# CHAPTER-1 INTRODUCTION

## INTRODUCTION TO MACHINE LEARNING

Machine learning is an application of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed. Machine learning focuses on the development of computer programs that can access data and use it learn for themselves.

**Fig 1.1** Machine Learning

The process of learning begins with observations or data, such as examples, direct experience, or instruction, in order to look for patterns in data and make better decisions in the future based on the examples that we provide. The primary aim is to allow the computers learn automatically without human intervention or assistance and adjust actions accordingly.

Machine learning enables analysis of massive quantities of data. While it generally delivers faster, more accurate results in order to identify profitable opportunities or dangerous risks, it may also require additional time and resources to train it properly. Combining machine learning with AI and cognitive technologies can make it even more effective in processing large volumes of information.

# Some machine learning methods

Machine learning algorithms are often categorized as supervised or unsupervised.

### Supervised machine learning algorithms

Supervised machine learning algorithms can apply what has been learned in the past to new data using labelled examples to predict future events. Starting from the analysis of a known training dataset, the learning algorithm produces an inferred function to make predictions about the output values. The system is able to provide targets for any new input after sufficient training. The learning algorithm can also compare its output with the correct, intended output and find errors in order to modify the model accordingly.

### Unsupervised machine learning algorithms

In contrast, Unsupervised machine learning algorithms are used when the information used to train is neither classified nor labelled. Unsupervised learning studies how systems can infer a function to describe a hidden structure from unlabelled data. The system doesn’t figure out the right output, but it explores the data and can draw inferences from datasets to describe hidden structures from unlabelled data.

### Semi-supervised machine learning algorithms

Semi-supervised machine learning algorithms fall somewhere in between supervised and unsupervised learning, since they use both labelled and unlabelled data for training – typically a small amount of labelled data and a large amount of unlabelled data. The systems that use this method are able to considerably improve learning accuracy. Usually, semi-supervised learning is chosen when the acquired labelled data requires skilled and relevant resources in order to train it or learn from it. Otherwise, acquiring unlabelled data generally doesn’t require additional resources.

### Reinforcement machine learning algorithms

Reinforcement machine learning algorithms is a learning method that interacts with its environment by producing actions and discovers errors or rewards. Trial and error search and delayed reward are the most relevant characteristics of reinforcement learning. This method allows machines and software agents to automatically determine the ideal behaviour within a

specific context in order to maximize its performance. Simple reward feedback is required for the agent to learn which action is best; this is known as the reinforcement signal.

* 1. **INTRODUCTION TO NATURAL LANGUAGE PROCESSING**

Natural language processing (NLP) is the ability of computers to analyse, understand and generate human language, including speech. It is the relationship between computers and human language. More specifically, natural language processing is the computer understanding, analysis, manipulation, and/or generation of natural language.

Natural language refers to speech analysis in both audible speech, as well as text of a language. NLP systems capture meaning from an input of words (sentences, paragraphs, pages, etc.) in the form of a structured output (which varies greatly depending on the application).

Basically, it is the ability of a computer program to understand human language as it is spoken. Natural language processing is a fundamental element of artificial intelligence. The next stage of NLP is natural language interaction, which enables people to communicate with computers using normal, everyday language to complete tasks.

The ultimate goal of natural language processing is for computers to achieve human-like comprehension of texts/languages. When this is achieved, computer systems will be able to understand, draw inferences from, summarize, translate and generate accurate and natural human text and language.

# Some Natural Language processing methods

Natural language processing, however, is more than just speech analysis. There are a variety of approaches for processing human language. These include:

### Symbolic Approach

The symbolic approach to natural language processing is based on human-developed rules and lexicons. In other words, the basis behind this approach is in generally accepted rules of speech within a given language which are materialized and recorded by linguistic experts for computer systems to follow.

### Statistical Approach

The statistical approach to natural language processing is based on observable and recurring examples of linguistic phenomena. Models based on statistics recognize recurring themes through mathematical analysis of large text corpora. By identifying trends in large samples of text the computer system can develop its own linguistic rules that it will use to analyse future input and/or the generation of language output.

### Connectionist Approach

The connectionist approach to natural language processing is a combination of the symbolic and statistical approaches. This approach starts with generally accepted rules of language and tailors them to specific applications from input derived from statistical inference.

# How Systems Interpret Language

Morphological Level: Morphemes are the smallest units of meaning within words and this level deals with morphemes in their role as the parts that make up word.

### Lexical Level

This level of speech analysis examines how the parts of words (morphemes) combine to make words and how slight differences can dramatically change the meaning of the final word.

### Syntactic Level

This level focuses on text at the sentence level. Syntax revolves around the idea that in most languages the meaning of a sentence is dependent on word order and dependency.

### Semantic Level

Semantics focuses on how the context of words within a sentence helps determine the meaning of words on an individual level.

### Discourse Level

How sentences relate to one another. Sentence order and arrangement can affect the meaning of the sentences.

### Pragmatic Level

Bases meaning of words or sentences on situational awareness and world knowledge.

Basically, what meaning is most likely and would make the most sense.

* 1. **EXISTING SYSTEM**

English is the globally recognized language and it is estimated that the number of people in the world that use in English to communicate is 2 billion. Learning English can be challenging and time consuming for people who are not educated in standard schools. But at the same time it is also very valuable to learn and it creates many opportunities. There are many skill building applications such as word game, knudge me, CAT Vocabulary etc..., you can type in what you want to say and the app will model the correct pronunciation for your phrase.

## PROPOSED SYSTEM

* It’s an application that teaches you to ‘pronounce English like a British through real- world conversations.
* It’s a great use of AI in language learning. With Fluency Pro, you can work through

a wide range of activities which mainly focus on the individual sounds of English.

* Most activities involve recording yourself saying a word (business, food, fashion, etc.).
* The app models the correct pronunciation then records your attempt. You can listen

back to a recording of your voice.

* You can retry as many times as you like.

# CHAPTER-2 LITERATURE SURVEY

## API’S & LIBRARIES REQUIRED

* + 1. **Speech recognition services**

The Speech recognition and its translation to digital text version is an area that many

programmers and developers have been dealing with for some years. It is a wide area, which interferes with many scientific disciplines such as linguistics, mathematics and computer science. To reach correct results of spoken words or sentences complex diagnostic of voice is needed; for example, identification of language, recognition of intonation for correctly specifying of sentence patterns, or statistical processing of verbalism.

Speech recognition is generally applied to services, where is not convenient to use ordinary input methods. Besides this group of devices, voice control is also used in common devices as mobile phones or computers for the purpose of simplification, facilitation or making work more effective. Today’s smart devices are dispose among other things this feature. This group includes devices such as televisions, computers, mobile phones, watches, domestic appliances such as fridges, freezers, intelligent controlling of households, in-car systems, offices as well as devices in industry or development. Speech recognition is also used for people with some type of handicap to help them to create text files and documents. It is also used in military, especially in air forces, transportation, education or autonomous systems (Most Common Uses of Voice Recognition Software 2016).

There are many ways to create an application using some Speech to text APIs. Major tech companies such as Google, Microsoft, Amazon or Apple use their own algorithm for voice processing and invest considerable resources to improve it. These services are no longer just about voice recording and trying to transpose it to the text form. The algorithms of speech recognition are very sophisticated. That is one of the reason, why Artificial intelligence is used. The algorithms must be able to separate speech and noise in the background, find out not just correct language but also specific dialect with intonation. These services are built to become more like a personal assistant; they should become some “new member” of household in the

future. AI can learn itself and so after a while, it is able to find out, when its user will come home, what to buy, how to set a thermostat and much more (The Past, Present, and Future of Speech Recognition Technology). It is not easy to build a software like this, and the best people in this field of research needed to be employed. To be more progressive and open for all possibilities of voice control, these companies offer their services for other developers, however, of course, with some restrictions, e.g. such as the length of one record or number of free minutes on their servers in a month.

There are three most used methods for Speech recognition (A Review on Different Approaches for Speech Recognition System 2018):

* + - * HMM – Hidden Markov Model
      * DTW – Dynamic Time Warping
      * Neural networks

“A Hidden Markov Model is a type of graphical model often used to mode temporal data. The HMM assume that the data observed is not the actual state of the model, but instead generated by the underlying hidden states. Because of the flexibility and computational efficiency, HMM have found wide application in many different fields. They are known for their use in temporal pattern recognition and generation, such as speech recognition, handwriting recognition, and speech synthesis.” (Hidden Markov Models 2018).

Dynamic Time Warping is an algorithm for measuring similarity between two sequences that may vary in time or speed. It is possible to compare it to two video records. For instance, similarities in walking patterns would be detected, even if in one video the person walked slowly and if in the other video the person was walking more quickly, or even if there were accelerations and deceleration during one observation. In general, DTW is a method that allows a computer to find an optimal match between two given sequences with certain restrictions (International Journal of Computer Applications 2015).

Simply put, while HMMs uses mechanism of parsing of voice signal and probability of occurrence parts of sentence, DTW uses recognition of individual words.

Neural networks are capable of solving more complicated recognition tasks but could not perform as excellent as HMM when it comes to large vocabularies. Neural network technology is used due the following reasons; It reduces the modeling unit to advance the recognition rate, depth learning, can be used to develop combine a hybrid system. This method of speech recognition has been significantly developed in recent years (Ibid).

# Cloud Speech API

Google Cloud Speech API enables developers to convert audio to text by applying powerful neural network models in an easy to use API. The API recognizes over 120 languages and variants to support a global user base. It also allows transcribing the users’ text by dictating to an application’s microphone. It also filters inappropriate content in text results for all languages, enables command-and-control through voice, or transcribes audio files, among many other use cases. Google Cloud Speech service enables to recognize the audio uploaded in the request, and integrates it to audio storage on Google Cloud Storage by using the same technology Google uses to power its own products (Documentation to Cloud Speech-to-Text API 2018).

Google Cloud Speech API applies most advanced deep learning neural network algorithms on user's audio for speech recognition with high accuracy. The advantage is that Speech API accuracy improves over time as Google company improves the internal speech recognition technology and software (Ibid).

One of the features is that Speech API can stream text results (return immediately), as they become available, so the recognized text appears immediately while speaking. There is also option to recognize text from audio file. The Google Cloud service can also handle noisy audio, so there is no need to filter audio record before processing. This service has been available for third-part developers since summer 2016, when Google allowed access to its speech recognition technology. After this step, Google Speech Cloud API has become the most used web recognition service

(Ibid).

Regarding pricing, the Google Cloud Speech API is priced monthly on the amount of audio or video successfully processed by the service, measured in increments rounded up to 15 seconds (Figure 1). For example, three separate requests, each containing 7 seconds of audio are billed

as 45 seconds (3 x 15 seconds) of audio. Fractions of seconds are included, when rounding up to the nearest increment of 15 seconds. That is, 15.14 seconds are rounded up and billed as 30 seconds. Pricing tiers are based on the total amount of audio processed by the service per month (Documentation to Cloud Speech-to-Text API 2018).

### Types of Speech requests

The Speech API has three main methods to perform speech recognition (Ibid):

* + Synchronous Speech Recognition
  + Asynchronous Speech Recognition
  + Streaming Speech Recognition

Synchronous Speech Recognition returns recognized text for short audio files (less than ~1 minute) in the response as soon as it is processed. Audio content can be sent directly to Cloud Speech-to-Text, or it can process audio content already residing in Google Cloud Storage.

Synchronous Speech Recognition is suitable for shorter audio records or for audio records stored locally because this method of Speech Recognition is the fastest and the simplest (Documentation to Cloud Speech-to-Text API 2018).

Asynchronous Speech Recognition starts a long running audio or video processing operation. Asynchronous Speech Recognition is used to recognize audio or video file, which is longer than a minute. Audio content can be sent directly to Cloud Speech-toText, or it can process audio content that already resides in Google Cloud Storage (Ibid).

Streaming Speech Recognition allows streaming audio to the Cloud Speech API and receiving the stream of speech recognition results in real time as the audio is processed. Streaming Speech Recognition is available only via gRPC. gRPC is explained later (Ibid).

The application is based on Streaming Speech Recognition method. The reason of using this method of speech recognition is that user can say such as short so long or multiple commands as well and the process of translating speech to text should be as fast as possible. The best way how to ensure this feature is to use Streaming Speech Recognition method, where the results are returned immediately.

# Google Protocol RPC Library (gRPC)

To work with real-time applications such as recording live audio from a microphone the streaming speech recognition has been developed. This method returns interim results while an audio sample is recorded, therefore allows a result to appear, while a user is speaking.

Streaming recognition method uses gRPC bi-directional stream for recognition on an audio data (Cloud Speech-to-Text, Basic 2018).

gRPC (Remote Procedure Calls) is a modern open source remote procedure call system initially developed at Google runs on HTTP/2 network protocol. It enables client and server applications to communicate transparently and makes it easier to build connected systems. As in many RPC systems, gRPC is based around the idea of defining a service, specifying the methods to be called remotely with their parameters and return types. On the server side, the server implements this interface and runs a gRPC server to handle client calls. On the client side, the client has a stub that provides the same methods as the server (gRPC Overview 2018).

The communication between server and client side is shown in Figure 2.

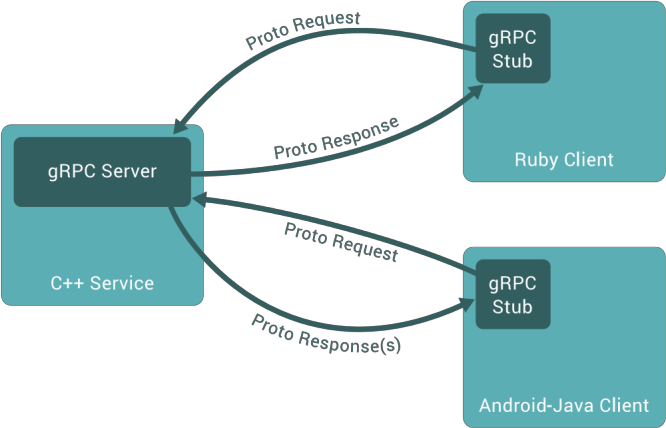
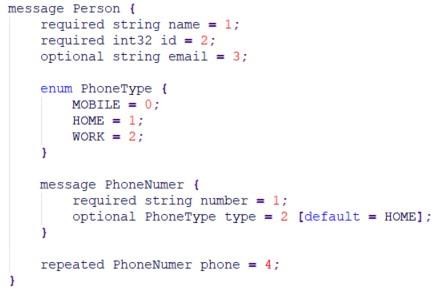


Figure 2. gRPC Client - Server communication

gRPC clients and servers can run and talk to each other in a variety of environments and can be written in any of gRPC supported languages thus for example, there can be gRPC server side

created in JAVA language and the client side in Go, Python or C++ language. By default, gRPC uses protocol buffer, Google’s mature open source mechanism for serializing structured data, however, it can be used also with other data formats, such as JSON (Ibid).

Protocol buffers are a flexible, efficient, automated mechanism for serializing structured data. Each protocol buffer message is a small logical record of information, containing a series of name-value pairs called fields. This information is saved in .proto file, as below is shown (gRPC Developer Guide 2018).



# TextToSpeech API

Synthesizes speech from text for immediate playback or to create a sound file. A TextToSpeech instance can only be used to synthesize text once it has completed its initialization. Implement the [TextToSpeech.OnInitListener](https://developer.android.com/reference/android/speech/tts/TextToSpeech.OnInitListener.html) to be notified of the completion of the initialization. When you are done using the TextToSpeech instance, call the [shutdown()](https://developer.android.com/reference/android/speech/tts/TextToSpeech.html#shutdown()) method to release the native resources used by the TextToSpeech engine.

# CHAPTER-3 SYSTEM-ANALYSIS

## 3.1 SYSTEM REQUIREMENTS

**3.1.1 Software Requirements**

1. **Java**

Java is a high level programming language and computing platform developed by Sun Microsystems in 1995. Since then, the language has been regularly updated with Java SE 8.0 version being the latest version, released in March 2014.

Based on the advantages of Java, it gained wide popularity and multiple configurations have been built to suit various types of platforms including Java SE for Macintosh, Windows and UNIX, Java ME for Mobile Applications and Java EE for Enterprise Applications.

With the growing importance of web based and mobile based applications, Java today is the foundation for most networked applications and is considered to be useful for scripting, web- based content, enterprise software, games and mobile applications.



**Fig 3.1** Java

# Android Studio

Android studio is the official IDE (Integrated Development Environment) or tool (layman terms) for developing application exclusively for Android platform. It has a strong editor tool for developing creative UI and emulators for different versions to test and simulate sensors without having actual Android devices. It also has a very useful Gradle plugin using which you can create application files (apks) with different configurations. Moreover it makes exporting and uploading apk on play store easy with a single click. It also has ANT build if you prefer that. In the recent updates Android studio has brought instant run which makes testing even faster and easier.



# Android SDK

**Fig 3.2** Android Studio

The Android software development kit (SDK) includes a comprehensive set of development tools. These include a debugger, libraries, a handset emulator based on QEMU, documentation, sample code, and tutorials. Currently supported development platforms include computers running Linux (any modern desktop Linux distribution),Mac OS X 10.5.8 or later, and Windows 7 or later. As of March 2015, the SDK is not available on Android itself, but software development is possible by using specialized Android applications.

Until around the end of 2014, the officially-supported integrated development environment (IDE) was Eclipse using the Android Development Tools (ADT) Plug-in, though IntelliJ IDEA IDE (all editions) fully supports Android development out of the box, and NetBeans IDE also supports Android development via a plug-in. As of 2015,Android Studio, made by Google and powered by IntelliJ, is the official IDE; however, developers are free to use others, but Google made it clear that ADT was officially deprecated since the end of 2015 to

focus on Android Studio as the official Android IDE. Additionally, developers may use any text editor to edit Java and XML files, then use command line tools (Java Development Kit and Apache Ant are required) to create, build and debug Android applications as well as control attached Android devices (e.g., triggering a reboot, installing software package(s) remotely).

Enhancements to Android's SDK go hand-in-hand with the overall Android platform development. The SDK also supports older versions of the Android platform in case developers wish to target their applications at older devices. Development tools are downloadable components, so after one has downloaded the latest version and platform, older platforms and tools can also be downloaded for compatibility testing.

Android applications are packaged in.apk format and stored under /data/app folder on the Android OS (the folder is accessible only to the root user for security reasons). APK package contains .dex files (compiled byte code files called Dalvik executables), resource files, etc…

# Firebase

Firebase is a Backend-as-a-Service — BaaS — that started as a YC11 start up and grew up into a next-generation app-development platform on Google Cloud Platform.

Firebase frees developers to focus crafting fantastic user experiences. You don’t need to manage servers. You don’t need to write APIs. Firebase is your server, your API and your data store, all written so generically that you can modify it to suit most needs. Yeah, you’ll occasionally need to use other bits of the Google Cloud for your advanced applications. Firebase can’t be everything to everybody. But it gets pretty close.

### Firebase Real time Database:

The Firebase Real time Database is a cloud-hosted database. Data is stored as JSON and synchronized in real time to every connected client. When you build cross-platform apps with our iOS, Android, and JavaScript SDKs, all of your clients share one Real time Database instance and automatically receive updates with the newest data.

# CHAPTER-4 SOFTWARE DESIGN

## 4.1 UML DIAGRAMS

Unified Modelling Language is a tool that helps a designer to present his ideas about the project to his client and his developer. Modelling plays a crucial role in designing a software. A poorly designed model can lead to a poorly developed software.

A UML system has using five different views that help in describing systems from different perspectives. Each view has a set of diagrams that and components that represent the real time objects.

## User Model View

* + It models the user behaviour in a system context.
  + All the diagrams are drawn keeping in mind the user’s response and reaction towards a system.

## Structural Model View

* + This view consists of class diagram and object diagram which is used to model the static structures.
  + It uses objects, attributes, operations and relationships.

## Behavioural Model View

* + It mainly consists of the sequence diagram, collaboration diagram, state chart diagram and activity diagram. They mainly represent flow of actions between different objects involved in the system
  + They are used to visualize various dynamic aspects of the system architecture.

## Implementation Model View

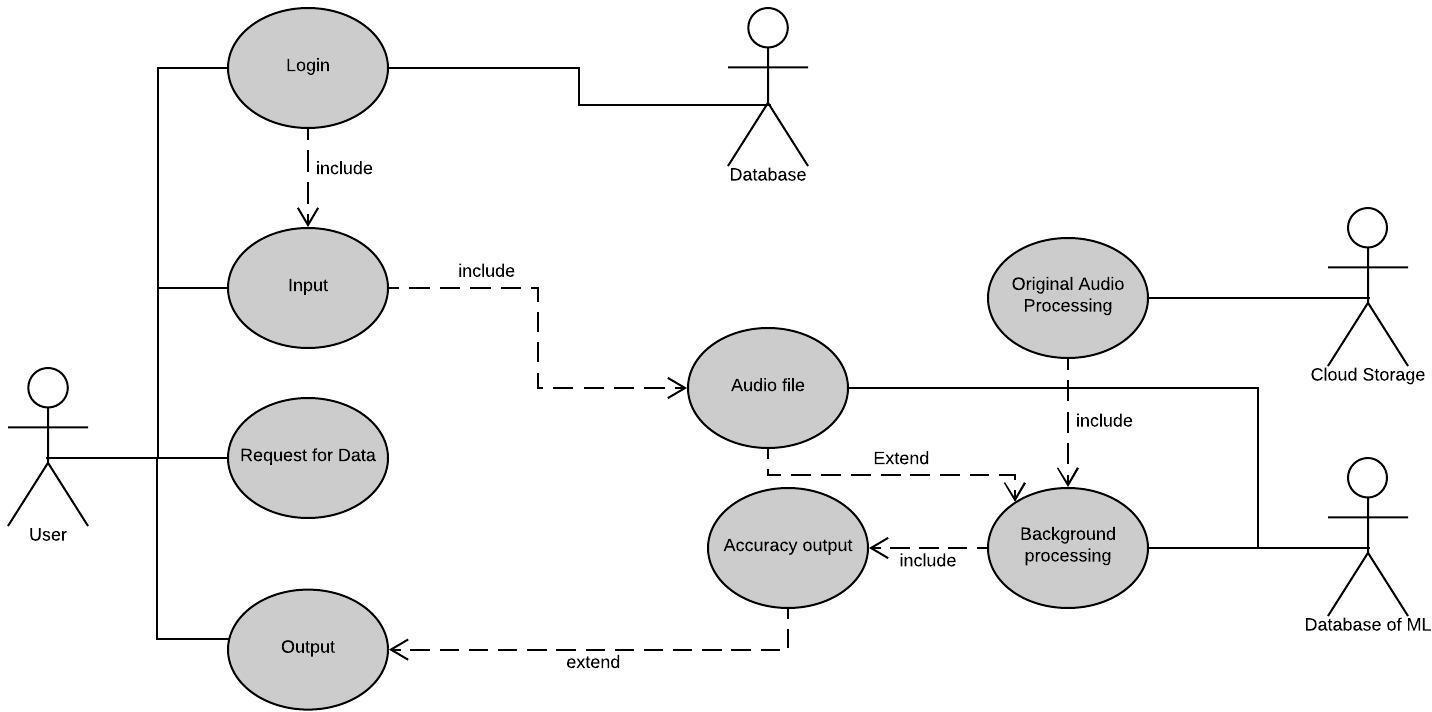
* + This view consists of component diagrams and deployment diagrams. This view models the static software modules for an organization.
  + This usually contains the data files, documentation, the executables and source code.
  + These are the physically replaceable components of the system. They are modelled using component diagrams.

# Use Case Diagram

The basic representation for the interaction of the user with the system is represented using the use case diagram. It involves the relationship between the user and various use cases with the actors being involved. There are different kinds of relationships that are involved between the use cases and the actors. They include:

* Association relationship
* Generalization
* Dependency
* Realizations
* Transitions

The following represents the use case diagram of the proposed system:



**Fig 4.1:** Use Case Diagram for Developed Model

# Class Diagram

They are static representation of an application. Only the class diagrams have the capability to be directly mapped with the OOP Languages because in OOPs everything is model in the form of classes and objects. Because of this reason these diagrams are used widely at the

time of construction. This is one of the most popularly used UML diagram in the designer community. A class diagram plays an essential role in forward and reverse engineering.

* + - * It acts as a base for the component and deployment diagrams.
      * It mainly describes and defines the basic responsibilities of a system’s application.
      * It implements the analysis and design view for a static application.

In a class diagram, each object is modelled as a class. Each class consists of section or compartments.

* Class name
* Attributes of a class or operations
* Methods or functions
* Documentation (optional section)

The following points ought to be recollected while drawing a class diagram:

* + The name of the class diagram must be meaningful to portray the aspect of the framework.
  + Each component and their connections must be distinguished ahead of time.
  + Each class has a responsibility (attributes and methods) that must be identified clearly.
  + Number of properties for each class must be minimum. Since pointless properties will make the diagram convoluted.
  + At whatever point required to depict some part of the diagram use notes since toward the finish of the diagram it must be justifiable to the designer/coder.
  + Before finalising the last version, the diagram must be drawn on plain paper and revise whatever number circumstances as would be prudent to make it redress.

### Scopes:

The UML diagrams have two different types of scopes for class members:

* + Instance members scope
  + Classifier members scope

### Classifier members:

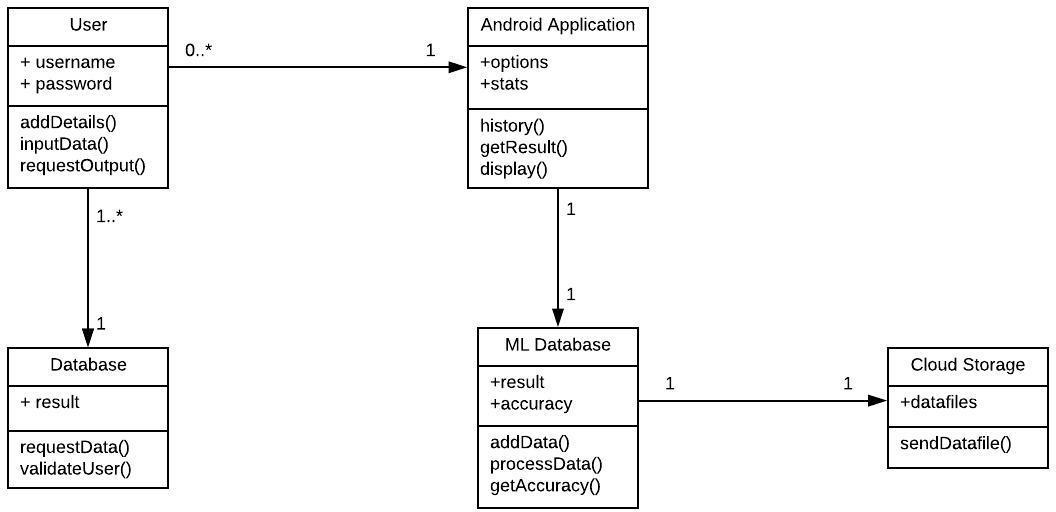
They are “static” members of a class in many programming languages. The scope is the class itself.

* + Static attributes are common to all other objects that invoke the class.
  + Static methods are not instantiated.

### Instance members:

They are nothing but the members that are local to an object.

* + The main purpose of instance members is to allow the objects to store their states.
  + Declarations outside the methods are usually known as instance members.



**Fig 4.2:** Class Diagram for Developed Model

# Sequence Diagram

The Sequence Diagram depicts the time sequence among various objects in an application. It depicts the sequence of messages with which objects communicate with each other so that they carry out the required functionality.

It consists of the lifelines which are usually parallel vertical lines. It consists of horizontal arrows which indicate the direction of the messages that are exchanged in a proper order which makes the user easy to understand.

The lifeline for a given object represents a role. The synchronous calls are represented with the help of a solid arrow head whereas the asynchronous messages are represented with the help of open arrow heads.

All objects are represented according to their time ordering. Timing of messages plays a major role in sequence diagrams. An object is killed immediately after its use in sequence diagrams.

## Common Properties:

An arrangement graph is much the same as unique sort of diagram and offers some indistinguishable properties from other diagrams. In any case, it varies from every single other diagram in its content.

## Contents

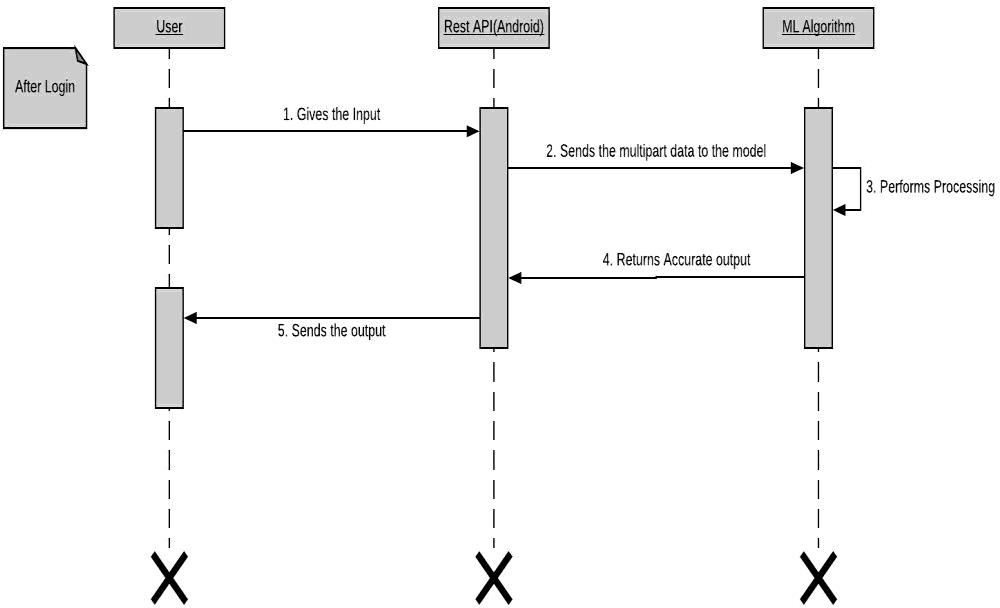
Objects are normally named or unknown instances of class, however may likewise speak to occurrences of different things, for example components, collaboration and nodes. Graphically, object is represented as a rectangle by underlying its name.

## Links

A link is a semantic association among objects i.e., an object of an affiliation is called as a connection. It is represented as a line.

## Messages

A message is a determination of a correspondence between objects that passes on the data with the desire that the action will follow.



**Fig 4.3** Sequence Diagram for Developed Model

# Activity Diagram

The flow from one activity to another activity can be represented in the form of a flow chart which is usually an activity diagram. It forms a backbone for the UML diagrams. It depicts the dynamic aspects for all the objects within the system.

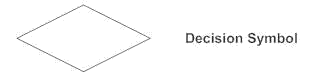
The control flow from one object to another object is drawn which shows the basic operations that are to be performed.

Activity diagrams are constructed using the following:

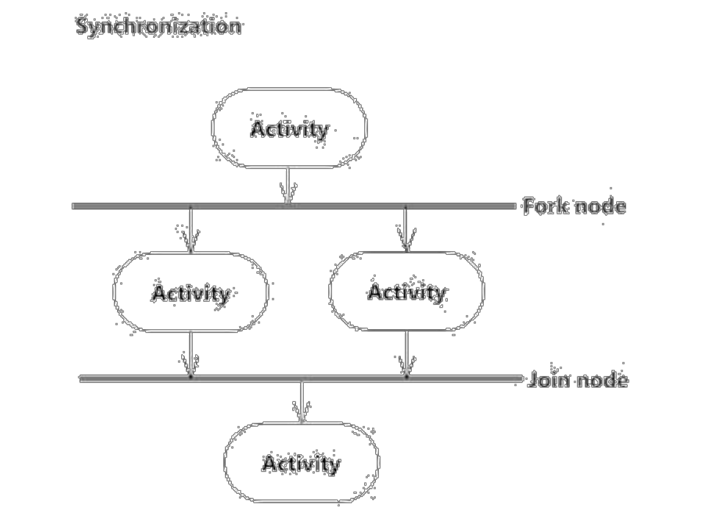
1. Actions are represented using rounded rectangles



1. Decisions are represented using diamonds



1. Concurrent activities bars are represented using the start (split) or end (join)



1. Time event is represented as



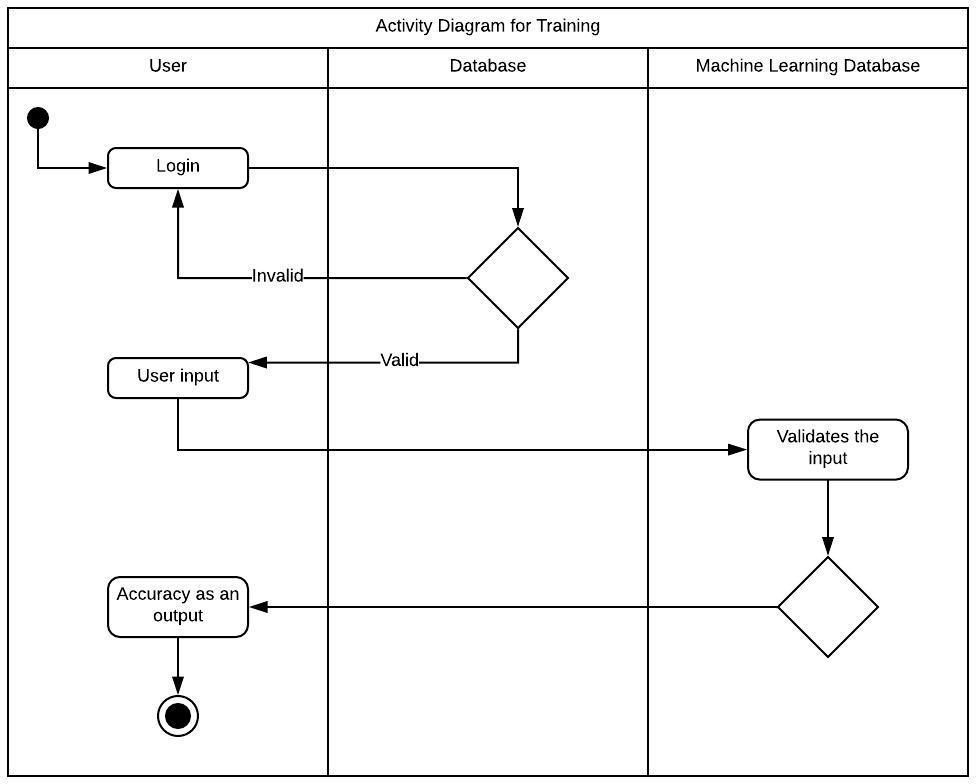
1. Final state is represented using encircled black circle.



The basic purpose of an activity diagram is same as that of other UML diagrams. The dynamic behaviour of the system is viewed by the activity diagram. They are used to construct a system using the backward and forward engineering mechanisms.

The purpose of an activity diagram is as follows:

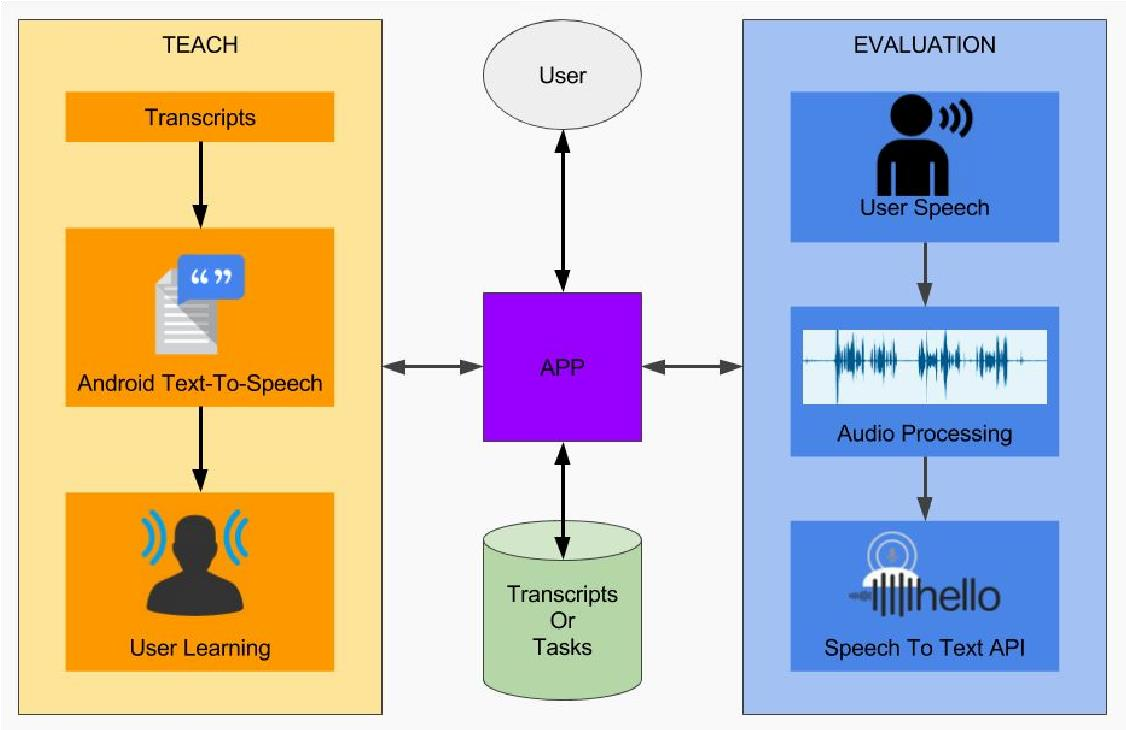
* + For drawing the flow (i.e. activity) in a system.
  + For showing the flow of sequence from one activity to another activity.
  + For showing the concurrent and parallel flow of actions in the system. The elements that are used in an activity diagram are as follows:
    - Association relationship
    - Activities
    - Conditions and Constraints



**Fig 4.4** Activity Diagram for Developed Model

# CHAPTER-5 IMPLEMENTATION

## IMPLEMENTATION METHOD

We intend to utilize the Google Speech API to help with speech recognition. This task is more effectively visualized in the following block diagram:

**Fig 5.1** Google Speech API Architecture

Google’s speech API is still in beta and can help recognise over 80 languages. Here is a short snippet demonstrating how we communicate



**Fig 5.2** Sample GRPC request to Google Speech API



**Fig 5.3** Sample Response from Google Speech API

The Google Speech API returns us with a transcript detected in the audio and a “Confidence Value”.Understanding this confidence score is the key to evaluating user performance. As per the documentation, the confidence score is defined as follows:

*“When returning an alternative, the Speech API will assign a confidence value to any given transcription, on a scale of 0.0 to 1.0, with 1.0 meaning absolute confidence. You can use these confidence values to compare alternatives or to decide whether to return results to a user (and/or ask for confirmation from the user)”.*

This essentially means that this is a probability assigned to generated transcript which implies we can make use of these values to evaluate user’s speech. Since the speech API focuses on returning alternatives we can make the following confusion matrix:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Parameters** | **Scenario 1** | **Scenario 2** | **Scenario 3** | **Scenario 4** |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Audio Spoken** | America | Table | Chair | Car |
| **Transcript** | America | Table | Hair | Cart |
| **Confidence** | 0.9 | 0.4 | 0.3 | 0.7 |
| **Alternatives** | None | Cable, Fable | Care, Fare | Bar, Far |
| **Evaluation** | Good | Good  if confidence>0.65 else Bad | Bad | Bad |

So we encounter four scenarios returned from Google Speech API:

### Transcript Matches with High Confidence:

In this case, we can safely assume that the user spoke the right word and we can evaluate this to Good.

### Transcript Matches with Low Confidence:

This is a pretty common scenario, in this case, the word sounds right, but the speech API is not confident enough. This is most likely because there are a lot of similar sounding words in that language. Which implies that we can safely assume this word to be correct if it’s above a certain threshold. A research was performed to come up with an average to the least accepted confidence level to consider the word pronounced correctly.

### Transcript Mismatches with Low Confidence:

This is likely a wrong pronunciation and we report this as bad.

### Transcript Mismatches with High Confidence:

This is definitely a miss-pronunciation and we report this as wrong.

# How to build Google Cloud Speech Application

1. **Implementation of Google Cloud Speech library**

To enable API speech in an application, there has to be a unique API key first and next service account key. These keys are used in an application for authentication to gain access to Google server and to use with Google Cloud Speech service. To gain these keys, sign-in to Google Cloud Platform is needed. There is necessary to set up a project first and choose right service. There are two options, how to use Service Account Key. It is possible to download this key as JSON format or P12 format, which is there for backward compatibility. Before downloading the Key, a user is redirected to Google page, where is required to fill in personal information and information about credit card as well, in the case of data limit overdrawing. The downloaded key contains information about users account as well as project id, generated private key, client id, security key and other important information for server (Documentation to Cloud Speech-to-Text API 2018).

The next code shows, how to credentials .json credential file to an Android application.



There is Google Cloud API v1 Client library, which allows developers a connection to Google Cloud service and use of all Google Speech-to-Text features. The library is free and there are more options on how to implement it to a project. The user can choose whether to use this library as a downloaded .zip file, define it in Android Maven file or in Gradle file (Ibid).

In this case, Google Cloud API v1 Client library is used and defined in Android build.gradle file.

As first, the new plugin must be added. The plugin specifies type of the project and indicates which plugin should be used.



After that the path of the class must be added to inform the system, where the project and used libraries are situated and.



To ensure that protocol buffer will used the plugin to the module needs to be add and the protbuf-lite library as well.



“A streaming Speech-to-Text API recognition call is designed for real-time capture and recording of audio within a bi-directional stream. That means that the application can send audio on the request stream and receive interim and final recognition results on the response stream in real time. Interim results represent the current recognition result for a section of

audio, while the final recognition result represents the lasts, best guess for that section of audio.” (Ibid). This is an advantage of gRPC service because bidirectional communication cannot be used in REST service.

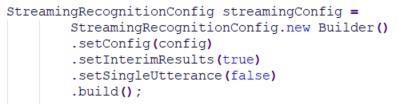
1. Streaming requests

In contrast with synchronous and asynchronous communication, where it is possible to send an audio sample and a configuration within a single request, Streaming Speech recognition requires to send these items in multiple requests. The first request – StreamingRecognizeRequest must contain a configuration without any added audio sample. As follow another request – StreamingRecognizeRequest is sent over the same stream and this request contains raw audio bytes (Documentation to Cloud Speech-to-Text API 2018).

A StreamingRecognitionCofnig consists of the following fields (Ibid):

* + config – (required) contains configuration information about the audio sample
  + SingleUtterance – (optional, defaults to “false”) indicates whether the request should end immediately after any speech is no longer detected. If is set, after Speech-to-Text will detect silence or pause in the recorded audio, the recognition will end. If it is not set, the stream continues with listening and processing an audio, until the stream is closed directly, or the limit of the stream recognition is reached. This implies, that to set single\_uterrance to true is advisable for processing voice commands.
  + InterimResults – (optional, defaults to “false”) by the setting this field to “true”, temporary results will be return, and these results may be modified after processing more audio. By setting InterimResults to “false”, temporary results will be noted inside responses by the setting of is\_final to “false”

Used StreamingRecognitionConfig method is shown in the following code snippet.

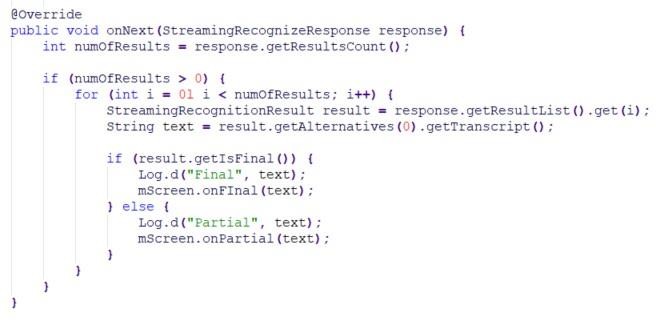


1. Streaming responses

Streaming speech recognition results are returned inside a several of responses of type StreamingRecognizeResponse and comprise of following fields (shown in the next code snippet) (Documentation to Cloud Speech-to-Text API 2018):

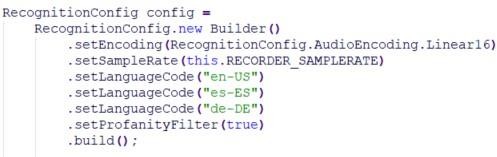
* + speechEventType – “value of this event indicates when a single utterance has been determined to have been completed.”
  + result – contains the list of results, which were returned and may be interim or final. This field contains of following sub-fields: o alternatives – includes a list of alternative transcriptions o isFinal – notifies whether the returned results inside this list are interim or final

o stability – indicates defectiveness of obtained results, where value 0.0 indicates total instability while value 1.0 indicates entire stability

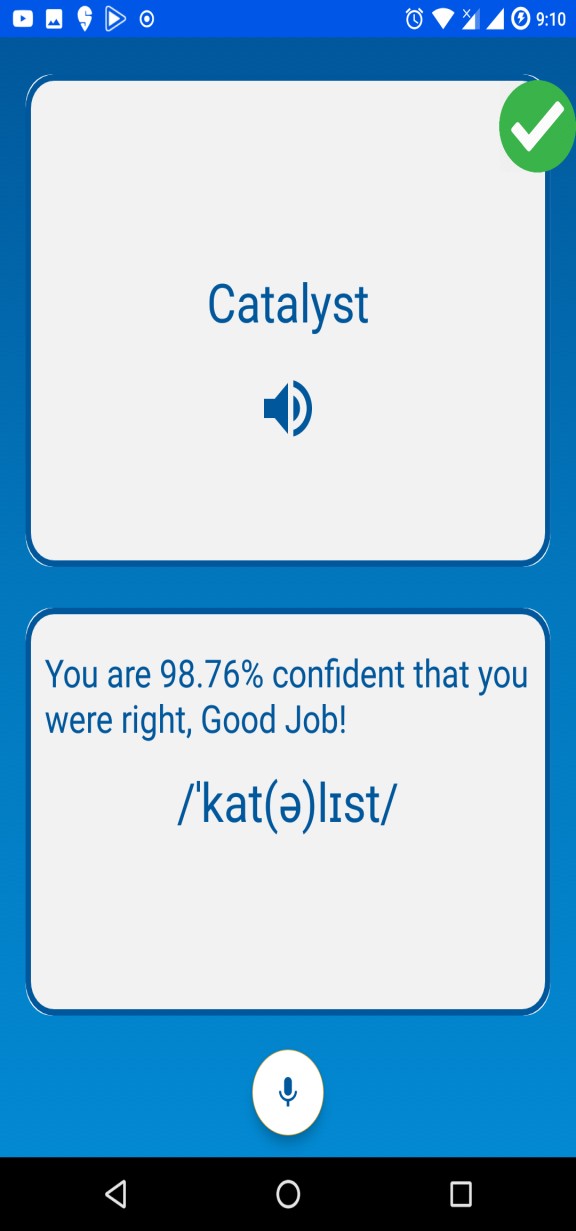
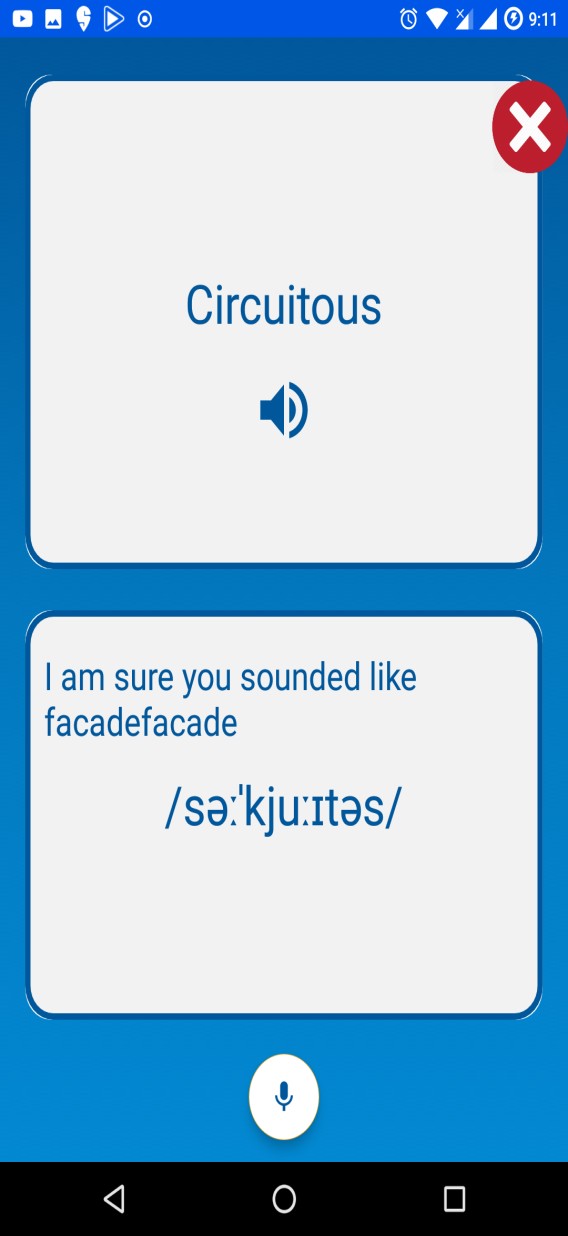


Besides these settings and attributes, it is possible in the configuration of API to set more important properties such as audio encoding, sample rate, supported languages or profanity filter as well. In this case, every profanity word is replaced by asterisks, except the first character (Ibid).

The setting of these attributes is the next code snippet.

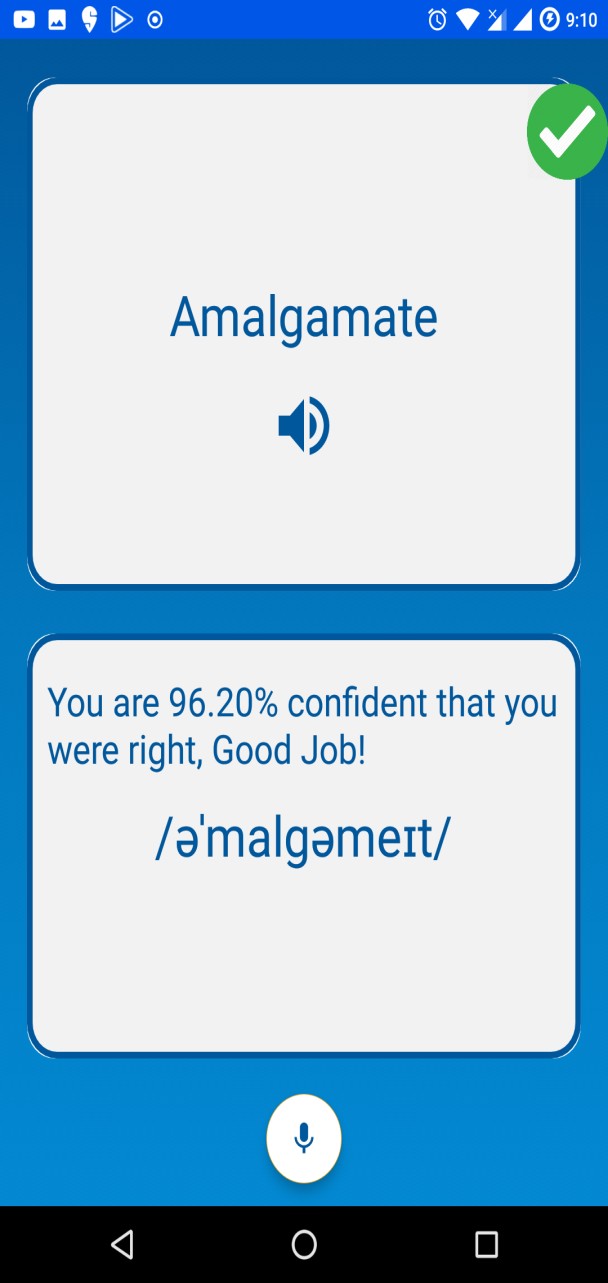
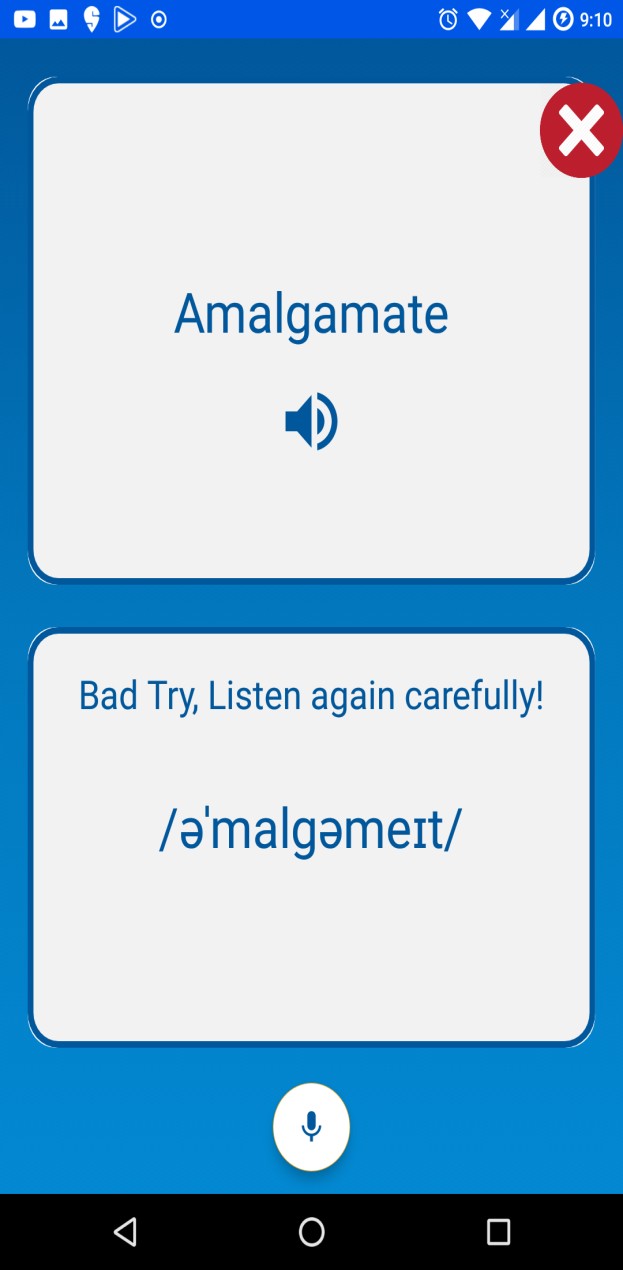


## USER INTERFACE



***Words highlighted red if Words highlighted green if***

***Pronounced wrong pronounced right***

***Words highlighted green if Words highlighted red if***

***Pronounced right pronounced wrong***

32

# CHAPTER-6 TESTING

Testing is an important stage in the System development life cycle. Careful planning is needed to get the most out of testing and to control testing cost. Test planning is concerned with setting out standard for the testing process rather than describing the product test. System testing is an important element of software quality assurance and ultimate review of specifications.

## INTRODUCTION

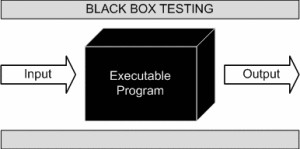
Software testing is a process used to identify the correctness, completeness and quality of the developed computer software. Testing a process is questioning a product in order to evaluate it, where the questions are things the tester tries to do with the product, and the product answers with its behaviour in reaction to probing of the tester. The testing phase is performed after the coding to detect all the errors and provide quality assurance and ensure reliability of the software. Testing is vital to the success of the system. During testing, the software to be tested is evaluated to determine if the system is performing as expected. Clearly, the success of testing in revealing errors depends critically on the test cases. The different testing strategies employed in this project are explained in this chapter.

## UNIT TESTING

In computer programming, unit testing is software verification and validation method in which a programmer tests if individual units of source code are fit for use. A unit is smallest testable part of an application. It is also called as module testing. The goal of unit testing is to isolate each part of the program first and then testing the sum of its parts, integration testing becomes much easier. In our project, we apply this by testing the various modules of the application and also each feature individually.

# Black Box Testing

In this approach, a few tests cases of experiments are created as entering circumstances that totally execute every single deliberate need for the program. This looking at has been utilized to discover bugs. In this looking at, just the yield is checked for accuracy. The sensible stream of the information isn't checked.



**Fig 6.1**: Black Box Testing

# White Box Testing

In this approach, the test instances are generated on the good judgment of every module via loading different words to that module and are tested on all test cases. It has been used to generate the take a look at test cases inside the following test cases:

* Guarantee that everyone impartial paths were accomplished.
* Execute all logical selections on their authentic and false aspects.
* Execute all test cases in the application.
* Execute all the instructions from mobile application.

## INTEGRATION TESTING

Integration testing is three phases in software testing in which individual software modules are combined and tested as group. It occurs after unit testing and before system testing. Integration testing takes as its input module that have been tested groups them in larger aggregates, applies tests defined in an integration test plan to those aggregate and delivers as its output the integrated system ready for the system testing. The purpose of the integration testing is to verify functional, performance and reliability requirement placed on major design items. All the different modules of the project are combined and tested.

# CHAPTER-7 CONCLUSION

## CONCLUSION

It has been a great pleasure for us to work on this exciting and challenging project. This project proved good for us as it provided practical knowledge of not only **programming in Java** and using **Android Studio** through which we come to know some extent of Mobile based applications and Firebase but also able to understand how to work with various API’s using “**Fluency Pro**”. It also provides knowledge about the latest technology used in developing cloud enabled application and client server technology that will be great demand in future. This will provide better opportunities and guidance in future in developing projects independently.

## FUTURE SCOPE

* + - Our Application in future can be introduced in schools, colleges and various organisations where people can learn English easily and implement their ideas.
    - This will be very helpful to every individual who wants to express their ideas and feelings.
    - It will also be helpful for the people who are preparing for GRE, GMAT, CAT and thinking to pursue higher studies.
    - Tutors will also get help from our app.
    - Our Application can be useful especially for students studying in government schools and colleges who want to learn English on their own.
    - This Application can be used in training and placements for enhancement in English speaking and learning.

# CHAPTER-8 BIBLIOGRAPHY

## 8.1 REFERENCES

[1] "Speech API - speech recognition | Google cloud platform," Google Cloud Platform. [Online] Available: <https://cloud.google.com/speech/>

Accessed: August 14, 2018

[2]"Cloud speech API basics," Google Cloud Platform, 2016. [Online] Available: <https://cloud.google.com/speech/docs/basics#confidence-values>

Accessed: September 24, 2018

1. "Package google.rpc | Google cloud speech API (limited preview) | Google cloud platform," Google Cloud Platform, 2016. [Online]

Available:<https://cloud.google.com/speech/reference/rpc/google.rpc>

Accessed: December 14, 2016

1. "Protocol Buffers | Google developers," Google Developers. [Online] Available: <https://developers.google.com/protocol-buffers/>

Accessed: October 12, 2018