**CHAPTER 1**

**INTRODUCTION**

**1.1 OVERVIEW OF THE PROJECT**

There are several voice translators available in the android market which helps users to ease their mode of communication with unknown languages. The applications are available for foreign languages but there is no existing application which translates any foreign language to any regional Indian language.

The application allows user to select an input language from a list of foreign languages available. The preferred language to which the input is to be translated is selected from the list of options available. The application uses the Google’s voice recognizer to recognize the input when the user clicks the ‘Click Me’ button. For translation, the application uses the Bing translator to translate the input language to the preferred output language (only selected foreign languages) selected.

Bing translator is the open source software, used for speech based applications and recognizes few foreign languages. For Indian languages, the application uses the Google App Engine which is the database to store the word and their translated text. Whenever user clicks on the translate button, the result is retrieved from the App Engine and displayed on the screen.

Google App Engine can store enormous words and hence can be easily maintained and updated.

**1.2 EXISTING SYSTEM**

There are many successful android applications for a voice translation but there are no applications applicable for regional languages of India. The pronunciation of words will be very difficult and it has to be in British English accent. It is not interactive and has poor user interface.

**1.3 PROPOSED SYSTEM**

In our model of voice translation, the application uses Bing translator to translate speech for foreign languages and Google App Engine database to translate the input to the preferred regional language. Voice recognition is used to recognize input and if the user cannot pronounce a word correctly, the spelling of the word can also be given. The history of searched words will help the user to get familiar with difficult translations. Finally, the users use the translated words to forward text messages, comment or post in some social networking websites. The application is developed for android users and application can be downloaded free of cost.

**1.4 OBJECTIVES**

There are several people across the globe who wants to get connected but lags in the mode of communication. This application facilitates the user to translate the input from one language to other. The application mainly focuses in the Indian languages but also has support for several foreign languages.

**1.5 ORGANIZATION OF THE PROJECT**

The report has been organized in such a way that the flow of the process and modules of the application coincides with the text. The report is split into chapters that the application demands. The succeeding chapters deal with the requirements specification, a detailed description about the design and the test plans, the implementations and the results and finally the conclusion of the project. The requirements specification includes the overall description of the product and its specific requirements. The detailed design and the test plan consist of the description about the way in which the modules are decomposed and their dependencies. It also includes the Object Oriented Analysis and Design (OOAD) diagrams associated with the project. The test plan deals with the different test cases the inputs, the actions and their corresponding outputs.

**CHAPTER 2**

**REQUIREMENTS SPECIFICATION**

**2.1 INTRODUCTION**

This chapter discusses the product perspective, functions, user characteristics and the operating environment. Also, the applications-specific requirements, including the system features and data flow diagrams are discussed. In addition, performance requirements and software quality attributes are also discussed.

**2.2 OVERALL DESCRIPTION**

**2.2.1 PRODUCT PERSPECTIVE**

This project is the college level project and is being implementing under the guidance of college professors. The application will provide users to translate speech given as input to the desired language selected through voice translator. The output helps user to analyse the way each word is pronounced. The given input can be translated to several different languages simultaneously. Also, the user can bookmark the frequently used complex words.

**2.2.2 PRODUCT FUNCTION**

The application is free of cost. The advanced feature of this product is that, it can be used by any person who intends to translate his/her speech to Indian or other foreign languages. The history of sentences that were used for translation can also be viewed.

**2.2.3 User Characteristics**

The users who will be using this product covers a wide range. The users include students who are interested to learn a new Indian language, people who are new to that particular region and even foreigners who know only English which is also the universal language in this product. Because this product can translate any text to English and vice versa.

**2.2.4. Operating Environment**

The operating environment for this application is any device that is running Android as its operating system. The version of Android ranges from Android version 2.2 to 4.2(JELLY BEAN).

**Hardware SPECIFICATIONS**

**Developer End:**

* + - PENTIUM 4 PROCESSOR (3.0GHz+) OR HIGHER.
    - 512 MB RAM.
    - 512 MB internal MEMORY.

**User End:**

* + - 32 MB RAM
    - 32 MB FLASH MEMORY (max.)
    - 200 MHz PROCESSOR
    - VOICE INPUT/OUTPUT DEVICE.

**Software SPECIFICATIONS**

**Developer End:**

* + - * WINDOWS XP/7/8
      * EMULATOR
      * ECLIPSE
      * JAVA SDK 6.0 +

**User End:**

* + - * ANDROID DEVICE WITH VERSION 2.2 TO 4.2(JELLY BEAN)
      * FONTS OF THE LANGUAGES THAT ARE TO BE USED AS INPUT AND OUTPUT FOR TRANSLATION.

2.2.5 **Design and Implementation Constraints**

There are a few design and implementation constraints that has to be overcome in the forthcoming enhancements. They are:

* The output voice of the application is constant and cannot be changed.
* This is because, the default library for text-to-speech has only one built-in voice in it.
* The input word that is being pronounced by the user must be as accurate as possible else the desired word will not be recognized.
* To overcome this drawback, we provide an option of word suggestions from which the user can select the intended word.

**2.3 SPECIFIC REQUIREMENTS**

**2.3.1 External Interface Requirements**

**User Interfaces**

The product has a predominantly clear and well understandable user interface. First there are two dropdown spinner fields that will get the input and output language from the user. Then there is a button which will open the pop up box that will get voice data as input. Then a list of suggestions will be displayed that closely matches the word pronounced by the user. From the list the desired word is selected and the word is translated. The displayed translated word is then made to be pronounced by the device with the help of another button. This is the user interface design of this product.

**Communication Interfaces**

* The Customer must connect to the Internet to provide voice input.
* 2G or 3G Internet connection.
* A GSM mobile network connection.

**2.3.2 System Architecture**

In the system architecture, the application runs on the android device. The user opens the application and the interface comprised of two drop down boxes specified as ‘From’ and ‘To’ to select the input and the output language from the list displayed.

After selecting the language, the user clicks on the ‘Speak’ button where user speaks his/her input to the voice recognizer. A list of words are displayed based on the voice recognized and the user selects the preferred one.

On clicking the translate button, if the output is for foreign languages, the Bing translator is used for translation. Else if the output is for Indian language, the translation is done using Google’s App Engine database.

On clicking the ‘Click to Translate’ button, the application speaks out the translated word.

2.3.3 **Other Non Functional Requirements**

**Performance Requirements**

The application that we are going to develop will be used as a major voice translator for the Indian regional languages. Therefore, it is expected that the application will provide functionally all the requirements as specified within the application description.

**Safety Requirements**

The application requires an active internet connection for voice translation. In account of network failure, the application would not provide the accurate result. Hence, an active internet connection is required.

OPEN THE APPLICATION IN AN ANDROID DEVICE.

USE GOOGLE APP-ENGINE DATABASE FOR TAMIL AND HINDI LANGUAGES.

CLICK THE “TRANSLATE” BUTTON TO GET THE DESIRED OUTPUT.

CLICK THE “SPEAK OUT” BUTTON TO LISTEN THE TRANSLATED WORD.

USE ‘BING TRANSLATOR’ IN CASE OF FOREIGN LANGUAGES.

GIVE THE VOICE INPUT AND CHOOSE THE REQUIRED WORD FROM THE LIST OF SUGGESTIONS

SELECT THE FROM AND TO LANGUAGES.

*Fig 2.1 System Architecture*

**Security Requirements**

The application is secure and ad free. The quality of the application is designed in such a way that it is user friendly and can be easily used by any user.

**2.3.4 Software Quality Attributes**

**Functionality**:

Functionality is the required functions available, including interoperability and security.

**Reliability:**

It is the ability of the system to recover from the failure after detection. The project is prone to fault tolerance and recoverability.

**Usability:**

It is effectiveness of the use of system. The user interface of this project is developed in such a way that the user can understand easily, learn at a faster rate and operate with great level of comfort.

**Efficiency:**

The project is having high performance and resource behaviour.

**Portability:**

The application is compatible only in phones and tablets running with android operating system with version varying from 2.2 to 4.0.

**Integrity:**

It is protection of the system from unauthorized access. This project provides a higher degree of integrity.

**CHAPTER 3**

**SOFTWARE DESCRIPTION**

**3.1 FRONT END**

The J2EE platform, built on the Java programming language and Java technologies, is the application architecture that is best suited for an enterprise-distributed environment. The J2EE platform is a standard that brings the following benefits to IT organizations, application developers, and product vendors.

* Vendors develop products that can run on any system that supports the J2EE platform. With virtually no extra effort, the products are available on a wide range of system platforms.
* Corporate IT developers benefit from the advantages of portable component technology. IT applications become vendor-independent and release the IT organizations from the clutches of vendor lock-in.
* IT developers can focus on supporting business process requirements rather than building in-house application infrastructure. The application servers handle the complex issues of multithreading, synchronization, transactions, resource allocation, and life-cycle management.
* IT organizations can take advantage of the best available products built on a standard platform. To choose among products and select the most suitable and cost-effective development products, deployment products, and deployment platforms based on the requirements.

**3.2 FEATURES**

**ANDROID**

**Android** is a [Linux](http://en.wikipedia.org/wiki/Linux)-based [operating system for mobile devices](http://en.wikipedia.org/wiki/Mobile_operating_system) such as [smart phones](http://en.wikipedia.org/wiki/Smartphone) and [tablet computers](http://en.wikipedia.org/wiki/Tablet_computer). It is developed by the [Open Handset Alliance](http://en.wikipedia.org/wiki/Open_Handset_Alliance) led by [Google](http://en.wikipedia.org/wiki/Google).

Android has a large community of developers writing applications ("[apps](http://en.wikipedia.org/wiki/Mobile_apps)") that extend the functionality of the devices. Developers write primarily in a customized version of [Java](http://en.wikipedia.org/wiki/Java_(programming_language)).

Android was listed as the best-selling Smartphone platform worldwide in Q4 2010 by analysis with over 200 million Android devices in use by November 2011. According to Google's [Andy Rubin](http://en.wikipedia.org/wiki/Andy_Rubin), as of December 2011 there are over 700,000 Android devices activated every day.

**FOUNDATION**

Android, Inc. was founded in [Palo Alto, California](http://en.wikipedia.org/wiki/Palo_Alto,_California), United States in October, 2003 by [Andy Rubin](http://en.wikipedia.org/wiki/Andy_Rubin) (co-founder of [Danger](http://en.wikipedia.org/wiki/Danger_(company))), [Rich Miner](http://en.wikipedia.org/wiki/Rich_Miner) (co-founder of Wildfire Communications, Inc.), Nick Sears (once VP at [T-Mobile](http://en.wikipedia.org/wiki/T-Mobile_USA)), and Chris White (headed design and interface development at [WebTV](http://en.wikipedia.org/wiki/WebTV)) to develop, in Rubin's words "...smarter mobile devices that are more aware of its owner's location and preferences".Despite the obvious past accomplishments of the founders and early employees, Android Inc. operated secretly, revealing only that it was working on software for mobile phones.

**ACQUISITION BY GOOGLE**

[Google](http://en.wikipedia.org/wiki/Google) acquired Android Inc. on August 17, 2005, making Android Inc. a wholly owned subsidiary of Google Inc. Key employees of Android, including Andy Rubin, Rich Miner and Chris White, stayed at the company after the acquisition. Not much was known about Android Inc. at the time of the acquisition, but many assumed that Google was planning to enter the [mobile phone](http://en.wikipedia.org/wiki/Mobile_phone) market with this move. Google purchased android in 1 million US dollars.

## DESIGN android layer

[http://bits.wikimedia.org/skins-1.18/common/images/magnify-clip.png](http://en.wikipedia.org/wiki/File:System-architecture.jpg) *Fig 3.1 Android System Design*

**Architecture diagram**

Android consists of a [kernel](http://en.wikipedia.org/wiki/Kernel_(software)) based on the [Linux kernel](http://en.wikipedia.org/wiki/Linux_kernel), with [middleware](http://en.wikipedia.org/wiki/Middleware), libraries and [APIs](http://en.wikipedia.org/wiki/Application_programming_interface) written in [C](http://en.wikipedia.org/wiki/C_(programming_language)) language and [application software](http://en.wikipedia.org/wiki/Application_software) running on an [application framework](http://en.wikipedia.org/wiki/Application_framework) which includes Java-compatible libraries based on [Apache Harmony](http://en.wikipedia.org/wiki/Apache_Harmony). Android uses the [Dalvik virtual machine](http://en.wikipedia.org/wiki/Dalvik_virtual_machine" \o "Dalvik virtual machine) with [just-in-time compilation](http://en.wikipedia.org/wiki/Just-in-time_compilation) to run Dalvik dex-code (Dalvik Executable), which is usually translated from [Java](http://en.wikipedia.org/wiki/Java_(programming_language)) bytecode.

### LINUX

Android's kernel is based on the [Linux kernel](http://en.wikipedia.org/wiki/Linux_kernel) and has further architecture changes by Google outside the typical Linux kernel development cycle. Android does not have a native [X Window System](http://en.wikipedia.org/wiki/X_Window_System) nor does it support the full set of standard [GNU](http://en.wikipedia.org/wiki/GNU) libraries, and this makes it difficult to port existing Linux applications or libraries to Android.

**CURRENT FEATURES AND SPECIFICATIONS**

* + **Handset layouts**

The platform is adaptable to larger, [VGA](http://en.wikipedia.org/wiki/Video_Graphics_Array), [2D graphics](http://en.wikipedia.org/wiki/2D_computer_graphics) library, [3D graphics](http://en.wikipedia.org/wiki/3D_computer_graphics) library based on [OpenGL ES](http://en.wikipedia.org/wiki/OpenGL_ES) 2.0 specifications, and traditional Smartphone layouts.

* + **Storage**

[SQLite](http://en.wikipedia.org/wiki/SQLite), a lightweight [relational database](http://en.wikipedia.org/wiki/Relational_database), is used for [data](http://en.wikipedia.org/wiki/Data) storage purposes.

* + **Connectivity**

Android supports connectivity technologies including [GSM](http://en.wikipedia.org/wiki/GSM)/[EDGE](http://en.wikipedia.org/wiki/Enhanced_Data_Rates_for_GSM_Evolution), [IDEN](http://en.wikipedia.org/wiki/Integrated_Digital_Enhanced_Network), [CDMA](http://en.wikipedia.org/wiki/Code_division_multiple_access), [EV-O](http://en.wikipedia.org/wiki/Evolution-Data_Optimized), [UMTS](http://en.wikipedia.org/wiki/Universal_Mobile_Telecommunications_System), [Bluetooth](http://en.wikipedia.org/wiki/Bluetooth), [Wi-Fi](http://en.wikipedia.org/wiki/Wi-Fi), [LTE](http://en.wikipedia.org/wiki/LTE_Advanced), [NFC](http://en.wikipedia.org/wiki/Near_field_communication) and [WiMAX](http://en.wikipedia.org/wiki/WiMAX).

* + **Messaging**

[SMS](http://en.wikipedia.org/wiki/SMS) and [MMS](http://en.wikipedia.org/wiki/Multimedia_Messaging_Service) are available forms of messaging, including threaded [text messaging](http://en.wikipedia.org/wiki/Text_messaging) and now [Android Cloud To Device Messaging](http://en.wikipedia.org/wiki/Android_Cloud_To_Device_Messaging) (C2DM) is also a part of Android Push Messaging service.

* + **Multiple language support**

Android supports multiple languages.

* + **Web browser**

The web browser available in Android is based on the open-source [Web Kit](http://en.wikipedia.org/wiki/WebKit) layout engine, coupled with [Chrome's](http://en.wikipedia.org/wiki/Google_Chrome) [V8 JavaScript engine](http://en.wikipedia.org/wiki/V8_JavaScript_engine). The browser scores 100/100 on the [Acid3](http://en.wikipedia.org/wiki/Acid3#Mobile_browsers) test on Android 4.0.

* + **Java support**

While most Android applications are written in [Java](http://en.wikipedia.org/wiki/Java_(programming_language)), there is no [Java Virtual Machine](http://en.wikipedia.org/wiki/Java_Virtual_Machine) in the platform and Java byte code is not executed. Java classes are compiled into Dalvik executables and run on [Dalvik](http://en.wikipedia.org/wiki/Dalvik_virtual_machine" \o "Dalvik virtual machine), a specialized virtual machine designed specifically for Android and optimized for battery-powered mobile devices with limited memory and CPU. [J2ME](http://en.wikipedia.org/wiki/J2ME) support can be provided via third-party applications.

* **Streaming media support**

RTP/RTSP streaming ([3GPP PSS](http://en.wikipedia.org/w/index.php?title=3GPP_PSS&action=edit&redlink=1), [ISMA](http://en.wikipedia.org/wiki/Internet_Streaming_Media_Alliance" \o "Internet Streaming Media Alliance)), HTML progressive download ([HTML5 <video> tag](http://en.wikipedia.org/wiki/HTML5_video)). Adobe Flash Streaming (RTMP) and HTTP Dynamic streaming are supported by the [Flash plug-in](http://en.wikipedia.org/wiki/Adobe_Flash_Player#Mobile_platforms). Apple HTTP Live Streaming is supported by [RealPlayer for Android](http://en.wikipedia.org/wiki/RealPlayer_for_Android), and by the operating system in Android 3.0 (Honeycomb).

* + **Multi-touch**

Android has native support for [multi-touch](http://en.wikipedia.org/wiki/Multi-touch) which was initially made available in handsets such as the [HTC Hero](http://en.wikipedia.org/wiki/HTC_Hero" \o "HTC Hero). The feature was originally disabled at the kernel level (possibly to avoid infringing Apple's patents on touch-screen technology at the time). Google has since released an update for the [Nexus One](http://en.wikipedia.org/wiki/Nexus_One) and the [Motorola Droid](http://en.wikipedia.org/wiki/Motorola_Droid) which enables multi-touch natively.

* + **Bluetooth**

Supports [A2DP](http://en.wikipedia.org/wiki/A2DP), [AVRCP](http://en.wikipedia.org/wiki/AVRCP" \o "AVRCP), sending files ([OPP](http://en.wikipedia.org/wiki/Object_Push_Profile" \o "Object Push Profile)), accessing the phone book ([PBAP](http://en.wikipedia.org/wiki/Bluetooth_profile" \l "Phone_Book_Access_Profile_.28PBAP.2C_PBA.29" \o "Bluetooth profile)), voice dialing and sending contacts between phones. Keyboard, mouse and joystick ([HID](http://en.wikipedia.org/wiki/Bluetooth_profile#Human_Interface_Device_Profile_.28HID.29)) support is available in Android 3.1+, and in earlier versions through manufacturer customizations and third-party applications.

* + **Video calling**

Android does not support native video calling, but some handsets have a customized version of the operating system that supports it, either via the [UMTS](http://en.wikipedia.org/wiki/UMTS" \o "UMTS) network (like the [Samsung Galaxy S](http://en.wikipedia.org/wiki/Samsung_Galaxy_S)) or over IP. Video calling through Google Talk is available in Android 2.3.4 and later. Gingerbread allows [Nexus S](http://en.wikipedia.org/wiki/Nexus_S) to place Internet calls with a SIP account. This allows for enhanced VoIP dialing to other SIP accounts and even phone numbers. Skype 2.1 offers video calling in Android 2.3, including front camera support.

* + **Multitasking**

Multitasking of applications is available.

* + **Voice based features**

Google search through voice has been available since initial release. Voice actions for calling, texting, navigation, etc. are supported on Android 2.2 onwards.

* + **Tethering**

Android supports [tethering](http://en.wikipedia.org/wiki/Tethering), which allows a phone to be used as a wireless/wired [Wi-Fi hotspot](http://en.wikipedia.org/wiki/Wi-Fi_hotspot). Before Android 2.2 this was supported by third-party applications or manufacturer customizations.

* + **Screen capture**

Android supports capturing a [screenshot](http://en.wikipedia.org/wiki/Screenshot) by pressing the power and volume-down buttons at the same time. Prior to Android 4.0, the only methods of capturing a screenshot were through manufacturer and third-party customizations or otherwise by using a PC connection (DDMS developer's tool). These alternative methods are still available with the latest Android.

* + **External storage**

Most Android devices include microSD slot and can read microSD cards formatted with [FAT32](http://en.wikipedia.org/wiki/FAT32), [Ext3fs](http://en.wikipedia.org/wiki/Ext3fs) or [Ext4fs](http://en.wikipedia.org/wiki/Ext4fs) file system. To allow use of high-capacity storage media such as [USB flash drives](http://en.wikipedia.org/wiki/USB_flash_drive) and [USB HDDs](http://en.wikipedia.org/wiki/USB_HDD), many Android tablets also include [USB](http://en.wikipedia.org/wiki/USB) 'A' receptacle. Storage formatted with [FAT32](http://en.wikipedia.org/wiki/FAT32) is handled by [Linux Kernel](http://en.wikipedia.org/wiki/Linux_Kernel) VFAT driver, while 3rd party solutions are required to handle other popular file systems such as [NTFS](http://en.wikipedia.org/wiki/NTFS" \o "NTFS), [HFS Plus](http://en.wikipedia.org/wiki/HFS%2B" \o "HFS+) and [exFAT](http://en.wikipedia.org/wiki/ExFAT" \o "ExFAT).

**USES**

While Android is designed primarily for smartphones and tablets, the open and customizable nature of the operating system allows it to be used on other electronics, including [laptops](http://en.wikipedia.org/wiki/Laptop) and [netbooks](http://en.wikipedia.org/wiki/Netbook), [smartbooks](http://en.wikipedia.org/wiki/Smartbook" \o "Smartbook), and [ebook readers](http://en.wikipedia.org/wiki/Ebook_reader" \o "Ebook reader). Further, Google intends to bring Android to televisions with [Google TV](http://en.wikipedia.org/wiki/Google_TV), and the OS has seen niche applications on [wristwatches](http://en.wikipedia.org/wiki/Wristwatch), [headphones](http://en.wikipedia.org/wiki/Headphones), car CD and DVD players, [digital cameras](http://en.wikipedia.org/wiki/Digital_cameras), [portable media players](http://en.wikipedia.org/wiki/Portable_media_player) and [landlines](http://en.wikipedia.org/wiki/Landlines).

The first commercially available phone to run Android was the [HTC Dream](http://en.wikipedia.org/wiki/HTC_Dream" \o "HTC Dream), released on 22 October 2008. In early 2010 Google collaborated with [HTC](http://en.wikipedia.org/wiki/HTC" \o "HTC) to launch its flagship Android device, the [Nexus One](http://en.wikipedia.org/wiki/Nexus_One). This was followed later in 2010 with the [Samsung](http://en.wikipedia.org/wiki/Samsung)-made [Nexus S](http://en.wikipedia.org/wiki/Nexus_S) and in 2011 with the [Galaxy Nexus](http://en.wikipedia.org/wiki/Galaxy_Nexus).

At present, Android mobile devices have undertaken the market by outrunning Apple’s iOS and Nokia’s Symbian and Windows mobiles. This is because of the collaboration with Samsung and providing higher level smart phones at lower prices.

**CHAPTER 4**

**DETAILED DESIGN AND TEST PLAN**

**4.1 DECOMPOSITION DESCRIPTION**

There are various components and modules present in our project. Everything in the project can’t be done at the same time; hence we had decomposed our project in to some of the modules. Decomposition gives us an easier way to solve any type of big problems. It has a power to convert very huge things in to number of smaller ones. Therefore the work stress will be reduced and much possibility to understand the project very easily.

**4.1.1 Module Decomposition**

Modules in the sense it is the part of the project. Module decomposition is that dividing the work in to smaller parts such a way that to make our work easier to understand and face the problems that occurs while handling the project work. So based up on the project the module description and decomposition had been done.

**4.2 DEPENDENCY DESCRIPTION**

**4.2.1 Inter Module Dependency**

The application is divided into three modules namely:

* 1. Getting Voice Input
  2. Translation
  3. Translated Voice Output

**4.2.2 Data Dependency**

The data which we are using in our project will depend on user’s voice input. The Google Voice recognizer helps to recognize the input from the user and displays the list of suggestions which enables user’s to select the preferred word for accuracy. At noisy places, voice recognizer may lack in its accuracy. At such times user can also type the required word to be translated.

**4.3 DETAILED DESIGN**

In our project the modules had been designed based up on its work. The design of our project is to develop an application for voice translation.

**Module 1: Getting Voice Input**

In this module, the application gets the user input using the Google’s Voice Recognizer which starts when the user clicks the ‘Click Me’ button in the application. Then user speaks out and the voice recognizer recognizes the voice input and displays a list of words as suggestions.

**Algorithm for Module 1:**

* Select ‘From’ language in the dropdown list box.
* Select ‘To’ language in the dropdown list box.
* Press ‘Click Me’ button to give voice input.
* After providing the voice input, select the appropriate word from the list of suggestions.
* The input can be edited by entering the text manually.
* For foreign languages the input text can be in the form of sentences but for regional languages the input must be only in the form of words.

**Module 2: Translation**

This module is designed for the working of the Bing translator for foreign languages and the Google’s Database App Engine for regional Indian languages. After the ‘Click to Translate’ button is clicked, the respective translator function works based on the language preferred.

**Algorithm for Module 2:**

* Click the ‘Click to Translate’ button.
* If the selected ‘To’ language is a foreign language then, the available library functions in the ‘Bing translator’ is used to translate the given input text.
* If the selected ‘To’ language is a regional Indian language then, the Google app-engine will be used to translate the input.

**Module 3: Translated Voice Output**

After the translation module ends, the translated word is displayed on the screen. A ‘Speak Out’ button is provided to enable user to hear the translated word when the button is clicked.

**Algorithm for Module 3:**

* After the translated text appears on the screen, click on the ‘Speak out’ button to listen to the translated text.
* If the same input text has to be translated to a different language then just change the ‘To’ language and click ‘Click to Translate’ and then ‘Speak out’.
* Click ‘History’ button to view the recently used words for translation.

**4.4 OOAD DIAGRAMS**

The typical object oriented analysis and design diagrams for the project description can be shown according to the work we have done on the project. The OOAD diagrams consist of the following categories.

**4.4.1 Use-case Diagram**

A use case diagram is a graph of actors, a set of use cases enclosed by a system boundary, a communication association between the actors and the use cases, and a generalization among use cases.



*Fig 4.1 Use Case Diagram*

Figure 4.1 represents the use case diagram. It describes the interaction between the users-query images and the test clip in order to perform the processing through the algorithm. After the processing of the algorithm the output will be received as the match.

**4.4.2 Activity diagram**

An activity diagram is a variation or special case of a state machine, in which the states are the activities representing the performance of operations and the transitions are triggered by the completion of operation.



*Fig 4.2 Activity Diagram*

**4.4.3 Sequence Diagram**



*Fig 4.3: Sequence Diagram*

**4.4.4 State Chart Diagram**

State chart diagram describes the flow of control from one state to another state. States are defined as a condition in which an object exists and it changes when some event is triggered. So the most important purpose of State chart diagram is to model life time of an object from creation to termination.



*Fig 4.4: State Chart Diagram*

**4.5 TEST PLAN**

**TESTING:**

The project is tested to verify its correctness and efficiency. The test plan includes following test cases:

|  |  |  |
| --- | --- | --- |
| **Test Case** | **Expected Output** | **Actual Output** |
| Mobile phone is not connected to the Internet. | Voice input cannot be recognized. | Can’t reach Google at this moment. |
| Mobile phone is connected to the Internet. | Voice input is recognized. | List of suggestions corresponding to the voice input is listed. |
| Changing input manually. | The input gets edited manually. | Input gets edited. |
| Wrong or mispronounced voice input. | Does not recognize. | Displays list of suggestion based on the input. |
| Unavailability of the required font. | Directed to download the font. | A page where link to Play Store will be given. |

**UNIT TESTING:**

**Unit testing** is a method by which individual units of [source code](http://en.wikipedia.org/wiki/Source_code), sets of one or more computer program modules together with associated control data, usage procedures, and operating procedures, are tested to determine if they are fit for use.

In this project, the unit testing can be carried out as follows:

**Providing valid input:**

After selecting the ‘From’ language from the dropdown list box, if the user gives the input in the specified language, the Google voice recognizer will display a list of suggestions related to the input word.

The user can select the correct word from the list of suggestions.

Eg: If the ‘From’ language is English and if the user gives ‘Hello’ then the text will be translated.

**Providing invalid input:**

After selecting the ‘From’ language from the dropdown list box, if the user gives the input in different language, the Google voice recognizer will display an error message.

The user needs to speak again.

Eg: If the ‘From’ language is English and if the user gives ‘Bonjour’ then an error message will be displayed.

**Translation:**

If the output in the desired language after the translation is obtained then it is a valid output.

**Voice output:**

The voice output will be valid if the word is pronounced correctly.

**CHAPTER 5**

**SYSTEM IMPLEMENTATION**

**5.1 IMPLEMENTATION**

The implementation involves implementation of three main phases, Getting Voice Input, Translation and Translated Voice Output.

Implementation is the stage of the project when the theoretical design is turned out into working system. Voice Input is recognized using Google’s Voice recognizer.

Translation is done using Bing Translator for foreign languages and Google’s Database App Engine for regional Indian languages.

Translated words can be heard by the user in their respective android phones.

There is an added advantage where the translated words will be stored in History and these words can be reviewed for later use.

**ALGORITHM:**

**STEP 1:** Run the application.

**STEP 2:** Select ‘From’ language in the dropdown list box.

**STEP 3:** Select ‘To’ language in the dropdown list box.

**STEP 4:** Press ‘Click Me’ button to give voice input.

**STEP 5:** After providing the voice input, select the appropriate word from the list of suggestions.

**STEP 6:** The input can be edited by entering the text manually.

**STEP 7:** For foreign languages the input text can be in the form of sentences but for regional languages the input must be only in the form of words.

**STEP 8:** Click the ‘Click to Translate’ button.

**STEP 9:** If the selected ‘To’ language is a foreign language then, the available library functions in the ‘Bing translator’ is used to translate the given input text.

**STEP 10:** If the selected ‘To’ language is a regional Indian language then, the Google app-engine will be used to translate the input.

**STEP 11:** After the translated text appears on the screen, click on the ‘Speak out’ button to listen to the translated text.

**STEP 12:** If the same input text has to be translated to a different language then just change the ‘To’ language and click ‘Click to Translate’ and then ‘Speak out’.

**STEP 13:** Click ‘History’ button to view the recently used words for translation.

**STEP 14:** Exit the application.

**CHAPTER 6**

**CONCLUSION AND FUTURE WORK**

**6.1 CONCLUSION**

The conclusion of this project report is that, we would like to make everybody to understand a language that they don’t know. Moreover the addition of regional languages such as Tamil and Hindi will serve as an added advantage.

However there is a limitation that, for regional languages the translation cannot be done in the form of sentences. Only word translation can be done.

**6.2 FUTURE WORK**

The limitations in this application and some more enhancements can be done in the future works. They are:

* Enable the sentence translation for regional Indian languages by using the strategy of splitting the words, identifying the meanings and then merging the words back.
* Add the option of sending text messages of the translated word directly to a friend.
* Addition of some more regional languages can be done.
* Enhance the data in the Google app-engine.
* Use Natural Language Processing (NLP) and phonetics to increase the accuracy in the pronunciation of regional languages.
* This can also help in providing the voice input for regional languages.

These are the few enhancements that can be carried out in the future.

**APPENDIX A**

**SAMPLE CODES**

**HOME SCREEN:**

**Homescreen.java:**

**package** com.example.voicerecognition;

**import** android.app.Activity;

**import** android.content.Intent;

**import** android.os.Bundle;

**public** **class** Homescreen **extends** Activity{

@Override

**protected** **void** onCreate(Bundle savedInstanceState) {

// **TODO** Auto-generated method stub

**super**.onCreate(savedInstanceState);

setContentView(R.layout.*home\_screen*);

Thread timer=**new** Thread(){

**public** **void** run() {

**try** {

*sleep*(5000);

} **catch**(InterruptedException e){

e.printStackTrace();

}**finally** {

Intent openHome = **new** Intent("com.example.voicerecognition.MainActivity");

startActivity(openHome);

}

}

};

timer.start();

}

}

**home\_screen.xml :**

<?xml version=*"1.0"* encoding=*"utf-8"*?>

<LinearLayout xmlns:android=*"http://schemas.android.com/apk/res/android"*

android:layout\_width=*"match\_parent"*

android:layout\_height=*"match\_parent"*

android:orientation=*"vertical"*

android:background=*"@drawable/bg"*

>

</LinearLayout>

**MainActivity.java:**

package com.example.voicerecognition;

import java.io.File;

import java.io.FileOutputStream;

import java.util.ArrayList;

import java.util.List;

import java.util.Locale;

import java.util.Scanner;

import org.apache.http.HttpResponse;

import org.apache.http.NameValuePair;

import org.apache.http.client.entity.UrlEncodedFormEntity;

import org.apache.http.client.methods.HttpPost;

import org.apache.http.impl.client.BasicResponseHandler;

import org.apache.http.impl.client.DefaultHttpClient;

import org.apache.http.message.BasicNameValuePair;

import org.apache.http.protocol.HTTP;

import android.app.Activity;

import android.app.AlertDialog;

import android.app.Dialog;

import android.content.Context;

import android.content.Intent;

import android.content.pm.PackageManager;

import android.content.pm.ResolveInfo;

import android.os.AsyncTask;

import android.os.Bundle;

import android.speech.RecognizerIntent;

import android.speech.tts.TextToSpeech;

import android.util.Log;

import android.view.LayoutInflater;

import android.view.View;

import android.view.View.OnClickListener;

import android.view.View.OnLongClickListener;

import android.widget.AdapterView;

import android.widget.AdapterView.OnItemClickListener;

import android.widget.AdapterView.OnItemSelectedListener;

import android.widget.ArrayAdapter;

import android.widget.Button;

import android.widget.EditText;

import android.widget.LinearLayout;

import android.widget.ListView;

import android.widget.Spinner;

import android.widget.TextView;

import android.widget.Toast;

public class MainActivity extends Activity implements

TextToSpeech.OnInitListener {

public static final String TAG = "TTS";

private static final int REQUEST\_CODE = 1234;

private static final int MY\_DATA\_CHECK\_CODE = 4321;

private static String source, target;

private static String[] sourceLangCodes, targetLangCodes;

private static String text;

private static String translatedText = "";

private static Spinner sourceSpinner, targetSpinner;

private ArrayAdapter<CharSequence> sourceAdapter, targetAdapter;

public static TextToSpeech tts;

private static ListView dialog\_list;

private static TextView output;

private static EditText input\_txt;

private static Button speakButton;

private static Button speakOutButton;

private static Button translateButton;

private static Button historyButton;

/\*\*

\* Called with the activity is first created.

\*/

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.voice\_recog);

speakButton = (Button) findViewById(R.id.speakButton);

speakOutButton = (Button) findViewById(R.id.speakOutButton);

input\_txt = (EditText) findViewById(R.id.input\_txt\_box);

translateButton = (Button) findViewById(R.id.translate\_btn);

historyButton = (Button) findViewById(R.id.historyButton);

output = (TextView) findViewById(R.id.translated\_text);

source = "English";

target = "French";

sourceLangCodes = getResources().getStringArray(R.array.input\_language\_code);

targetLangCodes = getResources().getStringArray(R.array.output\_language\_code);

initSpinners();

// Disable button if no recognition service is present

PackageManager pm = getPackageManager();

List<ResolveInfo> activities = pm.queryIntentActivities(new Intent(

RecognizerIntent.ACTION\_RECOGNIZE\_SPEECH), 0);

if (activities.size() == 0) {

speakButton.setEnabled(false);

speakButton.setText("Recognizer not present");

}

speakButton.setOnClickListener(new OnClickListener() {

@Override

public void onClick(View v) {

startVoiceRecognitionActivity();

}

});

translateButton.setOnClickListener(new OnClickListener() {

@Override

public void onClick(View v) {

String text = input\_txt.getText().toString();

File History = getApplicationContext().getDir("History", Context.MODE\_APPEND);

boolean newEntry = true;

try{

Scanner in = new Scanner(new File(History,"history.txt"));

while(in.hasNext()){

String temp = in.nextLine();

if(temp.equalsIgnoreCase(text)) {

newEntry = false;

}

}

if(newEntry){

File history = new File(History, "history.txt");

FileOutputStream fos = new FileOutputStream(history, true);

String content = text + "\n";

fos.write(content.getBytes());

fos.close();

}

}catch(Exception e) {

e.printStackTrace();

}

if(target.equalsIgnoreCase("tamil") || target.equalsIgnoreCase("hindi")) {

TranslatorBing.translate(text, source, target, output, true);

Log.d("TTS", "Translation over...");

Log.d(TAG, "Starting TTS event...");

Intent checkIntent = new Intent();

checkIntent.setAction(TextToSpeech.Engine.ACTION\_CHECK\_TTS\_DATA);

startActivityForResult(checkIntent, MY\_DATA\_CHECK\_CODE);

} else {

TranslatorBing.translate(text, source, target, output, false);

Log.d("TTS", "Translation over...");

Log.d(TAG, "Starting TTS event...");

Intent checkIntent = new Intent();

checkIntent.setAction(TextToSpeech.Engine.ACTION\_CHECK\_TTS\_DATA);

startActivityForResult(checkIntent, MY\_DATA\_CHECK\_CODE);

}

}

});

historyButton.setOnClickListener(new OnClickListener() {

@Override

public void onClick(View v) {

// TODO Auto-generated method stub

removeDialog(1);

showDialog(1);

}

});

historyButton.setOnLongClickListener(new OnLongClickListener() {

@Override

public boolean onLongClick(View v) {

// TODO Auto-generated method stub

File History = getApplicationContext().getDir("History", Context.MODE\_PRIVATE);

File history = new File(History, "history.txt");

String emptystring = "";

history.delete();

try {

FileOutputStream fos = new FileOutputStream(history, true);

fos.write(emptystring.getBytes());

fos.close();

}catch(Exception e) {

e.printStackTrace();

}

Toast.makeText(getApplicationContext(), "History deleted...", Toast.LENGTH\_SHORT).show();

return false;

}

});

speakOutButton.setOnClickListener(new OnClickListener() {

@Override

public void onClick(View v) {

if(init)

{

Log.d("TTS","Speak out clicked...");

Log.d("TTS",translatedText);

translatedText = output.getText().toString();

Log.d("TTS",translatedText);

tts.speak(translatedText, TextToSpeech.QUEUE\_FLUSH, null);

}

}

});

}

public void initSpinners() {

Log.d(TAG, "Initiallizing spinners");

sourceSpinner = (Spinner)findViewById(R.id.spinner1);

sourceAdapter = ArrayAdapter.createFromResource(MainActivity.this, R.array.languages\_input, android.R.layout.simple\_spinner\_item);

sourceAdapter.setDropDownViewResource(android.R.layout.simple\_spinner\_dropdown\_item);

sourceSpinner.setAdapter(sourceAdapter);

sourceSpinner.setOnItemSelectedListener(new SourceLanguageOnItemSelectedListener());

sourceSpinner.setSelection(sourceAdapter.getPosition(source));

targetSpinner = (Spinner)findViewById(R.id.spinner2);

targetAdapter = ArrayAdapter.createFromResource(MainActivity.this, R.array.languages\_output, android.R.layout.simple\_spinner\_item);

targetAdapter.setDropDownViewResource(android.R.layout.simple\_spinner\_dropdown\_item);

targetSpinner.setAdapter(targetAdapter);

targetSpinner.setOnItemSelectedListener(new TargetLanguageOnItemSelectedListener());

targetSpinner.setSelection(targetAdapter.getPosition(target));

Log.d(TAG, "Spinner initialized");

}

/\*\*

\* Fire an intent to start the voice recognition activity.

\*/

private void startVoiceRecognitionActivity() {

Intent intent = new Intent(RecognizerIntent.ACTION\_RECOGNIZE\_SPEECH);

String sourceCode = sourceLangCodes[sourceSpinner.getSelectedItemPosition()];

intent.putExtra(RecognizerIntent.EXTRA\_LANGUAGE, sourceCode);

intent.putExtra(RecognizerIntent.EXTRA\_PROMPT,

"Voice recognition Demo...");

startActivityForResult(intent, REQUEST\_CODE);

}

/\*\*

\* Handle the results from the voice recognition activity.

\*/

@Override

protected void onActivityResult(int requestCode, int resultCode, Intent data) {

if (requestCode == MY\_DATA\_CHECK\_CODE) {

if (resultCode == TextToSpeech.Engine.CHECK\_VOICE\_DATA\_PASS) {

// success, create the TTS instance

tts = new TextToSpeech(this, MainActivity.this);

} else {

// missing data, install it

Intent installIntent = new Intent();

installIntent.setAction(TextToSpeech.Engine.ACTION\_INSTALL\_TTS\_DATA);

startActivity(installIntent);

}

}

if (requestCode == REQUEST\_CODE && resultCode == RESULT\_OK) {

// Populate the wordsList with the String values the recognition

// engine thought it heard

matches = data.getStringArrayListExtra(RecognizerIntent.EXTRA\_RESULTS);

removeDialog(0);

showDialog(0);

}

super.onActivityResult(requestCode, resultCode, data);

}

public ArrayList<String> matches;

@Override

protected Dialog onCreateDialog(int id) {

// Get the layout inflater

AlertDialog.Builder builder = new AlertDialog.Builder(this);

LayoutInflater inflater = getLayoutInflater();

LinearLayout layout = new LinearLayout(this);

LinearLayout.LayoutParams lp = new LinearLayout.LayoutParams(

LinearLayout.LayoutParams.MATCH\_PARENT,

LinearLayout.LayoutParams.WRAP\_CONTENT);

layout.setLayoutParams(lp);

layout.setOrientation(LinearLayout.VERTICAL);

switch(id) {

case 0:

dialog\_list = new ListView(this);

dialog\_list.setAdapter(new ArrayAdapter<String>(this,android.R.layout.simple\_list\_item\_1, matches));

dialog\_list.setOnItemClickListener(new OnItemClickListener() {

@Override

public void onItemClick(AdapterView<?> parent, View view,

int position, long id) {

text = ((TextView) view).getText().toString();

input\_txt.setText(text);

removeDialog(0);

}

});

layout.addView(dialog\_list);

builder.setView(layout);

break;

case 1:

try{

matches.clear();

File History = getApplicationContext().getDir("History", Context.MODE\_APPEND);

Scanner in = new Scanner(new File(History,"history.txt"));

while(in.hasNext()){

String temp = in.nextLine();

matches.add(temp);

}

}catch(Exception e){

e.printStackTrace();

}

dialog\_list = new ListView(this);

dialog\_list.setAdapter(new ArrayAdapter<String>(this,android.R.layout.simple\_list\_item\_1, matches));

dialog\_list.setOnItemClickListener(new OnItemClickListener() {

@Override

public void onItemClick(AdapterView<?> parent, View view,

int position, long id) {

text = ((TextView) view).getText().toString();

input\_txt.setText(text);

removeDialog(1);

}

});

layout.addView(dialog\_list);

builder.setView(layout);

}

return builder.create();

}

boolean init = false;

@Override

public void onInit(int status) {

// TODO Auto-generated method stub

int result;

if (status == TextToSpeech.SUCCESS) {

result = -1;

String targetLanguage = targetSpinner.getSelectedItem().toString();

String targetCode = targetLangCodes[targetSpinner.getSelectedItemPosition()];

Locale locale = new Locale(targetCode);

result = tts.setLanguage(locale);

Log.d(TAG, "target = " +targetCode);

Toast.makeText(MainActivity.this, "TTS language set to : " + target +", code: " +targetCode, Toast.LENGTH\_SHORT).show();

if (result == TextToSpeech.LANG\_MISSING\_DATA

|| result == TextToSpeech.LANG\_NOT\_SUPPORTED) {

Log.e("TTS", "This Language is not supported");

Toast.makeText(getApplicationContext(), "Language not supported", Toast.LENGTH\_SHORT).show();

Intent installIntent = new Intent();

installIntent.setAction(TextToSpeech.Engine.ACTION\_INSTALL\_TTS\_DATA);

startActivity(installIntent);

} else {

Log.d("TTS", "init set true...");

speakOutButton.setVisibility(View.VISIBLE);

speakOutButton.setEnabled(true);

init = true;

// speakOut();

}

} else {

Log.e("TTS", "Initilization Failed!");

}

}

public class SourceLanguageOnItemSelectedListener implements OnItemSelectedListener {

@Override

public void onItemSelected(AdapterView<?> parent, View view, int pos, long id) {

source = (String) sourceSpinner.getItemAtPosition(pos);

Log.d(TAG, "Source language selected as :" +source);

Toast.makeText(MainActivity.this, "Voice input language: " +source, Toast.LENGTH\_SHORT).show();

}

@Override

public void onNothingSelected(AdapterView<?> arg0) {

// Do nothing

}

}

public class TargetLanguageOnItemSelectedListener implements OnItemSelectedListener {

@Override

public void onItemSelected(AdapterView<?> parent, View view, int pos, long id) {

// Save the target language preference

target = (String) targetSpinner.getItemAtPosition(pos);

}

@Override

public void onNothingSelected(AdapterView<?> arg0) {

// Do nothing

}

}

private class GetTamilTranslation extends AsyncTask<String, Void, String> {

@Override

protected String doInBackground(String... urls) {

String response = "";

for (String url : urls) {

DefaultHttpClient client = new DefaultHttpClient();

HttpPost post = new HttpPost(url);

try {

Log.d(TAG, "Starting DbSearch...");

List<NameValuePair> form=new ArrayList<NameValuePair>();

form.add(new BasicNameValuePair("english", input\_txt.getText().toString().toLowerCase()));

form.add(new BasicNameValuePair("language", target));

post.setEntity(new UrlEncodedFormEntity(form, HTTP.UTF\_8));

HttpResponse resp = client.execute(post);

String data = new BasicResponseHandler().handleResponse(resp);

response = data;

} catch (Exception e) {

e.printStackTrace();

Log.d(TAG, e.toString());

response = "unable\_to\_download";

}

}

return response;

}

@Override

protected void onPostExecute(String result) {

output.setText(result);

}

}

}

**MainActivity Manifest:**

**(AndroidManifest.xml)**

<?xml version=*"1.0"* encoding=*"utf-8"*?>

<manifest xmlns:android=*"http://schemas.android.com/apk/res/android"*

package=*"com.example.voicerecognition"*

android:versionCode=*"1"*

android:versionName=*"1.0"* >

<uses-sdk

android:minSdkVersion=*"14"*

android:targetSdkVersion=*"17"* />

<uses-permission android:name=*"android.permission.INTERNET"*/>

<application

android:allowBackup=*"true"*

android:icon=*"@drawable/ic\_launcher"*

android:label=*"@string/app\_name"*

android:theme=*"@style/AppTheme"* >

<activity

android:name=*"com.example.voicerecognition.MainActivity"*

android:label=*"@string/app\_name"*

android:screenOrientation=*"portrait"* >

<intent-filter>

<action android:name=*"com.example.voicerecognition.MainActivity"* />

<category android:name=*"android.intent.category.DEFAULT"* />

</intent-filter>

</activity>

<activity

android:name=*"com.example.voicerecognition.Homescreen"*

android:screenOrientation=*"portrait"*>

<intent-filter>

<action android:name=*"android.intent.action.MAIN"* />

<category android:name=*"android.intent.category.LAUNCHER"* />

</intent-filter>

</activity>

</application>

</manifest>

**voice\_recog.xml:**

<?xml version=*"1.0"* encoding=*"utf-8"*?>

<LinearLayout xmlns:android=*"http://schemas.android.com/apk/res/android"*

android:layout\_width=*"fill\_parent"*

android:layout\_height=*"fill\_parent"*

android:orientation=*"vertical"*

>

<LinearLayout

android:layout\_width=*"match\_parent"*

android:layout\_height=*"wrap\_content"*

android:orientation=*"horizontal"*

android:weightSum=*"2"* >

<TextView

android:id=*"@+id/textView1"*

android:layout\_width=*"0dp"*

android:layout\_height=*"wrap\_content"*

android:layout\_weight=*"1"*

android:textSize=*"35sp"*

android:text=*"@string/from"*

/>

<Spinner

android:id=*"@+id/spinner1"*

android:layout\_width=*"0dip"*

android:layout\_height=*"wrap\_content"*

android:layout\_weight=*"1"* />

</LinearLayout>

<LinearLayout

android:layout\_width=*"match\_parent"*

android:layout\_height=*"wrap\_content"*

android:orientation=*"horizontal"*

android:weightSum=*"2"* >

<TextView

android:id=*"@+id/textView2"*

android:layout\_width=*"0dp"*

android:layout\_height=*"wrap\_content"*

android:layout\_weight=*"1"*

android:textSize=*"35sp"*

android:text=*"@string/to"* />

<Spinner

android:id=*"@+id/spinner2"*

android:layout\_width=*"0dip"*

android:layout\_height=*"wrap\_content"*

android:layout\_weight=*"1"* />

</LinearLayout>

<TextView

android:layout\_width=*"fill\_parent"*

android:layout\_height=*"wrap\_content"*

android:paddingBottom=*"4dip"*

android:text=*"@string/click"* />

<Button android:id=*"@+id/speakButton"*

android:layout\_width=*"fill\_parent"*

android:layout\_height=*"wrap\_content"*

android:text=*"Click Me!"* />

<EditText

android:id=*"@+id/input\_txt\_box"*

android:layout\_width=*"match\_parent"*

android:layout\_height=*"wrap\_content"*

android:text=*"Text to be translated."* />

<Button

android:id=*"@+id/translate\_btn"*

android:layout\_width=*"match\_parent"*

android:layout\_height=*"wrap\_content"*

android:text=*"Click to translate"* />

<TextView

android:id=*"@+id/translated\_text"*

android:gravity=*"center"*

android:layout\_gravity=*"center"*

android:layout\_width=*"match\_parent"*

android:textIsSelectable=*"true"*

android:layout\_height=*"wrap\_content"* />

<Button

android:id=*"@+id/speakOutButton"*

android:layout\_width=*"match\_parent"*

android:layout\_height=*"wrap\_content"*

android:enabled=*"false"*

android:visibility=*"invisible"*

android:text=*"Speak Out"* />

</LinearLayout>

**TranslatorBingAsynTask.java:**

**package** com.example.voicerecognition;

**import** java.util.ArrayList;

**import** java.util.List;

**import** org.apache.http.HttpResponse;

**import** org.apache.http.NameValuePair;

**import** org.apache.http.client.entity.UrlEncodedFormEntity;

**import** org.apache.http.client.methods.HttpPost;

**import** org.apache.http.impl.client.BasicResponseHandler;

**import** org.apache.http.impl.client.DefaultHttpClient;

**import** org.apache.http.message.BasicNameValuePair;

**import** org.apache.http.protocol.HTTP;

**import** android.graphics.Typeface;

**import** android.os.AsyncTask;

**import** android.util.Log;

**import** android.util.TypedValue;

**import** android.widget.TextView;

**import** com.memetix.mst.language.Language;

**import** com.memetix.mst.translate.Translate;

**public** **final** **class** TranslateBingAsyncTask **extends** AsyncTask<String, Void, Boolean> {

**private** **static** **final** String *API\_KEY* = "FE55328FE94D3809B4C6F458F1C5E4E655FE47FF";

**private** Language sourceLanguage;

**private** Language targetLanguage;

**private** String text;

**private** String translatedText;

**private** TextView textView;

**private** **boolean** indianLanguage;

**private** String targetIndianLanguage;

**public** TranslateBingAsyncTask(String text, Language sourceLanguage, Language targetLanguage, TextView textView, **boolean** indianLanguage, String targetIndianLanguage) {

**this**.sourceLanguage = sourceLanguage;

**this**.targetLanguage = targetLanguage;

**this**.text = text;

**this**.textView = textView;

**this**.indianLanguage = indianLanguage;

**this**.targetIndianLanguage = targetIndianLanguage;

}

@Override

**protected** **synchronized** **void** onPreExecute() {

**super**.onPreExecute();

textView.setTypeface(Typeface.*defaultFromStyle*(Typeface.*NORMAL*), Typeface.*NORMAL*);

textView.setTextSize(14);

textView.setText("Translating...");

}

@Override

**protected** **synchronized** Boolean doInBackground(String... arg0) {

Translate.*setKey*(*API\_KEY*);

**try** {

// Request translation

translatedText = Translate.*execute*(text, sourceLanguage, targetLanguage);

**if**(indianLanguage){

DefaultHttpClient client = **new** DefaultHttpClient();

HttpPost post = **new** HttpPost("http://speakindia-saveetha.appspot.com/WordSearch");

**try** {

List<NameValuePair> form=**new** ArrayList<NameValuePair>();

form.add(**new** BasicNameValuePair("english", translatedText.toLowerCase()));

form.add(**new** BasicNameValuePair("language", targetIndianLanguage));

post.setEntity(**new** UrlEncodedFormEntity(form, HTTP.*UTF\_8*));

HttpResponse resp = client.execute(post);

String data = **new** BasicResponseHandler().handleResponse(resp);

translatedText = data;

} **catch** (Exception e) {

e.printStackTrace();

translatedText = "unable\_to\_download";

}

}

} **catch** (Exception e) {

e.printStackTrace();

**return** **false**;

}

**return** **true**;

}

@Override

**protected** **synchronized** **void** onPostExecute(Boolean result) {

**super**.onPostExecute(result);

**if** (result) {

// Reset the text formatting

**if** (textView != **null**) {

textView.setTypeface(Typeface.*defaultFromStyle*(Typeface.*NORMAL*), Typeface.*NORMAL*);

}

// Put the translation into the textview

textView.setText(translatedText);

// Crudely scale betweeen 22 and 32 -- bigger font for shorter text

**int** scaledSize = Math.*max*(22, 32 - translatedText.length() / 4);

textView.setTextSize(TypedValue.*COMPLEX\_UNIT\_SP*, scaledSize);

} **else** {

textView.setTypeface(Typeface.*defaultFromStyle*(Typeface.*ITALIC*), Typeface.*ITALIC*);

textView.setText("Unavailable");

}

}

}

**TranslatorBing.java:**

/\*

\* Copyright (C) 2011 Robert Theis

\*

\* Licensed under the Apache License, Version 2.0 (the "License"); you may not

\* use this file except in compliance with the License. You may obtain a copy of

\* the License at

\*

\* http://www.apache.org/licenses/LICENSE-2.0

\*

\* Unless required by applicable law or agreed to in writing, software

\* distributed under the License is distributed on an "AS IS" BASIS, WITHOUT

\* WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the

\* License for the specific language governing permissions and limitations under

\* the License.

\*/

package com.example.voicerecognition;

import android.graphics.Typeface;

import android.widget.TextView;

import com.memetix.mst.language.Language;

/\*\*

\* Provides machine translations using the Microsoft Bing Translator API.

\*

\* @author Robert Theis

\*/

class TranslatorBing {

private TranslatorBing() {

// Private constructor to enforce noninstantiability

}

/\*\*

\* Check if the supplied language pair is valid for this translation service, and start an

\* AsyncTask to request the translation.

\*

\* @param text The text to translate

\* @param source The source language for translation

\* @param target The target language for translation

\* @param textView The text field that will receive the completed translation

\*/

static void translate(String text, String source, String target, TextView textView) {

if (source == null || target == null) {

throw new IllegalArgumentException();

}

Language sourceLanguage = null;

Language targetLanguage = null;

try {

sourceLanguage = toLanguage(source);

targetLanguage = toLanguage(target);

} catch (IllegalArgumentException e) {

textView.setTypeface(Typeface.defaultFromStyle(Typeface.ITALIC), Typeface.ITALIC);

textView.setTextSize(14);

textView.setText("Unsupported language pair");

}

if (sourceLanguage != null && targetLanguage != null) {

// Start an AsyncTask to perform the translation request.

new TranslateBingAsyncTask(text, sourceLanguage, targetLanguage, textView).execute();

}

}

/\*\*

\* Convert the given name of a natural language into a Language from the enum of Languages

\* supported by this translation service.

\*

\* @param languageName The name of the language, for example, "English"

\* @return The Language object representing this language

\* @throws IllegalArgumentException

\*/

private static Language toLanguage(String languageName) throws IllegalArgumentException {

// Convert string to all caps

String standardizedName = languageName.toUpperCase();

// Replace spaces with underscores

standardizedName = standardizedName.replace(' ', '\_');

// Remove parentheses

standardizedName = standardizedName.replace("(", "");

standardizedName = standardizedName.replace(")", "");

return Language.valueOf(standardizedName);

}

}

**R.java:**

/\* AUTO-GENERATED FILE. DO NOT MODIFY.

\*

\* This class was automatically generated by the

\* aapt tool from the resource data it found. It

\* should not be modified by hand.

\*/

**package** com.example.voicerecognition;

**public** **final** **class** R {

**public** **static** **final** **class** array {

**public** **static** **final** **int** *input\_language\_code*=0x7f040002;

**public** **static** **final** **int** *languages\_input*=0x7f040000;

**public** **static** **final** **int** *languages\_output*=0x7f040001;

**public** **static** **final** **int** *output\_language\_code*=0x7f040003;

}

**public** **static** **final** **class** attr {

}

**public** **static** **final** **class** dimen {

/\*\* Default screen margins, per the Android Design guidelines.

Customize dimensions originally defined in res/values/dimens.xml (such as

screen margins) for sw720dp devices (e.g. 10" tablets) in landscape here.

\*/

**public** **static** **final** **int** *activity\_horizontal\_margin*=0x7f050000;

**public** **static** **final** **int** *activity\_vertical\_margin*=0x7f050001;

}

**public** **static** **final** **class** drawable {

**public** **static** **final** **int** *bg*=0x7f020000;

**public** **static** **final** **int** *ic\_launcher*=0x7f020001;

**public** **static** **final** **int** *innercolor*=0x7f020002;

}

**public** **static** **final** **class** id {

**public** **static** **final** **int** *action\_settings*=0x7f09000b;

**public** **static** **final** **int** *dialog\_list*=0x7f090009;

**public** **static** **final** **int** *input\_txt\_box*=0x7f090005;

**public** **static** **final** **int** *sample*=0x7f09000a;

**public** **static** **final** **int** *speakButton*=0x7f090004;

**public** **static** **final** **int** *speakOutButton*=0x7f090008;

**public** **static** **final** **int** *spinner1*=0x7f090001;

**public** **static** **final** **int** *spinner2*=0x7f090003;

**public** **static** **final** **int** *textView1*=0x7f090000;

**public** **static** **final** **int** *textView2*=0x7f090002;

**public** **static** **final** **int** *translate\_btn*=0x7f090006;

**public** **static** **final** **int** *translated\_text*=0x7f090007;

}

**public** **static** **final** **class** layout {

**public** **static** **final** **int** *home\_screen*=0x7f030000;

**public** **static** **final** **int** *voice\_recog*=0x7f030001;

**public** **static** **final** **int** *wordlist*=0x7f030002;

}

**public** **static** **final** **class** menu {

**public** **static** **final** **int** *main*=0x7f080000;

}

**public** **static** **final** **class** string {

**public** **static** **final** **int** *action\_settings*=0x7f060001;

**public** **static** **final** **int** *app\_name*=0x7f060000;

**public** **static** **final** **int** *bar\_content\_type*=0x7f060003;

**public** **static** **final** **int** *button\_go*=0x7f060004;

**public** **static** **final** **int** *button\_switch*=0x7f060005;

**public** **static** **final** **int** *click*=0x7f06000f;

**public** **static** **final** **int** *from*=0x7f06000d;

**public** **static** **final** **int** *hello\_world*=0x7f060002;

**public** **static** **final** **int** *input\_hint*=0x7f060006;

**public** **static** **final** **int** *menu\_settings*=0x7f060007;

**public** **static** **final** **int** *progress\_indicator*=0x7f060008;

**public** **static** **final** **int** *service1*=0x7f060009;

**public** **static** **final** **int** *service2*=0x7f06000a;

**public** **static** **final** **int** *service3*=0x7f06000b;

**public** **static** **final** **int** *service4*=0x7f06000c;

**public** **static** **final** **int** *to*=0x7f06000e;

}

**public** **static** **final** **class** style {

/\*\*

Base application theme, dependent on API level. This theme is replaced

by AppBaseTheme from res/values-vXX/styles.xml on newer devices.

Theme customizations available in newer API levels can go in

res/values-vXX/styles.xml, while customizations related to

backward-compatibility can go here.

Base application theme for API 11+. This theme completely replaces

AppBaseTheme from res/values/styles.xml on API 11+ devices.

API 11 theme customizations can go here.

Base application theme for API 14+. This theme completely replaces

AppBaseTheme from BOTH res/values/styles.xml and

res/values-v11/styles.xml on API 14+ devices.

API 14 theme customizations can go here.

\*/

**public** **static** **final** **int** *AppBaseTheme*=0x7f070000;

/\*\* Application theme.

All customizations that are NOT specific to a particular API-level can go here.

\*/

**public** **static** **final** **int** *AppTheme*=0x7f070001;

}

}

**BuildConfig.java:**

/\*\* Automatically generated file. DO NOT MODIFY \*/

**package** com.example.voicerecognition;

**public** **final** **class** BuildConfig {

**public** **final** **static** **boolean** *DEBUG* = **true**;

}

**wordlist.xml:**

<?xml version=*"1.0"* encoding=*"utf-8"*?>

<LinearLayout xmlns:android=*"http://schemas.android.com/apk/res/android"*

android:orientation=*"vertical"*

android:layout\_width=*"match\_parent"*

android:layout\_height=*"wrap\_content"*

android:layout\_gravity=*"center"* >

<ListView

android:id=*"@+id/dialog\_list"*

android:layout\_width=*"fill\_parent"*

android:layout\_height=*"wrap\_content"* />

<TextView

android:id=*"@+id/sample"*

android:layout\_width=*"match\_parent"*

android:layout\_height=*"wrap\_content"*

android:text=*"sample"* />

</LinearLayout>

**APPENDIX B**

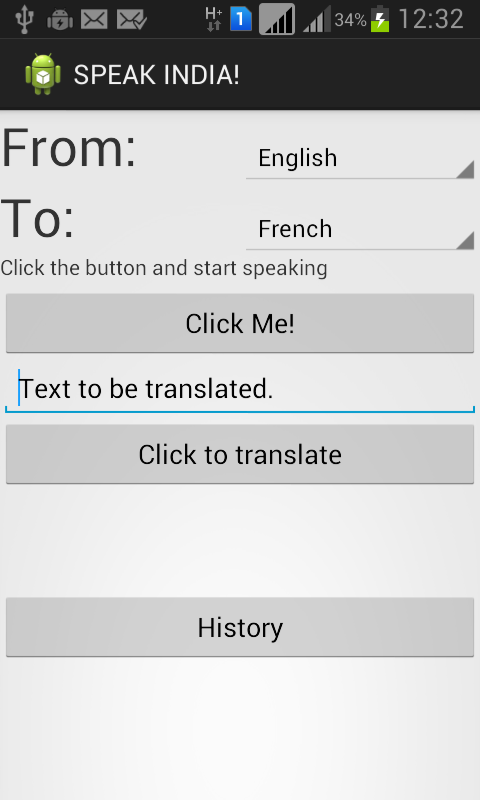
**SCREENSHOTS:**

1. **HOME SCREEN:**

****

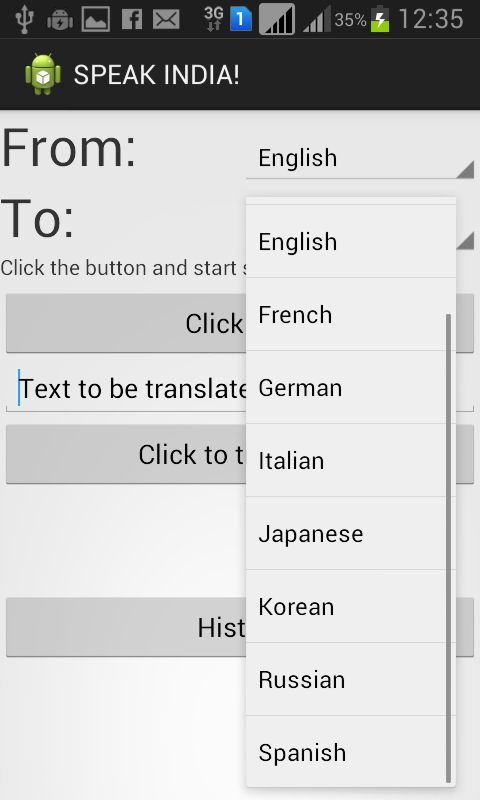
*Fig B.1 Home Screen*

1. **user interface:**

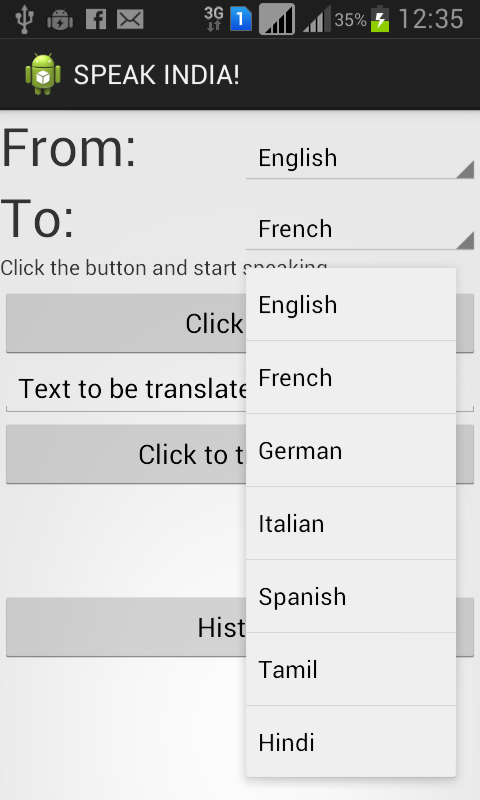
****

*Fig B.2 User Interface*

1. **LIST OF LANGUAGES:**

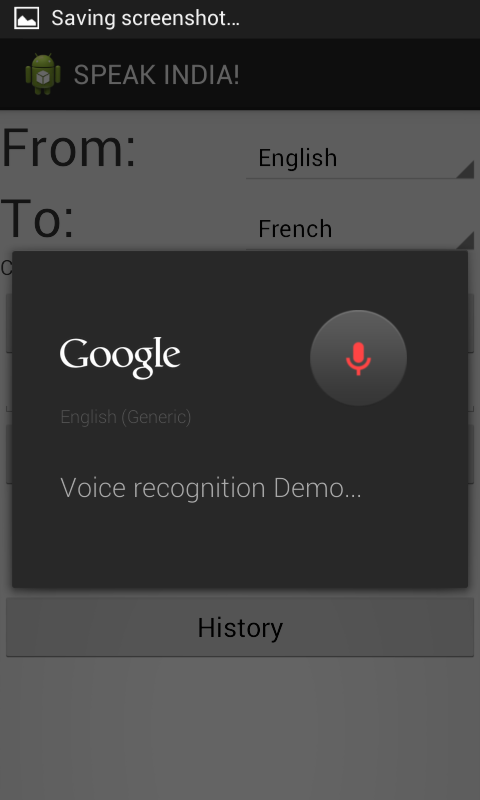
****

*Fig B.3 ‘From’ language list*

****

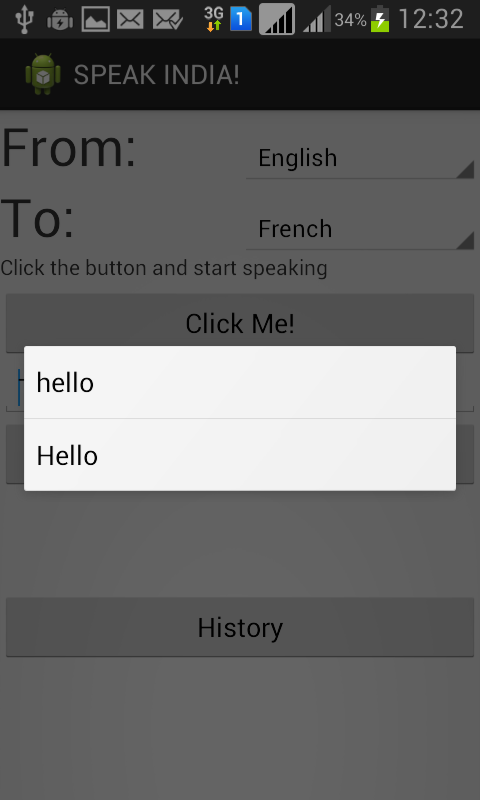
*Fig B.4 ‘To’ language list*

1. **AFTER CLICKING “CLICK ME!”:**



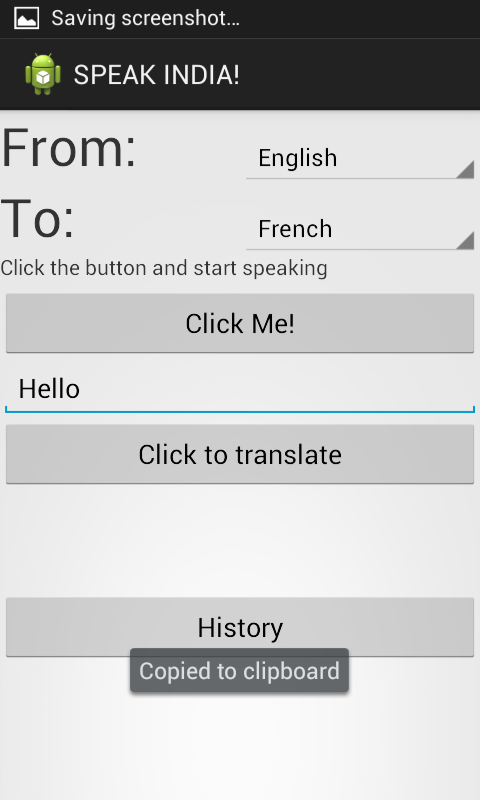
*Fig B.5 Google Voice Recognizer*

1. **VOICE INPUT:**

****

*Fig B.6 List of suggestions*

1. **AFTER SELECTING THE CORRECT WORD:**

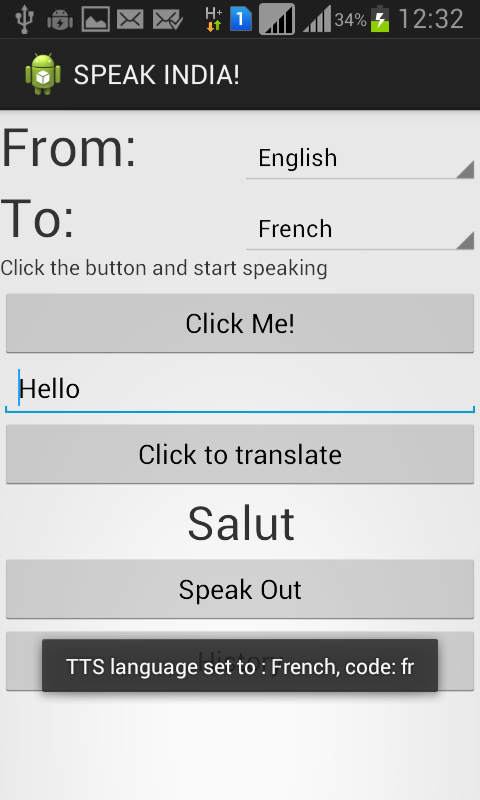
****

*Fig B.7 Selected text*

1. **AFTER CLICKING “Click to translate”:**

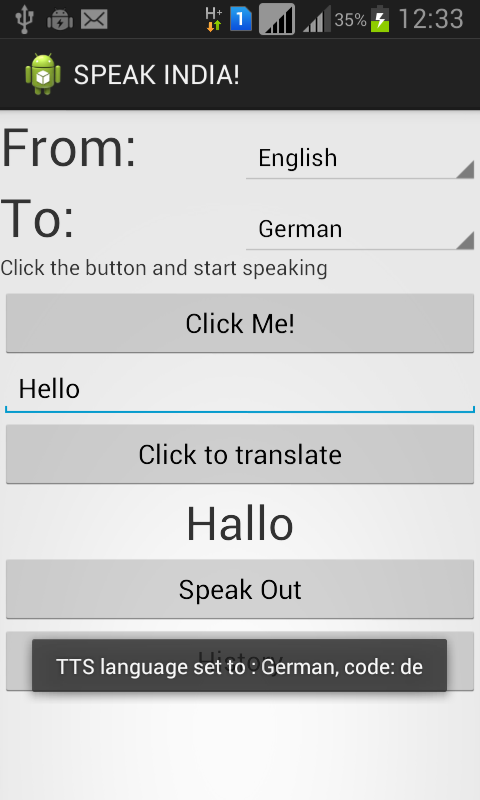
**OUTPUT IN DIFFERENT LANGUAGES:**

**FRENCH:**

****

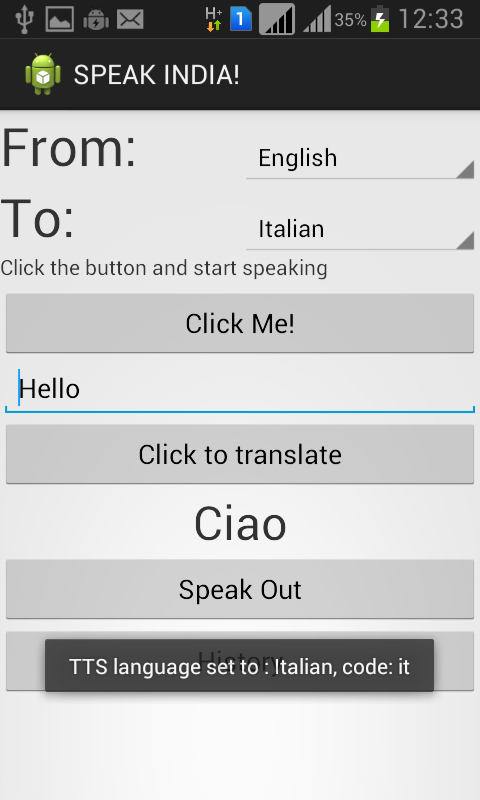
*Fig B.8 Translated text(French)*

**GERMAN:**

****

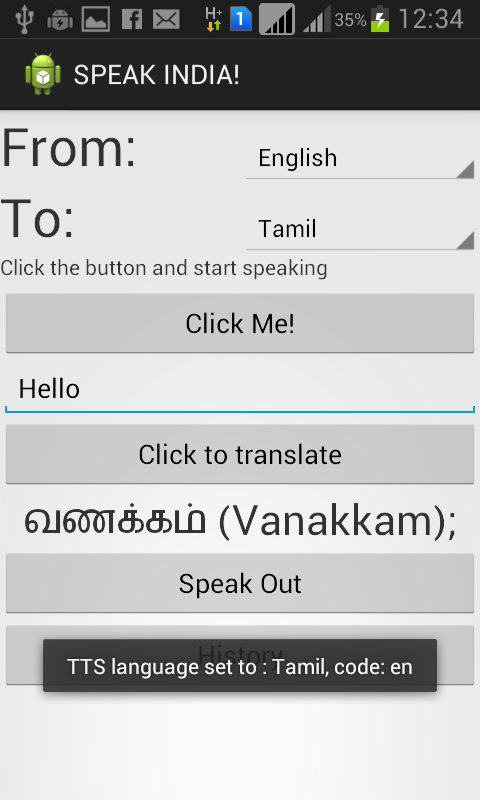
*Fig B.9 Translated text(German)*

**ITALIAN:**

****

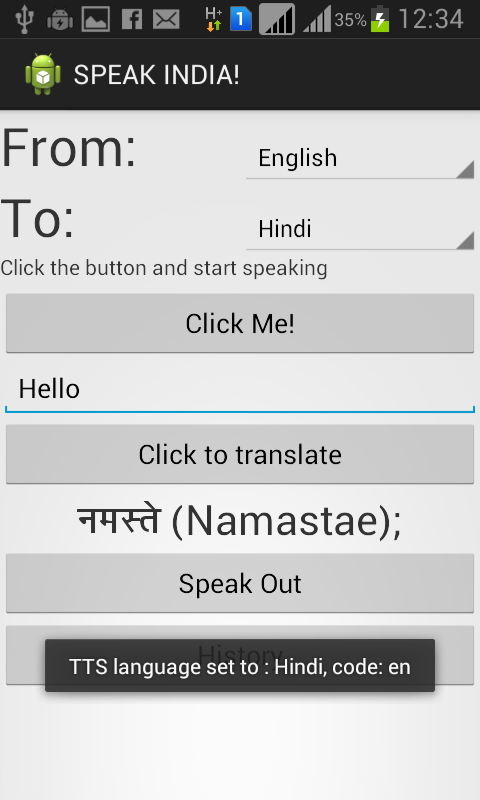
*Fig B.10 Translated text(Italian)*

**TAMIL:**

****

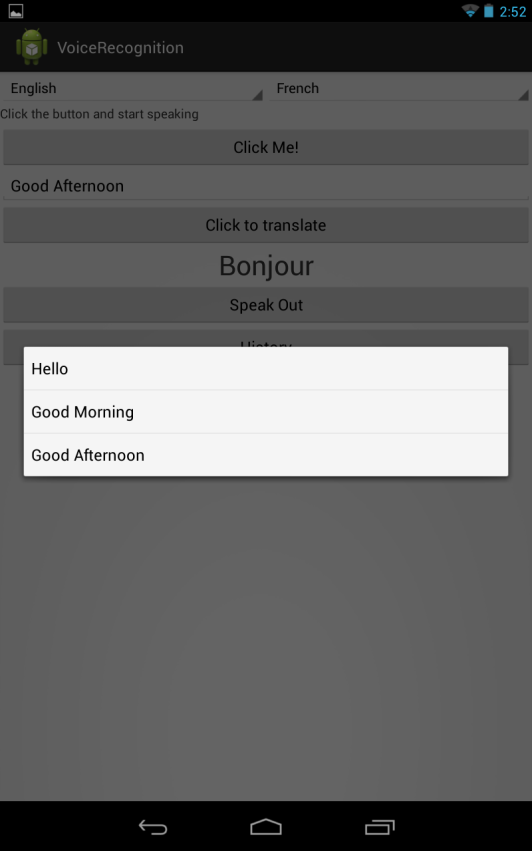
*Fig B.11 Translated text(Tamil)*

**HINDI:**

****

*Fig B.12 Translated text(Hindi)*

1. **HISTORY OF WORDS:**

****

*Fig B.12 Translated text(Hindi)*

**REFERENCES:**

|  |
| --- |
|  |
|  |  |

* A bi-lingual TTS engine system on Android devices by Saychum S and Thangathai A , May 2012.
* Trends in Speech and Language Processing [In the Spotlight] by  [Freng, J.](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=p_Authors:.QT.Freng,%20J..QT.&newsearch=true" \o ") [Ramabhadran, B.](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=p_Authors:.QT.Ramabhadran,%20B..QT.&searchWithin=p_Author_Ids:37284914500&newsearch=true" \o "); [Hansen, J.H.L.](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=p_Authors:.QT.Hansen,%20J.H.L..QT.&searchWithin=p_Author_Ids:37267645000&newsearch=true); [Williams, J.D , June 2011.](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=p_Authors:.QT.Williams,%20J.D..QT.&searchWithin=p_Author_Ids:37558106100&newsearch=true)
* Android-based automatic speech recognition front end to access Web APIs / M. Malcangi, P.Ferroni, M.E. Suzzani - In: Selected papers from proceedings of DSP application day 2010 : e-conference and webinars / [a cura di] M. Malcangi. - Milano : Libreria CLUP, 2011. - ISBN 9788863010411. - pp. 77-83 (( convegno DSP Application Day tenutosi a Milano, Italy nel 2010.

|  |  |
| --- | --- |
| |  | | --- | |  | |

|  |
| --- |
| * Proposal of a Method to Extract the Linguistic Information in Speech Based on Acoustic Separation of the Linguistic and Extra-Linguistic Aspects of Speech -- An Attempt toward Realizing Human-Like Speech Processing on Machines |

**The website the we referred are as follows:**

* <http://developer.android.com/training/index.html>
* <http://thenewboston.org/list.php?cat=6>
* <http://stackoverflow.com/>
* <http://www.talkandroid.com/android-forums>