

PROFESSIONAL TRAINING REPORT-II
at
Sathyabama Institute of Science and Technology
(Deemed to be University)

Submitted in partial fulfillment of the requirements for the award
of Bachelor of Technology Degree in

Information Technology

By

ANAND KUMAR RANJAN
(Reg. No.37120008)



DEPARTMENT OF INFORMATION TECHNOLOGY
SCHOOL OF COMPUTING

SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with Grade "A" by NAAC
JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI - 600 119

December 2020



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with "A" Grade by NAAC

Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai - 600 119.

Phone: 044 - 2450 3150 / 51 / 52 / 54 / 55 Fax: 044 - 2450 2344

www.sathyabama.ac.in



SCHOOL OF COMPUTING

BONAFIDE CERTIFICATE

This is to certify that this Professional Training Report is the bonafide work of **ANAND KUMAR RANJAN (Reg. No. 37120008)** who underwent the professional training in "TEXT-TO-SPEECH" under our supervision from July 2020 to December 2020.

Internal Guide

Dr. G. Mathivanan M.E., Ph.D.,

Head of the Department

Dr. R. SUBHASHINI M.E., Ph.D.,

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I, **ANAND KUMAR RANJAN (Reg. No. 37120008)** hereby declare that the Project Report entitled **“TEXT-TO-SPEECH”** done by me under guidance of **Dr. G. Mathivanan .,M.E., Ph.D.,** at Sathyabama Institute of Science and Technology is submitted in partial fulfillment of the requirements for the award of Bachelor of Technology degree in Information Technology.

DATE:

PLACE:

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

The satisfaction and elation that accompany the successful completion of any task would be incomplete without the mention of the people who have made it possible. It is our great privilege to express our gratitude and respect to all those who have guided me during the course of my Professional Training.

First and foremost, we would express our sincere gratitude to our beloved founder chancellor **Col. Dr. JEPPIAAR, M.A., B.L., Ph.D.**, and **Ms. THIRUMATHI REMIBAI JEPPIAR**. I extend my sincere thanks to our Chancellor **Dr. MARIAZEENA JOHNSON, B.E., M.B.A., M.Phil., Ph.D.**, and the president **Dr. MARIE JOHNSON, B.E., M.B.A., M.Phil., Ph.D.**, and for providing me the necessary facilities. I also acknowledge our Vice Chancellor **Dr. S. SUNDAR MANOHARAN, Ph.D.**, and the Pro Vice Chancellor **Dr. T. SASIPRABA, M.E., Ph.D.**, for their constant support and endorsement.

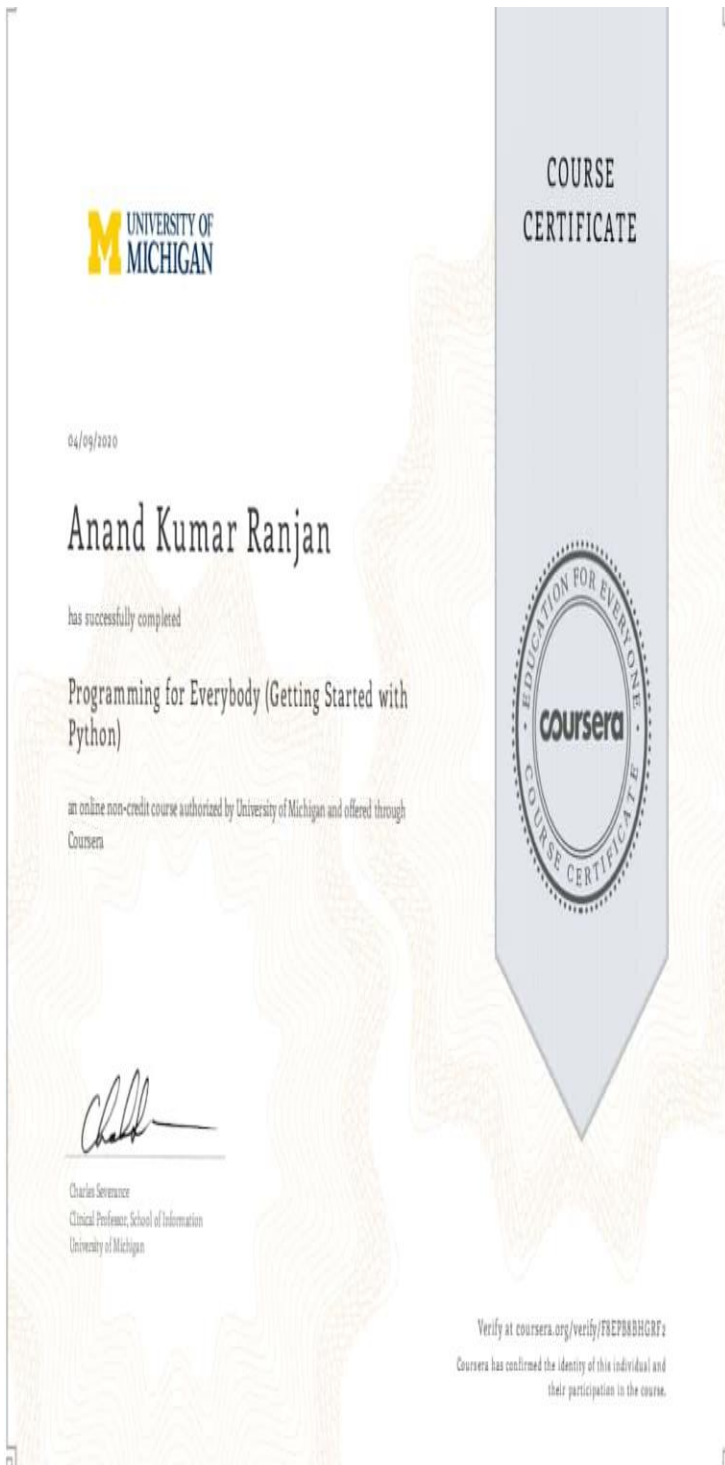
I would like to express our gratitude to our Registrar **Dr. S. S. RAU Ph.D.**, and Controller of Examinations **Dr. IGNI SABASTI PRABU M.E., Ph.D.**, for their valuable support offered to complete my professional training successfully.

I like to express our gratitude to **Dr. R. Subhashini M.E., Ph.D.**, Head of the Department of Information Technology, Sathyabama Institute of Science and Technology for having been a constant source of support and encouragement for the completion of the professional training.

I would like to express our sincere thanks to our internal guide **Dr. G. Mathivanan., M.E., Ph.D.**, who guided me in the preparation of the report.

I thank the Almighty and my Parents for supporting me in the completion of Professional Training.

Certification



ABSTRACT

Text-to-Speech (TTS) is a useful technology that converts any text into a speech signal. It can be utilized for various purposes, e.g. car navigation, announcements in railway stations, response services in telecommunications, visually impaired persons and e-mail reading. Corpus-based TTS makes it possible to dramatically improve the naturalness of synthetic speech compared with the early TTS. However, no general-purpose TTS has been developed that can consistently synthesize sufficiently natural speech. Furthermore, there is not yet enough flexibility in corpus-based TTS. This thesis addresses two problems in speech synthesis. One is how to improve the naturalness of synthetic speech in corpus-based TTS. The other is how to improve control of speaker individuality in order to achieve more flexible speech synthesis. To deal with the former problem, we focus on two factors: (1) an algorithm for selecting the most appropriate synthesis units from a speech corpus, and (2) an evaluation measure for selecting the synthesis units. Moreover, we focus on a voice conversion technique to control speaker individuality to deal with the latter problem. Since various vowel sequences appear frequently in Japanese, it is not realistic to prepare long units that include all possible vowel sequences to avoid vowel-to-vowel concatenation, which often produces auditory discontinuity.

TABLE OF CONTENT

CHAPTER NO	TITLE	PAGE NO
	ABSTRACT	iv
	LIST OF FIGURE	ix
	LIST OF ABBREVIATIONS	x
1	INTRODUCTION	1
	1.1 ABOUT THE CONCEPT	1
	2.2 SPEECH SYNTHESIS	2
2	AIM AND SCOPE OF PRESENT INVESTIGATION	3
	2.1 AIM	3
	2.2 SCOPE	4
	2.2.1 ACCESSIBILITY	4
	2.2.2 E-LEARNING AND SKILL TRAINING	5
	APPLICATION	
3	EXPERIMENTAL OR MATERIAL AND METHOD OR ALGORITHM USED	6
	3.1 SOFTWARE REQUIREMENT	6
	3.1.1 WINDOW 10	6
	3.1.2 NETBEANS IDE 8.2	6
	3.1.3 TTS JAR FILE	7
	3.2 HARDWARE REQUIREMENT	8
	3.2.1 HARD DISK	9
	3.2.2 RAM	9
	3.2.3 PROCESSOR	9
	3.2.4 MOUSE	10
	3.2.5 KEYBOARD	10
	3.2.6 MONITER	11
	3.2.7 SPEAKER	11

	3.3 JAVA SPEECH API	11
	3.4 ALGORITHM OF TEXT TO SPEECH SYSTEM	11
	vii	
	3.4.1 SYNTHESIZER TECHNOLOGIES	11
	3.4.2 CONCATINATION SYNTHESIS	12
	3.4.3 UNIT SELECTION SYNTHESIS	12
	3.4.4 DIPHONE SYNTHESIS	12
	3.4.5 DOMAIN SPECIFIC SYNTHESIS	13
	3.4.6 FORMAT SYNTHESIS	13
	3.4.7 ARTICULATORY SYNTHESIS	14
	3.4.8 HMM BASED SYNTHESIS	15
4	RESULT DISCUSSION AND PERFORMANCE ANALYSIS	15
	4.1 RESULT AND DISCUSSION	15
	4.2 ANALYSIS	16
	4.3 SCREENSHOTS	16
5	SUMMARY AND CONCLUSION	18
	5.1 FUTURE WORK	18
	REFERENCES	18
	APPENDIX	
	A.SOURCE CODE	19
	B.SAMPLE SCREEN	23

LIST OF FIGURES

SNO.	TITLE	PAGE NO.
1.1	SPEECH SYNTHESIS	2
3.1	NETBEANS IDE	7
3.2	TTS JAR FILE	8
3.3	ALGORITHM OF TTS	15
4.3.1	INPUT	
4.3.1 (a)	INPUT IMAGE	16
4.3.1 (b)	INPUT IMAGE	17
4.3.2	OUTPUT	
4.3.2 (a)	OUTPUT IMAGE	17

LIST OF ABBREVIATIONS

ACRONYM	EXPANSION
API Interface	Application programming
ASY	Articulatory Synthesis
CVS	Concurrent Version System
EJB	Enterprise Java Beans
HMM	Hidden Markov Model
IDE Environment	Integrated Development
JAVA ME	Java Micro Edition
JAVA SE	Java Standard Edition
LRC Check	Longitudinal Redundancy
OS	Operating System
SDV	Semicolon Delimited Value
TTS	Text to Speech System

CHAPTER 1

INTRODUCTION

1.1 ABOUT THE CONCEPT

Speech synthesis is the artificial production of human [speech](#). A computer system used for this purpose is called a speech computer or speech synthesizer, and can be implemented in [software](#) or [hardware](#) products. A text-to-speech (TTS) system converts normal language text into speech; other systems render [symbolic linguistic representations](#) like [phonetic transcriptions](#) into speech.

Synthesized speech can be created by concatenating pieces of recorded speech that are stored in a [database](#). Systems differ in the size of the stored speech units; a system that stores [phones](#) or [diphones](#) provides the largest output range, but may lack clarity. For specific usage domains, the storage of entire words or sentences allows for high-quality output. Alternatively, a synthesizer can incorporate a model of the [vocal tract](#) and other human voice characteristics to create a completely "synthetic" voice output.

The quality of a speech synthesizer is judged by its similarity to the human voice and by its ability to be understood clearly. An intelligible text-to-speech program allows people with [visual impairments](#) or [reading disabilities](#) to listen to written words on a home computer. Many computer operating systems have included speech synthesizers since the early 1990s.

A text-to-speech system (or "engine") is composed of two parts: a [front-end](#) and a [back-end](#). The front-end has two major tasks. First, it converts raw text containing symbols like numbers and abbreviations into the equivalent of written-out words. This process is often called text normalization, pre-processing, or [tokenization](#). The front-end then assigns [phonetic transcriptions](#) to each word, and divides and marks the text into [prosodic units](#), like [phrases](#), [clauses](#), and [sentences](#). The process of assigning phonetic transcriptions to words is called text-to-phoneme or [grapheme-to-phoneme](#) conversion. Phonetic transcriptions and prosody information together make up the symbolic linguistic representation that is output by the front-end. The back-end—often referred to as the synthesizer—then converts the symbolic linguistic representation into sound. In certain systems,

this part includes the computation of the target prosody (pitch contour, phoneme durations), which is then imposed on the output speech.

[Text-to-Speech](#) (TTS) is the ability of a computer to produce spoken words by converting text to voice. In other words Text-to-Speech software is a speech synthesizer that vocalizes text in real time in a natural way. Text to speech:

Telecommunications: Text to speech can be implemented in IVR systems to create an efficient self-service solution that improves customer satisfaction by informing and guiding callers while reducing costs. TTS can also be used in [automated outbound call systems](#) in order to provide information to customers.

Mobile: From navigators to mobile phones and from e-readers to tablets Text-to-Speech technology is used to vocalize contents in order to provide an easier and more natural interaction. It is also used in [mobile applications](#) for various developments.

Education: Text-to-Speech technology can be used in language teaching applications to vocalize any word in order to improve the pronunciation capabilities of users in the field of education to develop the text to speech technology.

Accessibility: Text to speech technology can read text out loud and enables the use of computers and mobile devices for the disabled and for people with special needs. For example menus of [ATMs](#) are vocalized with Text to Speech technology in order to provide enhanced customer experience especially for the disabled. Web site and newspaper vocalization can also be listed among accessibility-related use of TTS.

1.2 SPEECH SYNTHESIS

Speech synthesis is artificial simulation of human speech with by a computer or other device. The counterpart of the voice recognition, speech synthesis is mostly used for translating text information into audio information and in applications such as voice-enabled services and mobile applications. Apart from this, it is also used in assistive technology for helping vision-impaired individuals in reading text content..Speech synthesis has long been a vital [assistive technology](#) tool and its application in this area is significant and widespread. It allows environmental barriers to be removed for people with a wide range of disabilities. The longest application has been in the use of [screen readers](#) for people

with visual impairment, but text-to-speech systems are now commonly used by people with dyslexia and other reading difficulties as well as by pre-literate children.

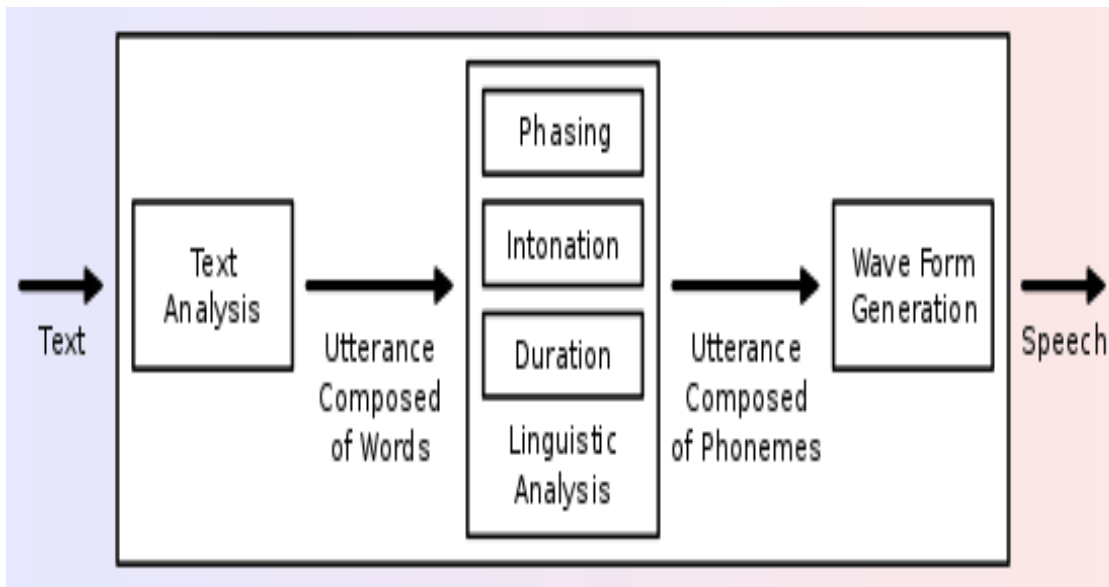


Fig-1.1

CHAPTER-2

AIM AND SCOPE OF THE PRESENT INVESTIGATION

2.1 AIM

Speech synthesis has long been a vital assistive technology tool and its application in this area is significant and widespread. It allows environmental barriers to be removed for people with a wide range of disabilities. The longest application has been in the use of screen readers for people with visual impairment, but text-to-speech systems are now commonly used by people with dyslexia and other reading difficulties as well as by pre-literate children. They are also frequently employed to aid those with severe speech impairment usually through a dedicated voice output communication aid.

2.2 SCOPE

Text to speech gives access to your content to a greater population, such as those with literacy difficulties, learning disabilities, reduced vision and those learning a language. It also opens doors to anyone else looking for easier ways to access digital content. The text is read and highlighted simultaneously so that the reader may easily follow along. 244 million people are foreign born across the globe (an increase of 70 million since 2000). Language proficiency and schooling in the host country's language is a very real problem for migrants and their families. Between 2015 and 2030, the number of people aged 60 years or over will grow by 56 percent, from 901 million to 1.4 billion. In the US alone, 59% of senior citizens use the Internet daily. Making digital content on the Internet accessible in multiple forms creates an easier user experience.

Some people are auditory learners, some are visual learners, and some are kinesthetic learners – most learn best through a combination of the three. Universal Design for Learning is a plan for teaching which, through the use of technology and adaptable lesson plans, aims to help the maximum number of learners comprehend and retain information by appealing to all learning styles.

In the following two years, human sounding text-to-speech services will also start to dominate the following voiceover recording applications.

2.2.1. Accessibility

There are already several software suites designed to convert on-screen messages to voice audio. Windows and MacOS both support this technology, to make their operating systems accessible to the blind. But TTS is also a boon to documentation accessibility. Think of long health care plans, instruction manuals, user agreements, or even warranties. TTS allows companies, or even users themselves, to quickly convert all of this content to accurate voiceover audio.

We probably won't start seeing it in audio description for movies and TV shows very soon, since it really requires a human voice, but for everything else – expect to hear TTS.

2.2.2. E-Learning and skills training applications

Captivate already provides TTS voices for its software in English and a couple other languages. Microsoft already uses TTS voices for its how-to video tutorials – you can see one of the Spanish-language videos [here](#), which has a blurb in which Microsoft explains that it switched to TTS because it allows them to offer their localized videos more quickly. More and more e-Learning developers are turning to TTS voices for their content, especially as a more user-friendly, but equally-low-cost alternative to subtitles. Text to speech, abbreviated as TTS, is a form of speech synthesis that converts text into spoken voice output. Text to speech systems were first developed to aid the visually impaired by offering a computer-generated spoken voice that would "read" text to the user.

TTS should not be confused with voice response systems. Voice response systems synthesize speech by concatenating sentences from a database of pre-recorded words and are used for different purposes than TTS systems, which form sentences and/or phrases based on a language's graphemes and phonemes. Voice response systems are limited to synthesizing sentences that contain only words that have been predetermined by the system.

TTS systems, in contrast, are theoretically capable of "reading" any string of text characters to form original sentences.

CHAPTER 3

EXPERIMENTAL OR MATERIALS AND METHODS USED

3.1 SOFTWARE REQUIREMENT

- Windows 10
- NetBeans IDE 8.2
- TTS.jar File

3.1.1 Window 10

Windows 10 is a Microsoft operating system for personal computers, tablets, embedded devices and internet of things devices.

Microsoft released Windows 10 in July 2015 as a follow-up to Windows 8. The company has said it will update Windows 10 in perpetuity rather than release a new, full-fledged operating system as a successor.

3.1.2 Netbeans IDE 8.2

NetBeans is an [integrated development environment](#) (IDE) for [Java](#). NetBeans allows applications to be developed from a set of modular [software components](#) called modules.

NetBeans run on [Windows](#), [macOS](#), [Linux](#) and [Solaris](#). In addition to Java development, it has extensions for other languages like [PHP](#), [C](#), [C++](#), [HTML5](#), and [JavaScript](#). Applications based on NetBeans, including the NetBeans IDE, can be extended by [third party developers](#).

NetBeans began in 1996 as Xelfi (word play on Delphi), a Java IDE student project under the guidance of the Faculty of Mathematics and Physics at Charles University in Prague. In 1997, Roman Staněk formed a company around the project and produced commercial versions of the NetBeans IDE until it was bought by Sun Microsystems in 1999. Sun open-sourced the NetBeans IDE in June of the following year. Since then, the NetBeans community has continued to grow.¹ In 2010, Sun (and

thus NetBeans) was acquired by Oracle Corporation. Under Oracle, NetBeans competed with JDeveloper, a freeware IDE that has historically been a product of the company. In September 2016, Oracle submitted a proposal to donate the NetBeans project to the Apache Software Foundation, stating that it was "opening up the NetBeans governance model to give NetBeans constituents a greater voice in the project's direction and future success through the upcoming release of Java 9 and NetBeans 9 and beyond". The move was endorsed by Java creator James Gosling. The project entered the Apache Incubator in October 2016.

NetBeans IDE is an open-source integrated development environment. NetBeans IDE supports development of all Java application types (Java SE (including JavaFX), Java ME, web, EJB and mobile applications) out of the box. Among other features are an Ant-based project System.

Modularity: All the functions of the IDE are provided by modules. Each module provides a well-defined function, such as support for the Java language, editing, or support for the CVS versioning system, and SVN. NetBeans contains all the modules needed for Java development in a single download, allowing the user to start working immediately.

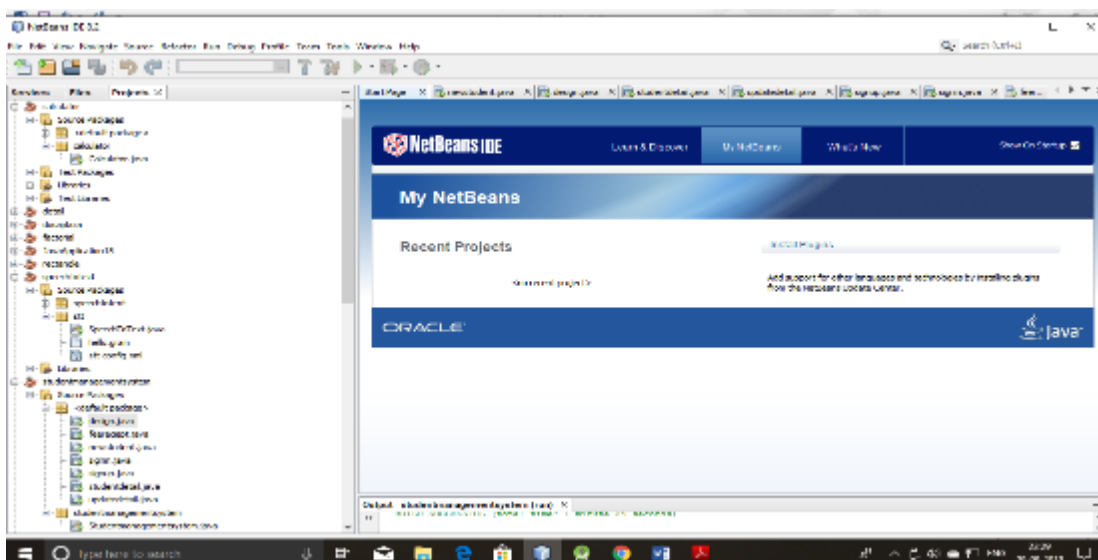


Fig-3.1

3.1.3 TTS.jar

FreeTTS is an open source speech synthesis system written entirely in the [Java](#) programming language. It is based upon Flite. FreeTTS is an implementation of Sun's Java Speech API.

FreeTTS supports end-of-speech markers. Gnopernicus uses these in a number of places: to know when text should and should not be interrupted, to better concatenate speech, and to sequence speech in different voices. Benchmarks conducted by Sun in 2002 on Solaris showed that FreeTTS ran two to three times faster than Flite at the time.

FreeTTS has been tested on the Solaris™ Operating Environment, Mac OS X, Linux and Win32 operating systems

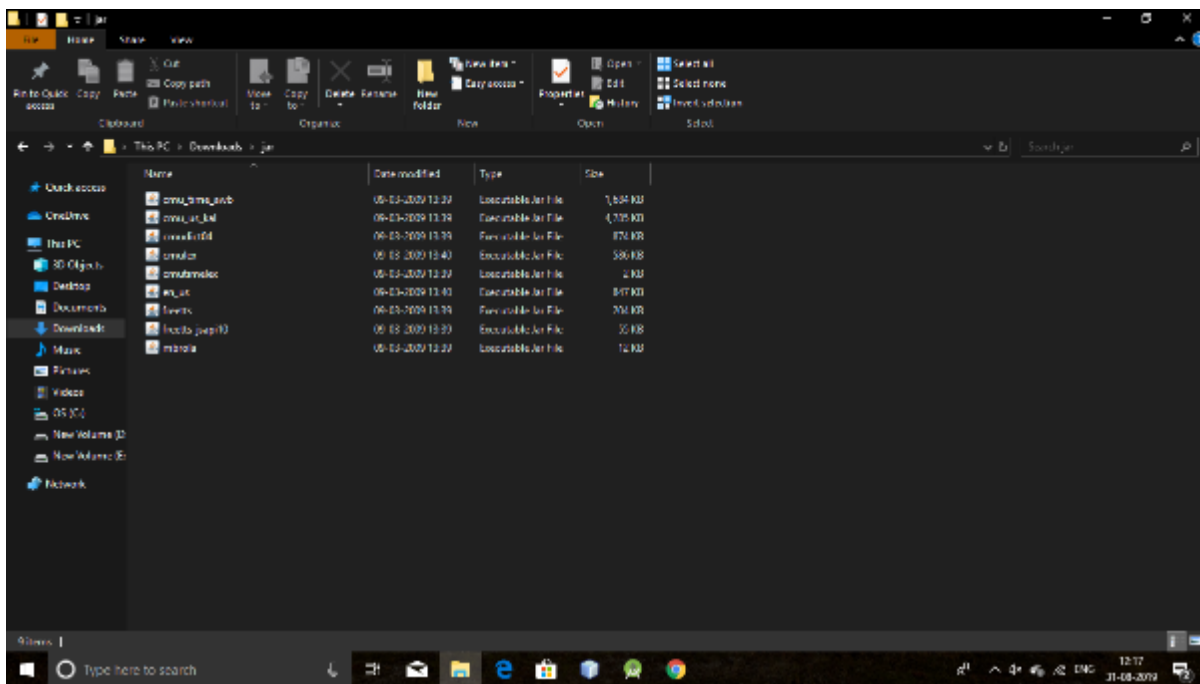


Fig-3.2

3.2 HARDWARE REQUIREMENT

- Hard Disk
- RAM – 1 GB
- Processor – Dual Core or Above
- Mouse
- Keyboard

- Monitor
- Speaker

3.2.1 Hard Disk

A hard disk or fixed disk, is an electro-mechanical data storage device that uses magnetic storage to store and retrieve digital information using one or more rigid rapidly rotating disks (platters) coated with magnetic material.

During the mid-1990s the typical hard disk drive for a PC had a capacity of about 1 gigabyte. As of January 2019, desktop hard disk drives typically had a capacity of 1 to 6 terabytes, with the largest-capacity drives reaching 15 terabytes

3.2.2 RAM

Random access memory (RAM) is a type of data storage used in computers that is generally located on the motherboard. This type of memory is volatile and all information that was stored in RAM is lost when the computer is turned off.

The RAM chip may be individually mounted on the motherboard or in sets of several chips on a small board connected to the motherboard. Older memory types were in the form of chips called dual in-line package (DIP). Although DIP chips are still used today, the majority of memory is in the form of a module, a narrow printed circuit board attached to a connector on the motherboard. The three main memory circuit boards types containing chips are: RIMMs (Rambus in-line memory modules), DIMMs (dual in-line memory modules) and SIMMs (single in-line memory modules). Most motherboards today use DIMMs. Sometimes, people get confused about the difference between memory and storage, in part because both can be measured in megabytes (MB), gigabytes (GB) and terabytes (TB).

By contrast, storage is slower, but it can retain data when the device is powered down. So, for example, if a document has been saved to a hard drive prior to a power outage or system crash, the user will still be able to retrieve it when the system is back up and running.

3.2.3 Processor

A processor is an integrated electronic circuit that performs the calculations that run a computer. A processor performs arithmetical, logical, input/output (I/O) and other basic instructions that are passed from an operating system (OS). Most other processes are dependent on the operations of a processor.

Dual core is a CPU that has two distinct processors that work simultaneously in the same integrated circuit. This type of processor can function as efficiently as a single processor but can perform operations up to twice as quickly.

3.2.4 Mouse

A mouse is a small handheld input device that controls a computer screen's cursor or pointer in conjunction with the way it is moved on a flat surface. The mouse term name originates from its likeness to a small, corded and elliptical shaped device that looks like a mouse tail.

Some mouse devices have integrated features, such as extra buttons that may be programmed and assigned with different commands.

Because the mouse reduces the use of a keyboard, its invention and continuous innovation is considered one of the most important breakthroughs in computer ergonomics.

3.2.5 Keyboard

A computer keyboard is an input device used to enter characters and functions into the computer system by pressing buttons, or keys. It is the primary device used to enter text. A keyboard typically contains keys for individual letters, numbers and special characters, as well as keys for specific functions. A keyboard is connected to a computer system using a cable or a wireless connection.

A computer keyboard is an input device used to enter characters and functions into the computer system by pressing buttons, or keys. It is the primary device used to enter text. A keyboard typically contains keys for individual letters, numbers and special characters, as well as keys for specific functions.

3.2.6 Monitor

A computer monitor is an electronic device that shows pictures for computers. Monitors often look similar to televisions. The main difference between a monitor and a television is that a monitor does

not have a television tuner to change channels. Monitors often have higher display resolution than televisions.

The advent of display technology has paved the way for the continuous evolution of the monitor, whether it's for computers, television, mobile devices or any device that has a display. The current contenders for top-tier technology being used for display devices includes Super LCD 3 (SLCD3) and Super AMOLED. It should be noted that LED displays are actually just a kind of LCD display that use LED lights as backlight illumination.

3.2.7 Speaker

Most Important Speaker is required for this project because when we enter any alphabetical word or paragraph it convert it into sound which will come through the speaker of the system.

3.2 JAVA SPEECH API

The Java Speech API allows Java applications to incorporate speech technology into their user interfaces. It defines a cross-platform API to support command and control recognizers, dictation systems and speech synthesizers.

Java Speech supports speech synthesis which means the process of generating spoken the language by machine on the basis of written input.

It is important to keep in mind that Java Speech is only a specification i.e. no implementation is included. Thus third-parties provide the implementations. The javax .speech package defines the common functionality of recognizers, synthesizers, and other speech engines the java speech api used in the system. The package javax.speech in java api. Synthesis extends this basic functionality for synthesizers.

3.3 ALGORITHM OF TEXT TO SPEECH SYSTEM

3.3.1 Synthesizer technologies

The most important qualities of a speech synthesis system are naturalness and intelligibility. Naturalness describes how closely the output sounds like human speech, while

intelligibility is the ease with which the output is understood. The ideal speech synthesizer is both natural and intelligible. Speech synthesis systems usually try to maximize both characteristics.

The two primary technologies generating synthetic speech waveforms are concatenative synthesis and formant synthesis. Each technology has strengths and weaknesses, and the intended uses of a synthesis system will typically determine which approach is used.

3.3.2 Concatenation synthesis

Concatenative synthesis is based on the [concatenation](#) (or stringing together) of segments of recorded speech. Generally, concatenative synthesis produces the most natural-sounding synthesized speech. However, differences between natural variations in speech and the nature of the automated techniques for segmenting the waveforms sometimes result in audible glitches in the output. There are three main sub-types of concatenative synthesis.

3.3.3 Unit Selection Synthesis

Unit selection synthesis uses large [databases](#) of recorded speech. During database creation, each recorded utterance is segmented into some or all of the following: individual [phones](#), [diphones](#), half-phones, [syllables](#), [morphemes](#), [words](#), [phrases](#), and [sentences](#). Typically, the division into segments is done using a specially modified [speech recognizer](#) set to a "forced alignment" mode with some manual correction afterward, using visual representations such as the [waveform](#) and [spectrogram](#). An [index](#) of the units in the speech database is then created based on the segmentation and acoustic parameters like the [fundamental frequency](#) ([pitch](#)), duration, position in the syllable, and neighboring phones. At [run time](#), the desired target utterance is created by determining the best chain of candidate units from the database (unit selection). This process is typically achieved using a specially weighted [decision tree](#).

3.3.4 Diphone Synthesis

Diphone synthesis uses a minimal speech database containing all the [diphones](#) (sound-to-sound transitions) occurring in a language. The number of diphones depends on the of the language: for example, Spanish has about 800 diphones, and German about 2500. In diphone synthesis, only one example of each diphone is contained in the speech database. At runtime, the target [prosody](#) of a

sentence is superimposed on these minimal units by means of [digital signal processing](#) techniques such as [linear predictive coding](#), [PSOLA](#) or [MBROLA](#), or more recent techniques such as pitch modification in the source domain using [discrete cosine transform](#)^[32] Diphone synthesis suffers from the sonic glitches of concatenative synthesis and the robotic-sounding nature of formant synthesis, and has few of the advantages of either approach other than small size.

3.3.5 Domain Specific Synthesis

Domain-specific synthesis concatenates words and phrases to create complete utterances. It is used in applications where the variety of texts the system will output is limited to a particular domain, like transit schedule announcements or weather reports. The technology is very simple to implement, and has been in commercial use for a long time, in devices like talking clocks and calculators. The level of naturalness of these systems can be very high because the variety of sentence types is limited, and they closely match the prosody and intonation of the original recording.

Because these systems are limited by the words and phrases in their databases, they are not general-purpose and can only synthesize the combinations of words and phrases with which they have been pre-programmed. The blending of words within naturally spoken language however can still cause problems unless the many variations are taken into account.

3.3.6 Format Synthesis

[Formant](#) synthesis does not use human speech samples at runtime. Instead, the synthesized speech output is created using [additive synthesis](#) and an acoustic model ([physical modelling synthesis](#)). Parameters such as [fundamental frequency](#), [voicing](#), and [noise](#) levels are varied over time to create a [waveform](#) of artificial speech. This method is sometimes called *rules-based synthesis*; however, many concatenative systems also have rules-based components. Many systems based on formant synthesis technology generate artificial, robotic-sounding speech that would never be mistaken for human speech. However, maximum naturalness is not always the goal of a speech synthesis system, and formant synthesis systems have advantages over concatenative systems. Formant-synthesized speech can be reliably intelligible, even at very high speeds, avoiding the acoustic glitches that commonly plague concatenative systems. High-speed synthesized speech is used by the visually impaired to quickly navigate computers using a [screen reader](#). Examples of non-real-time but highly accurate intonation control in formant synthesis include the work done in the late 1970s for the [Texas Instruments](#) toy [Speak & Spell](#), and in the early 1980s [Sega arcade](#) machines and in

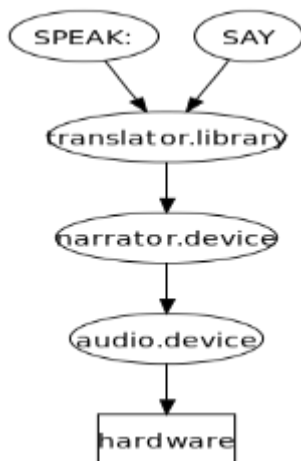
many [Atari, Inc.](#) arcade games using the [TMS5220 LPC Chips](#). Creating proper intonation for these projects was painstaking, and the results have yet to be matched by real-time text-to-speech interfaces.

3.3.7 Articulatory Synthesis

[Articulatory synthesis](#) refers to computational techniques for synthesizing speech based on models of the human [vocal tract](#) and the articulation processes occurring there. The first articulatory synthesizer regularly used for laboratory experiments was developed at [Haskins Laboratories](#) in the mid-1970s by [Philip Rubin](#), Tom Baer, and Paul. This synthesizer, known as ASY, was based on vocal tract models developed at [Bell Laboratories](#) in the 1960s and 1970s by Paul, Cecil Coker, and colleagues. Until recently, articulatory synthesis models have not been incorporated into commercial speech synthesis systems. A notable exception is the [NeXT](#)-based system originally developed and marketed by Trillium Sound Research.

3.3.8 HMM Based Synthesis

HMM-based synthesis is a synthesis method based on [hidden Markov models](#), also called Statistical Parametric Synthesis. In this system, the [frequency spectrum](#) ([vocal tract](#)), [fundamental frequency](#) (voice source), and duration ([prosody](#)) of speech are modeled simultaneously by HMMs. Speech [waveforms](#) are generated from HMMs themselves based on the [maximum likelihood](#) criterion.



CHAPTER 4

RESULTS AND DISCUSSION, PERFORMANCE ANALYSIS

4.1 RESULTS & DISCUSSION

Speech synthesis has long been a vital [assistive technology](#) tool and its application in this area is significant and widespread. It allows environmental barriers to be removed for people with a wide range of disabilities. The longest application has been in the use of [screen readers](#) for people with [visual impairment](#), but text-to-speech systems are now commonly used by people with [dyslexia](#) and other reading difficulties as well as by pre-literate children. They are also frequently employed to aid those with severe [speech impairment](#) usually through a dedicated [voice output communication aid](#). Speech synthesis has long been a vital [assistive technology](#) tool and its application in this area is significant and widespread. It allows environmental barriers to be removed for people with a wide range of disabilities. The longest application has been in the use of [screen readers](#) for people with [visual impairment](#), but text-to-speech systems are now commonly used by people with [dyslexia](#) and other reading difficulties as well as by pre-literate children. They are also frequently employed to aid those with severe [speech impairment](#) usually through a dedicated [voice output communication aid](#).

4.2 ANALYSIS

As mentioned, we analyze the speech recognition software using three approaches. First, we run the application on different platforms to understand how well the application scale with processor advancement. Speech input is captured in files and ran in batch mode in order to control the quality of the samples. We then use an Intel toolset called V tune to collect built-in performance counter values. Finally we use a full system simulator called Soft SDV together with a cache simulator to study the memory behavior in more depth.

4.3 SCREENSHOTS

4.3.1 Input

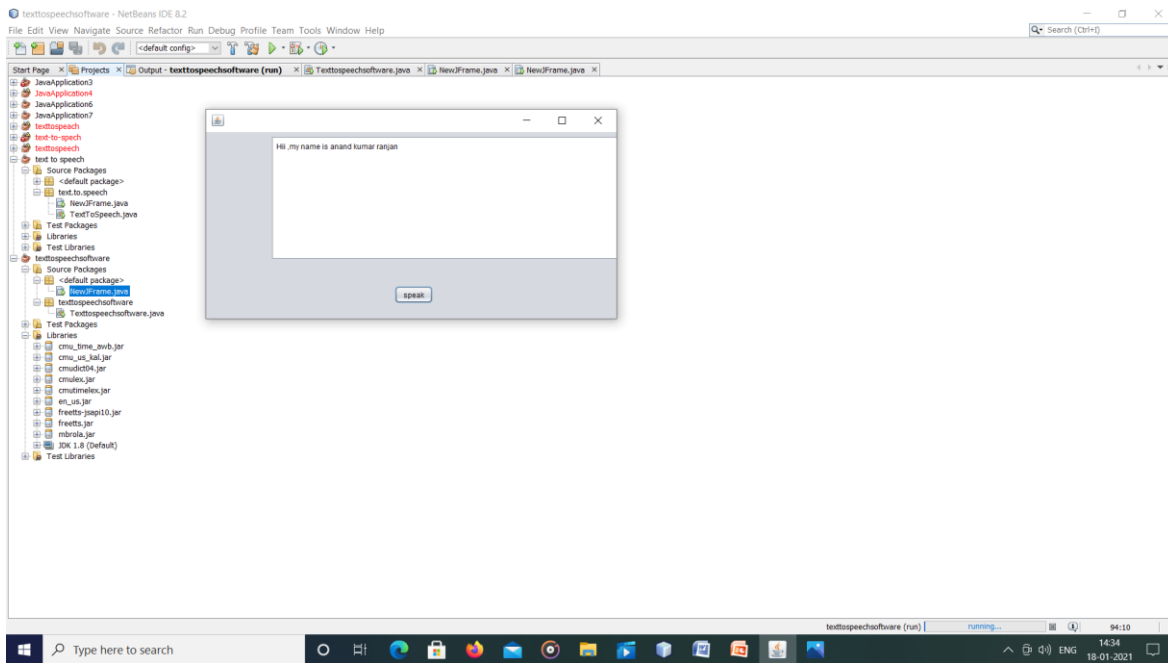


Fig-4.3.1 (a)



Fig-4.3.1 (b)

4.3.2 Output

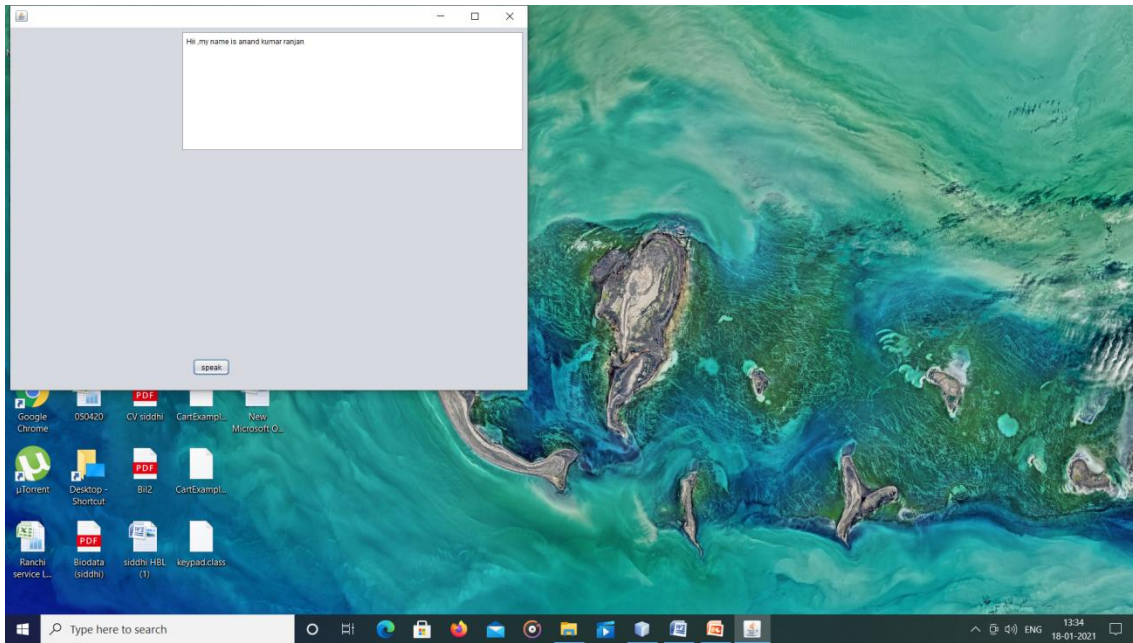


Fig-4.3.2 (a)

CHAPTER 5

SUMMARY AND CONCLUSIONS

Text to speech synthesis is a rapidly growing aspect of computer technology and is increasingly playing a more important role in the way we interact with the system and interfaces across a variety of platforms. We have identified the various operations and processes involved in text to speech synthesis. We have also developed a very simple and attractive graphical user interface which allows the user to type in his/her text provided in the text field in the application. Our system interfaces with a text to speech engine developed for American English. Another area of further work is the implementation of a text to speech system on other platforms, such as telephony systems, ATM machines, video games and any other platforms where text to speech technology would be an added advantage and increase functionality.

5.1 FUTURE WORK

Project can be extends to various application such as writing the text in different –different languages and getting the voice in the same language as written in the interface. Also project can be used in the e-learning in the different languages which can be useful for the speech less person ie who cannot speak.

REFERENCES

- 1.www.google.com
- 2.www.geeksfoegeeks.org
- 3.www.wikipedia.org/voicerecognition

4. www.wikipedia.org/netbeans
5. www.wikipedia.org/fts
6. www.wikipedia.org/synthesizer_technique
7. www.tutorialspointsg.com
8. www.javaworld.com

APPENDIX

SOURCE CODE

port the Packages

```
import com.sun.speech.freetts.Voice;  
import com.sun.speech.freetts.VoiceManager;  
public class text extends javax.swing.JFrame;
```

```
public text() {  
    initComponents();
```

```
private void initComponents() {
```

```
    jScrollPane1 = new javax.swing.JScrollPane();  
    ta = new javax.swing.JTextArea();  
    jButton1 = new javax.swing.JButton();  
    jButton2 = new javax.swing.JButton();  
    jLabel1 = new javax.swing.JLabel();  
    jLabel2 = new javax.swing.JLabel();  
    setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);  
    alwaysOnTop(true);
```

```

background(new java.awt.Color(51, 204, 255));
foreground(new java.awt.Color(0, 204, 255));

setColumns(20);
setFont(new java.awt.Font("Monospaced", 1, 24));
setRows(5);
jScrollPane1.setViewportViewView(ta);

jButton1.setBackground(new java.awt.Color(255, 255, 255));
jButton1.setFont(new java.awt.Font("Tahoma", 1, 18));
jButton1.setForeground(new java.awt.Color(255, 51, 51));
jButton1.setText("Speak");
jButton1.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        jButton1ActionPerformed(evt);
    }
});

jButton2.setFont(new java.awt.Font("Tahoma", 1, 18));
jButton2.setForeground(new java.awt.Color(255, 0, 51));
jButton2.setText("Clear");
jButton2.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        jButton2ActionPerformed(evt);
    }
});

jLabel1.setFont(new java.awt.Font("Tahoma", 1, 36));
jLabel1.setText("Text to Speech Software");

jLabel2.setFont(new java.awt.Font("Tahoma", 1, 18));

```

```

jLabel2.setText("for speech less people");

x.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
getContentPane().setLayout(layout);

out.setHorizontalGroup(
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(layout.createSequentialGroup()
            .addGap(35, 35, 35)
            .addComponent(jScrollPane1, javax.swing.GroupLayout.PREFERRED_SIZE,
                javax.swing.GroupLayout.PREFERRED_SIZE)
            .addGap(42, 42, 42)
            .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)
                .addComponent(jButton1, javax.swing.GroupLayout.DEFAULT_SIZE, 105, Short.MAX_VALUE)
                .addComponent(jButton2, javax.swing.GroupLayout.DEFAULT_SIZE,
                    javax.swing.GroupLayout.DEFAULT_SIZE,
                    Short.MAX_VALUE)))
            .addGroup(layout.createSequentialGroup()
                .addGap(25, 25, 25)
                .addComponent(jLabel1))
            .addGroup(layout.createSequentialGroup()
                .addGap(122, 122, 122)
                .addComponent(jLabel2, javax.swing.GroupLayout.PREFERRED_SIZE,
                    javax.swing.GroupLayout.PREFERRED_SIZE)))
        .addGap(31, Short.MAX_VALUE))
    );

out.setVerticalGroup(
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(layout.createSequentialGroup()
            .addGap(31, Short.MAX_VALUE)
            .addComponent(jLabel1, javax.swing.GroupLayout.PREFERRED_SIZE,
                javax.swing.GroupLayout.PREFERRED_SIZE)
            .addGap(31, Short.MAX_VALUE))
        .addGroup(layout.createSequentialGroup()
            .addGap(31, Short.MAX_VALUE)
            .addComponent(jLabel2, javax.swing.GroupLayout.PREFERRED_SIZE,
                javax.swing.GroupLayout.PREFERRED_SIZE)
            .addGap(31, Short.MAX_VALUE))
    );

```

```

        javax.swing.GroupLayout.PREFERRED_SIZE)
        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
Component(jLabel2,javax.swing.GroupLayout.PREFERRED_SIZE,
        javax.swing.GroupLayout.PREFERRED_SIZE)
Gap(50, 50, 50)
        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
IGroup(layout.createSequentialGroup())
Component(jScrollPane1,javax.swing.GroupLayout.PREFERRED_SIZE,
        javax.swing.GroupLayout.PREFERRED_SIZE)
ContainerGap(41, Short.MAX_VALUE))
        addGroup(layout.createSequentialGroup())
Gap(0, 0, Short.MAX_VALUE)
Component(jButton1,javax.swing.GroupLayout.PREFERRED_SIZE,
        javax.swing.GroupLayout.PREFERRED_SIZE)
Gap(33, 33, 33)
Component(jButton2,javax.swing.GroupLayout.PREFERRED_SIZE,
        javax.swing.GroupLayout.PREFERRED_SIZE)
Gap(124, 124, 124))))
);

pack();

```

sign the voice name variable as kevin16 type which diphone type
 ate static final String VOICENAME="kevin16";

```

private void jButton1ActionPerformed(java.awt.event.ActionEventevt) {
    Voice voice;
    VoiceManagervm=VoiceManager.getInstance();
    voice=vm.getVoice(VOICENAME);
    voice.allocate();
    try{

```

```

voice.speak(ta.getText());
    }
    catch(Exception e)
    {

    }
}

private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
ta.setText("");
}

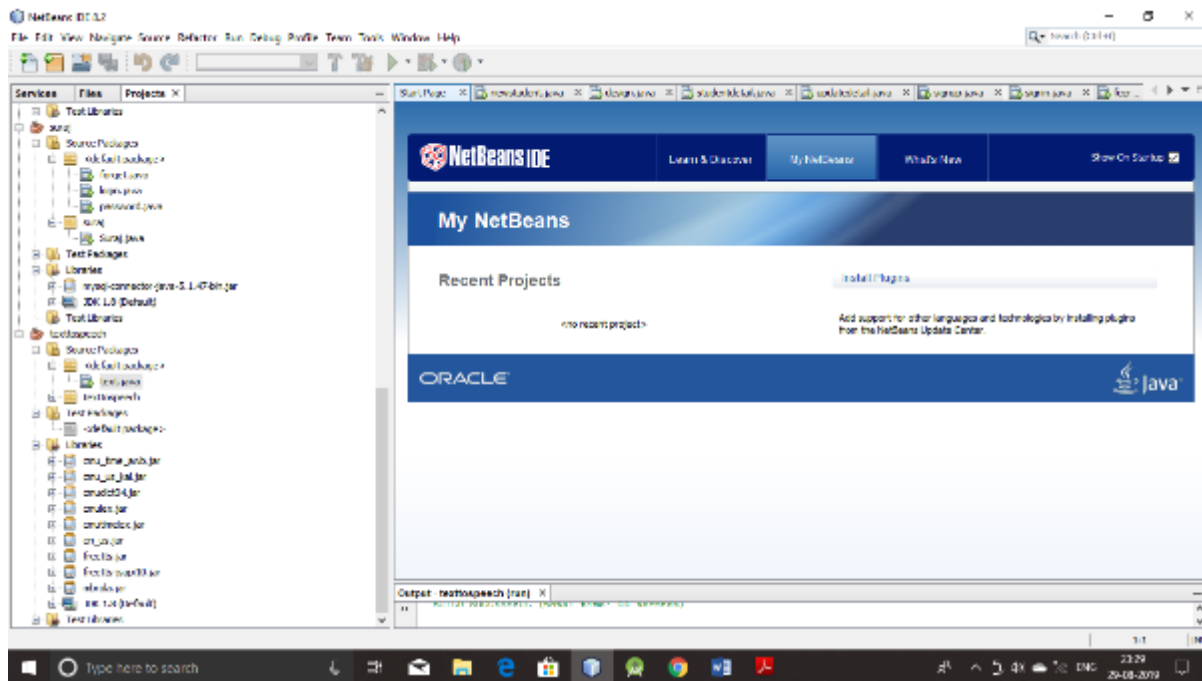
public static void main(String args[]) {
java.awt.EventQueue.invokeLater(new Runnable() {
    public void run() {
        new text().setVisible(true);
    }
});
}
private javax.swing.JButton jButton1;

private javax.swing.JButton jButton2;
private javax.swing.JLabel jLabel1;
private javax.swing.JLabel jLabel2;
private javax.swing.JScrollPane jScrollPane1;
private javax.swing.JTextArea ta;

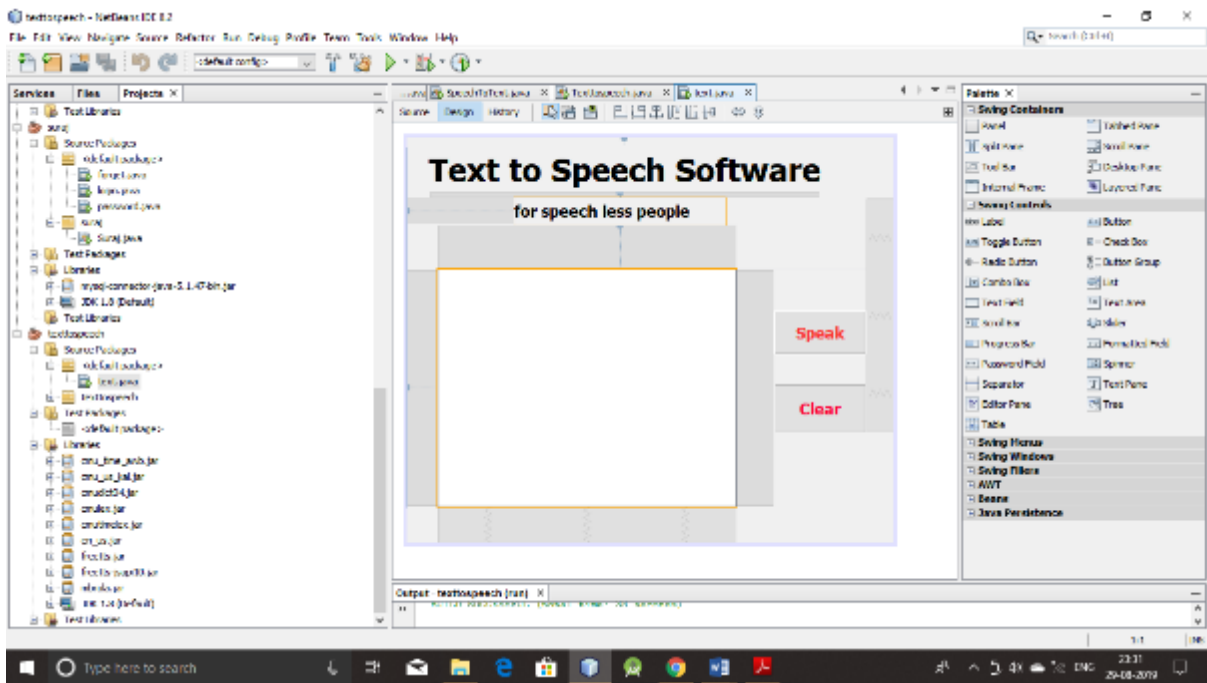
}

```

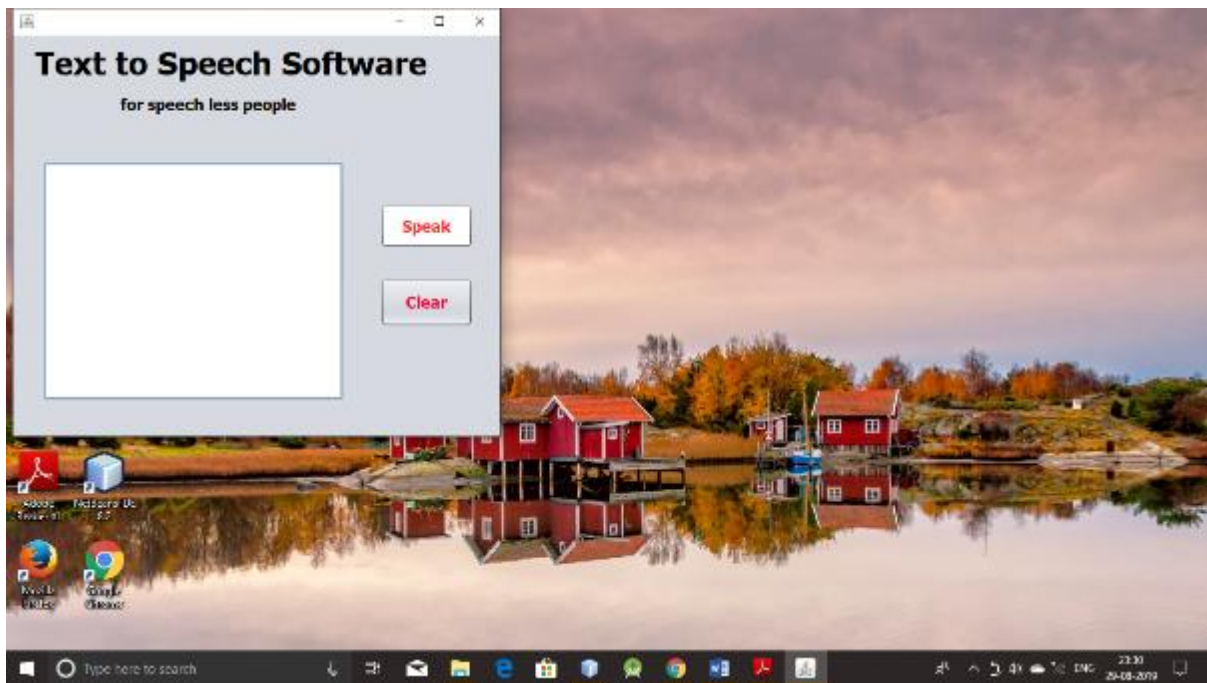
B.SAMPLE SCREEN



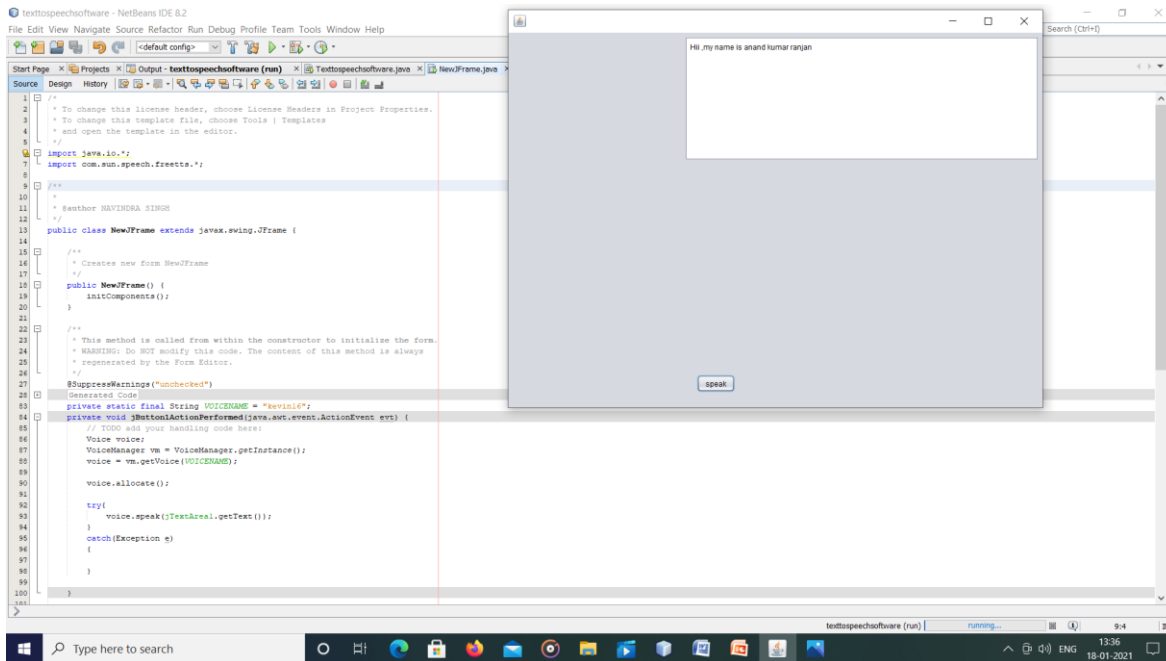
B.1



B.2



B.3



B.4