San José State University

CMPE-287 Software Quality Assurance

Updated Test Design Document

Alexa For Mobile Phones

Harsha Muktamath (011825138)
Anup VijayaKumar Kulkarni (011772475)
Sampath Lakkaraju (011818781)
Srivatsa Mulpuri (011431147)

Department of Computer Engineering
Spring 2018

Submitted to: Dr. Jerry Gao

Contents

Chapter 1 Targeted System Overview

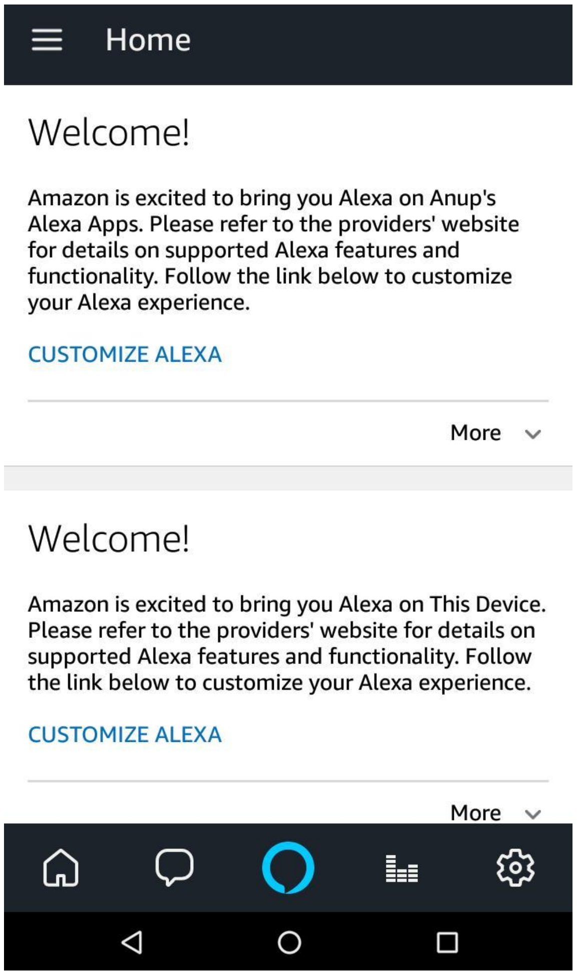
1.1 Introduction

Software testing is an important part of the any software development life cycle. There are numerous models and methods of software testing depending on the system's state, dependent environment, modifications or developments made to the system and user requirements. This project is intended to understand the various model and methods used and to apply them on practically. For the scope of the project we have used an Artificial intelligence based mobile application Amazon Alexa.

Artificial Intelligence in simple terms is intelligence demonstrated by the computer systems. Any device that perceives its environment and performs task accordingly to maximize its probability of achieving its goals is said to have artificial intelligence. Though we are eons away from creating a perfect artificial intelligence system, there has been huge progress towards one. One of the fields where this can be observed in conversations with computers. Alexa is one such conversational AI system hosted on the cloud. Conversational AI system is one which can communicate with people in a natural way while performing task, solve problems, learn over a period. The system is owned and managed by "amazon". The makers of alexa describes voice activated personal assistant.

Amazon Alexa is the mobile application allows mobile phone users to use the features of the Alexa. The application is free and can be installed using Play Store for Android. The iPhone application is not allowed to use the full functionalities of the Alexa. Amazon Alexa can get you weather updates, play songs, send SMS and even turn your lights on and off if you have a smart bulb that is connected to the internet. Amazon has recently introduced alexa skills which can be used to interact with third party application and can avail numerous services using the voice assistant.

We are planning to use some of these features of the amazon alexa and test system. Since the code is not open source, so we have performed Black Box testing on the application. We have considered all the methods for the system and have selected Scenario based testing and Decision table.



Welcome!

Amazon is excited to bring you Alexa on Anup's Alexa Apps. Please refer to the providers' website for details on supported Alexa features and functionality. Follow the link below to customize your Alexa experience.

CUSTOMIZE ALEXA

More

Welcome!

Amazon is excited to bring you Alexa on This Device. Please refer to the providers' website for details on supported Alexa features and functionality. Follow the link below to customize your Alexa experience.

CUSTOMIZE ALEXA

More

Screenshot of the Amazon Alexa application

1.2 Objectives of our project

The main objective of this project is to learn, understand and implement various test models and methods of black box testing. For the same we have come up with the below steps to achieve the same.

1. Understand all the methods of testing
2. Verify the compatibility of the method with our application.
3. Compare each other and come up with two methods to test the application.

Chapter 2 Test Requirements

2.1 Requirement

Below are the requirements that we are planning to test on Alexa mobile application.

1. Smart Light: Smart Lights should be configurable and controlled to Turn on and Off using Alexa Mobile App.
2. Find My Phone: Alexa should be able to find lost phone using these three ways
    a. TrackR
    b. Have Alexa call your phone
    c. Play a song from your Android phone
3. Send An SMS: Use Alexa to send messages.
4. Play Music: Alexa should be able to play music on Android phone with Amazon music options.

2.2 Tools

The below mentioned two methods are used to test Amazon Alexa.

1. VUI testing tool: This is the first of the two methods that we are going to use to test Amazon Alexa. VUI is Voice User Interface testing tool. This is a four step process
   a. Setting up voice user interface
   b. Setting up Lamba function using Amazon Web Services
   c. Connecting Voice user Interface to Lambda Function
   d. Testing Alexa Skill
2. Alexa Skill Testing Tool: This is the second method that we are going to use to test Amazon Alexa. We are going to use Echosim.io which is a community tool that simulates the look and feel of Amazon Alexa. It gives us the ability to experience a realistic interaction with Alexa capabilities.

Chapter 3 Selected Test Design Methods/Test cases

3.1 Different Black Box Testing methods explored

At a high level, below black box methods were analyzed and considered

Black box Testing methods	**Considered Y/N**	**Used Y/N**	**Comments**

Category Partition	Y	N	Good method but not very suitable for mobile Apps
Decision Table	Y	Y	Too many test cases so we had to limit the conditions. Ideally, we wanted to test with all conditions so, this method is not the best choice
Scenario	Y	Y	Used to develop scenarios from a user perspective even though too many scenarios. However good method for AI mobile app testing
State Transition	Y	N	More useful for communication based systems and GUI based

			systems with lots of different window states, etc.
Equivalence Partitioning	Y	N	This is a good method to use, however it is similar to Category partition testing, but not suitable for AI based mobile app
Boundary Value	Y	N	This is useful for enterprise based products

From Alexa App mobile testing and AI features testing, these testing methods were also looked at which are specific to Alexa. However, we choose to develop our test strategy based on Scenario based and Decision table based for this project.

1. Wizard of Oz Testing

WOZ testing allows designers to evaluate the program flow (i.e., test call flow, grammar, prompts, etc.) without investing a significant amount of resources into the project [8].

2. Prototype Testing

prototyping takes testing to the next level as it involves using an actual system prototype [12], which implies that coding has been performed. It is important to test using a functioning prototype and has been referred to as the “gold standard” in evaluating an application’s usability

3. VUI Review Testing

VUI Review Testing (VRT) is a type of “holistic” and “experiential” review where a VUI designer reverses roles and acts as the caller of the system exercising each use-case scenario previously determined. It is important that the tester be a VUI designer or usability specialist and not a developer because the goal of VRT is to verify the quality of the user experience [. VRT should be conducted after DTT has already been performed but before User Acceptance Testing (UAT) to catch possible usability problems, prompt quality problems, and to determine if and where pauses might need to be placed or extended in addition to other, more general, problems [10]. Because VRT catches these types of issues, it increases the informativity of usability testing.

4. Usability Inspection

Usability inspections are more empirical in nature and, thus, more objective

however, the number of established metrics for evaluating VUIs is minimal.

3.2 Method #1 Scenario based method

3.2.1 Test coverage

The test cases that are created depicts end users usage scenarios based on system requirements and use cases. Since the test scenarios can be independent test cases or a combination of test suites, this is helpful for our Alexa AI mobile based app testing.

3.2.2 Testing Strategy

Step #1 – Identify use cases/user operation scenarios

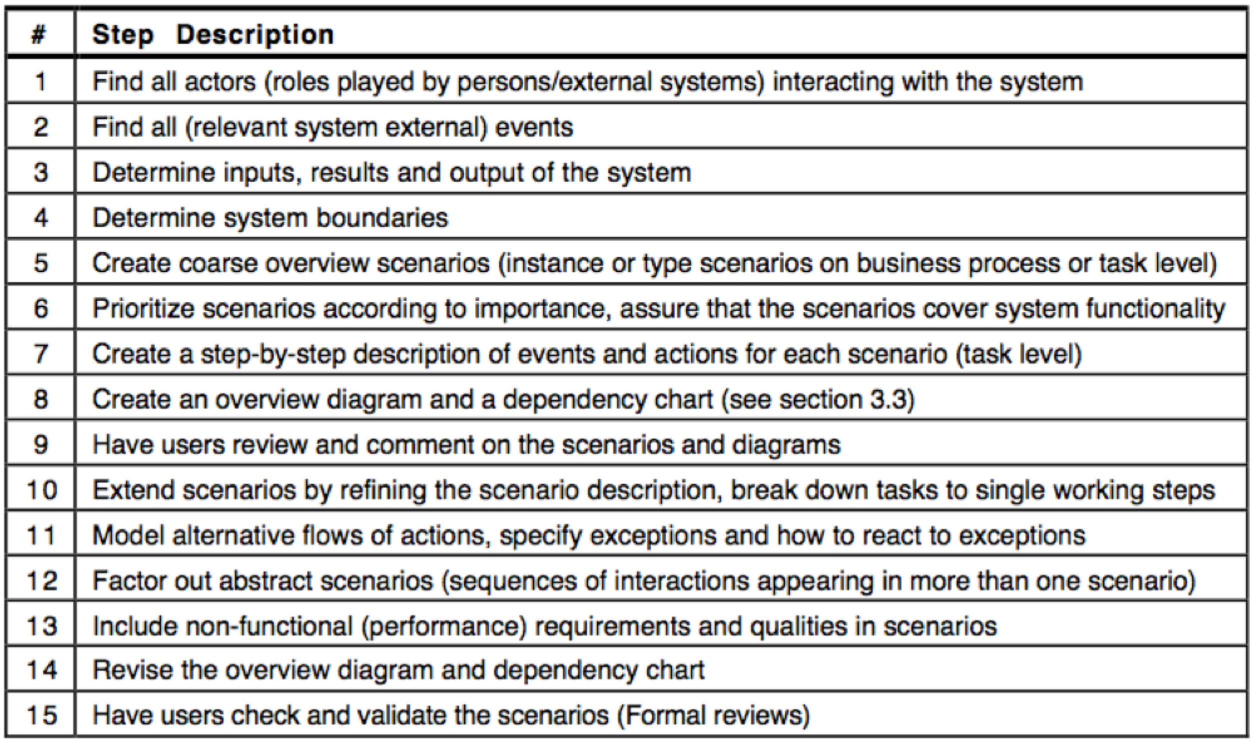
Step #2 - Define test cases based on the identified scenarios

Step #3 - Execute scenario tests

3.2.3 Our approach

We selected use case and role based approach for scenario testing as it suits our needs. We are using this method as it focuses on the interaction between user and the system having multiple roles and differing environmental conditions.

3.2.4 Test Procedure



#	Step Description
1	Find all actors (roles played by persons/external systems) interacting with the system
2	Find all (relevant system external) events
3	Determine inputs, results and output of the system
4	Determine system boundaries
5	Create coarse overview scenarios (instance or type scenarios on business process or task level)
6	Prioritize scenarios according to importance, assure that the scenarios cover system functionality
7	Create a step-by-step description of events and actions for each scenario (task level)
8	Create an overview diagram and a dependency chart (see section 3.3)
9	Have users review and comment on the scenarios and diagrams
10	Extend scenarios by refining the scenario description, break down tasks to single working steps
11	Model alternative flows of actions, specify exceptions and how to react to exceptions
12	Factor out abstract scenarios (sequences of interactions appearing in more than one scenario)
13	Include non-functional (performance) requirements and qualities in scenarios
14	Revise the overview diagram and dependency chart
15	Have users check and validate the scenarios (Formal reviews)

3.2.5 Scenario based test cases

Why we used?

It is helpful to document software specifications and requirements. In this case, for Alexa we have different scenarios such as connectivity to wifi, whether it is connected to microphone and, we had 4 scenario, Configuring Alexa to 1 Find phone. 2. Play Music 3. Turning on and off the Smart light 4. And sending SMS to other mobiles. Therefore, this model suits best for scenario-based testing. Therefore, we selected the scenario-based testing to writ our test cases for these individual user scenarios listed above. We also consider the aspect that for Alexa there are many data combinations and many possible paths. Therefore, it is important to consider scenario-based testing in these conditions. Sometimes, methods like equivalence partitions boundary value analysis do not provide enough to cover end to end quality. Therefore, it is important to consider scenario-based testing in these cases to ensure that the software is working for some of the very common scenarios. This helps finding bugs.

When we concluded all test cases passed?

When all scenario runs from end to end

Scenario 1		Device connectivity checks	
Test Preparation			
ID	State	Input/User actions/ Conditions	Expected Output
1	Alexa sensed	Check if device is connected to Wifi	Connected and response given
2	Alexa sensed	Device is disconnected	Response not received, Alexa not sensed
Scenario 2		Recognizing wake words	
1	Recognizing wake word (Hello Alexa)	Check if Alexa wakes up with wake word like "Alexa"	Listening "Blue bar flashing at bottom of the screen"
2	Recognizing wake word (Hello Alexa)	Check if Alexa wakes up with wake word like "Alexa"	Does not wake up
Scenario 3		Smart Light	
1	Paired with smart light	Check if smart light is connected and sync'd	Connected and smart light listed
2	Paired with smart light	Smart light is not connected	Nothing listed
Scneario 4		Smart light operations	
1	State of light (on/off)	Smart light turned on	On
2	State of light (on/off)	Smart light turned off	Off
Scenario 5		Find my phone	
1	Phone configured	Check if phone is configured for this feature	Phone should be listed under Alexa App
2	Phone configured	Find my phone feature is not configured	Phone should be listed under Alexa App
3	Phone configured	Call my phone	Ringing
Scenario 6		Send an SMS	
1	Access to contacts	Permission given to Alexa	Contacts visible
2	Access to contacts	No permission to Alexa	Contacts not visible
3	Access to message app	Permission given to Alexa	Messages App seen
4	Access to message app	No permission to Alexa	Messages App not visible
5	Send SMS	Has access to SMS App to send message	SMS sent
6	Send SMS	Has access to SMS app but couldn't send the message	SMS not sent
Scenario 7		Play Music	
1	Song recognized	Song name recognized	Song selected and seen in app
2	Song recognized	Song name not recognized	No songs selected
3	Play music	Selected song played	Music played
4	Play music	Slected song but couldn't be played	Music couldn't be played

3.3 Method #2 - Decision Table Testing Method

3.3.1 Test Coverage

The decision table value method assures the decision table test coverage.

For each rule in the table, there is a test case derived.

3.3.2 Creating Decision table

1. Step One – List All Stub Conditions

In this we take three inputs, and from those inputs we perform combination of tests to test the Alexa for its AI features.

2. Step Two – Calculate the Number of Possible Combinations (Rules)

So, in our table we have 3 condition stubs and we are developing a limited entry decision table, so we use the following formula:

Number of Rules = 2 (power) Number of Condition stubs, So therefore

Number of Rules = 2^3 = 8

So, we have 8 possible combinations in our decision table.

3.3.3 Categories chosen for Decision table

For our testing purpose we are using 3 important categories that affect the AI features the most. the categories we have chosen are as follows:

Decision table:

Decision tables, especially when coupled with AI models like data model or knowledge model would allow users to work from the same information, the decision tables themselves. Tools to render any programming loop statements from conventional programming languages into decision tables is easy and can be used for debugging. Decision tables have proven to be easier to understand

and review than code and have been used extensively and successfully to produce specifications for complex systems. With our AI feature set model, identifying input conditions was difficult. Hence, we chose limited input decision table which helped us in identifying rules and actions for the three input features we chose.

In AI testing, non-parametric methods in statistics is used and helpful for the end user. These methods are also helpful to analyze the decision tables.

1	Invoking the intent	
	With Intent	Y
	Without Intent	N
2	Invoking with questions and requests	
	With question and request	Y
	Without questions and request	N
3	Intent responses	
	One-shot	Y

	Multiple phrases	N

3.3.4 Decision table for “Alexa” application

Conditions	R1	R2	R3	R4	R5	R6	R7	R8
Invoke the skill without specifying an intent, for example: <Hello Alexa>. Respond to the prompt provided by the skill and verify that you get a correct response.	T	T	T	T	F	F	F	F
Test a variety of intents – both those that ask questions and those complete the user's request.	T	T	F	F	T	T	F	F
Test the skill's intent responses using different combinations of slot values. You can use one of the one-shot phrases for starting the skill.	T	F	T	F	T	F	T	F

Actions								
Correct Intent	T	X	F	X	T	X	T	T
Incorrect Intent	F	F	T	F	T	X	X	X

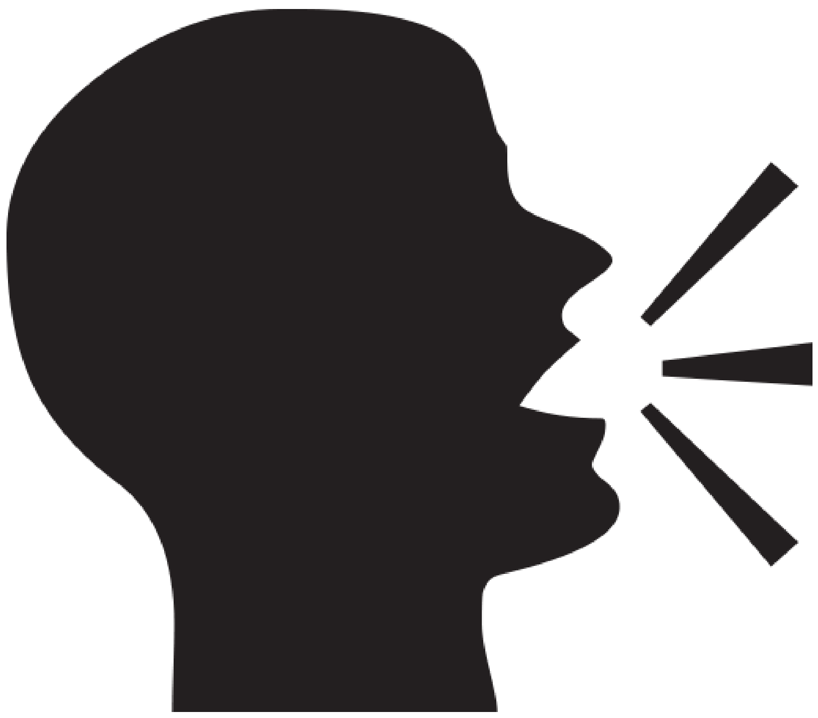
3.3.5 Test cases from decision table

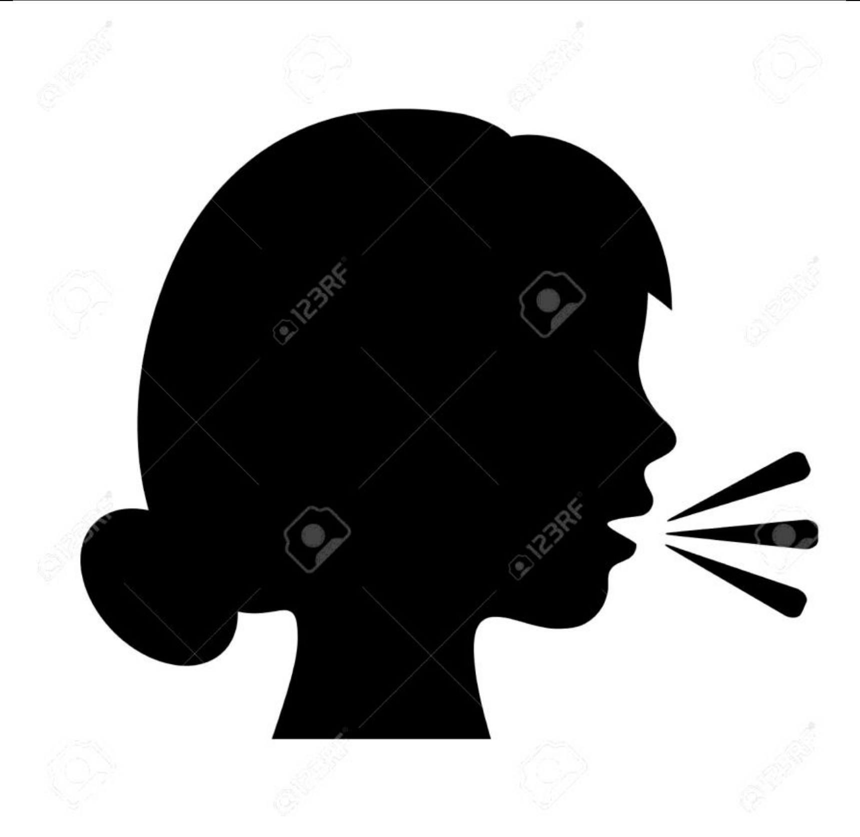
Open the Alexa app, click **Skills** and then scroll or search to find your skill. Review the skill's detail page. Inspect all example phrases listed in the skill's detail card.		R1	• At least one example phrase has been provided. • The first phrase clearly indicates how to begin using the skill, and includes both the wake word and invocation name.

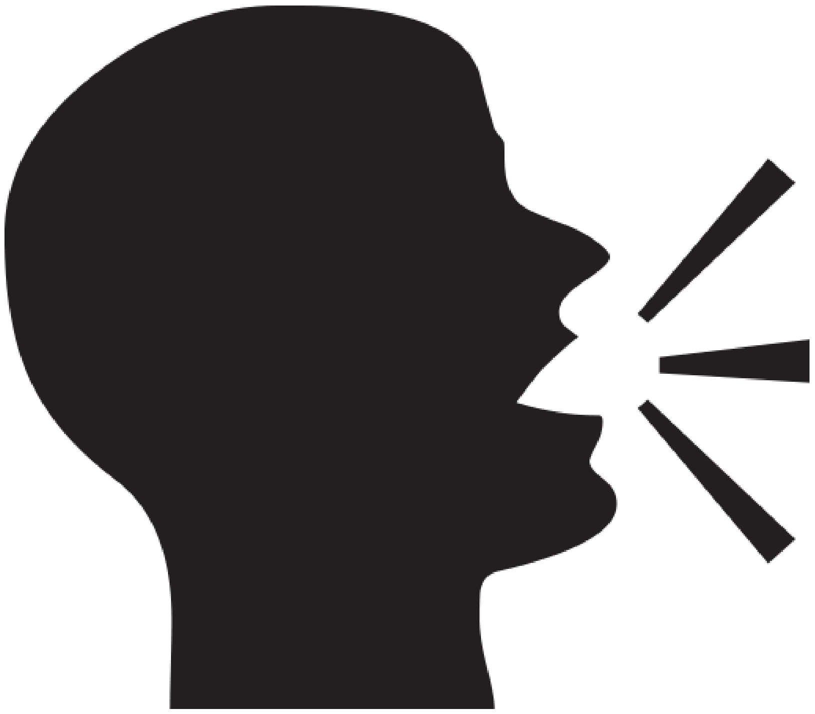
Open the Alexa app, click **Skills** and then scroll to your skill. Review the **short description**shown in the list of skills		R2	• The short description describes the skill's core functionality. • The short description is written in the same language used by the Alexa account. For instance, when using an account configured with German, the skill short description is displayed in German.
While listening to the audio playback, invoke the following built-in intents: • • AMAZON.PauseIntent (say "pause") • • AMAZON.ResumeIntent (say "resume")		R3	• The "pause" utterance stops the audio playback. • The "resume" utterance resumes the audio playback
While listening to the audio playback, use a remote control or other hardware device to pause and resume the audio.		R4	• Pressing a pause button while audio is playing stops the playback.

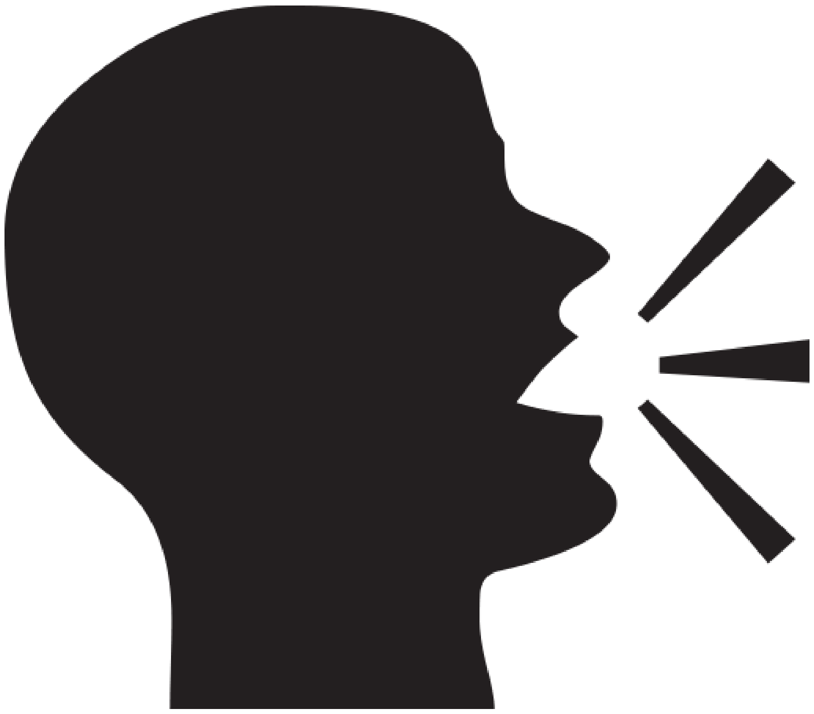
While listening to the audio playback, invoke the following built-in intents: • • AMAZON.CancelIntent (say "cancel") • AMAZON.LoopOffIntent (say "loop off") • AMAZON.LoopOnIntent (say "loop on") • AMAZON.NextIntent (say "next") • AMAZON.PreviousIntent (say "previous") • AMAZON.RepeatIntent (say "repeat that") • AMAZON.ShuffleOffIntent(say "shuffle off") • AMAZON.ShuffleOnIntent (say "shuffle") • AMAZON.StartOverIntent (say "start over") • AMAZON.StopIntent (say "stop")		R5	Each command either does something that makes sense for your skill (such as "next" advancing to the next track in a playlist) or is handled gracefully. No unexpected errors occur. See Configuring Your Skill for the AudioPlayer Directives.

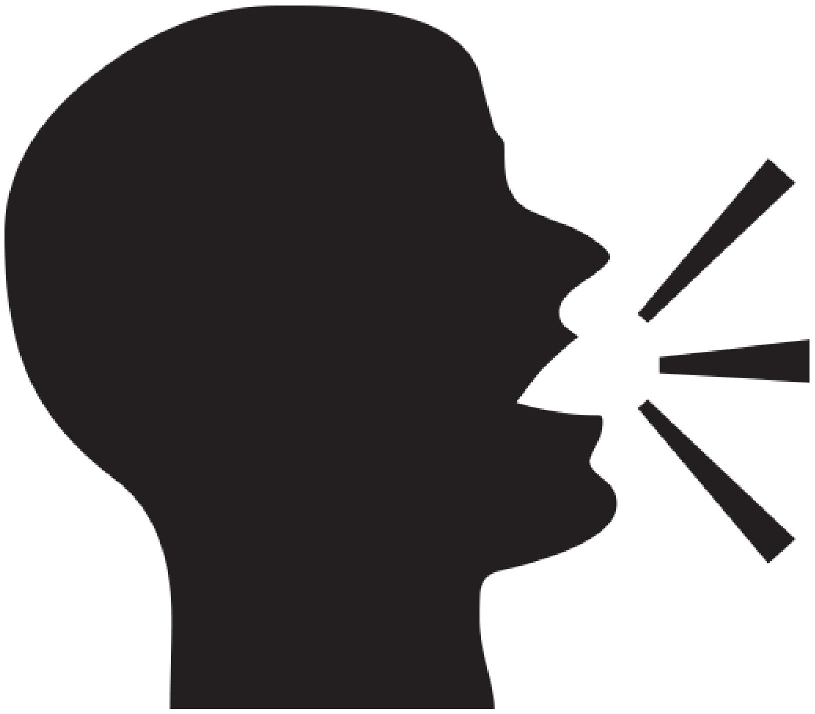
Chapter 4 Conventional testing results

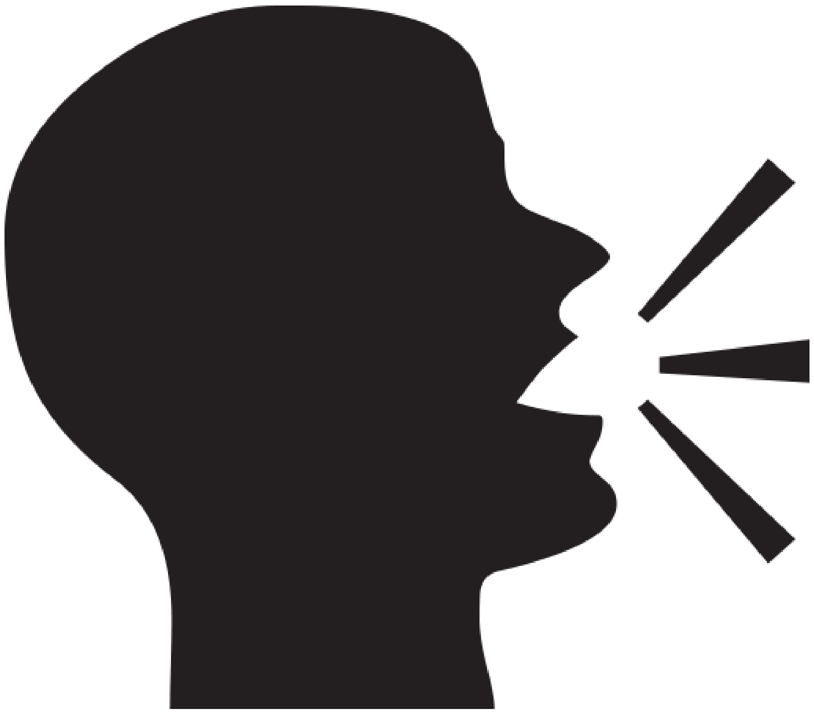
Test Case	 Pronunciation by man
Performed by	Anup
Execution Date	2/11/2018
Test type	Scenario based testing
Pre-requirements	Alexa App on android phone
Expected result	Alexa should recognize voice by the wake word
Actual result	Alexa recognized the voice
Test case status	Pass

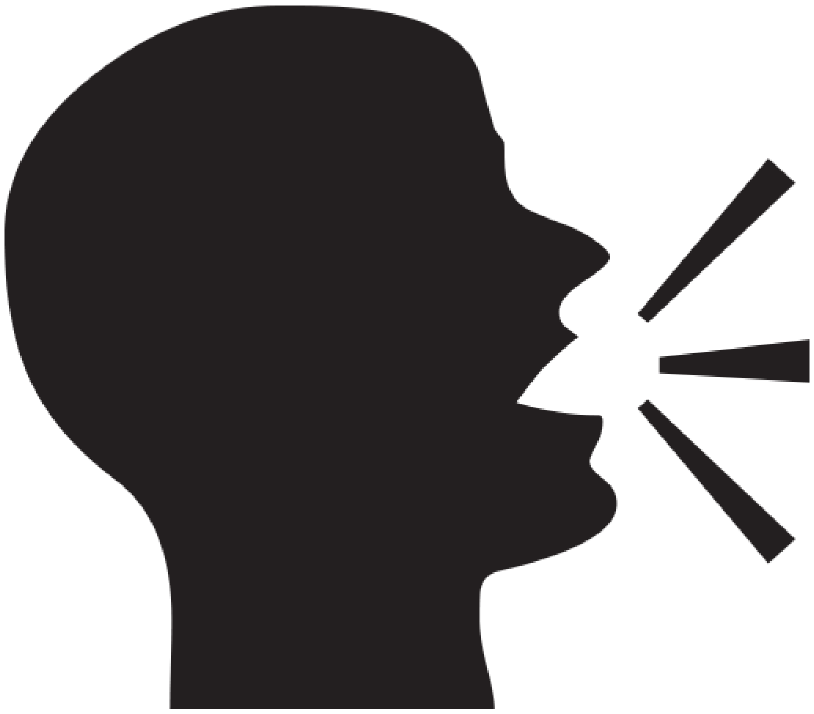
Test Case	 Pronunciation by women
Performed by	Rucha
Execution Date	2/11/2018
Test type	Scenario based testing
Pre-requirements	Alexa App on android phone
Expected result	Alexa should recognize voice by the wake word
Actual result	Alexa recognized the voice
Test case status	Pass

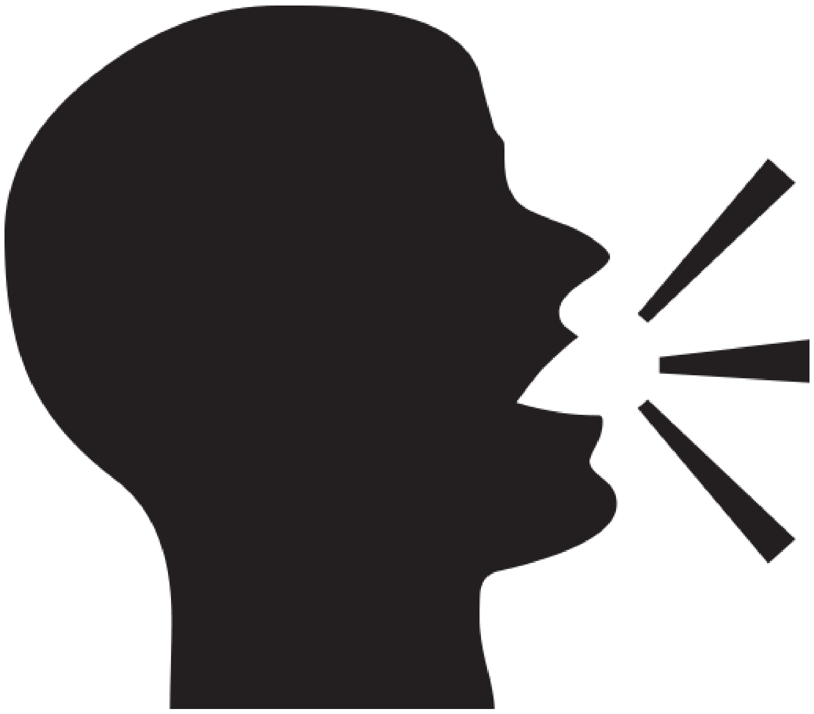
Test Case	 Pronunciation Accent
Performed by	Anup
Execution Date	2/11/2018
Test type	Scenario based testing
Pre-requirements	Alexa App on android phone
Expected result	Alexa should recognize Accent
Actual result	Alexa recognized the Accent
Test case status	Pass

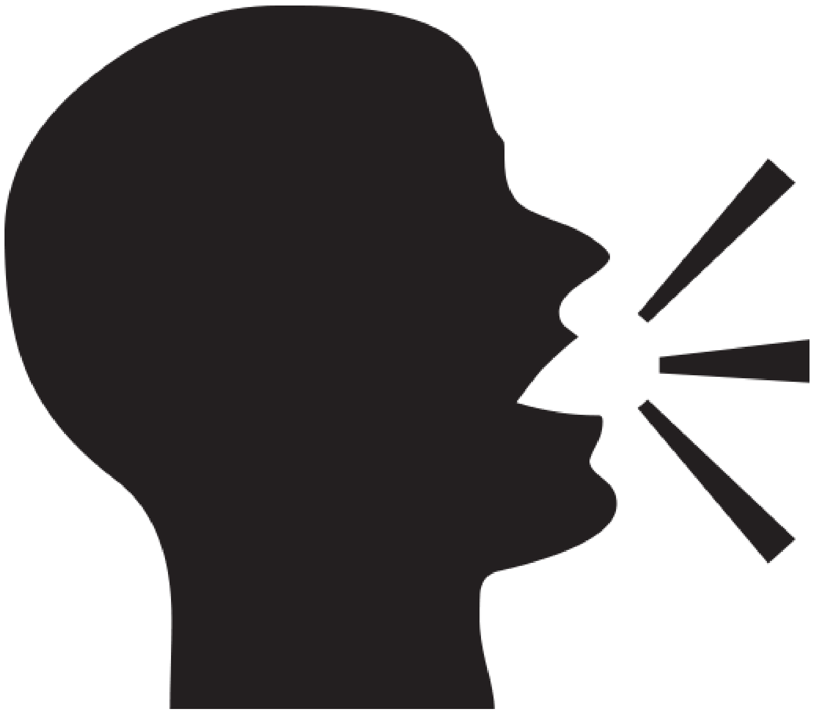
Test Case	 Pronunciation Accent
Performed by	Guoang Kim
Execution Date	2/11/2018
Test type	Scenario based testing
Pre-requirements	Alexa App on android phone
Expected result	Alexa should recognize Accent
Actual result	Alexa failed to recognize the Accent
Test case status	Fail

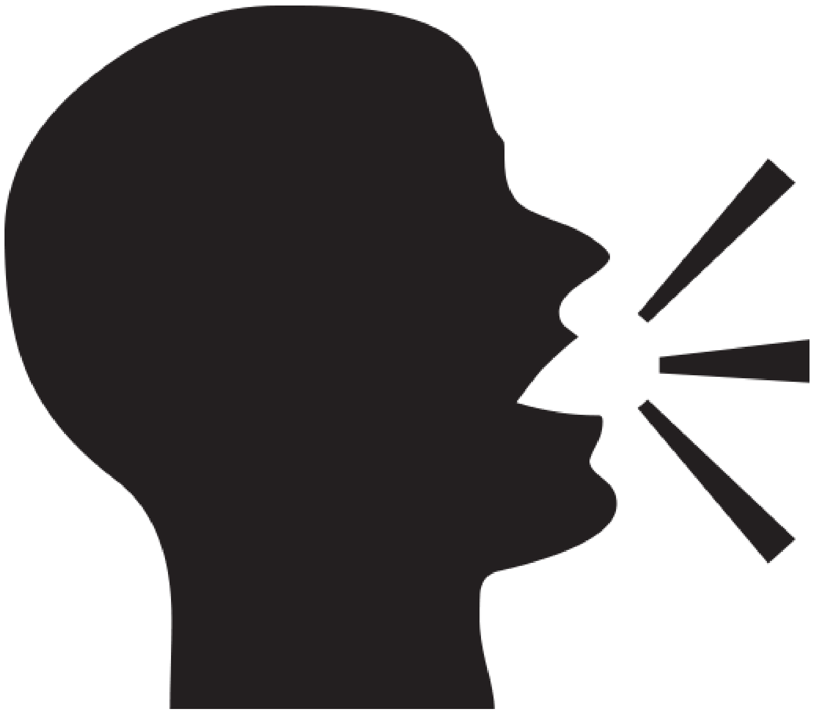
Test Case	 Pronunciation Speed 145 wpm
Performed by	Google Voice(computer)
Execution Date	2/11/2018
Test type	Scenario based testing
Pre-requirements	Alexa App on android phone
Expected result	Alexa should recognize the sentence if it had intent
Actual result	Alexa understood the sentence and responded
Test case status	Pass

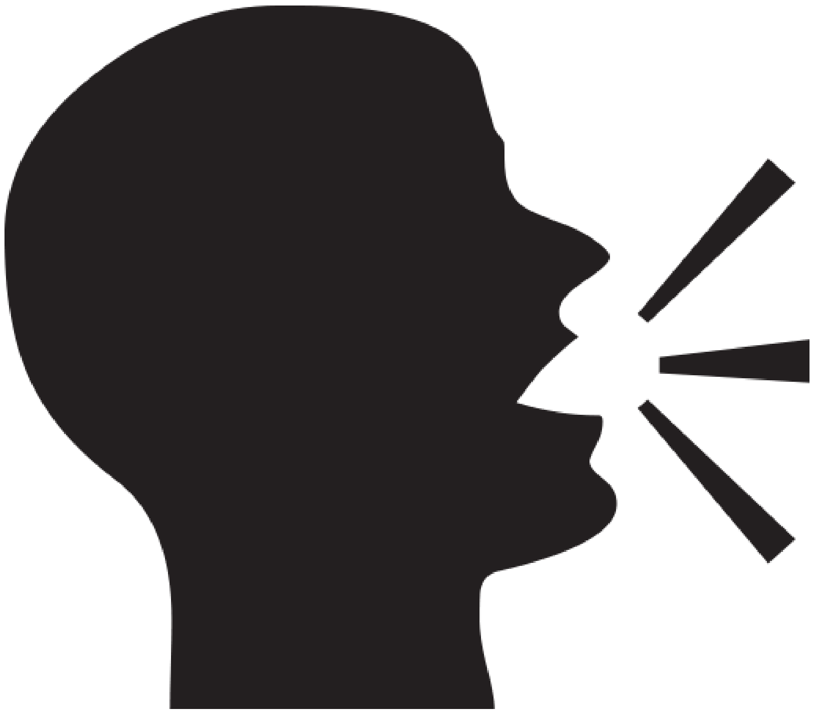
Test Case	 Pronunciation Speed 200 wpm
Performed by	Google Voice(computer)
Execution Date	2/11/2018
Test type	Scenario based testing
Pre-requirements	Alexa App on android phone
Expected result	Alexa should recognize the sentence if it had intent
Actual result	Alexa couldn't the to recognize the intent when spoken fast
Test case status	Fail

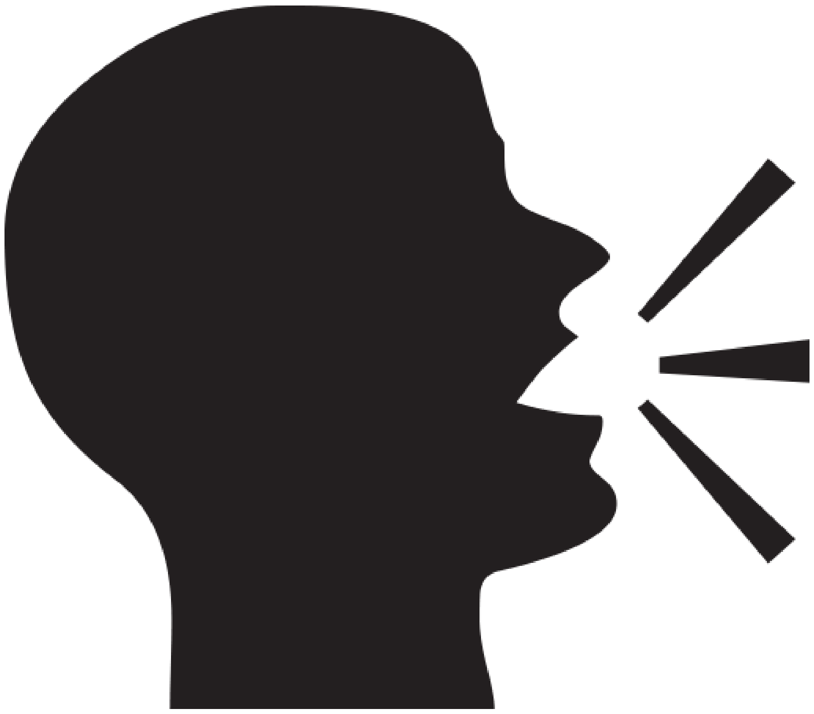
Test Case	 Pronunciation Distance Until 2 meters
Performed by	Anup
Execution Date	2/11/2018
Test type	Scenario based testing
Pre-requirements	Alexa App on android phone
Expected result	Alexa should recognize the sentence if it had intent from 2 meters
Actual result	Alexa recognized the intent when from 2 meters
Test case status	Pass

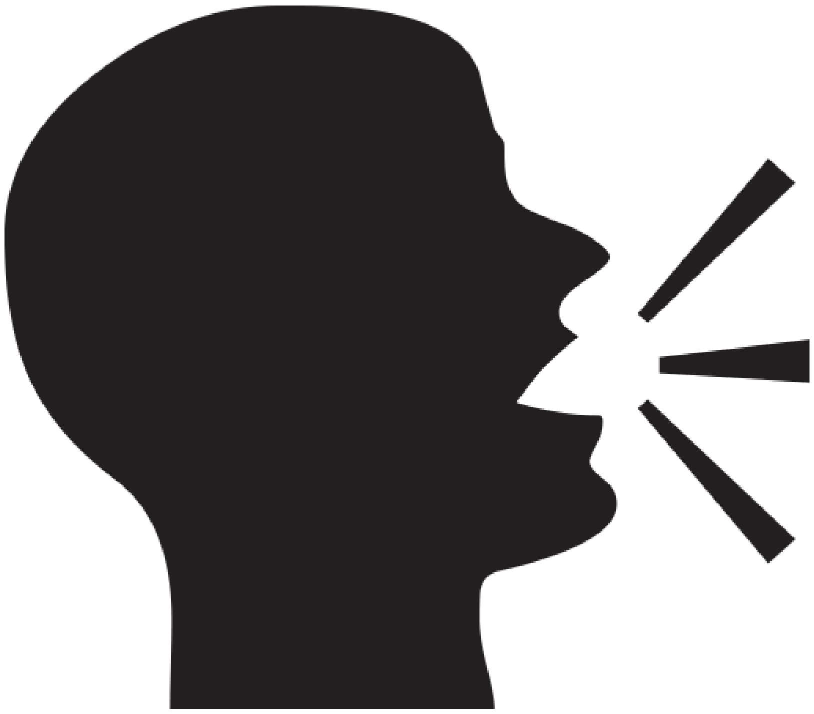
Test Case	 Pronunciation Distance More than 2 meters
Performed by	Anup
Execution Date	2/11/2018
Test type	Scenario based testing
Pre-requirements	Alexa App on android phone
Expected result	Alexa should recognize the sentence if it had intent distance more than 2 meters
Actual result	Alexa couldn’t recognize the intent when distance is more than 2 meters
Test case status	Fail

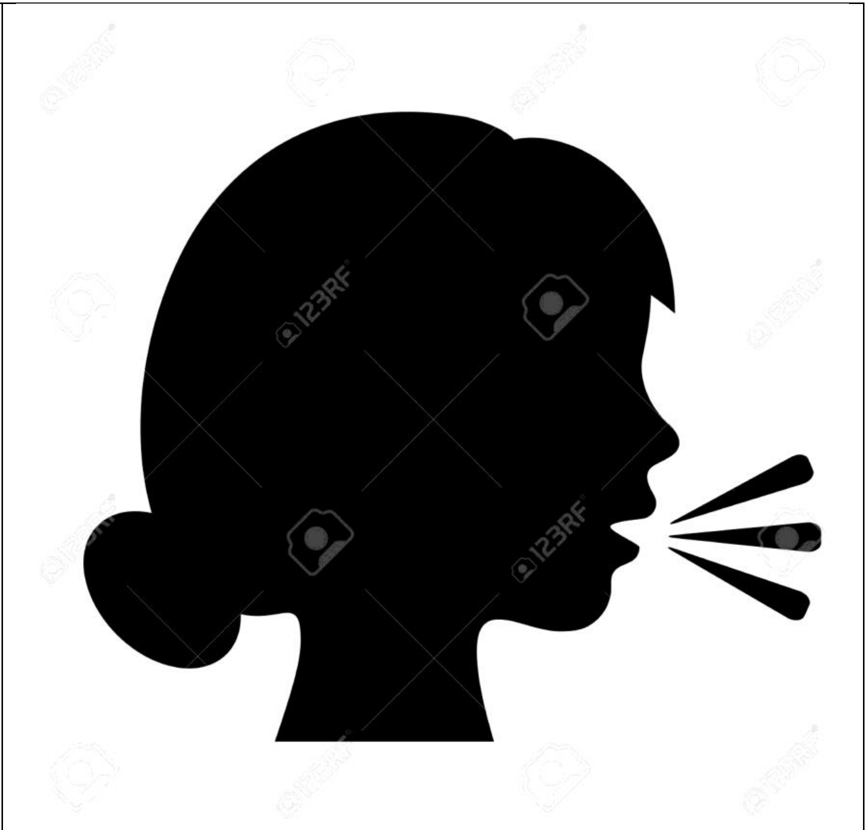
Test Case	 Language Content- Language- English
Performed by	Anup
Execution Date	2/11/2018
Test type	Scenario based testing
Pre-requirements	Alexa App on android phone
Expected result	Alexa should recognize the language English
Actual result	Alexa recognized the language English
Test case status	Pass

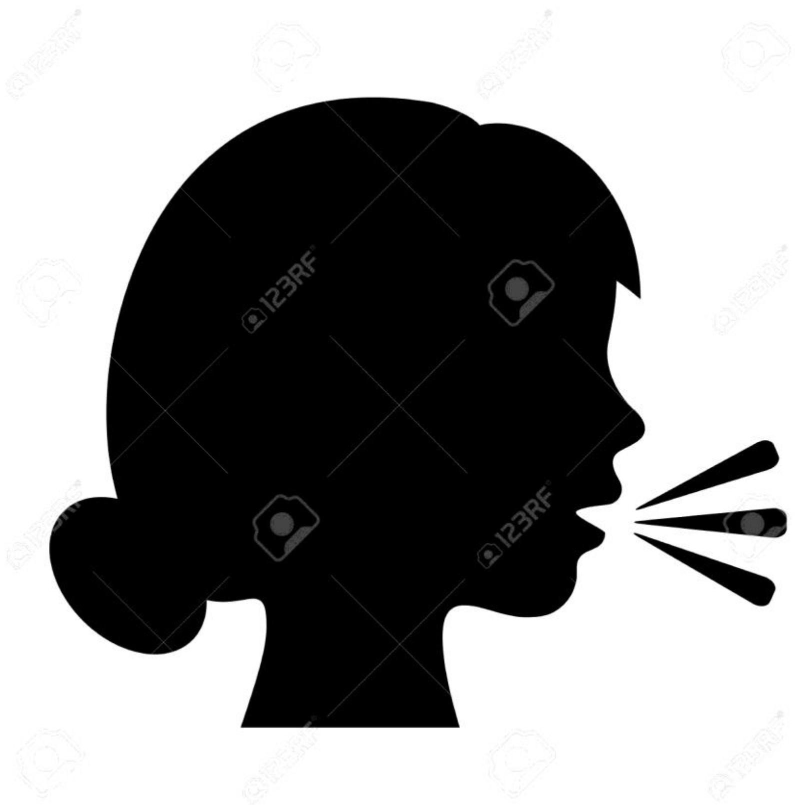
Test Case	 Language Content- Long Sentences
Performed by	Anup
Execution Date	2/11/2018
Test type	Scenario based testing
Pre-requirements	Alexa App on android phone
Expected result	Alexa should recognize long sentences
Actual result	Alexa failed to recognize long sentence when there is no intent
Test case status	fail

Test Case	 Language Content- Long Sentences
Performed by	Anup
Execution Date	2/11/2018
Test type	Scenario based testing
Pre-requirements	Alexa App on android phone
Expected result	Alexa should recognize long sentences with intent
Actual result	Alexa recognized long sentence when there is intent
Test case status	Pass

Test Case	 Language Content- Chinese and English mix
Performed by	Anup
Execution Date	2/11/2018
Test type	Scenario based testing
Pre-requirements	Alexa App on android phone
Expected result	Alexa should recognize long sentences with intent
Actual result	Alexa recognized long sentence when there is intent
Test case status	Pass

Test Case	 Language Content- English Chinese mix
Performed by	Guaong Kong
Execution Date	2/11/2018
Test type	Scenario based testing
Pre-requirements	Alexa App on android phone
Expected result	Alexa should recognize language content
Actual result	Alexa didn’t recognize language mixed
Test case status	Fail

Test Case	 Language Content- Special name
Performed by	Rucha
Execution Date	2/11/2018
Test type	Scenario based testing
Pre-requirements	Alexa App on android phone
Expected result	Alexa shouldn't recognize special names
Actual result	Alexa didn't recognize special names
Test case status	Pass

Test Case	 Noise Environment- Human Voice
Performed by	Rucha
Execution Date	2/11/2018
Test type	Scenario based testing
Pre-requirements	Alexa App on android phone
Expected result	Alexa should recognize Human Voice
Actual result	Alexa recognized human voice in all test cases
Test case status	Pass

Test Case	Language Content-Traffic
Performed by	Anup
Execution Date	2/11/2018
Test type	Scenario based testing
Pre-requirements	Alexa App on android phone
Expected result	Alexa should recognize intent with background noise of traffic
Actual result	Alexa didn’t recognize intent in 2 cases with background noise
Test case status	Passed 18 test cases and failed in 2.

Test Case	Language Content- Air Conditioning
Performed by	Anup
Execution Date	2/11/2018
Test type	Scenario based testing
Pre-requirements	Alexa App on android phone
Expected result	Alexa should recognize intent with background AC
Actual result	Alexa recognized the intent with background AC
Test case status	Passed

Test Case	Language Content- TV
Performed by	Anup
Execution Date	2/11/2018
Test type	Scenario based testing
Pre-requirements	Alexa App on android phone
Expected result	Alexa should recognize intent with background TV voice
Actual result	Alexa recognized the intent with background TV in 19 cases
Test case status	Passed 19 and failed 1

Test Case	Language Content- Music
Performed by	Anup
Execution Date	2/11/2018
Test type	Scenario based testing
Pre-requirements	Alexa App on android phone
Expected result	Alexa should recognize intent with background music
Actual result	Alexa recognized the intent with background music in 19 cases
Test case status	Passed 19 and failed 1

Test Case	Language Content- Communication Equipment - Laptop
Performed by	Anup
Execution Date	2/11/2018
Test type	Scenario based testing
Pre-requirements	Alexa App on android phone
Expected result	Alexa should recognize intent from communication device
Actual result	Alexa recognized the intent from a communication device
Test case status	Passed 20 test cases

Test Case	Application Scenarios-Disconnected from network
Performed by	Anup
Execution Date	2/11/2018
Test type	Scenario based testing
Pre-requirements	Alexa App on android phone
Expected result	Cannot connect with alexa
Actual result	Couldn’t connect with alexa as it is a cloud based application
Test case status	Pass

Test Case	Application Scenarios- Microphone off
Performed by	Anup
Execution Date	2/11/2018
Test type	Scenario based testing
Pre-requirements	Alexa App on android phone
Expected result	Cannot connect with alexa
Actual result	Couldn’t connect with alexa as it is a cloud based application
Test case status	Pass

Chapter 5 AI testing results

Our test coverage included the comparison between

1) Conventional testing
2) AI software testing

5.1 Coverage criteria for conventional testing

Since the object under test is a software produce, we performed black box testing for the same and all the identified test cases were executed as planned. Our coverage matrix for conventional testing is around 100%.

5.2 Coverage criteria for AI software testing

For any AI testing, the coverage criteria will have to be considered from:

- Statistical methods
- Performance evaluation

5.2.1 Statistical methods and measures

Any AI systems analyzed, there is no correct/single result which is universal. Consequently, when such a AI system is analyzed against human experts the outcome may come out to be different and hence in such cases the statistical techniques adds value and provides quantitative measures of relative performance.

"Skill" is the important measure in deriving a single statistic and in the performance of a task. The coefficients measuring the correlation between the performance of a knowledge-based system and that of human experts (or the objective facts if these are known) are often used.

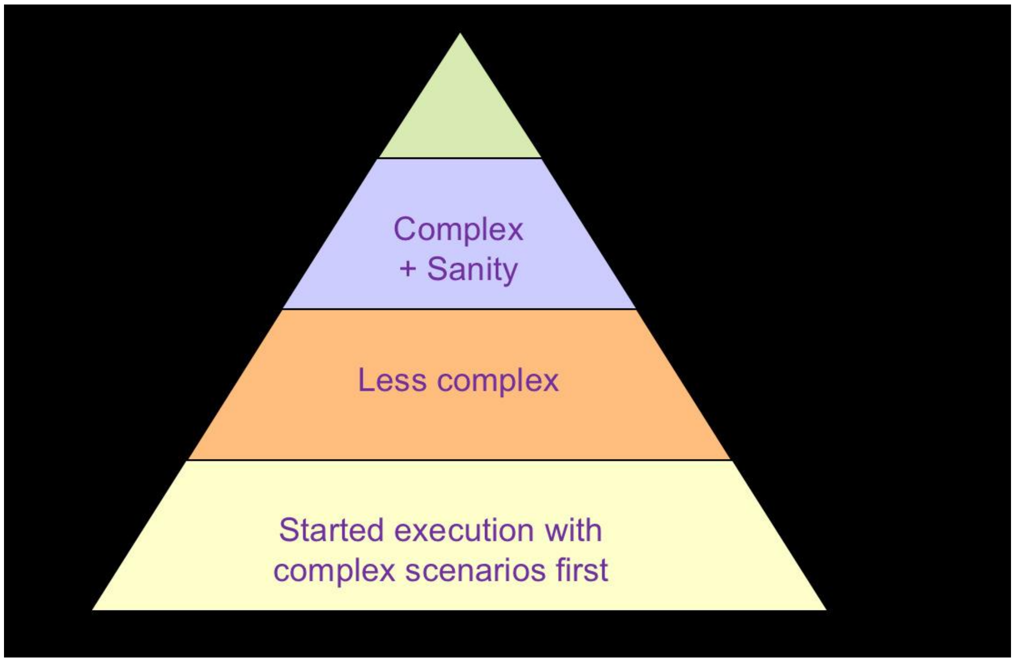
5.2.2 Performance evaluation

Performance evaluation considers the overall behavior of the AI system. It also considers the internal and external factors of any AI model and evaluates from the system behavior perspective. In the AI software testing, this also correlates to the system AI testing

5.3 Quality Factors

There are many influencing factors which will help in gauging the quality of a software product. To begin with let us first understand the AI quality pyramid :

This shows that it is always better to tackle the complex scenarios first which will uncover important issues and for which dev teams will have more time to fix. This is assuming the product is almost ready for testing. Once this is done the focus can be on less complex and then again the very high complex with sanity. This would yield a good quality product.



5.3.1 Consideration of test objects in AI solution

Dataset:

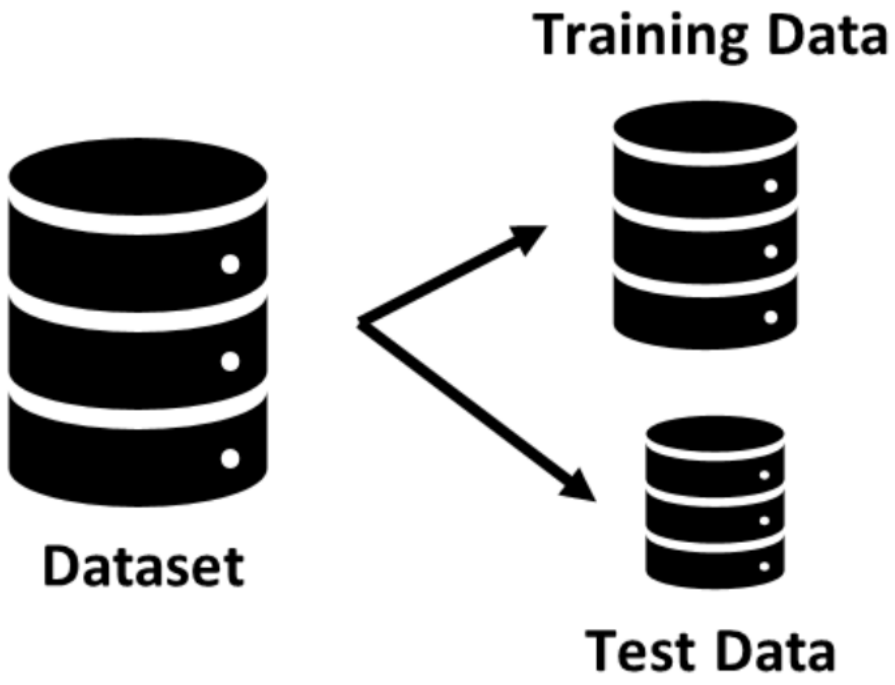
The dataset comprises of all the available data that the AI software can consume. This could be some facts (or historical data) or the similarity of available data.

Training data:

This is the subset of dataset. This is the data which we use to train the model and this data should ideally be available with or without randomness.

Test data:

In some instances, the actual data or the real time data might not be available for testing purpose. That is when the test data comes into picture. This is another subset of dataset. This data tells whether the model works as it is intended.



5.3.2 AI Test Models for "Alexa" Mobile App

Any AI software can be classified under different models. Out of the many available models, AI software typically is modelled based on

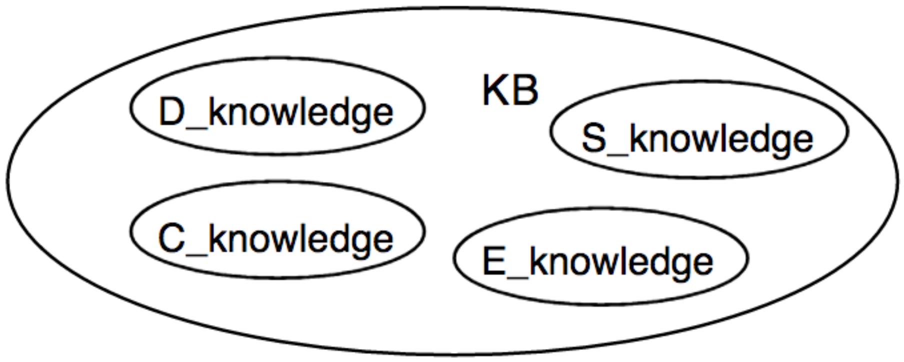
- Data Model
- Knowledge Model
- Feature set Model

5.3.2.1 Data Model

Data Modelling is used to represent the real-world concepts which are important in constructing the information which organizes uses. This modelling is more human friendly. This is getting evolved into Object oriented data modelling for AI software. The object-oriented data modelling is based on object oriented concepts of Aggregation, Association, Generalization and Inheritance.

5.3.2.2 Knowledge Model

Knowledge Model constitutes a cumulation of multiple aspects of a system. This involves a knowledge base of large, complex and aggregated objects. This is represented in the below diagram. D_knowledge is domain knowledge, it refers to domain knowledge, facts and artifacts. C_knowledge is control knowledge. It describes the system's problem-solving strategies and its functional model, and is more or less domain independent. E_knowledge denotes explanatory knowledge. It defines the contents of explanations and justifications of the system's reasoning process, as well as the way they are generated. System knowledge (the S_knowledge part) describes the contents and structure of the knowledge base, as well as pointers to some useful programs, which should be "at hand" during the knowledge base building process, since they can provide valuable information.



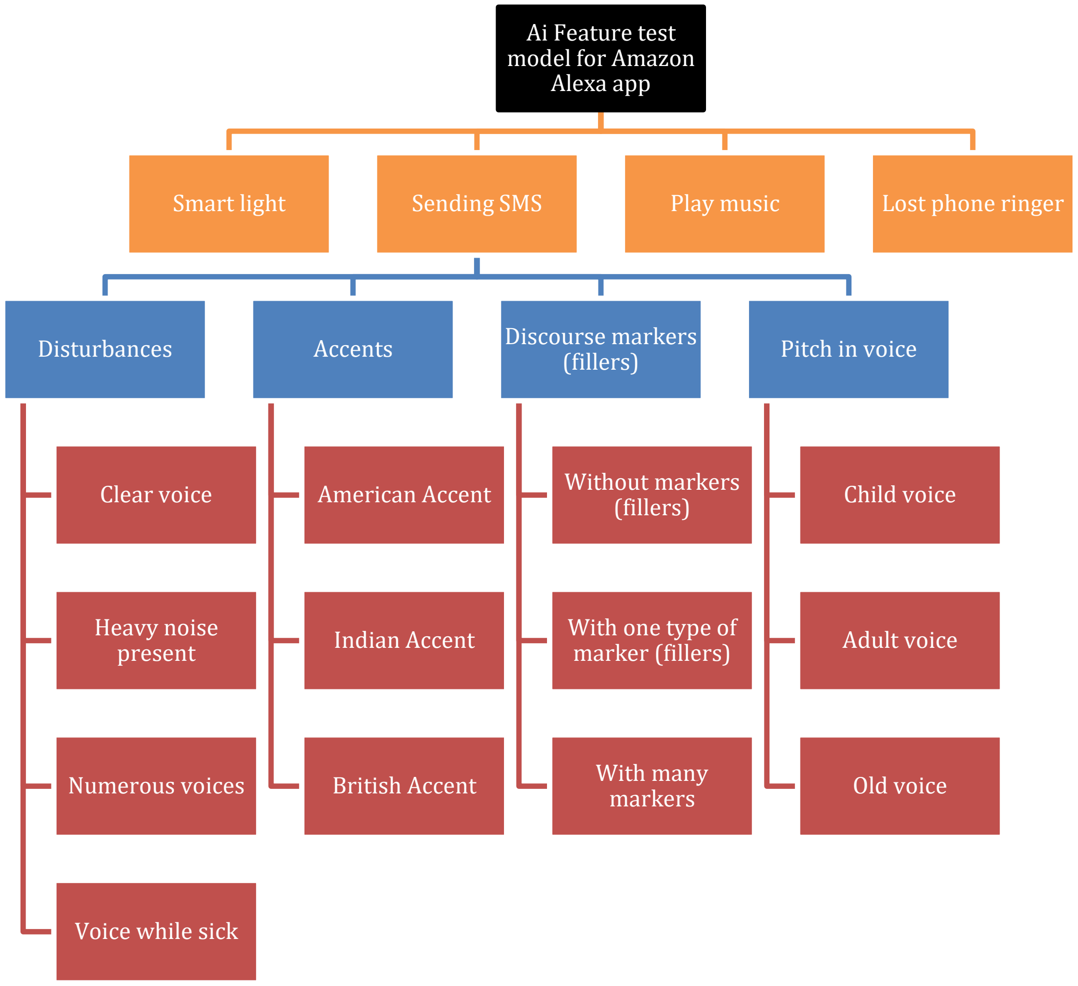
5.3.2.3 AI feature set Model

This model explains the AI features sets which are rendered in the AI software product. This is a subset of all the available feature sets.

For an Ai test model of the Amazon Alexa mobile application we have considered four features that is Toggling a Smart Light, sending an SMS through the same mobile, play music from an application and ringing a lost phone. Testing of these features have provided us with the quality of information of the application's features but not any of the AI methodologies used in the application. To test the same, we started to understand the type of AI that the application uses and the inputs it requires.

Amazon Alexa Mobile Application Is a part of Amazon Alexa which is digital personal assistant which is hosted on the cloud. The Amazon Alexa uses conversational AI, which is focused on making computer system communicate with humans in a natural way, solve problems and get smarter over time.

The major input for a conversational AI system is voice. So, we have focused our testing on the Voice input to the Amazon Alexa Mobile Application to test the AI features. The following depicts the overview of the categories that have been selected to test upon.



As mentioned above we have focused on four categories of input voice variation. They are

Disturbance:

Disturbances in the voice comprehends to the additional unwanted sounds in the input from either environment or with the voice itself. We have come up with different methods this can affect the functionalities of the application. They are Clear Voice, Heavy noise present, multiple voices, voice when one is sick.

Accents:

There are different types of accents in every once voice, this is usually dependent on the region of origin or the place of learning the language. We have considered Indian, British and American Accents.

Discourse Markers (fillers):

Even for the most proficient speakers, when they are talking spontaneously will have some gaps between words, to fill these people usually use fillers, words which have no real meaning in the context for example “like, um, AA” and so on. We have considered this for one of the categories. The types of the test performed on this category are without fillers, with one fillers, and with many filters.

Pitch:

The Pitch is the number of vibrations in one’s voice, we have considered Adult, child and old voice for testing with different pitches.

Chapter 6 Test Metrics

We have performed 120 test cases, 30 each based on features that is Smart light, Playing Music, sending an SMS and ringing a Mobile.

Each of the features has been taken up member of the team and tested upon on different categories based on Disturbance while using the app, accent of the person, fillers in the speech and pitch of the voice.

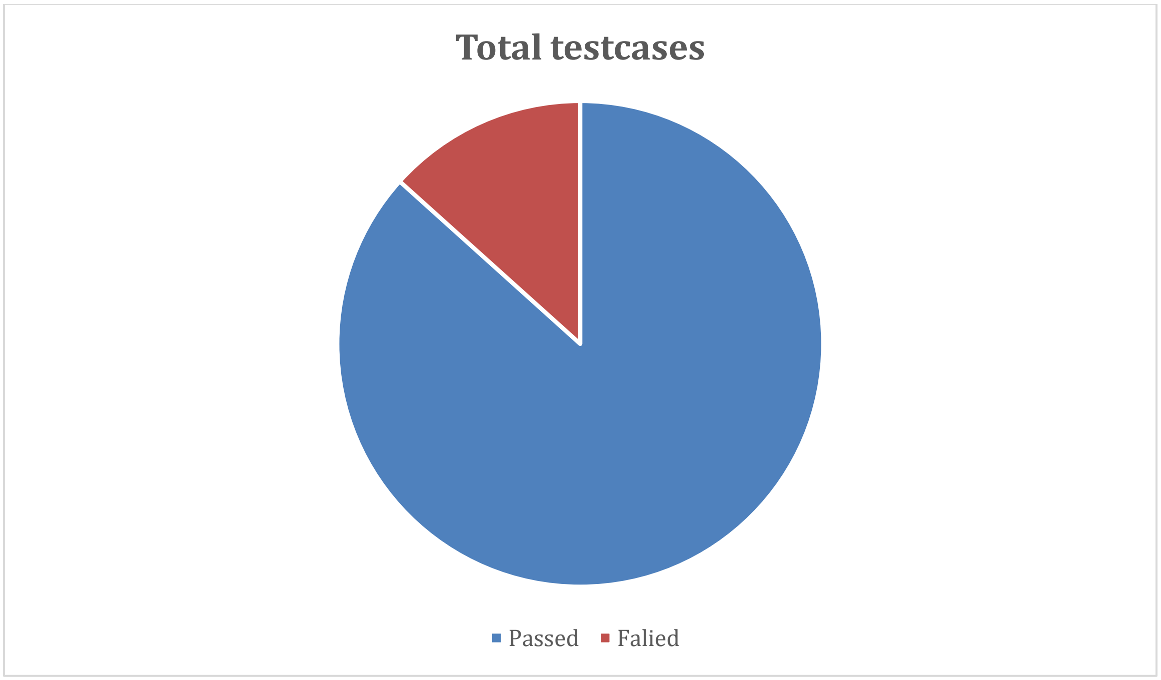
6.1 Total Test Cases

The following are the metrics of the test cases.

Total testcases metrics:

Total Testcase Performed: 120, Passed: 104 & Failed :16

.



6.2 Test Pass/Fail according to feature

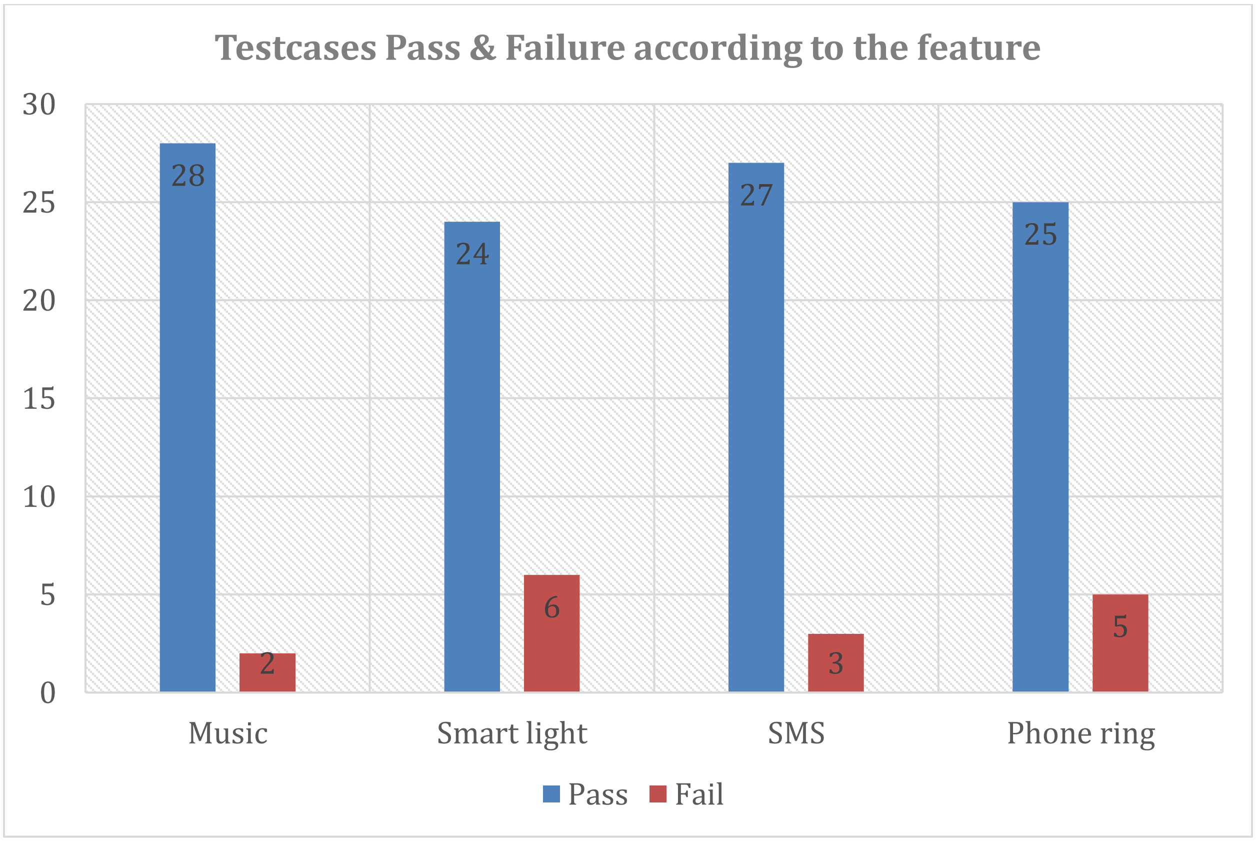
Testcases metrics according to features: total (pass/fail)

Playing Music: 30 (28/2) Test cases

Smart light:30 (24/6) Test cases

Sending an SMS:30 (27/3) Test cases

Mobile Ringer:30 (25/5) Test cases



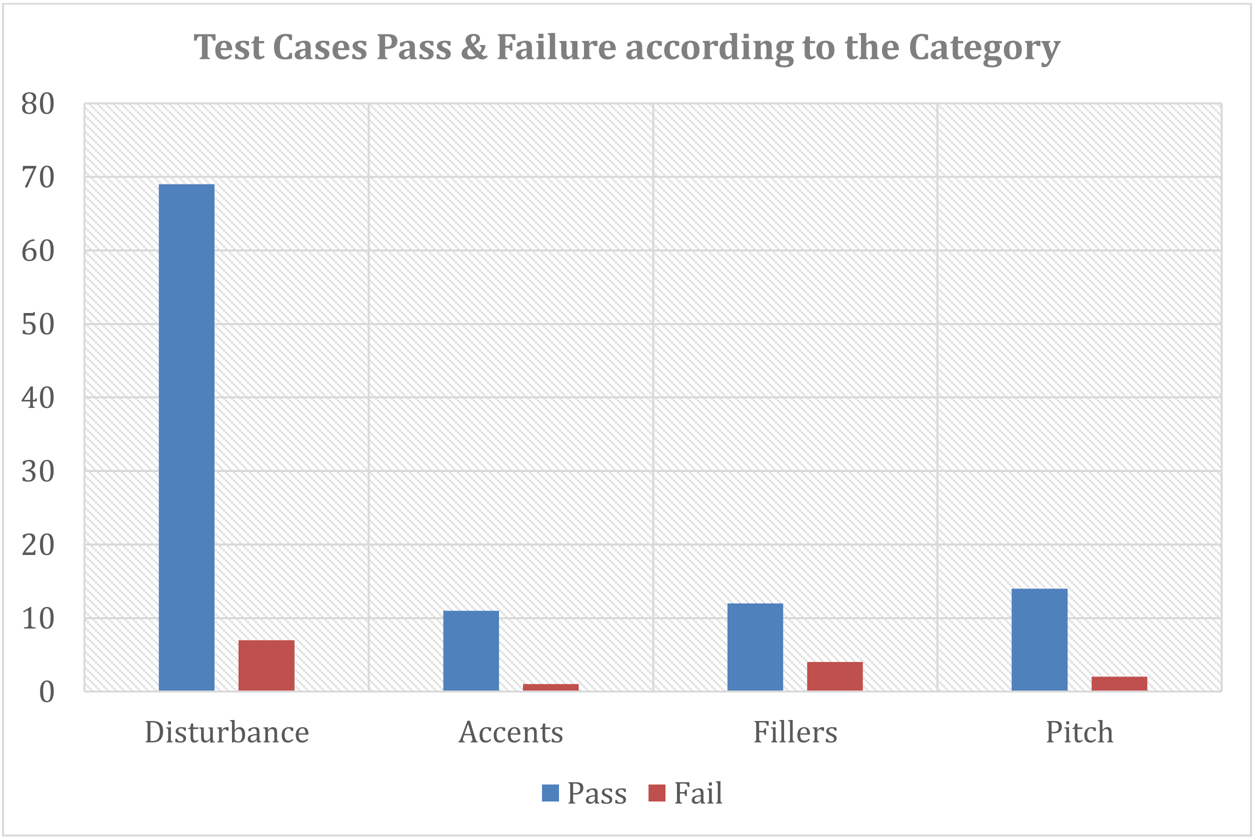
6.3 Test Pass/Fail according to category

Disturbance: 76(69/7) Test cases

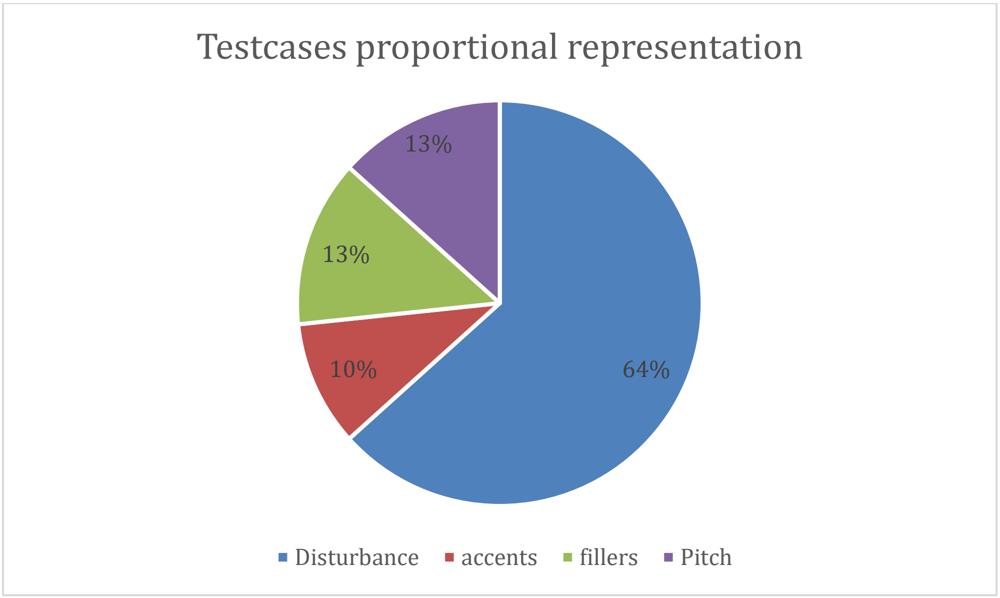
Accents: 12(11/1)Test cases

Fillers:16 (12/4) Test cases

Pitch: 16(14/2) Test cases

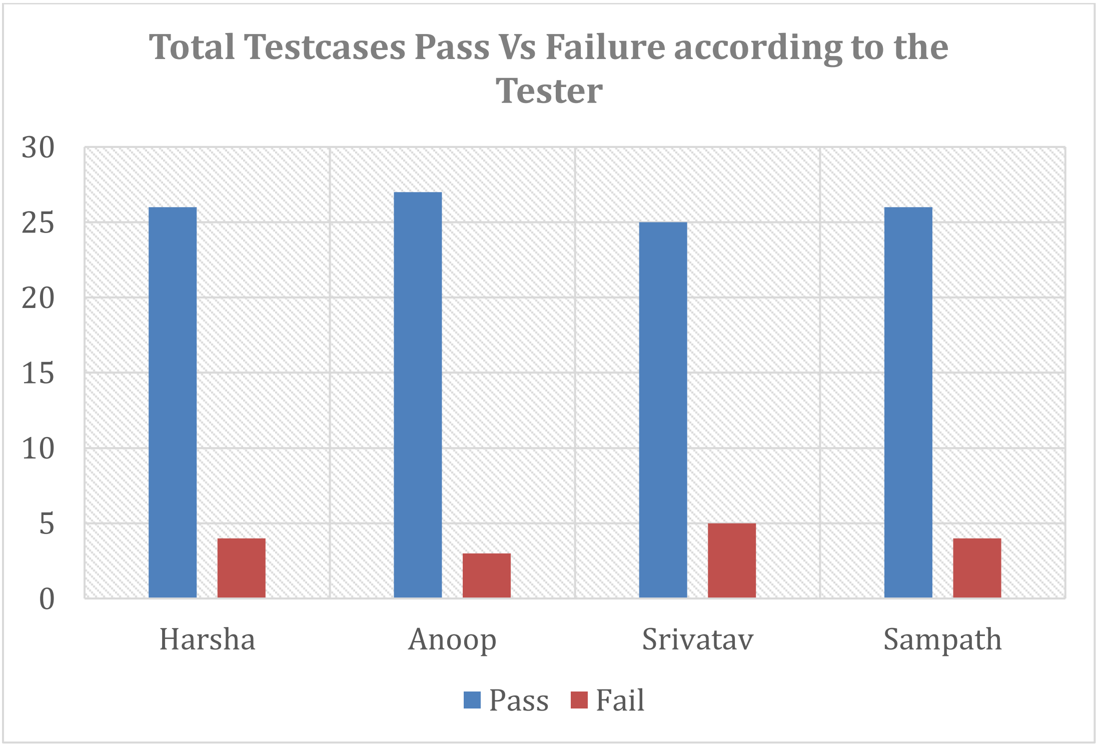


6.4 Proportional representation of the category partition:

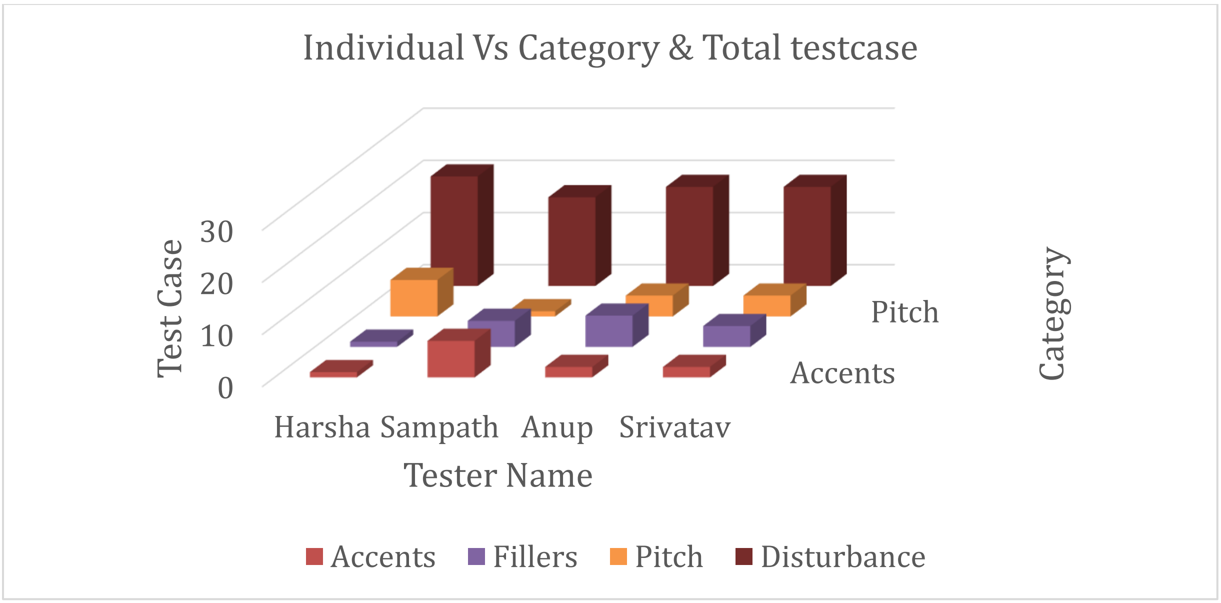


Individual Tester metrics:

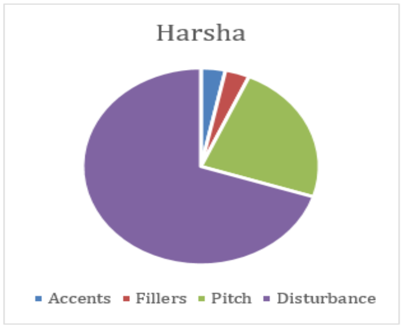
6.5 Metrics Pass & failure of testcases according to the tester.

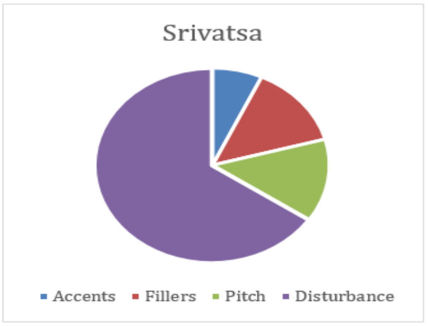


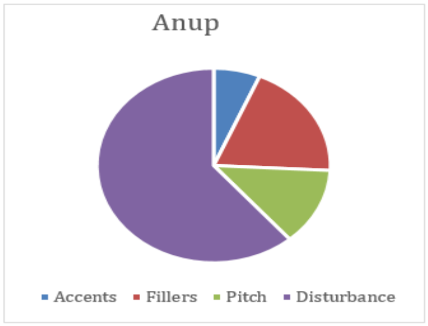
Metrics of total testcases handled by each team member of the team and category of the test case.

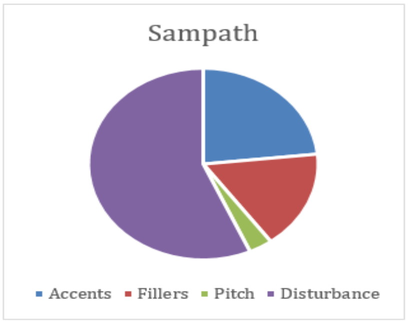


6.6 Individual team member category metrics.





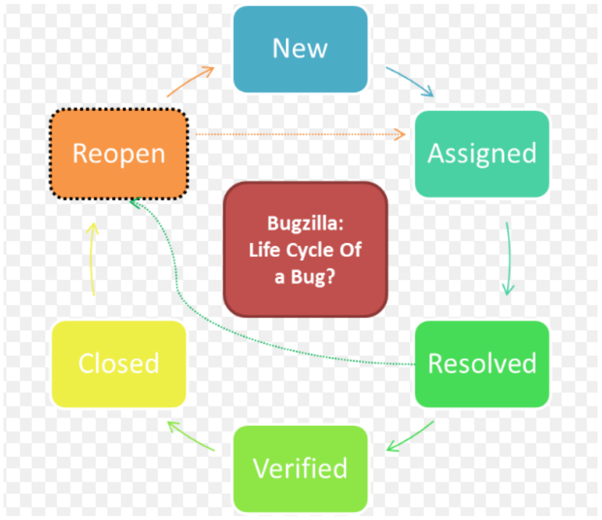




Chapter 7 Bug Report, Experience and Lesson Learned

7.1 Introduction

A software bug is a fault in the piece of code that causes the software to behave in an unexpected way. Bug report is a document or a draft that says, "What is the current behavior of the software system", "What should be the behavior as per the requirements" and "Well written steps to reproduce the test case".



7.2 Issues with bug reporting

There are many issues with bug reporting in general. Some of them are because of the tools that are used to report bugs and some of the issues are because of the forma in which the bugs are reported by the testers.

- There are too many fields to fill in Bugzilla or any bug tracking tool and it is confusing sometimes because of the poor description of what the tool is expecting the tester to input in a field

- Sometimes it is possible for a tester to report what could be the cause of the issue instead of what exactly is the problem and the speculation might go wrong more often than not. So, it is important to focus on facts
- Developers might face difficulty in understanding a bug if a proper format is not followed. As mentioned above, it is important to mention what is the expected behavior instead of just reporting what is the current behavior. You can also attach the screenshots as a proof so that the developer believes you even if he is not able to reproduce the bug in the very first attempt.

7.3 Specific concerns for Bug reporting for testing conversational ai

Other than the general problems discussed above, AI systems pose more issues with reporting bugs. Our AI system is Alexa, a voice recognition system which performs several tasks based up on the task issued to the system by human voice. The below mentioned are some of the problems we faced in our project

1) It is very challenging to reproduce the bug consistently as it is dependent on many factors. There are large number of factors that affect the voice recognition and it is close to impossible to try and test all these possibilities.
2) It is hard to reproduce the same accent to replicate the bug.
3) It is difficult to reproduce the error as the environment that the developers get is not exactly same as the one that testers get.
4) It is very difficult to reproduce the accent with which we have tested the system to produce bug.

7.4 Bug reporting tool – Bugzilla

Bugzilla is the tool that we opted for reporting the bugs we identified.

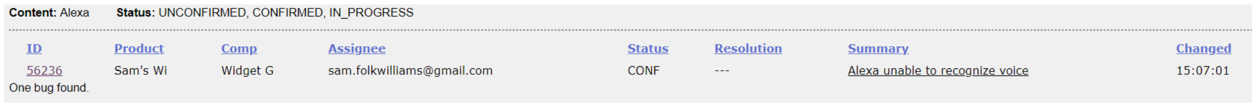
We choose Bugzilla because

- It is widely used and opensource bug tracker
- It's user interface makes it easy to use without any technical knowledge
- It is very easy to keep track of bugs in Bugzilla
- The documentation is automated in Bugzilla

7.5 Bug reported in Bugzilla

We have reported the bugs we identified in Bugzilla to the developers of Alexa application. The bug that we identified is that if a person talks with music in the background, Alexa does not understand the intent some of the times. Below attached is the pdf of the bug we reported in Bugzilla

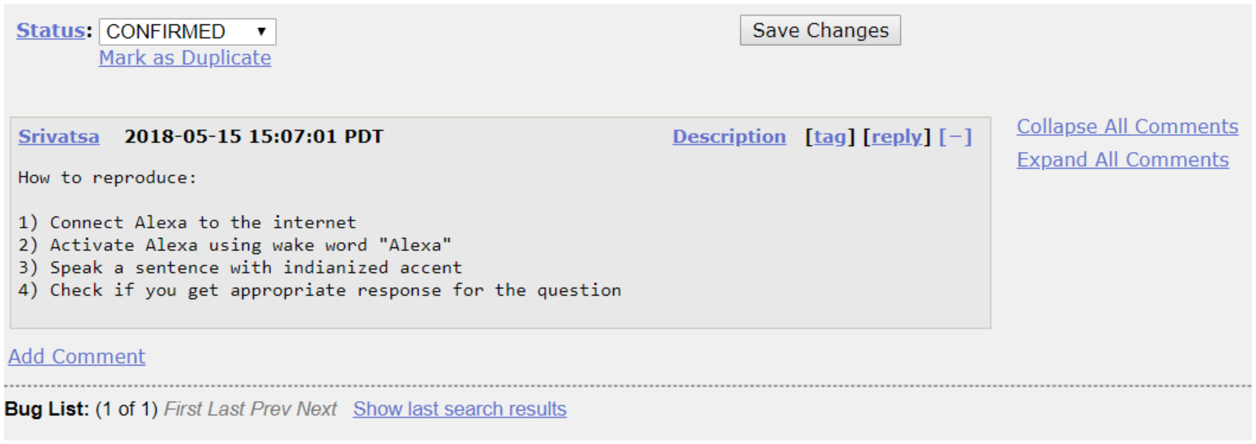
BUG ID – 56236



Content: Alexa **Status:** UNCONFIRMED, CONFIRMED, IN_PROGRESS

ID	Product	Comp	Assignee	Status	Resolution	Summary	Changed
56236	Sam's Wi	Widget G	sam.folkwilliams@gmail.com	CONF	---	Alexa unable to recognize voice	15:07:01

One bug found.



Status: CONFIRMED

Srivatsa **2018-05-15 15:07:01 PDT**

```
How to reproduce:

1) Connect Alexa to the internet
2) Activate Alexa using wake word "Alexa"
3) Speak a sentence with indianized accent
4) Check if you get appropriate response for the question
```

Bug List: (1 of 1) *First Last Prev Next* Show last search results

7.6 Lessons Learnt

We as a team learnt many interesting things in the process of testing the ai application. Below are some of them.

- We evaluated and compared different testing methods and came up with the best method that suits our Alexa conversational AI testing.
- We learnt how to come up with limited entry decision table from hundreds of inputs
- We got to research about conversational ai
- It was exciting to perform black box testing on a real AI application
- We also have learnt how to use tools like Echoism.io, VUI (Voice user Interface) that are used to test Alexa.