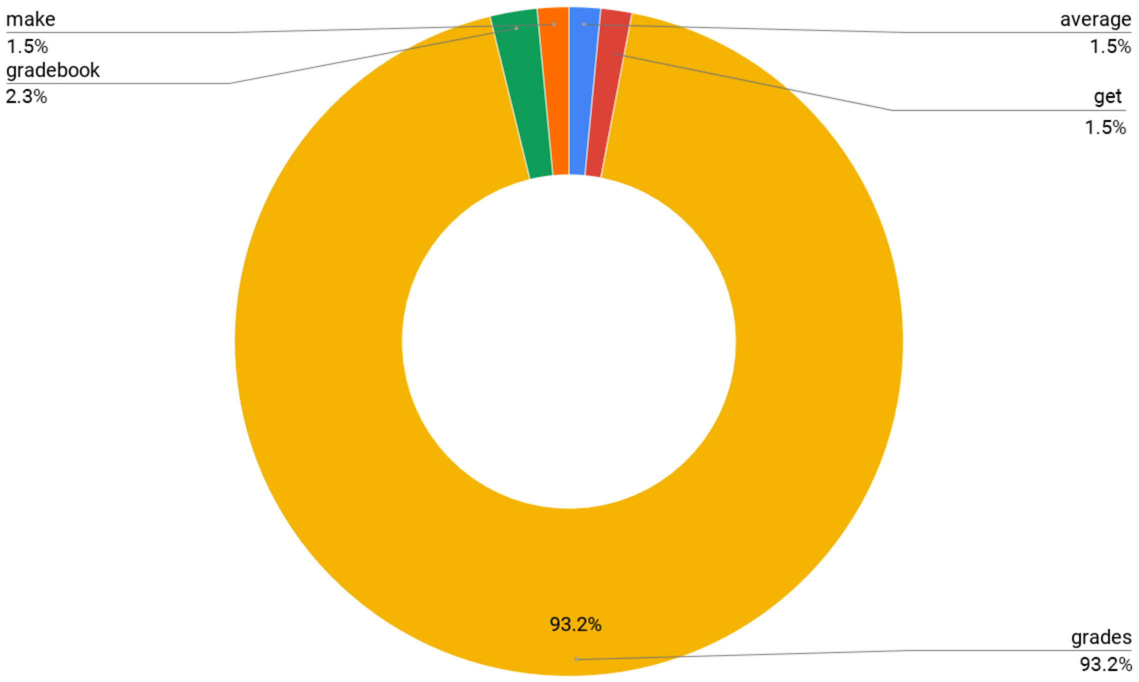


Figure 4.5: Survey results for the ways students refer to "grades."

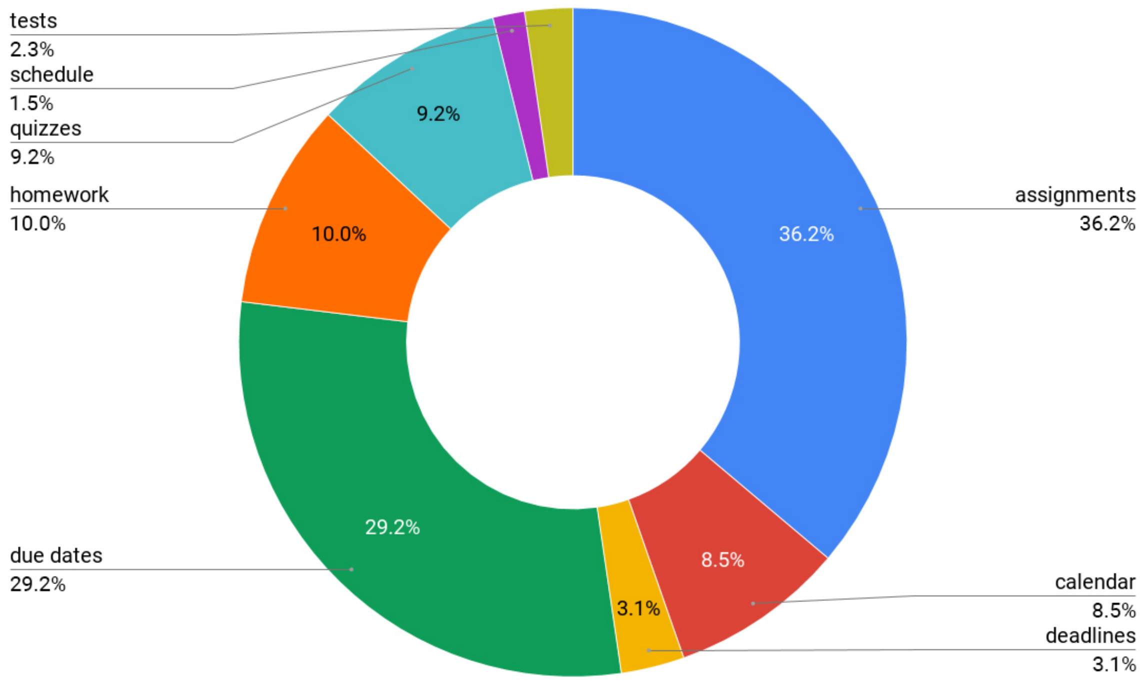


Figure 4.6: Survey results for the ways students refer to "due dates."

Table 4.1 shows the supported phrase formats and a sampling of the invocation phrases for this skill. Alexa-defined connecting words are shown in bold. Each block in the table groups similar invocation phrases. The first line in each block shows an abstract phrase format and subsequent lines give specific examples.

Table 4.1: *Supported phrases and examples*

PHRASE FORMATS AND INVOCATION EXAMPLES
<*Sample utterance*> **<connecting word: by, from, in, using, with>** <invocation name>
Get site announcements **from** As You Learn
Get course announcements **from** As You Learn
Get my grades **from** As You Learn
Get my due dates **from** As You Learn

Table 4.1: *Supported phrases and examples*

PHRASE FORMATS AND INVOCATION EXAMPLES
Ask <invocation name> **<connecting word: to, about, for, if, whether>** <*sample utterance*>
Ask As You Learn **to** *give me site announcements*
Ask As You Learn **for** *site announcements*
Ask As You Learn **to** *give me course announcements*
Ask As You Learn **for** *course announcements*
Ask As You Learn **to** *give me my grades*
Ask As You Learn **for** *my grades*
Ask As You Learn **to** *give me my due dates*
Ask As You Learn **for** *my due dates*
Ask <invocation name> <*sample utterance*>
Ask As You Learn *site announcements*
Ask As You Learn *course announcements*
Ask As You Learn *my grades*
Ask As You Learn *my due dates*
Ask <invocation name> <*question beginning with a supported question: what, how, as part of utterance*>
Ask As You Learn *what are the site announcements?*
Ask As You Learn *what are my course announcements?*
Ask As You Learn *what are my grades?*
Ask As You Learn *what are my due dates?*
Tell <invocation name> **<connecting word: to, that>** <*sample utterance*>
Tell As You Learn **to** *give me site announcements*
Tell As You Learn **to** *give me my course announcements*
Tell As You Learn **to** *give me my grades*
Tell As You Learn **to** *give me my due dates*
Tell <invocation name> <*sample utterance*>
Tell As You Learn *I want site announcements*
Tell As You Learn *I want my course announcements*
Tell As You Learn *I want my grades*
Tell As You Learn *I want my due dates*

Table 4.1: *Supported phrases and examples*

PHRASE FORMATS AND INVOCATION EXAMPLES
Search, Open <invocation name> for <*sample utterance*>
Search As You Learn for *site announcements* **Search** As You Learn for *course announcements* **Search** As You Learn for *my grades* **Search** As You Learn for *my due dates*
Talk to, Open, Launch, Start, Resume, Run, Load, Begin <invocation name> and <*sample utterance*>
Open As You Learn and *give me the site announcements* **Open** As You Learn and *give me my course announcements* **Open** As You Learn and *give me my grades* **Open** As You Learn and *give me my due dates*
Use <invocation name> **<connecting word: and, to>** <*sample utterance*>
Use As You Learn **and** *get the site announcements* **Use** As You Learn **and** *get my course announcements* **Use** As You Learn **and** *get my grades* **Use** As You Learn **and** *get my due dates*

4.1.3 Design of Alexa's Responses

The last step in the design process was planning how Alexa would respond to user requests. Formatting the responses so they sound natural took priority over using proper grammar to make sure Alexa sounds like a person when a user is interacting with the AsULearn skill [3].

Responses that need an answer from the user were designed to end with a prompting question to serve as a cue for the user to begin speaking. Acknowledgments such as "thanks," "okay," and "great" were planned for inclusion in responses to let users know they have been understood and to make the interaction more conversational. Multiple variations of responses were designed for each intent. One of the variations is randomly selected as the response. This design feature makes using the skill feel more like a natural conversation with a person and not an interaction with a computer [3].

To give the user a clear choice and to prevent confusion, the voice interface design included providing specific choices when options are part of the response. For example, instead of responding with "Which course do you want announcements for?" the skill lists available courses and asks the user to select one. For user-requested information returned in a list format, the design specifies a limit of five items to help with comprehension, since the interface is typically entirely spoken [3]. If a user has more than five upcoming due dates, only the most recent five will be included in the response.

Another element of the design focused on adding a layer of access protection to the voice interaction. When an account has been linked from an Alexa device to a service account, anyone near the device can use the skills enabled on it. For multi-user environments, open access to a student's grades, assignments, or courses presented a privacy issue. To address this concern, the design incorporated the ability to set an optional PIN during account linking. After authenticating on the Moodle site during account linking, the user is presented with the option to create a 4-digit PIN. If the PIN is set for a user, any skill requests for personal information, such as grades, course announcements, or due dates, will require the user to speak the PIN to verify their identity before the information is returned.

4.1.4 Evaluation of Design Prototype

Prototypes exposed a few usability shortcomings in the original design of the skill. Sessions were conducted with three users: one who had no experience with a voice assistant or a learning management system (LMS); one who had experience with an LMS but very little experience with a voice assistant; and one who had average experience with both. Four test courses with varying content for site announcements, course announcements, due dates, and grades for each user were created for the sessions. An Echo Dot was used instead of an Alexa simulator to more accurately test the user experience. The sessions were recorded on video for review and analysis, and several changes were made to the application after reviewing the results.

The ability to set an optional PIN for increased access security was not clear to users during the account linking process. To resolve this issue, the label for the PIN field on the account linking form was edited to clarify that it is for the creation of a PIN (not to enter an existing one) and that it is optional.

Rather than responding to a one-shot invocation for the GetCourseAnnouncementsIntent for the specified course as described earlier, the skill did not process the request properly and returned the list of the user's courses as options. This response required the user to repeat the course selection. The utterances configured for the intent were reviewed, and it was discovered that there were insufficient utterances with the COURSE slot included to be able to process these one-shot invocations well. To resolve this issue, a variety of utterances including the COURSE slot were added to the GetCourseAnnouncementsIntent to better implement one-shot invocations.

When one of the courses included as an option in the GetCourseAnnouncementsIntent response was selected, the skill was unable to parse the course name into a valid response for the web service, resulting in the error message that no record for the course could be found. A review of the values configured for the COURSE slot revealed that all demo site courses had not been input as valid options. Since the skill needs all the possible values a user could speak for a slot to properly process the data, all the preferred names for courses on the demo AsULearn site were added to the COURSE slot to make sure they would be valid COURSE slot responses. Documentation was also included with the web service plugin to make skill installers aware of this important aspect of the front-end implementation.

The interface was initially designed to allow a user to invoke multiple intents during a single session by ending each response with the prompt, "Would you like anything else?" Prototype sessions revealed several usability issues with this approach. One user had to think about what capabilities the skill offered and did not respond in the 8-second time window, causing the session with the skill to end. When the user finally replied with a new utterance, Alexa confusingly answered with a generic error response because the skill was no longer active. Another user attempted to respond to the prompt with "No," "I'm done," and "Nothing." The skill did not recognize these responses and repeated the skill's help response instead of closing the session. Deeper research revealed this approach was not a recommended practice; open-ended questions tend to make using a voice interface more challenging because they expect the user to know or remember the interface's capabilities. To improve usability, the web service design was modified to provide the specific information the user asked for and end the session with the skill [3].

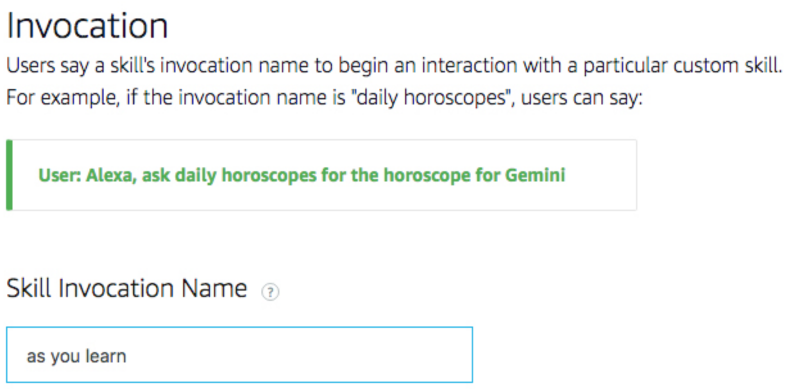
4.2 Alexa Front-end

Within the Alexa skill interaction model, the overall skill invocation name was set to "as you learn" since this is how users speak the branded name of the Appalachian State University Moodle site. Figure 4.7 shows the skill invocation configuration screen as seen by the skill developer.

To enable the four primary capabilities of the skill, four intents were created: GetSiteAnnouncementsIntent, GetCourseAnnouncementsIntent, GetGradesIntent, and GetDueDatesIntent. Sample utterances for each intent were added to the interaction model of the Amazon Alexa developer console. Table 4.1 previously showed sample utterances users can speak to invoke these intents and Figures 4.8, 4.9, 4.10, and 4.11 show utterance configurations for each of the intents in the interaction model of the developer console.

Utterances were designed according to Amazon Alexa voice design documentation and based on the results of the student survey. Between 50 and 250 utterances were added to each intent, as Amazon recommends at least 30 utterances per intent to enhance skill performance [3].

A custom COURSE slot type was created so users can say the name of a course for which they would like to hear announcements. Figure 4.9 shows the use of this slot in the configuration of utterances. The custom slot was populated with course preferred names from the demo AsULearn site so they would be available as request options for



Invocation

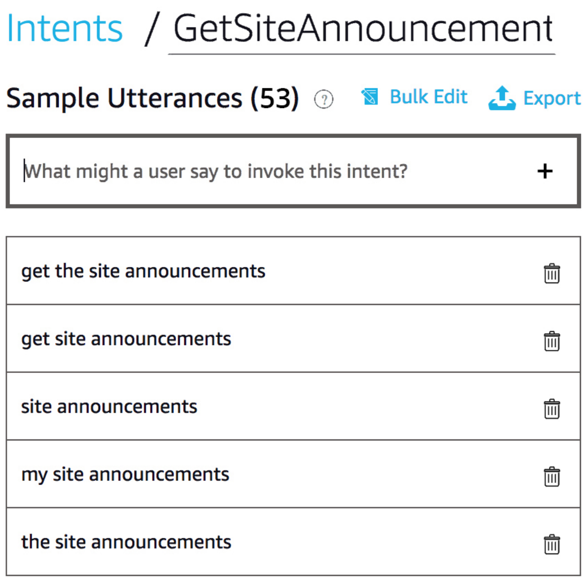
Users say a skill's invocation name to begin an interaction with a particular custom skill.
For example, if the invocation name is "daily horoscopes", users can say:

User: Alexa, ask daily horoscopes for the horoscope for Gemini

Skill Invocation Name

as you learn

Figure 4.7: Alexa skill invocation configuration.

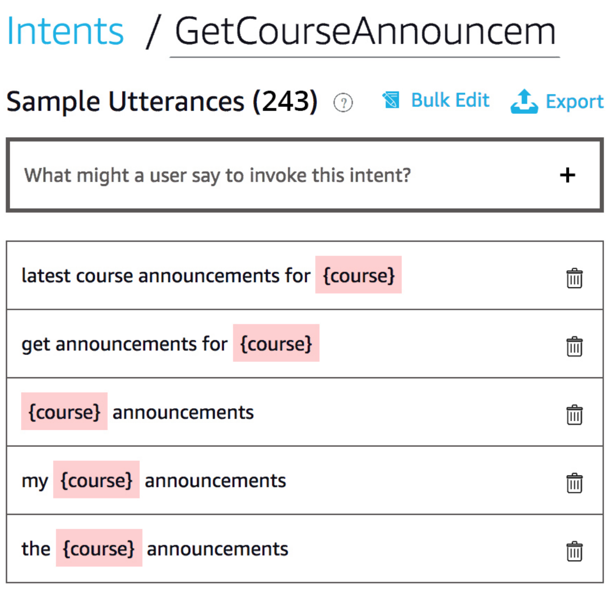


Intents / GetSiteAnnouncement

Sample Utterances (53)

get the site announcements
get site announcements
site announcements
my site announcements
the site announcements

Figure 4.8: Alexa skill GetSiteAnnouncementsIntent configuration.


Intents / GetCourseAnnouncem

Sample Utterances (243)

latest course announcements for {course}

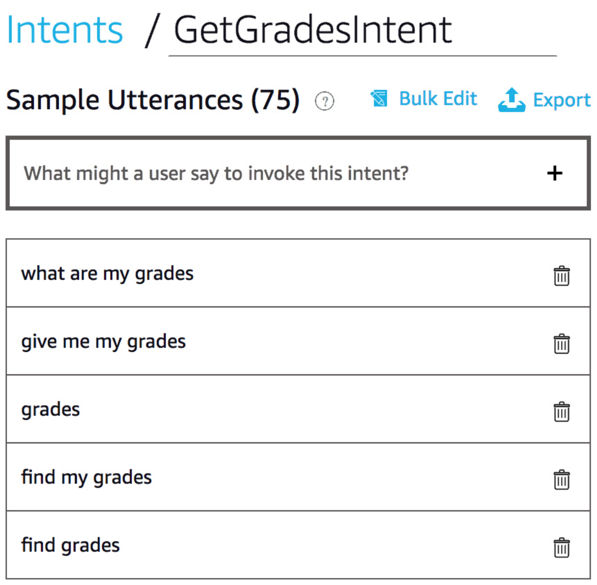
get announcements for {course}

{course} announcements

my {course} announcements

the {course} announcements

Figure 4.9: Alexa skill GetCourseAnnouncementsIntent configuration.


Intents / GetGradesIntent

Sample Utterances (75)

what are my grades

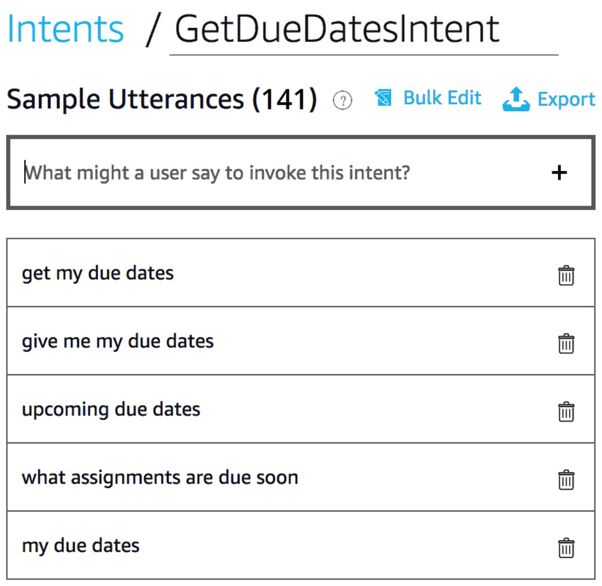
give me my grades

grades

find my grades

find grades

Figure 4.10: Alexa skill GetGradesIntent configuration.


Intents / GetDueDatesIntent

Sample Utterances (141)

get my due dates

give me my due dates

upcoming due dates

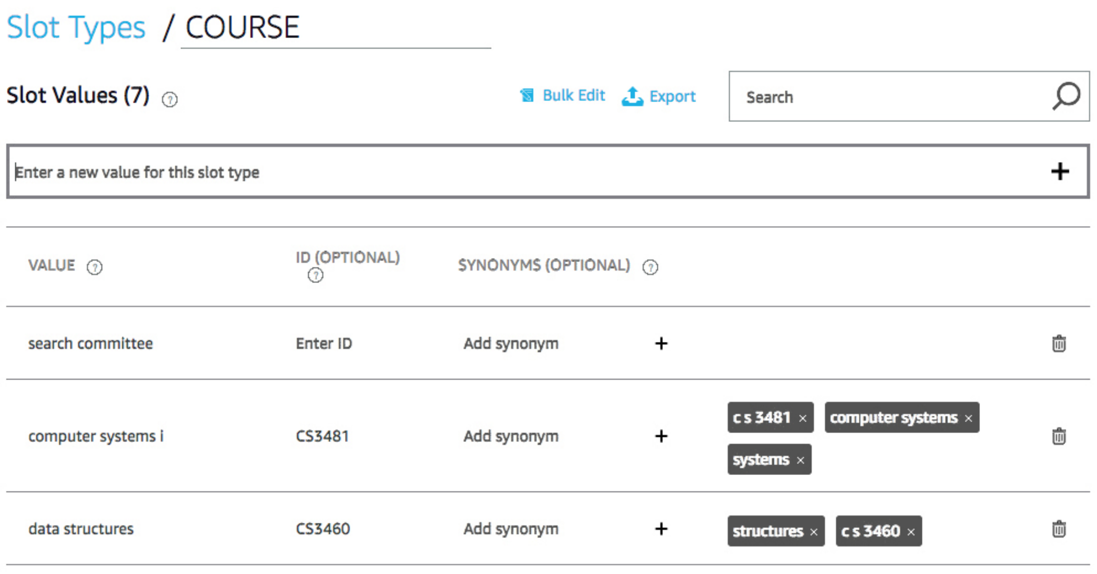
what assignments are due soon

my due dates

Figure 4.11: Alexa skill GetDueDatesIntent configuration.

users. The COURSE slot type was configured for the GetCourseAnnouncementsIntent to enable the dialog prompt for this information when needed. Figure 4.12 shows how the skill developer defines the COURSE slot values in the Amazon Alexa developer console. The slot configuration includes inputting values the user might say, setting optional IDs that are sent with the intent request that may help the web service process the slot value, and entering synonyms, or other ways a user might say the slot value.

To handle PIN responses from the user for intents that provide personal information from Moodle, the AMAZON.FOUR_DIGIT_NUMBER slot type was implemented. The GetCourseAnnouncementsIntent, the GetDueDatesIntent, and the GetGradesIntent respond with a PIN prompt if the user has set a PIN for Alexa access to the Moodle account. The AMAZON slot type provides built-in recognition of the variety of ways four-digit numbers are spoken, such as “nineteen twenty-one” or “one nine two one”, and sends the digits to the web service for processing [4].



Slot Types / COURSE

Slot Values (7)

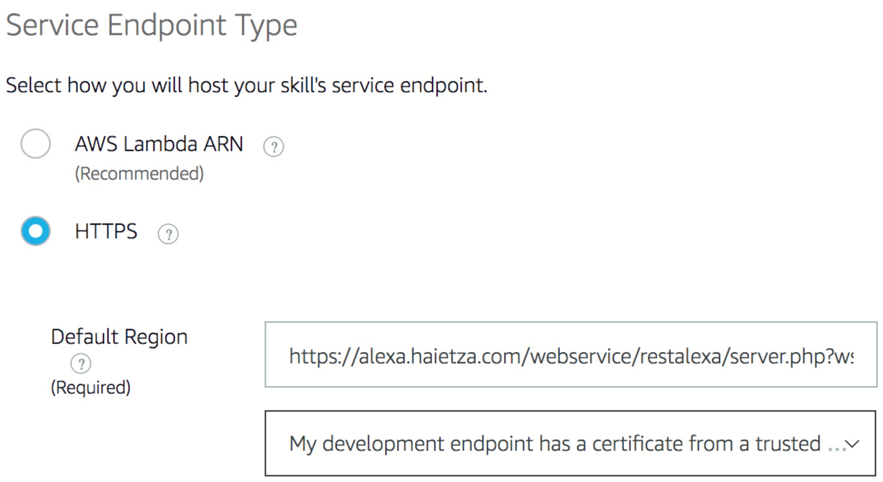
VALUE	ID (OPTIONAL)	SYNONYMS (OPTIONAL)	
search committee	Enter ID	Add synonym	
computer systems i	CS3481	Add synonym	c s 3481 × computer systems × systems ×
data structures	CS3460	Add synonym	structures × c s 3460 ×

Figure 4.12: Alexa skill COURSE slot configuration.

Several Alexa built-in intents were also implemented in the skill front-end to provide for the processing of standard commands. The AMAZON.CancelIntent and AMAZON.StopIntent were enabled to handle the typical ways users end a skill session. The AMAZON.HelpIntent was enabled to process requests for help from the user. The AMAZON.FallbackIntent was implemented to process utterances that do not map to any of the existing intents. Amazon Alexa analytics provide reports from the FallbackIntent that list unmapped utterances users are invoking in the skill. These reports can be used to guide future design and development work [4].

To establish the connection between the Alexa skill front-end and the web service back-end that receives and processes the skill requests, the address of the Moodle web service was input as the endpoint. Figure 4.13 shows the endpoint configuration.

Account linking was enabled to use OAuth 2.0 implicit grant authorization, and the address of the custom login for the Alexa skill for Moodle plugin was set as the authorization URI. Figure 4.14 shows this configuration.


Service Endpoint Type

Select how you will host your skill's service endpoint.

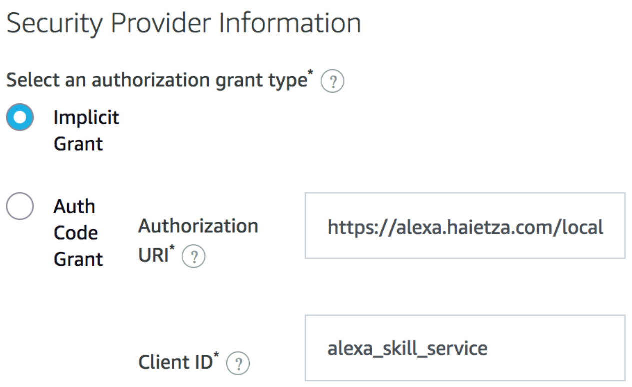
○ AWS Lambda ARN (Recommended)

[✓ HTTPS]

Default Region (Required): https://alexa.haietza.com/webservice/restalexa/server.php?ws

My development endpoint has a certificate from a trusted ...

Figure 4.13: Alexa skill endpoint configuration.


Security Provider Information

Select an authorization grant type*

[✓ Implicit Grant]

○ Auth Code Grant

Authorization URI*: https://alexa.haietza.com/local

Client ID*: alexa_skill_service

Figure 4.14: Alexa skill account linking configuration.

Several types of testing were performed as recommended by the skill certification checklist [4]. Policy testing encompassed reviewing Amazon policy examples, such as trademarks, advertising, violence, and content; it was determined that the AsULearn skill adheres to the Amazon Alexa content guidelines. Security testing involved reviewing Amazon's requirements for the protection of customer data for skills hosted as web services, skills with account linking, and privacy requirements, and verifying that the AsULearn skill meets these requirements.

Functional testing was performed to verify that the skill's functionality works and provides the information as described on the skill's detail card in the Amazon Alexa app. This testing comprised three steps. (1) Example phrases displayed on the skill's detail card were reviewed and tested with a simulator. (2) The skill description on the detail page was reviewed to confirm that it was accurate and complete. (3) Account linking settings were reviewed and tested through the Amazon Alexa app.

Voice interface and user experience testing was conducted to ensure a high-quality experience for users. The Amazon checklist [4] suggested the following eleven items, all of which were performed and confirmed.

- Session management was tested to confirm that sessions were kept open while they were in process and were ended when they were complete.
- Different combinations of utterances and slot values were tested on a simulator to verify the expected response was returned.
- Prompts for user responses were reviewed and tested.
- Invocation name only interactions were tested.

- One-shot utterances were tested for the GetCourseAnnouncementsIntent.
- Utterances for each intent were tested to make sure they mapped to the correct intent.
- Both custom and AMAZON slot type values were tested.
- Error handling was tested for no responses to prompts, invalid slot values, and invalid utterances.
- Help was tested to confirm the help response was returned when requested.
- Stopping and canceling were tested to verify that these utterances received a good-bye response and the session was ended.
- The FallbackIntent was tested to make sure that unmapped utterances elicited the fallback response.

4.3 Moodle Web Service Back-end

For hosting the web service, a virtual server was configured with the components required for running a Moodle application: Linux, Apache, MySQL, and PHP. HTTPS through the *Let's Encrypt* [1] certificate authority was installed on the server. Moodle version 3.4.4 (the latest stable release at the time of development) was installed and configured on the server. The following subsections review the web service plugin developed for

[1] *Let's Encrypt* is a free and open certificate authority provided by the Internet Security Research Group.

Alexa integration with Moodle, the handling of intent requests by the web service, and the installation of the web service plugin on a Moodle site.

4.3.1 Web Service Plugin

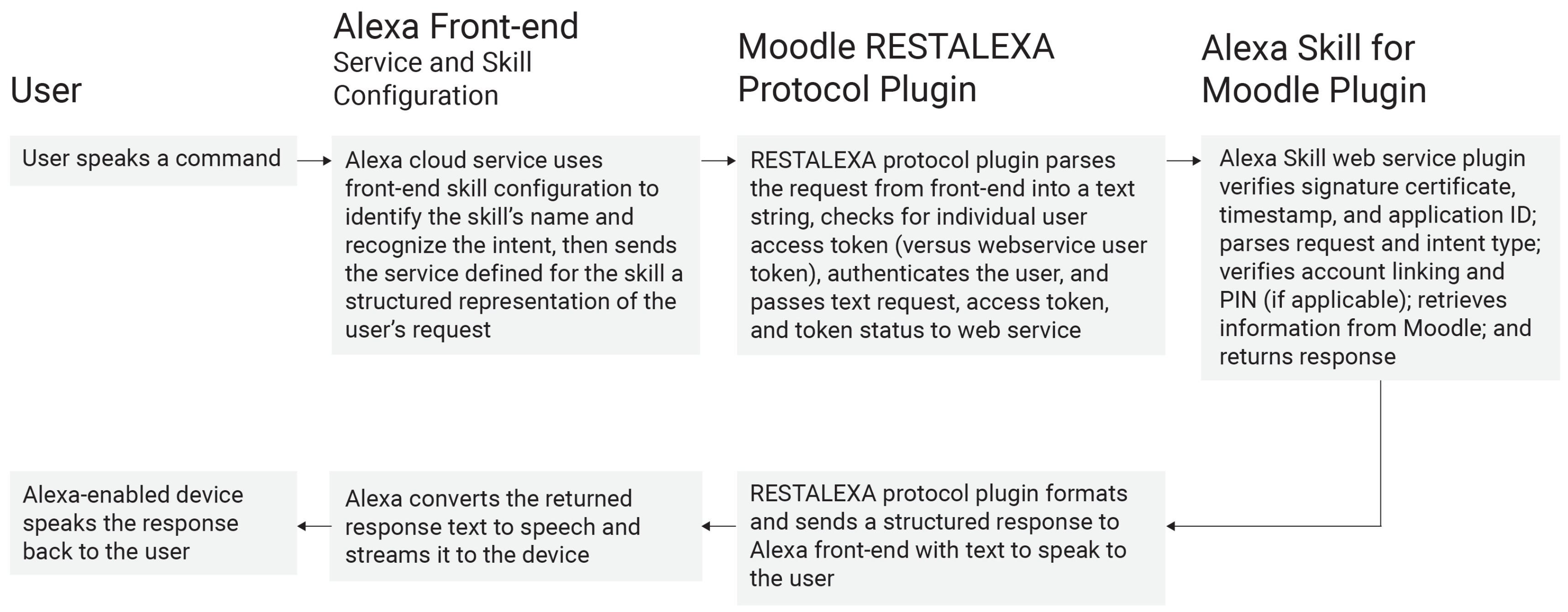
The custom Alexa Skill plugin for Moodle was developed and coded to serve as the web service endpoint for the skill. Moodle already provides a web service API enabling third-party customization. However, several deviations from the standard API were necessary to adhere to the Alexa Skills Kit (ASK) requirements. The Moodle core web service that custom plugins extend only allows the passing of arguments via URL query strings. In order to receive the JSON documents sent by Alexa, a custom plugin providing the REST protocol with JSON payload support was installed and set as a dependency of the Alexa Skill plugin.

The RESTJSON protocol plugin code [20] was forked and customized to support the goals of this thesis because it did not sufficiently meet the Alexa Skills Kit requirements. It was also missing a few key components for a fluid request and response process between Alexa and the web service. This new RESTALEXA plugin was designed to work specifically with the Alexa skill web service plugin. The original RESTJSON plugin sends the JSON document as an object to the web service plugin. Moodle's web service API requires that the parameters for the web service be pre-defined in the plugin, which would involve declaring all the JSON request properties in the plugin code. This specification posed a problem because the Alexa Skills Kit states that new properties may be added to the request and response formats, and web service endpoints must not break when receiving requests with additional properties [4]. In order to meet this specification and to accommodate other needs for the Alexa web service, the RESTALEXA plugin was designed to send the JSON request to the Alexa plugin as a text string. This adjustment allows the protocol plugin to send an unmodified version of the request to the web service plugin. The web service needs the unmodified request for signature certificate validation, an ASK security requirement.

The RESTALEXA plugin was also built to check the access token provided in the request to determine if it is for the Moodle webservice user (token provided in the web service endpoint URL for all Alexa requests) or for an individual user (token provided in the Alexa request JSON if the user has a linked account). The token is authenticated in each case; however, if the token is for an individual user, the RESTALEXA protocol plugin adds a valid token attribute to the request before it passes it to the web service so the web service can process account linking verification as needed. Figure 4.15 provides an overview of the skill interaction flow, outlining the interaction of the RESTALEXA plugin with the skill front-end when Moodle receives the request and returns the response. Sample code for the method that handles the parsing of the request is available in Appendix A.1.

4.3.2 Skill Linking to Moodle Account

To enable account linking on the front-end, the Alexa Skills Kit requires that the web service login accept a username, password, state, client ID, response type, and redirect URI. The web service needs to generate a token for the specified user and return it


User
Alexa Front-end
Service and Skill Configuration
Moodle RESTALEXA Protocol Plugin
Alexa Skill for Moodle Plugin

User speaks a command

Alexa cloud service uses front-end skill configuration to identify the skill's name and recognize the intent, then sends the service defined for the skill a structured representation of the user's request

RESTALEXA protocol plugin parses the request from front-end into a text string, checks for individual user access token (versus webservice user token), authenticates the user, and passes text request, access token, and token status to web service

Alexa Skill web service plugin verifies signature certificate, timestamp, and application ID; parses request and intent type; verifies account linking and PIN (if applicable); retrieves information from Moodle; and returns response

RESTALEXA protocol plugin formats and sends a structured response to Alexa front-end with text to speak to the user

Alexa converts the returned response text to speech and streams it to the device

Alexa-enabled device speaks the response back to the user

Figure 4.15: Overview of skill interaction flow.

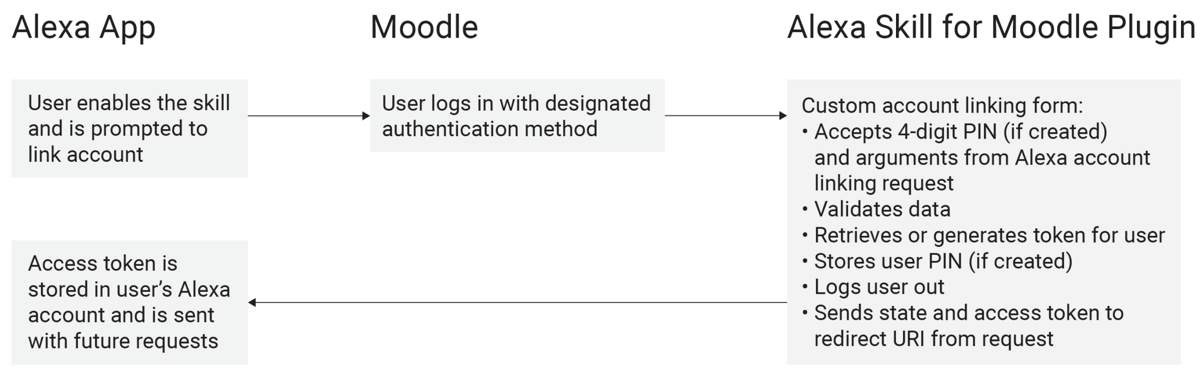

Alexa App
Moodle
Alexa Skill for Moodle Plugin
User enables the skill and is prompted to link account
User logs in with designated authentication method
Custom account linking form:
• Accepts 4-digit PIN (if created) and arguments from Alexa account linking request
• Validates data
• Retrieves or generates token for user
• Stores user PIN (if created)
• Logs user out
• Sends state and access token to redirect URI from request
Access token is stored in user's Alexa account and is sent with future requests

Figure 4.16: Overview of account linking process.

to Alexa at the provided redirect URI with the saved state from the request [4]. The Moodle core token request is similar to the core web service request in that arguments are passed to it via URL query strings. It also only provides the user's web service token in the response. This response structure was not sufficient to meet the ASK requirements, so a custom login and account linking process was created. Due to the variety of authentication methods available for Moodle, the account linking custom login was developed to allow integration with any authentication method available in Moodle while still meeting the ASK requirements. Figure 4.16 provides an overview of the account linking process, including the interaction with the custom account linking form.

A PIN verification option was implemented for users who want an added layer of security for accessing personal information in Moodle from Alexa. After users login to Moodle via their specified authentication method, they are able to create an optional 4-digit PIN that is stored in Moodle as user data. Figure 4.17 shows the account linking form for optional PIN creation after successful login via the specified authentication method in Moodle. If the web service receives a request from a user with a linked account and a PIN set, Alexa will prompt for PIN verification before providing user-

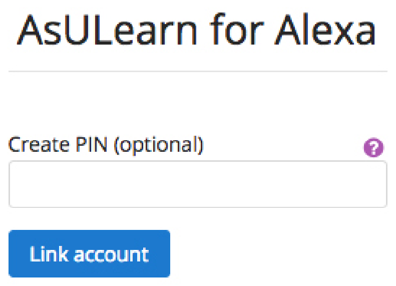


Figure 4.17: Custom account linking login.

specific information. PIN verification is only required for the first request for personal information during a session. The valid PIN status is stored within the session, so subsequent requests during that session will not prompt for verification again. If an invalid PIN is provided, the user is informed and the session is closed.

The Moodle Forms API, which provides a structure for validating input before form submission, was used to develop the account linking form. Data validation includes:

- The Alexa web service is installed and enabled.
- The redirect URI is one of the valid options provided in the Alexa developer console.
- The response type sent in the request is valid.
- The user token exists or can be created (the user has the capability to create a token for the Alexa web service).
- The PIN, if provided, is exactly 4-digits.
- The PIN user profile field exists.
- The state, response type, and redirect URI have been provided as query strings.

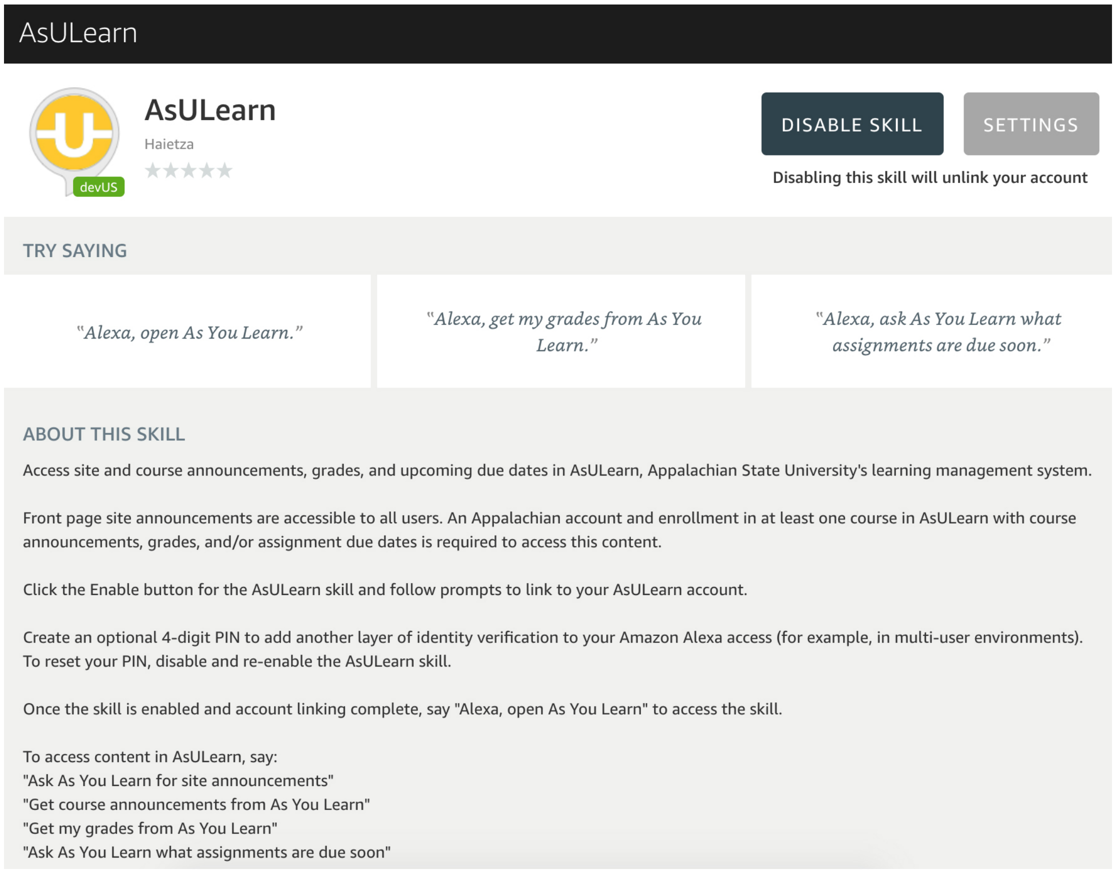
This validation confirms that the Moodle web service only processes valid account linking requests from Alexa. Users are automatically logged out of Moodle after completing the account linking form so their sessions are not left open.

Account linking is only required if a request is made for user-specific information, such as course announcements, grades or due dates. Since site announcements are public, a user who has enabled the AsULearn skill can retrieve this information without completing account linking. If the web service receives a request for user-specific information without a linked account, it responds with a LinkAccount card and message requesting that the Alexa account be linked with the Moodle account. After an account is linked, Alexa includes the user's access token in all requests. The plugin checks all requests for this token to perform user verification and determine capabilities.

In order to test account linking, beta testing was enabled for the skill in the Alexa Skills Store. As development work was completed, it was tested on Alexa simulators and devices with accounts authorized for the beta test. Figure 4.18 shows the display for enabling the AsULearn dev skill on Alexa devices.

4.3.3 Web Service Processing of Requests

When the web service receives an Alexa skill request, it validates several pieces of data as part of the security checks required by the ASK: signature certificate URL, signature certificate, timestamp, and application ID. The web service parses the JSON request text into an associative array and calls an internal function based on the request type specified. When the web service receives a LaunchRequest, it sends a response that



Disabling this skill will unlink your account

TRY SAYING

"Alexa, open As You Learn."

"Alexa, get my grades from As You Learn."

"Alexa, ask As You Learn what assignments are due soon."

ABOUT THIS SKILL

Access site and course announcements, grades, and upcoming due dates in AsULearn, Appalachian State University's learning management system.

Front page site announcements are accessible to all users. An Appalachian account and enrollment in at least one course in AsULearn with course announcements, grades, and/or assignment due dates is required to access this content.

Click the Enable button for the AsULearn skill and follow prompts to link to your AsULearn account.

Create an optional 4-digit PIN to add another layer of identity verification to your Amazon Alexa access (for example, in multi-user environments). To reset your PIN, disable and re-enable the AsULearn skill.

Once the skill is enabled and account linking complete, say "Alexa, open As You Learn" to access the skill.

To access content in AsULearn, say:
"Ask As You Learn for site announcements"
"Get course announcements from As You Learn"
"Get my grades from As You Learn"
"Ask As You Learn what assignments are due soon"

Figure 4.18: Enabling the AsULearn skill on Alexa.

includes a welcome message, as well as the options available to the user. The response ends with a prompt for the user's choice. If the user has a linked Moodle account, the response will be personalized with the user's first name.

The web service parses the specific intent from an IntentRequest. The implemented intents include GetSiteAnnouncementsIntent, GetCourseAnnouncementsIntent, GetGradesIntent, and GetDueDatesIntent.

For the GetSiteAnnouncementsIntent, the web service will respond with the site announcements from the front page. These posts are publicly accessible, so no account linking or PIN verification is required. The number of site announcements retrieved is determined by the front page settings for number of announcements. If the setting is over five, the number is limited to five for usability.

For the GetCourseAnnouncementsIntent, the web service performs account linking and PIN verification. If these steps are completed successfully, the list of courses for which the user has enrollments is retrieved. If there are no courses or if a single course with no announcements is found, these respective messages are returned. If there are announcements for a single course, they are provided. Similar to the GetSiteAnnouncementsIntent, the number of announcements retrieved is determined by the course setting for number of announcements up to five.

If more than one course is found for a user, the web service responds with the list of course names. Users are then prompted to select the course for which they want announcements. The user's course name response is parsed from the COURSE slot value in the request JSON from Alexa and checked against the list of course enrollments for the user. If a match is found, the announcements for that course are returned. If no match is found, a message to that effect is returned to the user. Both announcement intents only retrieve parent posts from the site or course news forum, as this forum is the one typically used for instructor announcements. Students are not able to post in a news forum, so there are usually no replies to these posts.

The GetGradesIntent returns user information so the web service performs account linking and PIN verification. Upon successful completion of these steps, a response is returned with the grade report overview for the user, which provides overall course grades for each of the student's courses.

The web service performs account linking and PIN verification for the GetDueDatesIntent as well. If these steps are completed successfully, the course enrollments and group memberships for the user are determined and site, category, course, group, and user events are retrieved. If no events are found, the web service responds with that message. If there are events, they are returned in the response. The number of events retrieved is determined by the site setting for number of events to show to users. If this setting is over five, the number is limited to five for VUI usability. The site setting for number of days in the future to look for upcoming events is also used in the evaluation of events to return to the user. If no setting is found, the default of 3 weeks is used.

For LaunchRequests and IntentRequests, the web service sends the requested content back as a JSON response formatted according to the Alexa Skills Kit interface requirements. For responses that do not need a reply from the user, the session is ended. For the LaunchRequest and GetCourseAnnouncementsIntent when multiple courses are found for a user, the web service returns a prompt for the user's selection and the session is kept open to enable processing of the user's response.

If the web service receives a SessionEndedRequest, it indicates the user session with Alexa has ended and it is not possible for the web service to send back a response. However, the plugin will log the reason for the closing of the session from the request JSON, in case the site administrators need that information for future debugging.

Several Alexa built-in intents were also implemented in the Alexa web service. The AMAZON.CancelIntent and AMAZON.StopIntent end the user's session with a goodbye response. The AMAZON.HelpIntent and AMAZON.FallbackIntent respond with the skill capabilities and prompt the user for a choice.

Responses to the LaunchRequest and IntentRequest are randomly chosen from several variations so the user experience is more personal and conversational. SSML was also used when providing list options in responses to include appropriate pauses between options to make it easier for users to hear, understand, and respond accordingly. The responses to these requests also include a reprompt, which Alexa speaks if no response is heard from the user within 8 seconds, or if the response is not understood. The reprompt text is based on the original response so it makes sense in context. Figure 4.15 provides an overview of the skill interaction flow, including the role of the Alexa Skill for Moodle plugin in performing verifications and retrieving the requested information from Moodle. Sample code for the function that handles the verifications and parsing of the request and intent types is available in Appendix A.2.

4.3.4 Moodle Plugin Installation

A README.md file was created for documentation and installation instructions for the web service plugin. A JSON file of the interaction model was also included with the plugin code for quickly building the base skill in the Alexa developer console with the JSON Editor import feature.

There are several GUI settings that are automatically configured for the Moodle site administrator on installation of the plugin:

- The webservice role is created with system level assignability if it does not already exist.
- The capabilities to create tokens and use the RESTALEXA protocol are enabled for the webservice role if they do not already exist.
- The site webservice user is created if it does not already exist. This user makes initial web service calls from the Alexa front-end to the web service endpoint on the Moodle site.
- The webservice role is assigned to the webservice user if it does not already exist.
- Web services are enabled on the site if they were not enabled.
- The RESTALEXA protocol is enabled if it was not already enabled.
- The Amazon Alexa skill default user profile category and PIN field are created if they do not already exist.

The plugin includes several configurable settings, to facilitate installation and use on any instance of Moodle:

- Application ID, for verification of the application ID in the request;
- Possible redirect URIs that Amazon could send with an account linking request for verification of these values before returning a response;
- Development setting that enables skipping signature validation to allow testing of the web service via command line CURL [2] requests on a site not internet accessible by Alexa (a valid signature cannot be simulated since it is based on the request data and the encryption method); and
- Regular expression for parsing course fullnames to the preferred format in requests and responses.

The plugin settings include a list of the course fullnames from the Moodle site formatted according to the regular expression setting. An administrator who is setting up the Alexa front-end can copy and paste these values in the COURSE custom slot bulk editor.

PHPUnit tests were implemented for the web service and account linking functions. Table 4.2 lists code coverage results of the unit test suite.

[2]CURL is software that provides command-line tools for transferring data.

Table 4.2: *Code coverage report*

CLASSES	**METHODS**	**LINES**
@account.linking::account_linking_form	100.00% (2/2)	100.00% (45/45)
@local.alexaskill::local_alexaskill_external	80.00% (20/25)	79.05% (298/377)
@local_alexaskill:: local_alexaskill_account_linking_form_testcase	100.00% (16/16)	100.00% (379/379)
@local_alexaskill:: local_alexaskill_externallib_testcase	100.00% (74/74)	100.00% (1515/1515)

The only functions not included in the local_alexaskill_external tests were the web service API required parameters and returns functions that are already tested by the Moodle core unit tests and the public web service function. This function only serves as a gateway for the internal functions that perform the actual operations, and all the internal functions are tested with individual unit tests. Instructions for setting unit test configurations are provided in the README.md file.

To comply with GDPR, both the RESTALEXA protocol plugin and the Alexa skill for Moodle web service plugin implement the Moodle Privacy API. The Privacy API functions enable the plugins to describe the data they store or send to an external service about a user, to export the data for a specific user on request, to delete data for all users based on the site retention policy, or to delete data for a specific user on request.

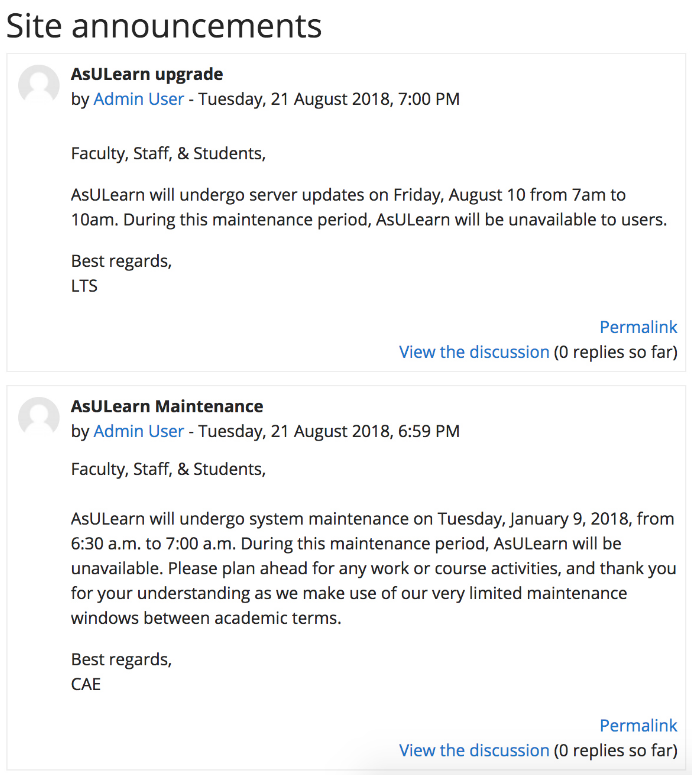
Chapter 5

Results

The overall objective of this thesis was to build a voice user interface that enhances the speed and convenience of accessing information in a learning management system (LMS). This goal was achieved by implementing an Amazon Alexa skill for the Moodle LMS that provides voice access to site announcements, course announcements, grades, and due dates.

5.1 Technical Contributions

Within the overall objective to allow more convenient and faster access to the Moodle learning management system with an Amazon Alexa skill, there are several technical contributions provided by the development of the voice interface. Specific goals for the project included implementing the ability for users to complete several primary tasks via voice commands.


Site announcements

AsULearn upgrade
by Admin User - Tuesday, 21 August 2018, 7:00 PM

Faculty, Staff, & Students,

AsULearn will undergo server updates on Friday, August 10 from 7am to 10am. During this maintenance period, AsULearn will be unavailable to users.

Best regards,
LTS

Permalink
View the discussion (0 replies so far)

AsULearn Maintenance
by Admin User - Tuesday, 21 August 2018, 6:59 PM

Faculty, Staff, & Students,

AsULearn will undergo system maintenance on Tuesday, January 9, 2018, from 6:30 a.m. to 7:00 a.m. During this maintenance period, AsULearn will be unavailable. Please plan ahead for any work or course activities, and thank you for your understanding as we make use of our very limited maintenance windows between academic terms.

Best regards,
CAE

Permalink
View the discussion (0 replies so far)

Figure 5.1: Moodle front page site announcements.

5.1.1 Intents Without Account Linking

Allow users to hear site announcements. The GetSiteAnnouncementsIntent does not require account linking or user verification of any kind, because the content is publicly available. Figure 5.1 shows site announcements from the front page of a Moodle site. Several items were completed to enable users to get site announcements.

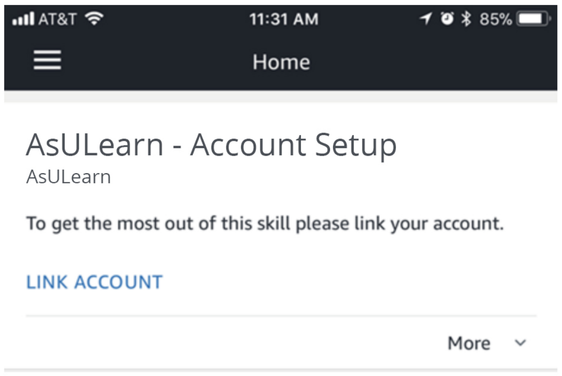
- Invocation phrases were designed for what a user might say to get site announcements from the skill.
- The GetSiteAnnouncementsIntent was added to the interaction model of the Alexa skill.

- Sample utterances (52) from the invocation phrases were added to the intent so Alexa can map what the user says to the GetSiteAnnouncementsIntent.
- The GetSiteAnnouncementsIntent was mapped to a web service function in the plugin that retrieves the site announcements content.
- The web service was developed to send Alexa a structured JSON response with the site announcements content for Alexa to speak to the user.

Verification is performed by requesting access to the demo Moodle site and the AsULearn skill beta test, enabling the skill in the Alexa app, and using an Alexa device or simulator to open the AsULearn skill and ask for site announcements. A video demonstration of the successful operation of the GetSiteAnnouncementsIntent is available at `https://youtu.be/CdOd5YleQpU`.

5.1.2 Alexa to Moodle Account Linking

Allow students and instructors to link their Amazon Alexa accounts with their Moodle accounts. To verify that users have authorization to access certain content in Moodle, it must be possible to link their Alexa and Moodle accounts. Figure 5.2 shows the Link Account card in the Alexa app. To provide account linking, the following technical tasks were completed to implement the OAuth 2.0 implicit grant.

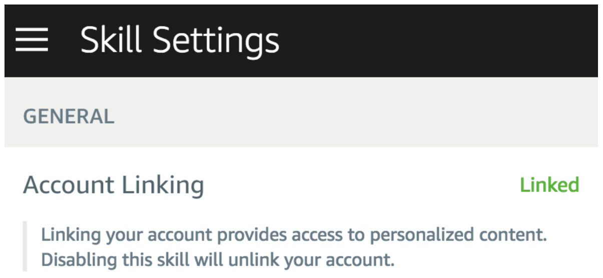


AsULearn - Account Setup
AsULearn
To get the most out of this skill please link your account.
LINK ACCOUNT
More

Figure 5.2: Link Account card.

- Account linking was enabled and configured for the AsULearn skill in the developer console.
- HTTPS was enabled on the Moodle site.
- A custom login form was developed for the web service that allows a user to login to Moodle and set an optional PIN for added security.
- The login form was built to store the state argument passed from Alexa.
- The web service was designed to retrieve the Moodle web service token for the user.
- The web service was developed to return a structured JSON response with the state, client ID, response type, token type, and token to the redirect URI provided as an argument from the Alexa request.
- If the web service receives an invalid token for a user, it was designed to return a LinkAccount card to re-link the Moodle account with Alexa.

Verification of account linking is performed by requesting access to the demo Moodle site and the AsULearn skill beta test, enabling the skill in the Alexa app, and



Account Linking — Linked

Linking your account provides access to personalized content.
Disabling this skill will unlink your account.

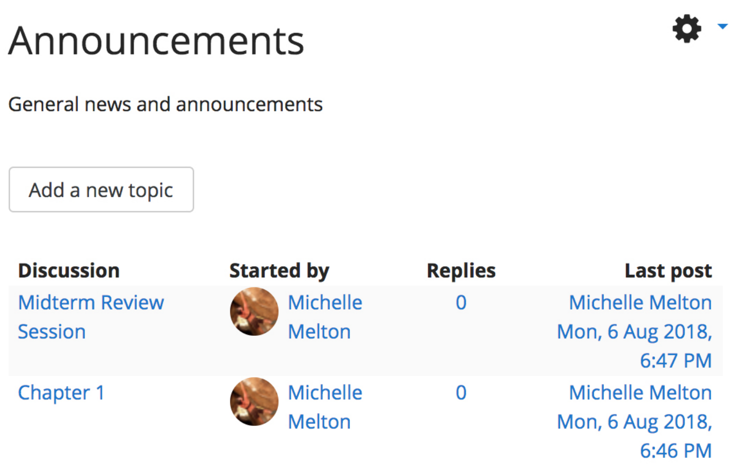
Figure 5.3: Account linked.

linking the Alexa account to the Moodle account. Figure 5.3 shows the linked account in the Alexa app. A video demonstration of the successful operation of account linking is available at `https://youtu.be/svtkDocS-lM`.

5.1.3 Intents With Account Linking

Most of the skill intents access user-specific information within Moodle and as a result, these intents require account linking and user verification. Several items were completed for each of these intents.

- Invocation phrases were designed for what a user might say to invoke each of the intents.
- Each intent was configured in the interaction model of the Alexa skill.
- Sample utterances from the invocation phrases were configured in each intent so Alexa can map what the user says to the appropriate intent.
- The AMAZON.FOUR_DIGIT_NUMBER slot type was added to each intent to store PIN responses from users.


Announcements

General news and announcements

Discussion	Started by	Replies	Last post
Midterm Review Session	Michelle Melton	0	Michelle Melton Mon, 6 Aug 2018, 6:47 PM
Chapter 1	Michelle Melton	0	Michelle Melton Mon, 6 Aug 2018, 6:46 PM

Figure 5.4: Moodle course announcements.

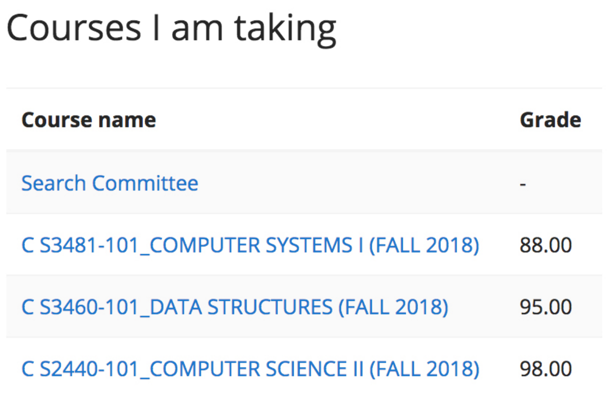
- Web service functions for each intent were developed to perform account linking and PIN verification, get the requested information, and send a structured JSON response with the content for Alexa to speak to the user.

Allow students to hear course announcements. Figure 5.4 shows announcements from a news forum in a course in a Moodle site. In addition to the work completed for all intents with account linking, several items were completed to enable users to get course announcements.

- Sample utterances (243) from the invocation phrases were added to the intent so Alexa can map what the user says to the GetCourseAnnouncementsIntent.

- The COURSE slot type was added to the GetCourseAnnouncementsIntent and populated with course names from the demo Moodle site.

- The web service function for the GetCourseAnnouncementsIntent retrieves the user's courses, prompts the user for a course selection, and retrieves the announcements content for the requested course.



Courses I am taking

Course name	Grade
Search Committee	-
C S3481-101_COMPUTER SYSTEMS I (FALL 2018)	88.00
C S3460-101_DATA STRUCTURES (FALL 2018)	95.00
C S2440-101_COMPUTER SCIENCE II (FALL 2018)	98.00

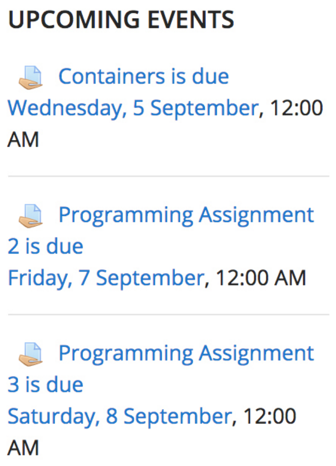
Figure 5.5: User grade display in Moodle.

Verification is performed by requesting access to the demo Moodle site and the AsULearn skill beta test, requesting enrollment in at least one course with at least one announcement, enabling the skill in the Alexa app, linking the Alexa account to the Moodle account, and using an Alexa device or simulator to open the AsULearn skill and ask for course announcements. A video demonstration of the successful operation of the GetCourseAnnouncementsIntent is available at `https://youtu.be/EeBeiu7O9Xo`.

Allow students to hear current overall grades. Figure 5.5 shows a user's grade display from a Moodle site. In addition to the work completed for all intents with account linking, several items were completed to enable users to get overall grades.

- Sample utterances (75) from the invocation phrases were added to the intent so Alexa can map what the user says to the GetGradesIntent.
- The web service function for the GetGradesIntent verifies the user has authorization to access the grade content, and retrieves the student's grades.

Verification is performed by requesting access to the demo Moodle site and the AsULearn skill beta test, requesting enrollment in at least one course with at least one


UPCOMING EVENTS

Containers is due
Wednesday, 5 September, 12:00 AM

Programming Assignment 2 is due
Friday, 7 September, 12:00 AM

Programming Assignment 3 is due
Saturday, 8 September, 12:00 AM

Figure 5.6: User upcoming events in Moodle.

grade, enabling the skill in the Alexa app, linking the Alexa account to the Moodle account, and using an Alexa device or simulator to open the AsULearn skill and ask for grades. A video demonstration of the successful operation of the GetGradesIntent is available at `https://youtu.be/-H6T__UNIIU`.

Allow students to hear upcoming due dates. Figure 5.6 shows upcoming events for a user on a Moodle site. In addition to the work completed for all intents with account linking, several items were completed to enable users to get due dates.

- Sample utterances (141) from the invocation phrases were added to the intent so Alexa can map what the user says to the GetDueDatesIntent.
- The web service function for the GetDueDatesIntent verifies the user has authorization to access the calendar content, and retrieves the student's upcoming due dates.

Verification is performed by requesting access to the demo Moodle site and AsULearn skill beta test, requesting enrollment in at least one course with at least one due date, enabling the skill in the Alexa app, linking the Alexa account to the Moodle account, and using an Alexa device or simulator to open the AsULearn skill and ask for due dates. A video demonstration of the successful operation of the GetDueDatesIntent is available at `https://youtu.be/gSTXaxCHJv0`.

5.1.4 Supporting Work

Additional work completed to carry out the aforementioned tasks includes:

- A JSON file for importing a pre-built Alexa skill front-end was included with the plugin. This file allows a Moodle admin to quickly populate the Alexa skill configuration in the developer console.
- An installation script was created to configure most of the Moodle site settings required for the web service plugin.
- Configurable settings were implemented to make the web service plugin installable on any Moodle instance.
- A COURSE custom slot values page was built to simplify the front-end development by outputting all course names in the preferred format.
- A third-party web service protocol plugin was modified to specifically parse Alexa web service requests.
- A custom account linking form with data validation and optional PIN creation was developed.

- The web service was built to verify that requests were sent by Alexa by checking the signature of the request and the request timestamp.
- The web service was built to verify that the request was intended for its service by matching the application ID sent with the request.
- The SessionEndedRequest was implemented in the web service.
- Error handling was implemented in the web service.
- Built-in intents for Cancel, Help, Stop, and Fallback were implemented in the web service.
- Unit tests (86) were written to support future development.

5.2 Usability Testing

5.2.1 Participant Profiles

Upon development of the four primary intents for the Alexa skill, usability testing was performed to evaluate the voice application. Eleven Appalachian State students from a variety of different colleges (Figure 5.7) and grade levels (Figure 5.8), and exhibiting a diverse familiarity with Amazon Alexa skills and AsULearn (Figure 5.9) were recruited to participate in usability testing of the skill.

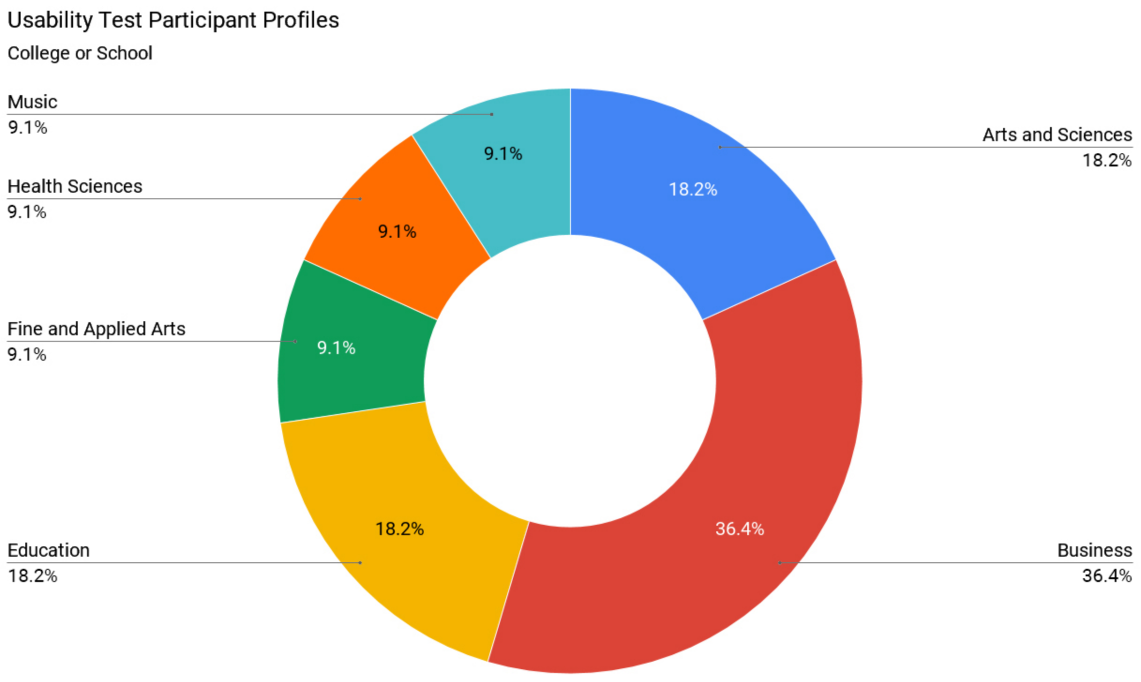


Figure 5.7: College or school of usability test participants.

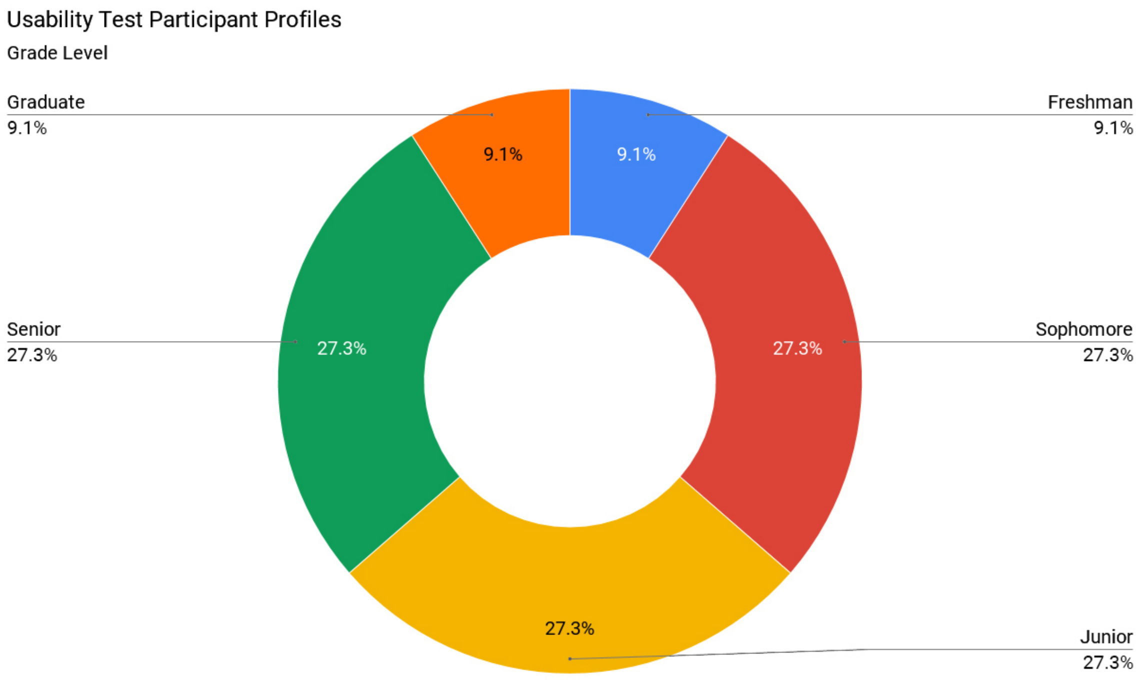


Figure 5.8: Grade level of usability test participants.

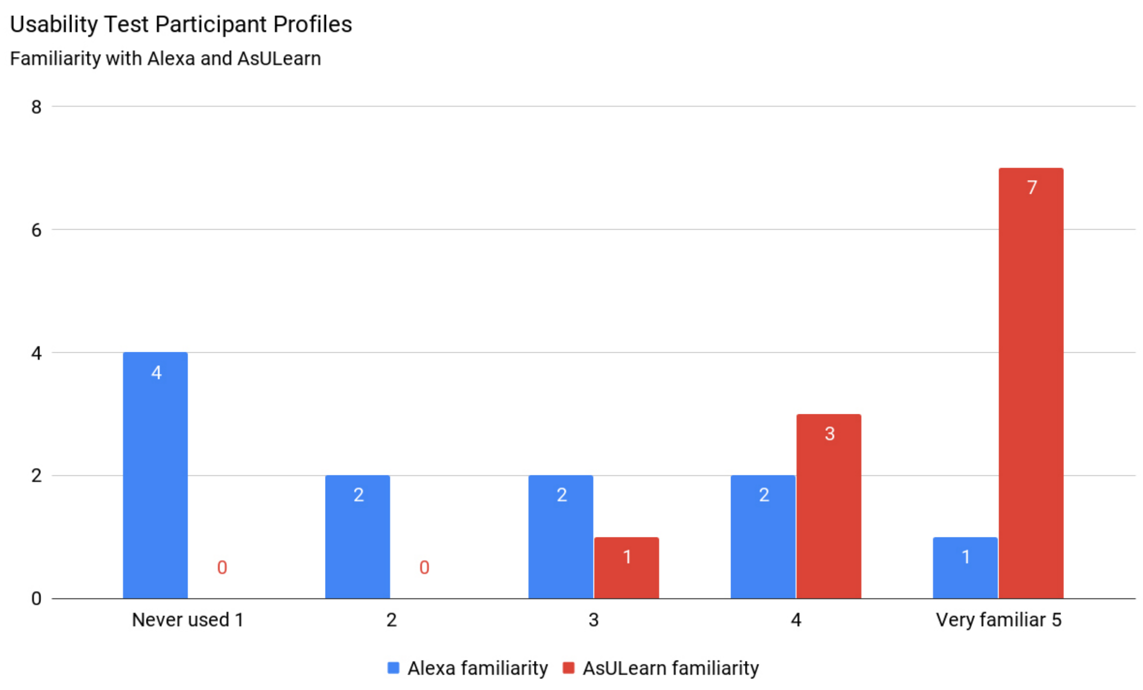


Figure 5.9: Alexa and AsULearn familiarity of usability test participants.

5.2.2 Testing Sessions

Participants were set up as users on a demo AsULearn/Moodle site with site and course announcements, grades, and due date content pre-populated. The developer was present as the test administrator during the sessions, but no assistance was provided during participants' interaction with the skill. The skill details card from the Amazon Alexa app was provided along with the following list of tasks to complete:

- Read the skill details card in the Alexa app.
- Enable the AsULearn skill and follow prompts to link to your AsULearn account.
- Open the AsULearn skill and try to get the following information:

- Site announcements,
- Course announcements,
- Grades, and
- Due dates.

The interaction with the skill portion of the usability sessions took, on average, five minutes to complete. After using the skill, participants were asked to complete an online survey to rate their experience. A follow-up interview was also conducted to get additional, open-ended feedback. The usability tests and interviews were recorded for post-test analysis.

5.2.3 Analytics Evaluation

Interestingly, the Alexa developer console analytics did not log any errors during the usability tests. Figure 5.10 from the skill analytics shows the percent distribution of total sessions into three outcome types: 1) Successful session: user-ended or skill-ended; 2) Incomplete sessions: ended due to user non-response; 3) Failed session: ended due to an error. However, observation and analysis of the usability test videos did show the occurrence of errors; the majority of errors were Alexa or Echo Dot errors, not errors with the skill design or implementation. Figure 5.11 shows the percent distribution of total sessions into two outcome types: 1) Successful session 2) Failed session; Figure 5.12 shows the source of the session error. Both figures are based on the video analysis.

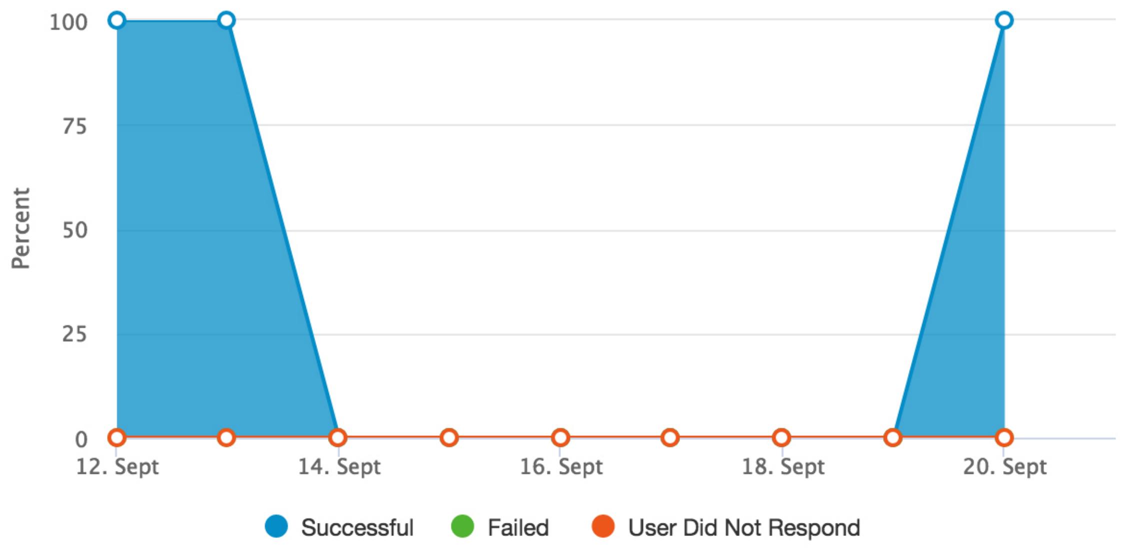


Figure 5.10: Session type distribution from Alexa analytics.

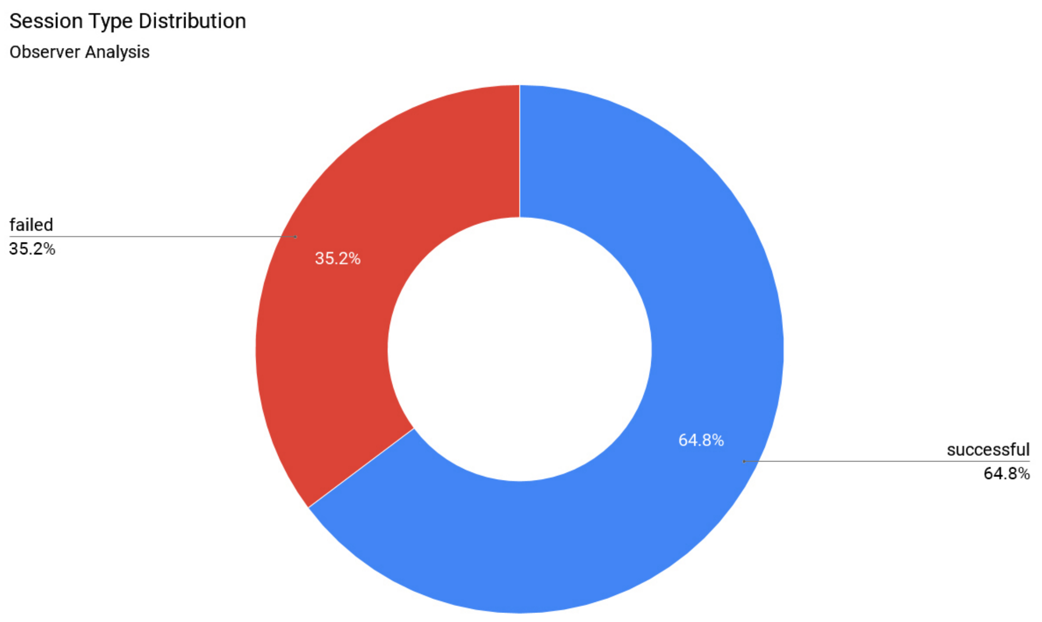


Figure 5.11: Session type distribution from observer analysis.

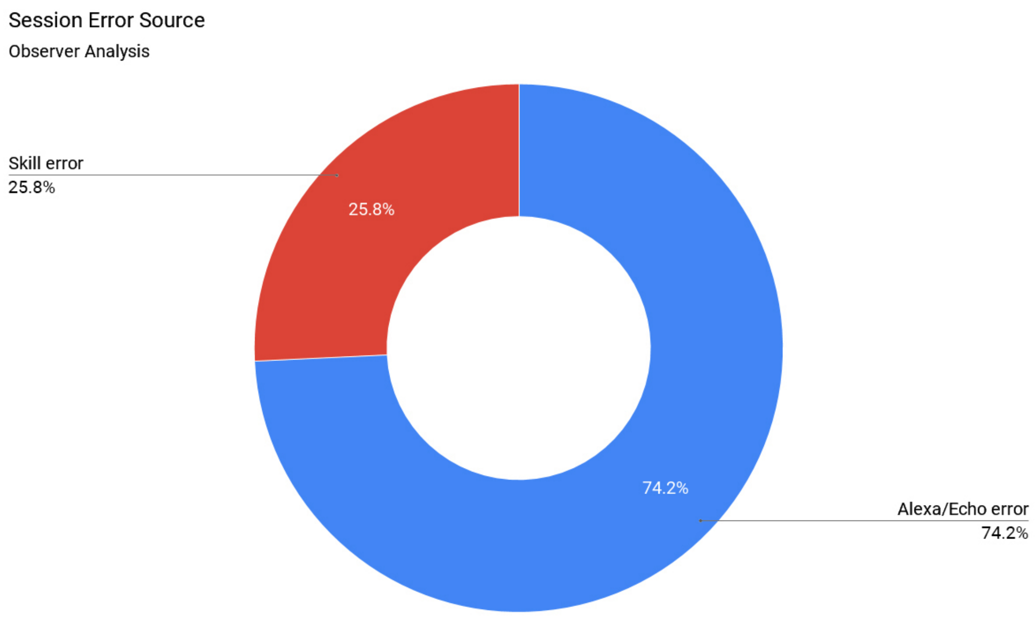


Figure 5.12: Session error source from observer analysis.

The Alexa app history and the usability testing video analysis revealed that the Alexa/Echo errors were because the device did not hear or understand the utterance correctly (either due to the device, the room acoustics, or the speech to text translation) or because the utterance used did not conform to any available invocation phrase formats (for example, “on” is not a valid connecting word). The skill errors that did occur were a result of a few utterances that were not planned in the original design. These utterances were added to the front-end interaction model of the skill in the Alexa developer console after usability testing was complete.

One participant used utterances for intents that have not yet been implemented in the skill, such as asking for course- or assignment-specific due dates. Unmapped utterances should invoke the FallbackIntent; however, the errors that were generated from these utterances were Alexa/Echo errors due to the commands not being heard, understood, or translated properly, so the unmapped utterances never reached the skill to be processed as FallbackIntent requests.

5.2.4 Objective Survey

The feedback survey completed by participants after using the skill was designed and built based on the SUISQ-MR. The four usability factors were distributed across the survey as user goal orientation (questions 1-2), customer service behavior (questions 3-4), speech characteristics (question 5), and verbosity (questions 6-8). A 5-point Likert scale was used, with 1 being "Strongly disagree" and 5 being "Strongly agree." Figure 5.13 shows the results of the survey completed by participants after using the skill. Participants rated the skill above average for user goal orientation, customer service behavior, and speech characteristics. The survey results demonstrate that the system's verbosity has some room for improvement. An overall score of 4.10 for the first version of the skill is very promising.

The survey was designed to allow for anonymous feedback so participants would feel comfortable providing their honest evaluation of the skill. In addition to the rating questions, an open-ended text area was provided for additional comments. Many users chose to discuss their open-ended feedback during the interview. The feedback that was provided in the comments text area matched the discussions conducted during the interviews.

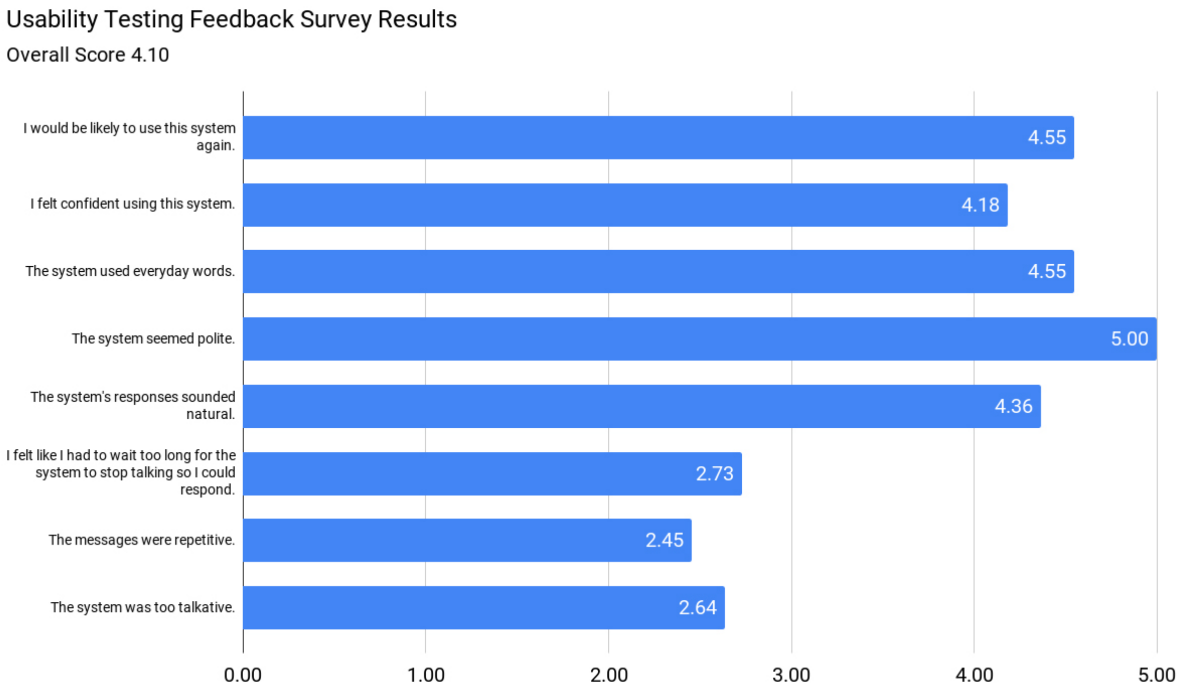


Figure 5.13: Usability testing survey results.

5.2.5 Participant Interview Evaluation

The interview consisted of several open-ended questions to allow participants to discuss their opinion of the skill in greater detail. Participants were asked what they found easy about using the skill, what they found difficult about using the skill, if they encountered anything unexpected during the use of the skill, and if there were other features or capabilities they would find useful to have in the skill. They were also asked about the PIN section of the account linking process; specifically, if it was obvious that it was optional, as well as its purpose.

All but one participant enabled the PIN option during the usability tests. Interviews revealed that most participants realized the optionality of the PIN after they had already created it during the account linking process, and they assumed its purpose was for an additional layer of access protection. However, they expressed that additional clarity on the account linking form would be helpful. All users reacted positively to the PIN feature.

Feedback regarding what was easy about using the skill tended to vary based on the user's familiarity with Alexa. Participants with little prior Alexa experience included voice interface characteristics in addition to comments specific to the AsULearn skill. These comments referred to the system's responsiveness, understandability, and ability to understand the user, as well as the ease of not having to use a computer and the ability to use it while doing other activities. Users who were more familiar with Alexa communicated that the skill was very similar to and even easier to use than other Alexa skills. Comments included that it was simple to learn how to open and use the commands, it was easy to figure out how to ask for things, it used natural wording, the listing of options was helpful, and the PIN was easy to create and use. Participants liked that the skill provided emails from teachers (course announcements) and expressed that the due dates feature was really helpful.

Participant evaluation of difficulty using the skill was also somewhat based on previous exposure to Alexa. Those with less Alexa experience discussed difficulty figuring out what they needed to say to use the skill; however, they also indicated that any difficulty with the utterances would be easily overcome with a little practice using the system. A common pain point with using the skill was the length of some of the responses, especially the site and course announcements. Participants suggested only listing the most recent announcement since that was likely what they were most interested in hearing. One user indicated difficulty annunciating the word “announcement” clearly, and a couple mentioned it would be nice to only have to say the PIN one time.

For user feedback regarding expectations of the skill, everyone mentioned that any errors experienced during the use of the skill were unexpected. Two participants mentioned that the comprehensive listing of information was a surprise (number of due dates and announcements); however, one user discussed this point as a positive feature. Another participant expressed initial confusion between site and course announcements, as she did not regularly pay much attention to the front page site announcements on AsULearn.

Many of the suggestions for additional features or capabilities for the skill were ideas discovered in previous research, such as the student survey conducted to aid in the design of the skill. Most of these features are already planned for future development work and include:

- Get grades or due dates for a specific class;
- Get due dates for a specific time frame (today, this week);
- Get the last grade in a specific class;
- Get a specific assignment grade;
- Send email;
- Get the description of an assignment;
- Get syllabus content, such as attendance policy, grading policy, office hours and location, classroom;

- Get number of replies to a forum post;
- Get student discussion forum updates;
- Get messages (an alternative way for faculty to communicate with students);
- Get class participants;
- Take a quiz;
- Speak an answer to an open-ended question with the ability to go in and edit it later; and
- Get due date reminders.

5.2.6 Usability Testing Conclusions

Usability testing provided several valuable learning experiences, both about the skill design and implementation, as well as the usability testing process itself. Since the skill is currently in development and not available on production AsULearn, a demo AsULearn/Moodle instance was used for the testing process. AsULearn uses Shibboleth single sign-on authentication; however, this system is only available on Appalachian supported applications. Since Appalachian accounts are actually Google GSuite accounts on the back-end, Google OAuth was implemented as an alternative on the demo site. While users are familiar with logging in to Google applications with their Appalachian credentials, they are not familiar with this process in the context of AsULearn. The inconsistency between authentication on the production system and the system used for testing made for a choppy account linking process. Caching of previous Google sessions resulted in the need to use a brand new incognito browser window between each usability test to ensure previous logins were not still active.

The first test participant was not informed that he would be testing the skill with a demo site. This omission caused a bit of initial confusion, as the courses listed were not familiar to him. Subsequent test participants were advised about the demo site to prevent this confusion that was unrelated to the use of the skill.

Site announcement content was pulled directly from AsULearn, as this is publicly available from the front page of the site. However, course announcements, grades, and due dates were fabricated based on an approximation of their content and length. The use of placeholder content may have affected the results of the usability testing because it was material that was not familiar to the students. If a participant's actual courses, announcements, and assignments were able to be used in testing, there may have been more precise feedback with regard to the evaluation of the skill.

The skill details card, which provides a description of the skill, its capabilities, and example utterances, was provided to participants during testing. Users with less Alexa experience tended to rely on the details card for most of the tasks they were asked to complete. All users appeared to refer to the card when errors were encountered. While there is typically no printed help documentation during the standard use of a voice interface, the skill details card in the Alexa app is the primary source of information about how to use a skill. It is likely a user encountering errors during the regular use of a skill would also refer to the skill details card for help.

The majority of the errors generated during the usability testing were a result of Alexa or the Echo device not hearing, understanding, or translating the request accurately. It is not evident if this lack of performance was due to the device quality, the device location, or the test room acoustics. Most of the challenges encountered specific to the usability test process did not appear to have an impact on users' evaluation of the application; however, they did provide knowledge about how to improve on the process for the future. Further experimentation with these variables would be valuable to reduce these types of avoidable errors.

Usability testing did reveal some areas for further research and possible enhancement of the skill. During a few of the sessions, the site and course announcement utterances triggered the Alexa Announcement feature. One of the ways this feature is invoked is by saying "Alexa, announce that ..." followed by a message; the message is then broadcast from all linked Alexa devices. Given this very similar Alexa feature and the possibility of it being confused with the site and course announcements intent, it may be worth promoting alternative utterances for these intents.

Users familiar with Alexa seemed to speak more clearly, they were more familiar with how to phrase commands, and they experienced fewer errors during the sessions. This observation indicates that there may be a steeper learning curve for inexperienced users; however, this is irrespective of the usability of the skill and indicates that more frequent use of Alexa should make the use of the skill easier as well.

All users expressed surprise and delight that Alexa knew and used their name in the response to the LaunchRequest. Personalization of the skill interaction appears to be appreciated and highly valuable for the usability of the interface.

The AMAZON.FOUR_DIGIT_NUMBER slot type worked very well for variations on spoken numbers for PIN verification. While most users read their PIN numbers as four individual digits, one user spoke it as two 2-digit numbers. Even with this alternative version of the number, the PIN was verified and the request was completed.

At the end of each testing session, all participants expressed their enjoyment in using the skill and the capabilities it provided, as well as their hope that it will be implemented on AsULearn in production.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

Since the skill is in a relatively complete, useable first version, and given that some of the feedback from the usability testing seemed to vary based on personal preference, the next step will be to release the skill in production and let a broader audience of students use and interact with the skill. This increased usage will enable additional usability research, as well as more accurate and complete skill analytics from the Alexa developer console. More comprehensive user feedback will be sought before significant changes to the interface are implemented, to ensure the changes align with the needs of most of the skill's user base.

Upon completion of this thesis, the plugin code for the Alexa skill will be available on GitHub at `https://goo.gl/jCJGLG`, and the plugin code for the RESTALEXA protocol plugin will be available at `https://goo.gl/eMdmBT`. The code for the Moodle plugins and demonstration videos is provided on the enclosed CD, and a list of the files is available in Appendix A.3.

With the implementation of a few final modifications and enhancements based on usability test feedback, the skill will be submitted to the Appalachian State University Center for Academic Excellence Learning Technology Services team for review before submitting for certification and launch in the Alexa Skills Store.

6.2 Future Work

The initial development of a voice application for accessing information in the Moodle learning management system was the core of this research; however, there are additional, further reaching implications to investigate in the future. With the goals outlined in this thesis accomplished and an initial version of the Alexa skill for Moodle functioning, there are several ideas for expanding the skill's capabilities as well as other areas of usability research to explore.

Future work may include adding instructor-specific tasks such as the ability to hear a list of assignments that need grading, as well as the ability to create voice activities in Moodle. Allowing students to complete quizzes verbally is a feature that would offer added value for instructors and students alike.

Increasing the granularity of the current intents would improve and expand the existing capabilities of the skill:

- Allow users to request more site or course announcements beyond the latest five;
- Allow users to request other announcement content, from forums other than the primary course announcements forum;
- Allow users to request forum post replies, especially to discussion posts the user has authored;
- Allow users to request grades from specific courses;
- Allow users to request grades for specific assignments;
- Allow users to request due dates for specific courses;
- Allow users to request specific event types (site, category, course, group, or user);
- Allow users to request due dates for specific assignments;
- Allow users to request due dates for a specific time period (today, this week); and
- Use the number of events and days to lookahead from the user preferences, if set, instead of the sitewide configuration to determine which due dates are returned.

It would also be interesting to explore some of the other standard built-in intents provided by Amazon. The AMAZON.NextIntent, AMAZON.PreviousIntent, and AMAZON.RepeatIntent could be helpful for responses with list content. Instead of placing a setting or hard limit on the number of site and course announcements returned, these intents would allow the user to browse through the list of announcements as desired, providing more control over the interaction.

Responses to the student survey suggested adding the ability for students to confirm assignment submissions, as well as to find out what assignments were missing submissions. The students also recommended adding the ability to set due date reminders. Currently, the Alexa Skills Kit (ASK) front-end only allows skill integration with Amazon Alexa Lists, so it would be possible to add an item to a user's shopping or to-do list in the Alexa app, but not to set a Reminder. If a user configures Alexa Lists to sync with a third-party service that provides date capabilities for list items, or if Amazon adds Reminders and Alarms integration with skills, then due date reminders would be a possible feature to add. Survey results also showed other features students would be interested in include notifications of new assignment postings, the ability to get attendance records, and the ability to hear resources, such as syllabi, documents, and instructor contact information and office hours.

Preliminary user research also indicated that users would like to submit assignments or activities like forum posts through the Alexa skill, as well as to send email messages to instructors. Since possible user responses must be input in the Alexa front-end (in the form of utterances mapped to an intent or a slot), there would need to be some advancement in the ASK natural language processing capability before features that interpreted unmapped user input could be considered.

Usability testing revealed areas for further voice user interface (VUI) design research, including reducing the site and course announcement responses to just the most recent post, changing the primary utterances for site and course announcements to use an alternative synonym for "announcement," and revisiting the implementation of multi-utterance sessions so a user only has to speak the PIN verification one time per interaction. During session interviews, participants suggested additional ideas for consideration. Some of these features would involve further design research, as well as some standardization of content location across AsULearn courses.

- Get the description of an assignment.
- Get syllabus content, such as attendance policy, grading policy, office hours and location, classroom.
- Get number of replies to a forum post.
- Get student discussion forum updates.
- Get messages (an alternative way for faculty to communicate with students).
- Get class participants.

Many of the newer Alexa devices incorporate some type of graphic display in addition to the voice interface. Adding Card responses to the web service for visual content like images or video to support the spoken responses is another avenue for future development.

Exploring the VUI implementation with other voice assistants, like Google Home and Apple Siri, would also be interesting and support even more voice integration with Moodle. Other types of Alexa skills that could be built would also be worth researching. For example, a Flash Briefing skill that provided reminders about upcoming assignments could be another means of access for users. It would also be beneficial to conduct more usability testing to reach a larger and more diverse group of users. Beta test invitations could be sent to a wider audience for independent skill use and testing, with a more in-depth follow-up survey to rate the interface.

In addition to expanding the functionality of the skill, research on the usability impact would be interesting to explore. The spoken/auditory access to Moodle may enhance the accessibility of the application for users with disabilities. Measuring the impact of the added speech modality to the learning management system for users with disabilities is a possible area of further research.

Providing students access to their current performance may also have a positive impact on student success. Research to find out if the increased access to academic status and learning materials afforded by the voice interface positively affects overall student success is another area of interest. With the development of the initial application and usability testing complete, these extended areas of development and research can be explored in the future.

Bibliography

[1] 'alexa, teach me!' voice-activated features coming to canvas and blackboard lms. World Wide Web electronic publication, https://www.moodlenews.com/2017/alexa-teach-me-voice-activated-features-coming-to-canvas-and-blackboard-lms, August 2017.

[2] Canvas announces skill for amazon alexa. World Wide Web electronic publication, https://www.prnewswire.com/news-releases/canvas-announces-skill-for-amazon-alexa-300495496.html, July 2017.

[3] Amazon Alexa. *Alexa Voice Design Guide*. World Wide Web electronic publication, https://developer.amazon.com/designing-for-voice.

[4] Amazon Alexa. *Understand Custom Skills*. World Wide Web electronic publication, https://developer.amazon.com/docs/custom-skills/understanding-custom-skills.html.

[5] Lyndon Cerejo. Designing voice experiences. World Wide Web electronic publication, https://www.smashingmagazine.com/2017/05/designing-voice-experiences, May 2017.

[6] Lee Yen Chaw and Chun Meng Tang. The voice of the students: Needs and expectations from learning management systems. In *Proceedings*. European Conference on Games Based Learning, 2017.

[7] Michael H. Cohen, Jennifer Balogh, and James P. Giangola. *Voice User Interface Design*. Addison-Wesley, 2004.

[8] Major League Hacking. Hacking with amazon alexa. In *AppHack*, 2018.

[9] Randy Allen Harris. *Voice Interaction Design: Crafting the New Conversational Speech Systems*. Morgan Kaufmann, 2005.

[10] Susan L. Hura. Usability testing of spoken conversational systems. *Journal of Usability Studies 12: 155 - 163*, August 2017. http://www.uxpajournal.org/usability-spoken-systems.

[11] Instructure Canvas. *Digital Assistant: Canvas Skill for Alexa*, August 2017. World Wide Web electronic publication, https://community.canvaslms.com/docs/DOC-11957-digital-assistant-canvas-skill-for-alexa.

[12] Internet Engineering Task Force. *OAuth 2.0 Authorization Framework*, October 2012. World Wide Web electronic publication, https://tools.ietf.org/html/rfc6749.

[13] Doug Lederman. The learning management landscape. World Wide Web electronic publication, https://www.insidehighered.com/digital-learning/article/2017/05/17/where-trends-are-going-lms-market, May 2017.

[14] James R. Lewis. *Practical Speech User Interface Design*. CRC Press, 2011.

[15] James R. Lewis. Standardized questionnaires for voice interaction design. *The Journal of the Association for Voice Interaction Design 1: 1 - 16*, April 2016.

[16] Szymon Machajewski. *My Blackboard - Amazon Alexa Skill*. Gamification and Play :: Experience Design for Learning, June 2017. World Wide Web electronic publication, https://szymonmachajewski.wordpress.com/2017/06/05/my-blackboard-amazon-alexa-skill.

[17] Jenni McKienzie, David Attwater, Jon Bloom, Karen Kaushansky, and Julie Underdahl. 49 vui tips in 45 minutes. In *Conference and Exhibition*. SpeechTEK, 2010.

[18] Moodle. *Moodle Developer Documentation*. World Wide Web electronic publication, https://docs.moodle.org/dev.

[19] Moodle. *Moodle Site Administrator Documentation*. World Wide Web electronic publication, https://docs.moodle.org/34/en.

[20] Moodle. *REST protocol (with JSON/XML payload support)*, February 2016. World Wide Web electronic publication, https://moodle.org/plugins/webservice_restjson.

[21] Clifford Nass and Li Gong. Speech interfaces from an evolutionary perspective. *Communications of the ACM 43: 36 - 43*, September 2000.

[22] Jakob Nielsen. 10 usability heuristics for user interface design. World Wide Web electronic publication, https://www.nngroup.com/articles/ten-usability-heuristics, January 1995.

[23] Cathy Pearl. *Designing Voice User Interfaces: Principles of Conversational Experiences*. O'Reilly Media, 2016.

[24] Walter Quesada and Bob Lautenbach. *Programming Voice Interfaces: Giving Connected Devices a Voice*. O'Reilly Media, 2018.

[25] John Rome. Alexa goes to college: Asu's innovative use of voice technology. In *Annual Conference*, 2017.

[26] Colin Tankard. What the gdpr means for businesses. *Network Security*, June 2016.

[27] Kathryn Whitenton. Voice interaction ux: Brave new world...same old story. World Wide Web electronic publication, https://www.nngroup.com/articles/voice-interaction-ux, January 2016.

[28] Kathryn Whitenton. Audio signifiers for voice interaction. World Wide Web electronic publication, https://www.nngroup.com/articles/audio-signifiers-voice-interaction, September 2017.

Appendix A

Appendix

A.1 RESTALEXA Protocol Plugin Sample Code

Sample PHP code from the RESTALEXA protocol plugin is provided below.

```
/**
 * This method parses the php input for the JSON request, web
    service token, and web service function.
 */
protected function parse_request() {
    // Retrieve and clean the POST/GET parameters from the
       parameters specific to the server.
    parent::set_web_service_call_settings();

    // Get JSON request as string and object for processing.
    $datastring = file_get_contents('php://input');
    $data = json_decode(file_get_contents('php://input'), true);

    // Add GET parameters.
    $methodvariables = array_merge($_GET, (array) $data);

    // Set REST format to JSON.
    $this->restformat = 'json';

    // Set token to query string argument.
    $this->token = isset($methodvariables['wstoken']) ?
       $methodvariables['wstoken'] : null;
    unset($methodvariables['wstoken']);

    // Set web service function to query string argument.
    $this->functionname = isset($methodvariables['wsfunction'])
       ? $methodvariables['wsfunction'] : null;
    unset($methodvariables['wsfunction']);

    // Prepare request to send to web service.
    $request = array('request' => $datastring, 'token' => '');

    // Check if user accessToken is in request.
    if ($data['context']['System']['user']['accessToken']) {
        try {
            // Save web service user token passed in query
               string.
            $webserviceusertoken = $this->token;

            // Get user token from request.
            $this->token = $data['context']['System']['user']['
               accessToken'];

            // Check if user token is valid.
            $this->authenticate_user();
            $request['token'] = 'valid';
        } catch (Exception $ex) {
            // Provided user accessToken is invalid.
            // Pass web service user token to plugin for account
                linking request.
            $this->token = $webserviceusertoken;
        }
    }

    $this->parameters = $request;
}
```

A.2 Alexa Skill for Moodle Plugin Sample Code

Sample PHP code from the Alexa skill for Moodle plugin is provided below.

```
/**
 * Main function to process web service request.
 *
 * @param string $request
 * @param string $token
 * @return mixed web service response
 */
public static function alexa($request, $token = '') {
    self::$requestjson = json_decode($request, true);
```

```
self::initialize_response();

// Check the URL of the signature certificate.
if (!self::signature_certificate_url_is_valid($_SERVER['
    HTTP_SIGNATURECERTCHAINURL'])) {
     debugging('Invalid␣signature␣certificate␣URL',
        DEBUG_DEVELOPER);
     return http_response_code(400);
}

// Only perform signature validation on live, internet
    accessible server that can receive requests directly from
     Alexa.
// Signature is encrypted version of request, no way to
    simulate.
if (!self::is_development_site()) {
     // Check the signature of the request.
     if (!self::signature_is_valid($_SERVER['
        HTTP_SIGNATURECERTCHAINURL'], $_SERVER['
        HTTP_SIGNATURE'], $request)) {
          debugging('Invalid␣signature', DEBUG_DEVELOPER);
          return http_response_code(400);
     }
}

// Check the request timestamp.
if (!self::timestamp_is_valid()) {
     debugging('Invalid␣timestamp', DEBUG_DEVELOPER);
     return http_response_code(400);
}

// Verify request is intended for my service.
if (!self::applicationid_is_valid()) {
     debugging('Invalid␣application␣id', DEBUG_DEVELOPER);
     return http_response_code(400);
}

// Process request.
if (self::$requestjson['request']['type'] == 'LaunchRequest'
    ) {
     return self::launch_request($token);
} else if (self::$requestjson['request']['type'] == '
    IntentRequest') {
     switch(self::$requestjson['request']['intent']['name'])
        {
            case "GetSiteAnnouncementsIntent":
                return self::get_announcements(1, 'the site');
                break;
            case "GetCourseAnnouncementsIntent":
                return self::get_course_announcements($token);
                break;
            case "GetGradesIntent":
                return self::get_grades($token);
                break;
            case "GetDueDatesIntent":
                return self::get_due_dates($token);
                break;
            case "AMAZON.CancelIntent":
            case "AMAZON.StopIntent":
                return self::say_good_bye();
                break;
            case "AMAZON.FallbackIntent":
                return self::get_help(true);
                break;
            case "AMAZON.HelpIntent":
            default:
                return self::get_help();
                break;
        }
    } else if (self::$requestjson['request']['type'] == '
        SessionEndedRequest') {
          self::session_ended_request();
    }
}
```

A.3 Demonstrations and Complete Code Listing

Demonstration videos and source code for both Moodle plugins is provided on the enclosed CD. The playlist of demonstration videos is available at `https://www.youtube.com/playlist?list=PLD76zgi_Ubzlg1NvLNjkPs81wuf9sK1AX`. Upon completion of this thesis, the plugin code for the Alexa skill will be available on GitHub at `https://goo.gl/jCJGLG`, and the plugin code for the RESTALEXA protocol plugin will be available at `https://goo.gl/eMdmBT`. The following is a list of the files that appear on the CD:

- Demo videos
  - account-linking.mp4: Video demonstration of the account linking process
  - site-announcements.mp4: Video demonstration of getting site announcements from the skill
  - course-announcements.mp4: Video demonstration of getting course announcements from the skill
  - grades.mp4: Video demonstration of getting grades from the skill
  - due-dates.mp4: Video demonstration of getting due dates from the skill
- Plugins
  - restalexa
    - classes/privacy/provider.php: Class which implements privacy providers of Moodle's Privacy API for GDPR compliance
    - db/access.php: Capabilities definition for plugin
    - lang/en/webservice_restalexa.php: Definition of English strings for plugin
    - locallib.php: Plugin's internal logic
    - README.md: Information about installing and using the plugin
    - server.php: Endpoint for web service protocol
    - version.php: Metadata about the plugin

- alexaskill
  - account_linking_form.php: Class which extends Moodle's Form API to define account linking form and perform data validation
  - account_linking.php: Output and display of account linking form
  - classes/privacy/provider.php: Class which implements privacy providers of Moodle's Privacy API for GDPR compliance
  - course_slot_values.php: Formatting and display of course names from Moodle site for COURSE slot configuration in Alexa skill front-end
  - db/install.php: Automatic configuration of majority of GUI settings for plugin on installation
  - db/services.php: Definition of external functions and web services provided by plugin
  - externallib.php: Class which extends Moodle's External functions API to create methods that can be accessed by external programs/web services
  - interaction-model.json: JSON file definition of Alexa skill configuration for import into Alexa skill front-end
  - lang/en/local_alexaskill.php: Definition of English strings for plugin
  - README.md: Information about installing and using the plugin
  - settings.php: Configurable settings for plugin
  - tests/account_linking_form_test.php: PHPUnit tests for account linking form
  - tests/externallib_test.php: PHPUnit tests for external functions and web services
  - version.php: Metadata about the plugin

Vita

Michelle Layne Melton was born in Annapolis, Maryland, to David and Donna Goeman. She graduated from Glen Burnie Senior High School in Maryland in June 1993. The following autumn, she entered Anne Arundel Community College to study Business Administration and Marketing and in December 1995, she graduated summa cum laude and was awarded the Associate of Arts degree. In the fall of 1995, she continued her studies at the University of Baltimore in Maryland and in May 1998, she graduated summa cum laude and was awarded the Bachelor of Science degree.

Ms. Melton spent the next several years working full-time in marketing, advertising, graphic design, and web development. In the fall of 2013, she began undergraduate study in computer science. In the spring of 2017, she began study toward a Master of Science degree in Computer Science at Appalachian State University in North Carolina. The M.S. was awarded in May 2019. Ms. Melton is a web developer and programmer for Learning Technology Services at Appalachian State University. She is also an active rock climber and lives with her husband in Boone, North Carolina.