A MINI PROJECT REPORT

ON

SPEECH RECOGNITION SYSTEM USING GOOGLE API

Submitted in the partial fulfilment of the requirements for the award of

**BACHELOR
OF
INFORMATION TECHNOLOGY**

SUBMITTED BY

VANJARAPU BHANU CHARAN

21BK1A12C3

Under the guidance of

Dr. K. LITTLE FLOWER, PhD

Associate Professor

DEPARTMENT OF CSE

Giving Wings To Thoughts

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

St. Peter's Engineering College (UGC Autonomous)

Approved by AICTE, New Delhi, Accredited by NBA and NAAC with 'A' Grade,

Affiliated to JNTU, Hyderabad, Telangana

2021-2025

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that a Mini Project entitled **"SPEECH RECOGNITION SYSTEM USING GOOGLE API"** is carried out by **VANJARAPU BHANU CHARAN (21BK1A12C3)**, in partial fulfilment for the award of the degree of **BACHELOR OF INFORMATION TECHNOLOGY** is a record of Bonafide work done by her/him under my supervision during the academic year 2024– 2025.

INTERNAL GUIDE
Dr. K. Little Flower, PhD
Associate Professor
Department of CSE
St. Peter's Engineering College,
Hyderabad

HEAD OF THE DEPARTMENT
Department of CSE
St. Peter's Engineering College,
Hyderabad

PROJECT COORDINATOR
Mr. A. Senthil Murugan, M.E., (PhD)
Assistant Professor
Department of CSE
St. Peter's Engineering College,
Hyderabad

EXTERNAL EXAMINER

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

ACKNOWLEDGEMENT

We sincerely express our deep sense of gratitude to **Dr. K. LITTLE FLOWER**, for her valuable guidance, encouragement and cooperation during all phases of the project.

We are greatly indebted to our Project Coordinator **Mr. A. SENTHIL MURUGAN**, for providing valuable advice, constructive suggestions and encouragement without whom it would not been possible to complete this project.

It is a great opportunity to render our sincere thanks to Head of the Department, Computer Science and Engineering for her timely guidance and highly interactive attitude which helped us a lot in successful execution of the Project.

We are extremely thankful to our Principal **Dr. K. SREE LATHA**, who stood as an inspiration behind this project and heartfelt for her endorsement and valuable suggestions.

We respect and thank our secretary **Sri. T. V. REDDY**, for providing us an opportunity to do the project work at **St. PETER'S ENGINEERING COLLEGE** and we are extremely thankful to him for providing such a nice support and guidance which made us to complete the project.

We also acknowledge with a deep sense of reverence, our gratitude towards our parents, who have always supported us morally as well as economically. We also express gratitude to all our friends who have directly or indirectly helped us to complete this project work. We hope that we can build upon the experience and knowledge that we have gained and make a valuable contribution towards the growth of the society in coming future.

VANJARAPU BHANU CHARAN

(21BK1A12C3)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INSTITUTE VISION

To be a renowned Educational Institution that moulds Students into Skilled Professionals fostering Technological Development, Research and Entrepreneurship meeting the societal needs.

INSTITUTE MISSION

IM1: Making students knowledgeable in the field of core and applied areas of Engineering to innovate Technological solutions to the problems in the Society.

IM2: Training the Students to impart the skills in cutting edge technologies, with the help of relevant stake holders.

IM3: Fostering conducive ambience that inculcates research attitude, identifying promising fields for entrepreneurship with ethical, moral and social responsibilities.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

DEPARTMENT VISION

To be a vibrant nodal centre for Computer Science Engineering Education, Research that make the students to contribute to technologies for IT, IT-Enabled Services; to involve in innovative research on thrust areas of industry and academia; to establish start-ups supporting major players in the industry.

DEPARTMENT MISSION

DM1: Emphasize project-based learning by employing the state-of art technologies, algorithms in software development for the problems in inter-disciplinary avenues.

DM2: Involve stakeholders to make the students industry ready with training in skill-oriented computer application software.

DM3: Facilitate to learn the theoretical nuances of Computer Science, Computer Engineering courses and motivate to carry out research in both core and applied areas of CSE.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

PROGRAM EDUCATIONAL OBJECTIVES (PEOs)

PEO1: Graduates shall involve in research & development activities in industry and government arenas to conceive useful products for the society.

PEO2: Graduates shall be entrepreneurs contributing to national development in the fields of Computer Science based technologies.

PEO3: Graduates shall be team leaders working for software development, maintenance in the fields of software industry and government agencies.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

PROGRAM OUTCOMES (POs)

Engineering Graduates will be able to:

1: ENGINEERING KNOWLEDGE: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

2: PROBLEM ANALYSIS: Identify, formulate, research literature, and analyze complex engineering problems reaching substantiated conclusions using the first principles of mathematics, natural sciences, and engineering sciences.

3: DESIGN/DEVELOPMENT OF SOLUTIONS: Design solutions for complex engineering problems and design system components or processes that meet the specified needs withappropriate consideration for public health and safety, and the cultural, societal, and environmental considerations.

4: CONDUCT INVESTIGATIONS OF COMPLEX PROBLEMS: Use research-based knowledge and research methods including design of experiments, analysis, interpretation of data, and synthesis of the information to provide valid conclusions.

5: MODERN TOOL USAGE: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

6: THE ENGINEER AND SOCIETY: Apply reasoning informed by the contextual knowledgeto assess societal, health, safety, legal and cultural issues, and the consequent responsibilities relevant to the professional engineering practice

7: ENVIRONMENT AND SUSTAINABILITY: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of,and need for sustainable development.

8: ETHICS: Apply ethical principles and commit to professional ethics and, responsibilities andnorms of the engineering practice.

9: INDIVIDUAL AND TEAM WORK: Function effectively as an individual, and as a memberor leader in diverse teams, and multidisciplinary settings.

10: COMMUNICATION: Communicate effectively on complex engineering activities with theengineering community and with society at large, such as being able to comprehend and draft effective reports and design documentation, make an effective presentation, give, and receive clear instructions.

11: PROJECT MANAGEMENT AND FINANCE: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's work, as amember and leader in a team, to manage projects and in a multidisciplinary environment.

12: LIFE-LONG LEARNING: Recognize the need for, and have the preparation and ability toengage in independent and lifelong learning in the broadcast context of technological changes.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

PROGRAM SPECIFIC OBJECTIVES (PSO'S)

PSO1

Design and develop computing subsystems for data storage, communication, informationprocessing, and knowledge discovery.

PSO2

Design algorithms for real world problems focusing on execution, complexity analysisconsidering the security, cost, quality, and privacy parameters in software development.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

DECLARATION

We declare that a Mini Project entitled "**SPEECH RECOGNITION SYSTEM USING GOOGLE API**" is an Original Work submitted by the following group memberswho have actively contributed and submitted in partial fulfilment for the award of degree in **"Bachelor of Information Technology",** at **St. Peter's Engineering College**, Hyderabad, and this project work has not been submitted by me to any other college or university for the award of any kind of degree.

Group No: 04

Program: B. Tech

Branch: Information Technology

Mini Project Title: Speech Recognition System Using Google API

Date Submitted:

Name	Roll Number	Signature
VANJARAPU BHANU CHARAN	**21BK1A12C3**	

ABSTRACT

This project proposes an accurate system for speech recognition, which has been developed with ease and accessibility in transcribing spoken words into text, thereby making it possible for both personal and professional purposes in speech. Both interfaces of user preference are provided by this system built with the Google Cloud Speech API: Notepad and Web. The Notepad Interface interfaces well with the Notepad application, so transcriptions will be viewed and saved as .txt files. This interface itself uses keyboard input simulation by pynput and pywinauto to interact with Notepad, giving the choice for text-based document storage in a quite straightforward manner. The Web Interface provides a facility to browser-based facility; users would be authorized to download a transcription file as a Portable Document format to make their transcriptions easier to download and exchange. The System Homepage would allow the user, according to his or her requirement, to switch easily between options between Notepad or Web Interface. These include audio preprocessing, feature extraction, and language modeling that power the critical Google Cloud Speech API processes for reliable high-accuracy transcriptions. This dual interface solution provides flexibility and precision in voice-to-text experiences, catering to a diverse range of accessibility and industry-specific requirements.
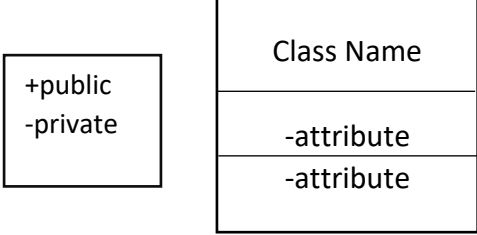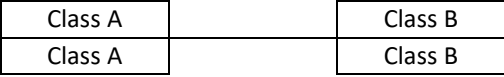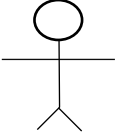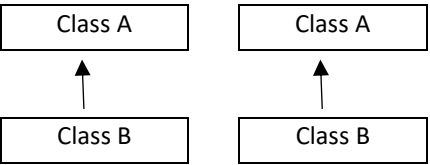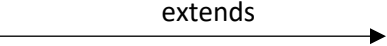
LIST OF FIGURES
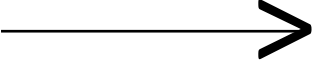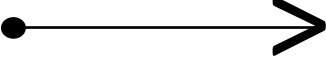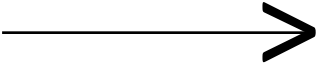
LIST OF TABLES

LIST OF SYMBOLS

S.No.	NOTATION NAME	NOTATION	DESCRIPTION
1.	Class	+public -private / Class Name -attribute -attribute	Represents a collection of similar entities grouped together.
2.	Association	Class A — Class B / Class A — Class B	Association represents static relationships between classes. Role represents the way the two classes see each other.
3.	Actor		It aggregates several classes into a single class.
4.	Aggregation	Class A ← Class B / Class A ← Class B	Interaction between the system and external environment.
5.	Relation (extends)	extends →	Extends relationship is used when one use case is similar to another use case but does a bit more.
6.	Communication	——————	Communication between various use cases.
7.	State	State	State of the processes.

8.	Initial State		Initial state of the object
9.	Final state		Final state of the object
10.	Control flow		Represents various control flow between the states

TABLE OF CONTENTS

CHAPTER 1
INTRODUCTION

It completely revolutionizes the accessibility of speech recognition technology with an accurate, easy-to-use solution for transcribing recorded or live speech to text through the application of the Google Cloud Speech API. Two very different interfaces are in place: with the Notepad Interface, transcriptions are saved directly as .txt files, while the Web Interface lets users download their transcriptions as shareable PDF files. The system's thorough design with respect to accessibility and convenience ensures that there are perfect transitions from interface to interface, hence also suitable for both work and personal life. Whether you're transcribing meetings, lectures, or even just personal notes, powerful accuracy and flexibility may help streamline the process using the system. This system boasts audio pre-processing and language modelling as guarantees for high-quality results, providing exceedingly potent functionality designed to bring about better productivity, accessibility, and communication capabilities in multiple industries.

1.1 PROBLEM STATEMENT

Most industries need speech-to-text transcription to be accurate and efficient, but the solutions developed so far are error-prone, expensive, and painstaking. Advanced tools fail with complicated speech or noise in the background. There is thus a great need for a very robust yet quite straightforward system to work by quick and accurate speech-to-text conversion while allowing flexibility about where you can store and access the transcriptions. This project is intended to provide a dual-interface solution that would have the functionality of saving text into Notepad, in addition to an option for downloading that could be saved as a PDF file. It will address all the needs of the user, personal as well as professional applications, while working on making the application accessible for users with speech impairments.

1.2 OBJECTIVES

The objective of this project:

- This project aims to design a speech-to-text-friendly system

- Integrate the Google Cloud Speech API for accurate transcription

- Text storage into two interfaces: Notepad and Web

- Higher accuracy and reliability in transcriptions;

- Increased accessibility with a speech-impaired user; and

- Productivity in professional and personal use cases.

1.3 MOTIVATION

The motivation for developing our system is its use of the Google Cloud Speech API for providing accurate speech-to-text conversions, and offering users dual interface options for convenience. With both a Notepad interface to store fast text and a Web interface to download PDFs, the application comfortably suits diverse user preferences. It will also target making access easy for people having speech impairments so that they, too, can easily translate their speech into text. The system will introduce, with simplicity, accuracy, and flexibility, improved productivity and communication across work and personal contexts.

1.4 EXISTING SYSTEM

There are several existing systems of speech recognition that are widely used today such as Google Speech-to-Text, Microsoft Azure Speech to Text, Apple siri etc. The Google Cloud Speech-to-Text API offers reliable and scalable speech recognition capabilities, making it suitable for a wide range of applications, including transcription services, voice assistants, call center analytics, voice command processing, and more.

DRAWBACKS OF EXISTING SYSTEM

- Accuracy Limitations
- Privacy and Data Security
- Language Support and Model Limitations
- Customization Complexity

1.5 PROPOSED SYSTEM

The proposed system makes use of an easy to use speech-to-text tool provided by Google Cloud Speech API. This tool gives two kinds of interfaces: the Notepad Interface, which translates speech into a Notepad file saved as `.txt` document, and the Web Interface, whereby users can download their transcription as PDF files for easier sharing and archiving. The flexibility of the system allowed users

to switch interfaces according to current needs. It provided high accuracy and reliability even in noisy environments with advanced speech recognition, audio pre-processing, and language modelling. Designed with accessibility in mind, this system is supportive of people with speech impairments and thus fosters better cross-professional and private communication. Combining the aspects of accurate transcription with versatile output options, it constitutes an efficient, powerful solution for vast numbers of users.

1.6 SCOPE

The goal of this project is to design an Adaptive and Accessible Speech-to-Text System using the Google Cloud Speech API. It will offer two interfaces: Notepad Interface for transcribing directly into `.txt` files; Web Interface that lets people take transcriptions as PDFs. With high precision in its design, the system is accessible for people with impairments in their speech, thereby ensuring that the system caters to diverse needs of its users. As a multi-purpose tool, it is adaptable to use in education, business, or personal documentation, which increases productivity and accessibility across any point of use case.

CHAPTER 2

LITERATURE SURVEY

NAME	YEAR	AUTHOR NAME	Tools	ADVANTAGES	LIMITATIONS
Automatic speech recognition: An evaluation of Google Speech	2015	Stenman, M.	Google Speech API	Evaluates Google Speech in real-world conditions, provides insights into system performance	Limited to Android, may face challenges in noisy environments
Speech recognition systems: A comparative review	2017	Matarneh, R., Maksymova, S., Lyashenko, V., & Belova, N.	Various speech recognition systems	Provides a comprehensive comparison of multiple systems, highlighting strengths and weaknesses	Focused on a wide range of systems, lacking in-depth technical details for individual tools
Speech recognition application for the speech impaired using the Android-based Google Cloud Speech API	2018	Anggraini, N., Kurniawan, A., Wardhani, L. K., & Hakiem, N.	Google Cloud Speech API, Android Platform	Improves communication for speech-impaired individuals, accessible on mobile platforms	Limited to Android, may face challenges in noisy environments
Development of the Speech-to-Text Chatbot Interface Based on Google API	2019	Shakhovska, N., Basystiuk, O., & Shakhovska, K.	Google Cloud Speech API, Chatbot Interface	Facilitates easy user interaction via a conversational interface, integrates seamlessly with Google API	Challenges in implementing speech recognition for complex, multi-turn conversations
Speech Recognition System using Natural Language Processing	2024	Jindam, S., Devaraju, G., Sindhu, T., Mudiraj, K. A., & Chittiprolu, A.	Google Cloud Speech API, Notepad	Demonstrates integration with desktop applications, effective for hands-free text input	Limited to text input (no voice commands or complex operations beyond text)

CHAPTER 3

SYSTEM REQUIREMENTS SPECIFICATIONS

3.1 HARDWARE REQUIREMENTS

- Processor   :   I3/Intel Processor
- Hard Disk   :   160GB
- RAM   :   8Gb

3.2 SOFTWARE REQUIREMENTS

- Operating System   :   Windows 10,11
- IDE   :   PyCharm
- Libraries Used   :   Pynput, Pywinauto, Pyaudio, Flask
- Technology   :   PYTHON, HTML, CSS, JAVA SCRIPT

CHAPTER 4
SYSTEM DESIGN

4.1 SYSTEM ARCHITECTURE

System Architecture
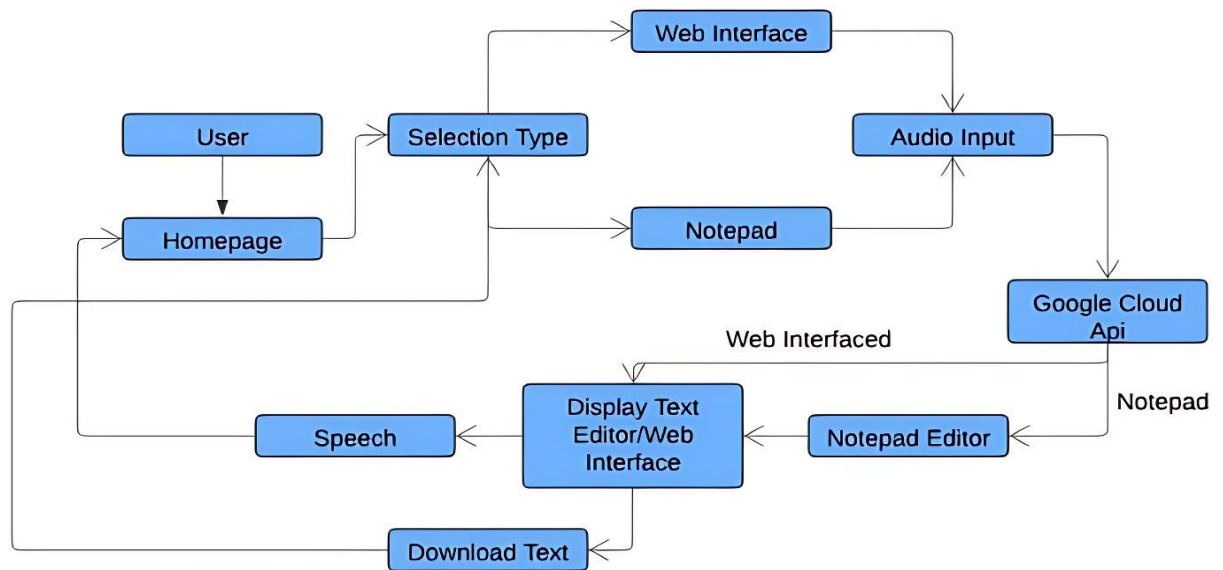
Fig-4.1 SYSTEM ARCHITECTURE

4.1.1 Architecture Description

- User: The human who interacts with the system.

- Start: Responsible for starting the system.

- Audio Input: Responsible for capturing the user's speech.

- Google API: Responsible for interacting with the Google Cloud Platform.

- Home Page: Displays the home page to the user.

- Speech: Converts the user's text into speech.

- Display Text: Displays the user's speech to the user.

- Notepad: A text editor that allows the user to edit text.

- The system architecture is designed to be modular and scalable.

4.2 UML DIAGRAMS

- UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

- The goal is for UML to become a common language for creating models of object-oriented computer software. In its current form UML is comprised of two major components: A Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

- The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

- The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

- The UML is a very important part of developing objects-oriented software. The UML uses mostly graphical notations to express the design of software projects.

4.2.1 Use Case Diagram

- A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases.
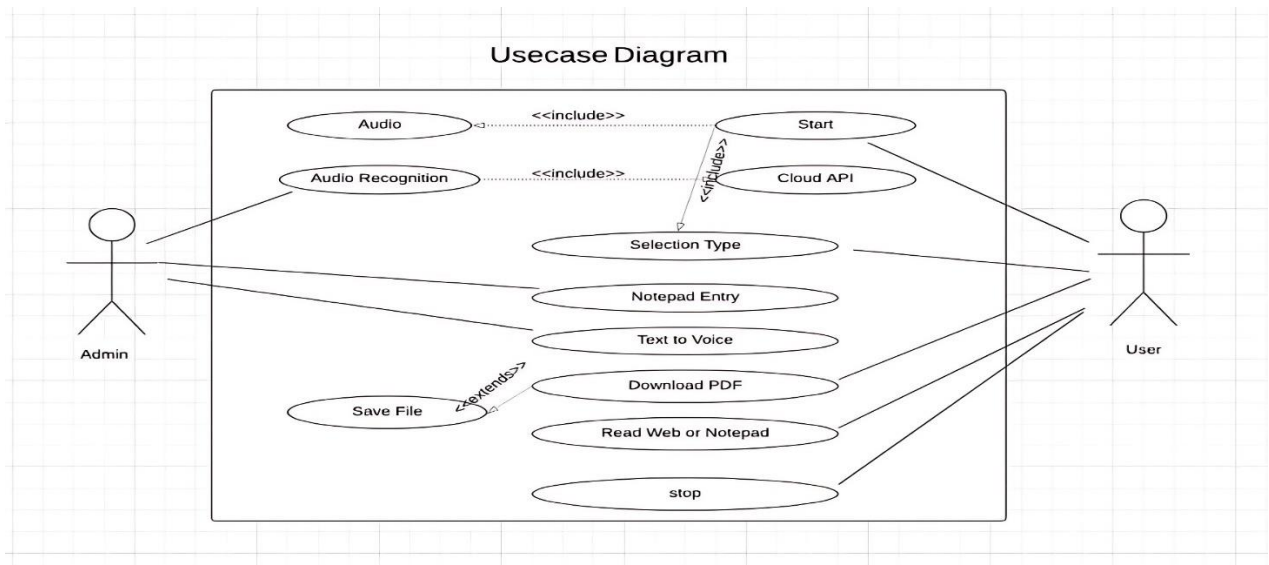
Fig-4.2.1 USE CASE DIAGRAM

4.2.2 Class Diagram

- In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains which information.
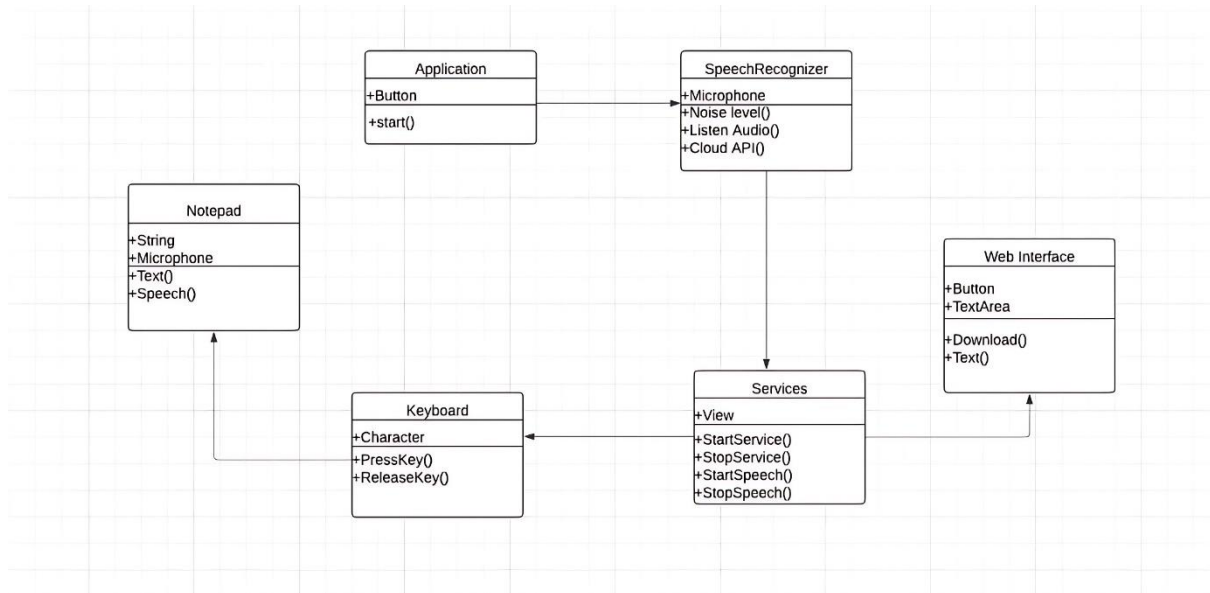
Fig-4.2.2 CLASS DIAGRAM

4.2.6 Activity Diagram

- Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modelling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.
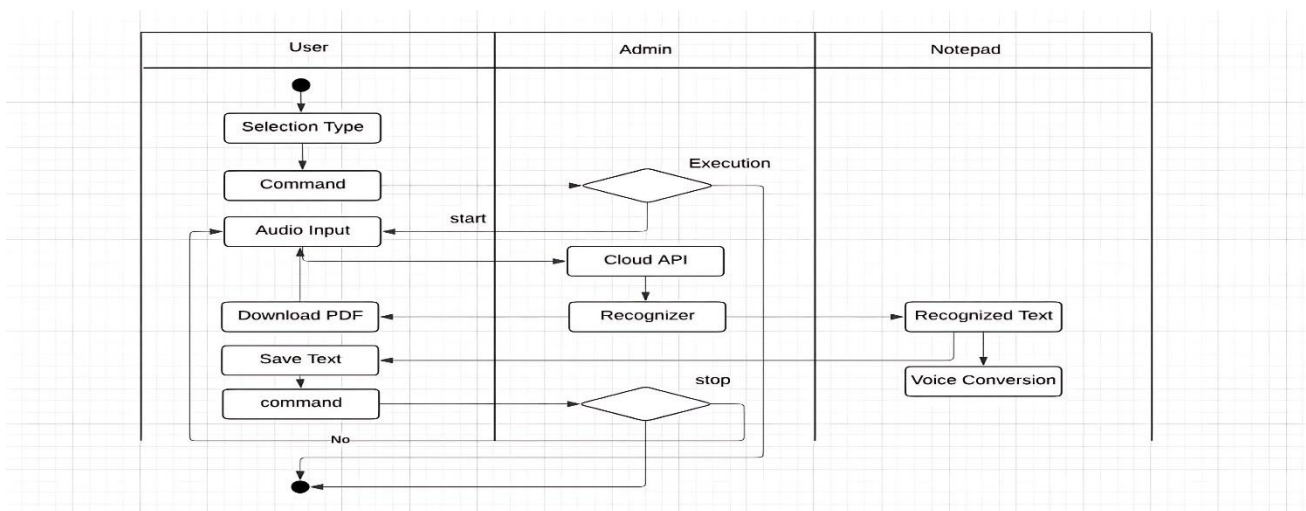
Fig-4.2.3 ACTIVITY DIAGRAM

CHAPTER 5

IMPLEMENTATION

5.1 ENVIRONMENTAL SETUP

Installing Python:

1. To download and install Python visit the official website of

2. Python https://www.python.org/downloads/ and choose your version.

Click on the "Download" button next to the chosen edition to start the download.
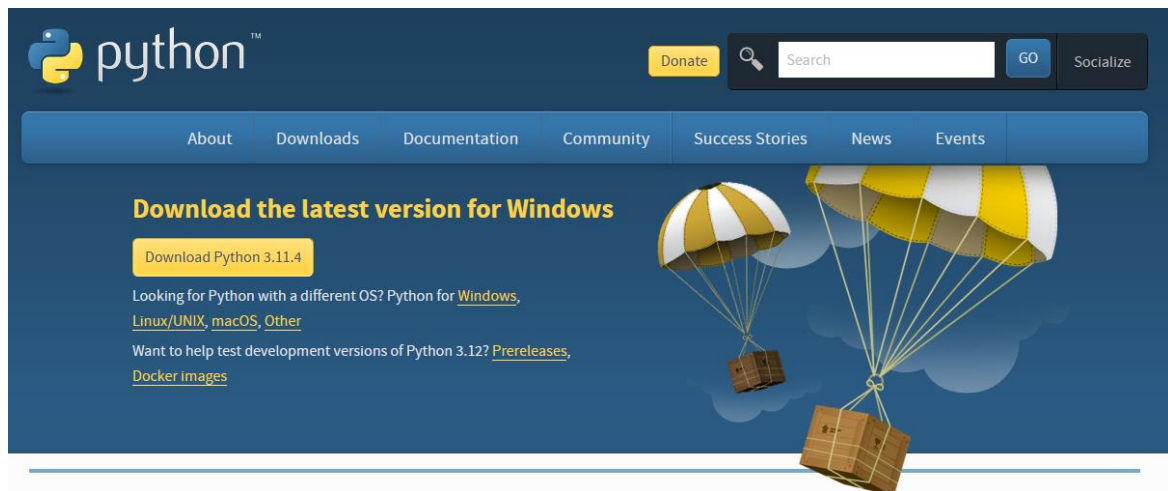
Fig-5.1 Python installation

1. Once the download is complete, locate the downloaded setup file and run it.

2. Once the download is complete, run the exe for install Python. Now click on Install Now.

3. You can see Python installing at this point.

4. When it finishes, you can see a screen that says the Setup was successful. Now click on "Close".

Installing PyCharm:

Fig-5.1.1 PyCharm installation

1. To download PyCharm visit the website https://www.jetbrains.com/pycharm/download/ and click the"DOWNLOAD" link under the Community Section.

2. Once the download is complete, run the exe for install PyCharm. The setup wizard should have started.   Click "Next".

3. On the next screen, Change the installation path if required. Click "Next".

4. On the next screen, you can create a desktop shortcut if you want and click on "Next".

5. Choose the start menu folder. Keep selected JetBrains and click on "Install".

6. Wait for the installation to finish.

7. Once installation finished, you should receive a message screen that PyCharm is installed. If you want to go ahead and run it, click the "Run PyCharm Community Edition" box first and click "Finish" .

8. After you click on "Finish" the Following screen will appear.
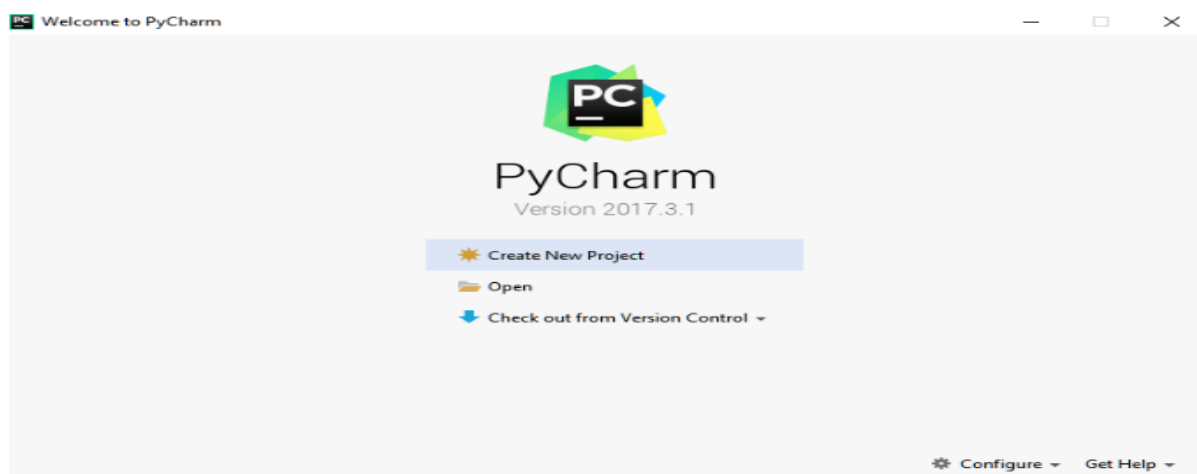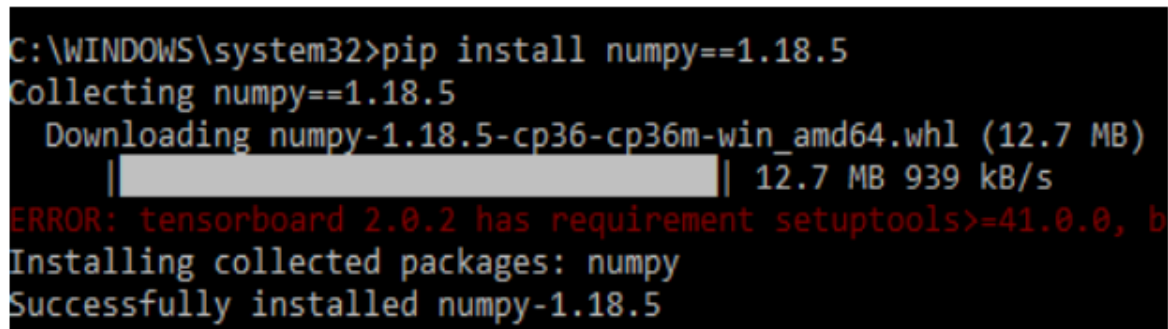
Fig-5.1.2 PyCharm interface

9. You need to install some packages to execute your project in a proper way.

10. Open the command prompt/ anaconda prompt or terminal as administrator.

11. The prompt will get open, with specified path, type "pip install package name" which you want to install (like numpy, pandas, seaborn, scikit-learn, matplotlib.pyplot)

Ex: pip install numpy.

```
C:\WINDOWS\system32>pip install numpy==1.18.5
Collecting numpy==1.18.5
  Downloading numpy-1.18.5-cp36-cp36m-win_amd64.whl (12.7 MB)
    |                                              | 12.7 MB 939 kB/s
ERROR: tensorboard 2.0.2 has requirement setuptools>=41.0.0, b
Installing collected packages: numpy
Successfully installed numpy-1.18.5
```

Fig-5.2 Installing libraries

5.2 EXPLAINATION

1.System

- **1.1 Record Voice:** The system captures audio input from the user through a microphone or similar device.

- **1.2 Convert Voice:** The audio captured is transmitted to the Google Cloud Speech API, which processes it and converts it into text, using advanced speech recognition techniques.

- **1.3 Outputting Options:** The text output is passed on to either the Notepad Interface or the Web Interface that a user may wish to select, therefore offering varied output options.

- **1.4 Notepad Interface:** If the Notepad Interface is selected, then the text is shown within the Notepad application using pynput and pywinauto to send keyboard inputs, so that it may be saved as a `.txt` file.

- **1.5 Web Interface:** Once the Web Interface has been selected, the transcribed text is available for download as a PDF file, thereby saving a great format for preservation and sharing.

2. User:

- **2.1 Launch:** From the System Homepage, the user launches the application and chooses their preferred interface: either Notepad or Web.

- **2.2 Record Voice:** After choosing the interface, the user initiates voice recording to begin the transcription process.

- **2.3 Save:** The user gets an option to save the output text. In Notepad Interface, texts are saved as `.txt`, whereas in Web Interface, the transcription can be downloaded as PDF.

- **2.4 Result:** The final output will be there in the converted text and will be shown on the selected interface by the user, that is either Notepad or Web.

5. 3 ALGORITHMS :

HMM:

- Hidden Markov Models (HMMs) are one of the cornerstones to be found in the Google Speech API Python library and present a framework for speech recognition as this model a sequence of observable outputs, namely speech features and hidden states that would presumably capture underlying linguistic or acoustic information.

- HMM is applied in audio-to-text conversions to make precise predictions. HMM employs a probabilistic model based on how source sequences of spoken language can be represented.

- HMMs do not rely entirely on the Markov property, where they assume each hidden state depends only on the preceding state. This makes the model simpler while preserving the accuracy.

- A training dataset is applied to the library to estimate critical HMM parameters, including state transition probabilities and emission probabilities.

- HMM acoustics: Typically, observable HMM outputs include acoustic features like MFCCs.

- MFCCs efficiently represent speech sounds in order to process them.

- The Viterbi algorithm is used to determine a sequence of the underlying states, say words or phonemes, given the observed features, in HMMs.

- Much of the complexity in HMM implementation is abstracted out by the Google Speech API, which enables developers to apply sophisticated speech recognition without massive technical expertise in probabilistic modeling.

- The HMMs are very good for speech recognition because it can treat sequences of any length

and thus could be very adaptable to different speech patterns.

- The quality of the training data is crucial; the more diverse and representative dataset provides a robust model in terms of accent, speaking style, and language.
- Adaptation techniques could be used to further adjust HMMs to specific speakers or environments with higher accuracy for specific users or environments.
- HMMs are widely used for speech recognition because they are stable and scalable, though often combined with language models to increase accuracy.
- HMMs cannot capture complex patterns, therefore, modern systems introduce more advanced techniques including deep learning models.
- Regardless of that HMMs remain as an essential component within the Google Speech API, which

  guarantees a strong foundation for developers when developing reliable and effective applications for speech recognition.
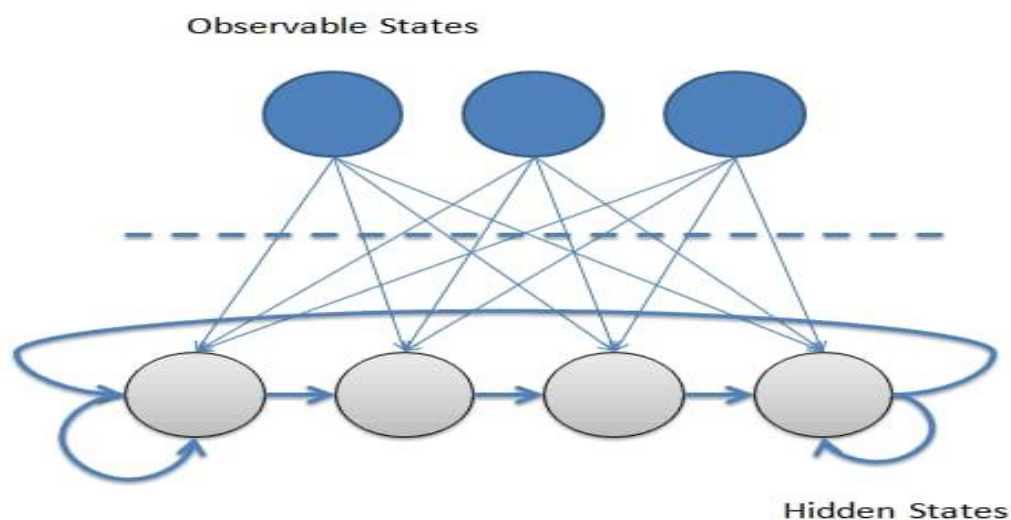
Fig-5.3.1 HMM Model

DEEP LEARNING AND CTC:

- Deep learning is a subset of machine learning that is really powerful for processing complex data, such as speech using neural networks with lots of layers. Below is the Python programming code that uses the Google API to convert speech into text by processing the data on a computer using deep learning.

- Deep Neural Networks: It captures deep audio patterns and representations, bettering the accuracy and robustness of the transcriptions of spoken words.

- The deep learning models of Google for speech recognition are built on a massive diverse set of data in this way generalizing well across different accents, languages, and speaking styles.

- This deep approach to learning has enabled the API to make transcription near real-time and suitable for the application such as voice assistants, live transcription, and much more.

- Such models frequently use recurrent layers: LSTM or GRU in dealing with variable-length sequences of audio.

- Convolutional layers are also applied to obtain sound features from spectrograms of audio to increase the accuracy of the representation of speech.

- Google speech recognition models are constantly upgraded due to research in deep learning, as these models update themselves continuously for best performance.

- Deep models show high accuracy and fast response time if processing is done before and after with proper work as compared to traditional methods.

- Deep learning is very flexible in adapting these models to specific tasks for speech recognition, so reliable performance can be gained on different applications.

- Though deep learning provides better accuracy improvements, it also demands massive computational resources along with large training data to be the best performer.

5.3 MODULE DESCRIPTION

- Pynput
- Pywinauto
- Speech Recognition
- Flask

Pynput:

- Pynput is a Python library used in the control and monitoring of input devices such as a keyboard and a mouse.

- It allows simulation of key presses and releases and mouse actions, making it very useful in automation for interactive applications.

- The engine can listen for and respond to keyboard and mouse events in real-time and cross-platform to all three Windows, macOS, and Linux.

- The user can determine the position of the mouse cursor and execute mouse actions like clicking, scrolling, and sending text keystrokes into any foreground application.

- Such simplicity in API makes it a great fit for automation of GUI applications, games, and other automations.

Pywinauto:

- Pywinauto is a Python library aimed at automating graphical interfaces of Windows applications.

- The program can programmatically control windows, dialogs, buttons, and more GUI elements- to work with any Windows application ranging from legacy to modern applications.

- It allows one to automate tasks, do application tests, and manipulate software interfaces without manual input through pywinauto

- It has simple syntax and allows the detection and interaction of GUI elements with attributes like a class name, title or control type.

- From simulating keyboard and mouse actions to checking window states, pywinauto streamlines GUI automation for Windows in Python.

Speech Recognition:

- Speech Recognition is an interface to various speech recognition services for Python. It makes it quite simple to add voice recognition support to applications.

- It supports multiple engines, including Google Web Speech API, Microsoft Bing Voice Recognition, and CMU Sphinx.

- It enables transcription of speech to text, aiding its functionalities, such as voice commands, transcription, and natural language processing.

- Audio recording by means of microphones or upload audio files are very helpful with a non-intrusive API and to retrieve the recognized text.

- Speech Recognition is helpful for building applications or voice-activated features using Python for an automated transcription.

Flask

- Flask is a lightweight and modern web framework for Python, perfectly suitable for speedy development of small web applications.

- Flask is WSGI compliant and utilizes Werkzeug toolkit alongside Jinja2 template engine for building of the web application. It can thus be applied in various projects in web development.

- Flask is minimalist, yet modestly flexible; thus, it is great for small to medium-sized applications. Flask offers important components in routing, processing HTTP requests, and processing responses.

- With minimalist approach, it is providing developers an avenue to include only the features that may be needed for them, and high on customizing.

- Flask's innumerable documentation and friendly community make it such an approachable yet powerful thing in Python web development.

5.4 SOURCE CODE

Home.html

```html
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Home - Speech-to-Text Application</title>
  <style>

    body {
      font-family: Arial, sans-serif;
      display: flex;
      flex-direction: column;
      justify-content: center;
      align-items: center;
      min-height: 100vh;
      margin: 0;
      background: linear-gradient(135deg, #6a11cb, #2575fc);
      color: #333;
    }

    .container {
      background: #ffffff;
      border-radius: 16px;
      box-shadow: 0 6px 15px rgba(0, 0, 0, 0.15);
      padding: 20px;
      max-width: 800px;
      width: 90%;
      overflow: hidden;
    }

    h1 {
      text-align: center;
      color: #333;
      margin-bottom: 30px;
      font-size: 2rem;
    }

    .row {
      display: flex;
      align-items: center;
      border: 1px solid #eaeaea;
      border-radius: 12px;
      overflow: hidden;
      margin-bottom: 20px;
      transition: transform 0.3s ease, box-shadow 0.3s ease;
```

```css
    }

    .row:hover {
      transform: scale(1.02);
      box-shadow: 0 8px 20px rgba(0, 0, 0, 0.15);
    }

    .row img {
      width: 50%;
      height: 100%;
      object-fit: cover;
    }

    .text-content {
      padding: 20px;
      width: 50%;
    }

    .text-content h2 {
      color: #2575fc;
      font-size: 1.5rem;
      margin-bottom: 10px;
    }

    .text-content p {
      color: #666;
      line-height: 1.6;
      font-size: 0.95rem;
    }

    .option {
      display: inline-block;
      margin-top: 15px;
      padding: 10px 20px;
      background-color: #6a11cb;
      color: #ffffff;
      font-weight: bold;
      text-decoration: none;
      border-radius: 6px;
      transition: background-color 0.3s ease, transform 0.2s ease;
      text-align: center;
    }

    .option:hover {
      background-color: #2575fc;
      transform: scale(1.05);
    }
  </style>
</head>
<body>
<h1 style="color:black ; font-size: 40px;">SPEECH TRANSCRIPTION</h1>
```

```html
<div class="container">
    <h1>Welcome to Speech-to-Text Application</h1>
    <div class="row">
        <img style="border-radius: 15px; padding: 5px;" src="webstt.jpg" alt="Web Interface"/>
        <div class="text-content">
            <h2>Web Interface</h2>
            <p>The Web Interface offers a seamless, dynamic online experience, perfect for users
looking to convert speech to text directly in a web environment. It's accessible from any device with
internet connectivity and integrates all major functionalities of the app.</p>
            <a href="http://127.0.0.1:5556/transcript" class="option">Explore Web Interface</a>
        </div>
    </div>

    <div class="row">
        <div class="text-content">
            <h2>Notepad Interface</h2>
            <p>The Notepad Interface is designed for users who prefer a direct text integration. Convert
speech to text quickly and efficiently in a Notepad-style view, ideal for rapid note-taking and
editing.</p>
            <a href="http://127.0.0.1:5555/notepad" class="option">Explore Notepad Interface</a>
        </div>
        <img style="border-radius: 15px; padding:5px;" src="notepadstt.jpg" alt="Notepad Interface"/>
    </div>
</div>

</body>
</html>
```

Speech.html

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Home - Speech-to-Text Application</title>
    <style>

        body {
            font-family: Arial, sans-serif;
            display: flex;
            flex-direction: column;
            justify-content: center;
            align-items: center;
            min-height: 100vh;
            margin: 0;
            background: linear-gradient(135deg, #6a11cb, #2575fc);
            color: #333;
        }
```

```css
.container {
  background: #ffffff;
  border-radius: 16px;
  box-shadow: 0 6px 15px rgba(0, 0, 0, 0.15);
  padding: 20px;
  max-width: 800px;
  width: 90%;
  overflow: hidden;
}

h1 {
  text-align: center;
  color: #333;
  margin-bottom: 30px;
  font-size: 2rem;
}

.row {
  display: flex;
  align-items: center;
  border: 1px solid #eaeaea;
  border-radius: 12px;
  overflow: hidden;
  margin-bottom: 20px;
  transition: transform 0.3s ease, box-shadow 0.3s ease;
}

.row:hover {
  transform: scale(1.02);
  box-shadow: 0 8px 20px rgba(0, 0, 0, 0.15);
}

.row img {
  width: 50%;
  height: 100%;
  object-fit: cover;
}

.text-content {
  padding: 20px;
  width: 50%;
}

.text-content h2 {
  color: #2575fc;
  font-size: 1.5rem;
  margin-bottom: 10px;
}

.text-content p {
  color: #666;
```

```css
        line-height: 1.6;
        font-size: 0.95rem;
      }

      .option {
        display: inline-block;
        margin-top: 15px;
        padding: 10px 20px;
        background-color: #6a11cb;
        color: #ffffff;
        font-weight: bold;
        text-decoration: none;
        border-radius: 6px;
        transition: background-color 0.3s ease, transform 0.2s ease;
        text-align: center;
      }

      .option:hover {
        background-color: #2575fc;
        transform: scale(1.05);
      }
   </style>
</head>
<body>
<h1 style="color:black ; font-size: 40px;">SPEECH TRANSCRIPTION</h1>
<div class="container">
   <h1>Welcome to Speech-to-Text Application</h1>
   <div class="row">
     <img style="border-radius: 15px; padding: 5px;" src="webstt.jpg" alt="Web Interface"/>
     <div class="text-content">
        <h2>Web Interface</h2>
        <p>The Web Interface offers a seamless, dynamic online experience, perfect for users
looking to convert speech to text directly in a web environment. It's accessible from any device with
internet connectivity and integrates all major functionalities of the app.</p>
        <a href="http://127.0.0.1:5556/transcript" class="option">Explore Web Interface</a>
     </div>
   </div>

   <div class="row">
     <div class="text-content">
        <h2>Notepad Interface</h2>
        <p>The Notepad Interface is designed for users who prefer a direct text integration. Convert
speech to text quickly and efficiently in a Notepad-style view, ideal for rapid note-taking and
editing.</p>
        <a href="http://127.0.0.1:5555/notepad" class="option">Explore Notepad Interface</a>
     </div>
     <img style="border-radius: 15px; padding:5px;" src="notepadstt.jpg" alt="Notepad Interface"/>
   </div>
</div>

</body>
```

</html>

Style.css

```css
*,*:after,:before{
    -webkit-box-sizing: border-box;
    -moz-box-sizing: border-box;
    -ms-box-sizing: border-box;
    box-sizing: border-box;
}
body{
    font-family: arial;
    font-size: 16px;
    margin: 0;
    background: linear-gradient(to right bottom,#d13cff,#031f6a);
    display: flex;
    align-items: center;
    justify-content: center;
    min-height: 100vh;
    color: #000;
}
.voice_to_text{
    width: 600px;
    text-align: center;
}
#convert_text{
    width: 100%;
    height: 200px;
    border-radius:10px;
    resize:none;
    padding:10px;
    font-size: 20px;
    margin-bottom: 10px;
}
h1{
    font-size:50px;
    color: #fff;
}
button{
    padding: 12px 20px;
    background: #0ea4da;
    border: 0;
    border-radius: 5px;
    cursor:pointer;
    color:#fff;

}
```

Script.js

```javascript
click_to_convert.addEventListener('click',function(){
    var speech=true;
    window.SpeechRecognition=window.webkitSpeechRecognition;
    const Recognition = new SpeechRecognition();

    Recognition.addEventListener('result',e=>{
        const transcript=Array.from(e.results)
        .map(result=>result[0])
        .map(result=>result.transcript)

        convert_text.innerHTML+= ' ' + transcript;
    })
    if(speech==true){
        Recognition.start();
    }
})

document.getElementById("download_pdf").addEventListener("click", downloadPDF);

function downloadPDF() {
    const { jsPDF } = window.jspdf;
    const doc = new jsPDF();
    const text = document.getElementById("convert_text").value;

    doc.text(text, 10, 10);
    doc.save("Voice_to_Text.pdf");
}
```

web.py

```python
from flask import Flask, render_template, request
apply = Flask(__name__)


@apply.route("/transcript")
def home():
    return render_template("speech.html")


if __name__ == "__main__":
    apply.run(debug=True, port=5556)
```

Speech.py

```python
import time

import speech_recognition as sr
from pynput.keyboard import Key, Controller
from pywinauto import application

from flask import Flask, render_template, request
import pyttsx3

apply = Flask(__name__)


@apply.route("/notepad")
def home():
    return render_template("index.html")


@apply.route("/speech", methods=['POST'])
def speech():
    output = request.form.get('button')
    if output == 'button':
        r = sr.Recognizer()
        keyboard = Controller()
        app = application.Application()

        with sr.Microphone() as source:
            r.adjust_for_ambient_noise(source)
            print("Now You Can Speak")
            audio = r.listen(source)
            text = r.recognize_google(audio) + " "
            if text == 'start ':
                app.start("Notepad.exe")

                while text != 'stop ':
                    print("Speak")
                    audio = r.listen(source)
                    try:
                        text = r.recognize_google(audio) + " "
                        if text == 'stop ':
                            pass
                        elif text == 'next line ':
                            keyboard.press(Key.enter)
                            keyboard.release(Key.enter)
                        elif text == 'select up ':
                            keyboard.press(Key.shift)
                            keyboard.press(Key.up)
                            keyboard.release(Key.up)
                            keyboard.release(Key.shift)
```

```python
    elif text == 'select down ':
        keyboard.press(Key.shift)
        keyboard.press(Key.down)
        keyboard.release(Key.down)
        keyboard.release(Key.shift)
    elif (text == 'select left 5 ') or (text == 'select left five '):
        keyboard.press(Key.shift)
        for x in range(0, 6):
            keyboard.press(Key.left)
            keyboard.release(Key.left)
        keyboard.release(Key.shift)
    elif (text == 'select right 5 ') or (text == 'select right five '):
        keyboard.press(Key.shift)
        for x in range(0, 6):
            keyboard.press(Key.right)
            keyboard.release(Key.right)
        keyboard.release(Key.shift)
    elif text == 'select left ':
        keyboard.press(Key.shift)
        keyboard.press(Key.left)
        keyboard.release(Key.left)
        keyboard.release(Key.shift)
    elif text == 'select right ':
        keyboard.press(Key.shift)
        keyboard.press(Key.right)
        keyboard.release(Key.right)
        keyboard.release(Key.shift)
    elif text == 'left ':
        keyboard.press(Key.left)
        keyboard.release(Key.left)
    elif text == 'right ':
        keyboard.press(Key.right)
        keyboard.release(Key.right)
    elif text == 'up ':
        keyboard.press(Key.up)
        keyboard.release(Key.up)
    elif text == 'down ':
        keyboard.press(Key.down)
        keyboard.release(Key.down)
    elif (text == 'left five ') or (text == 'left 5 '):
        for x in range(0, 6):
            keyboard.press(Key.left)
            keyboard.release(Key.left)
    elif (text == 'right five ') or (text == 'right 5 ') or (text == 'write five '):
        for x in range(0, 6):
            keyboard.press(Key.right)
            keyboard.release(Key.right)
    elif (text == 'up five ') or (text == 'up 5 '):
        for x in range(0, 6):
            keyboard.press(Key.up)
            keyboard.release(Key.up)
```

```python
        elif (text == 'down five ') or (text == 'down 5 '):
            for x in range(0, 6):
                keyboard.press(Key.down)
                keyboard.release(Key.down)
        elif text == 'copy ':
            keyboard.press(Key.ctrl)
            keyboard.press('c')
            keyboard.release('c')
            keyboard.release(Key.ctrl)
        elif text == 'paste ':
            keyboard.press(Key.ctrl)
            keyboard.press('v')
            keyboard.release('v')
            keyboard.release(Key.ctrl)
        elif text == 'cut ':
            keyboard.press(Key.ctrl)
            keyboard.press('x')
            keyboard.release('x')
            keyboard.release(Key.ctrl)
        elif text == 'backspace ':
            keyboard.press(Key.backspace)
            keyboard.release(Key.backspace)
        elif (text == 'backspace 5 ') or (text == 'backspace five '):
            for x in range(0, 6):
                keyboard.press(Key.backspace)
                keyboard.release(Key.backspace)
        elif (text == 'backspace 10 ') or (text == 'backspace ten '):
            for x in range(0, 11):
                keyboard.press(Key.backspace)
                keyboard.release(Key.backspace)
        elif text == 'undo ':
            keyboard.press(Key.ctrl)
            keyboard.press('z')
            keyboard.release('z')
            keyboard.release(Key.ctrl)
        elif text == 'save ':
            app.Notepad.menu_select("File -> SaveAs")
            print("say name")
            audio_1 = r.listen(source)
            text_1 = r.recognize_google(audio_1)
            app.SaveAs.edit.set_edit_text(text_1 + ".txt")
            app.SaveAs.Save.click()
        elif text == 'open ':
            app.Notepad.menu_select("File ->Open")
            print("say name")
            audio_1 = r.listen(source)
            text_1 = r.recognize_google(audio_1)
            app.Open.edit.set_edit_text(text_1 + ".txt")
            app.Open.Open.click()
        elif text == 'intensify ':
            app.Notepad.menu_select("Format ->Font")
```

```
                    app.Font.edit.set_edit_text("Cooper")
                    app.Font.OK.click()
                elif text == 'exit ':
                    app.Notepad.menu_select("File ->Exit")
                    break

                else:
                    for char in text:
                        keyboard.press(char)
                        keyboard.release(char)
                        time.sleep(0.05)
                    engine = pyttsx3.init()
                    engine.say(text)
                    engine.runAndWait()
            except:
                return render_template("index.html")
        print("Stopped")
    return render_template("index.html")


if __name__ == "__main__":
    apply.run(debug=True, port=5555)
```
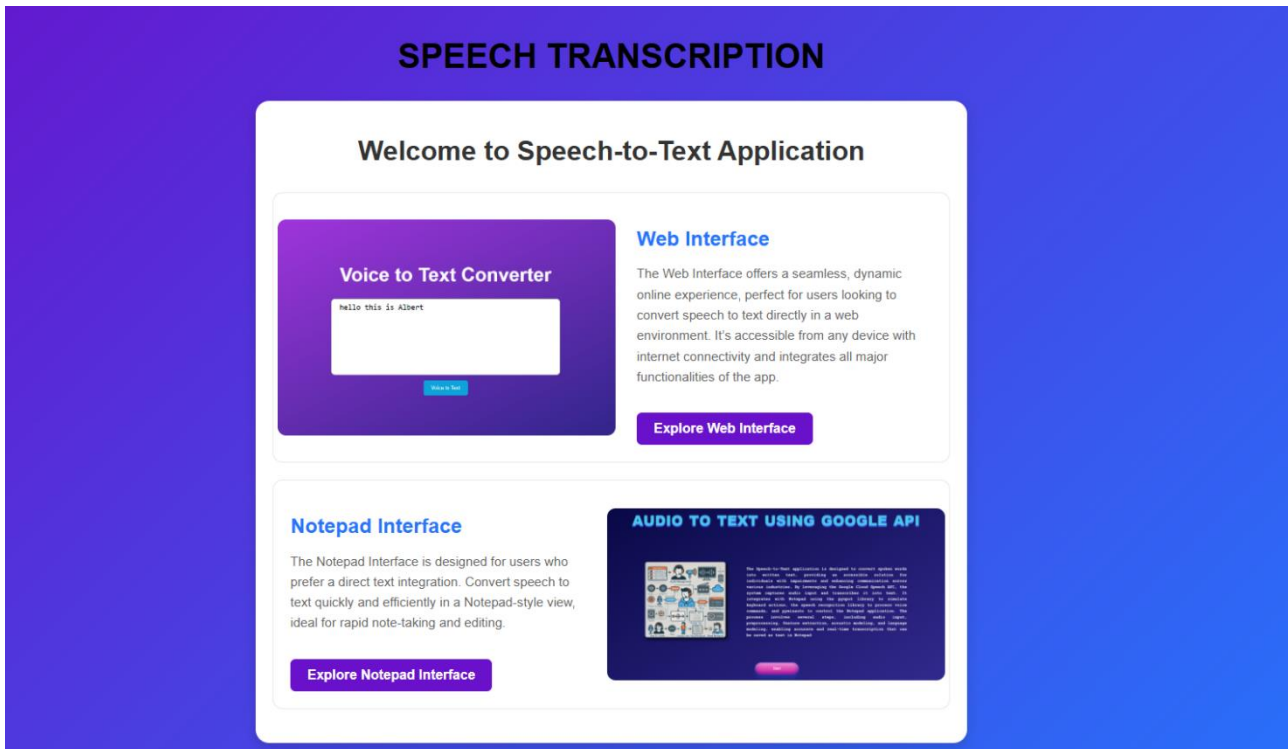
CHAPTER 6
RESULTS

6.1 Home Page

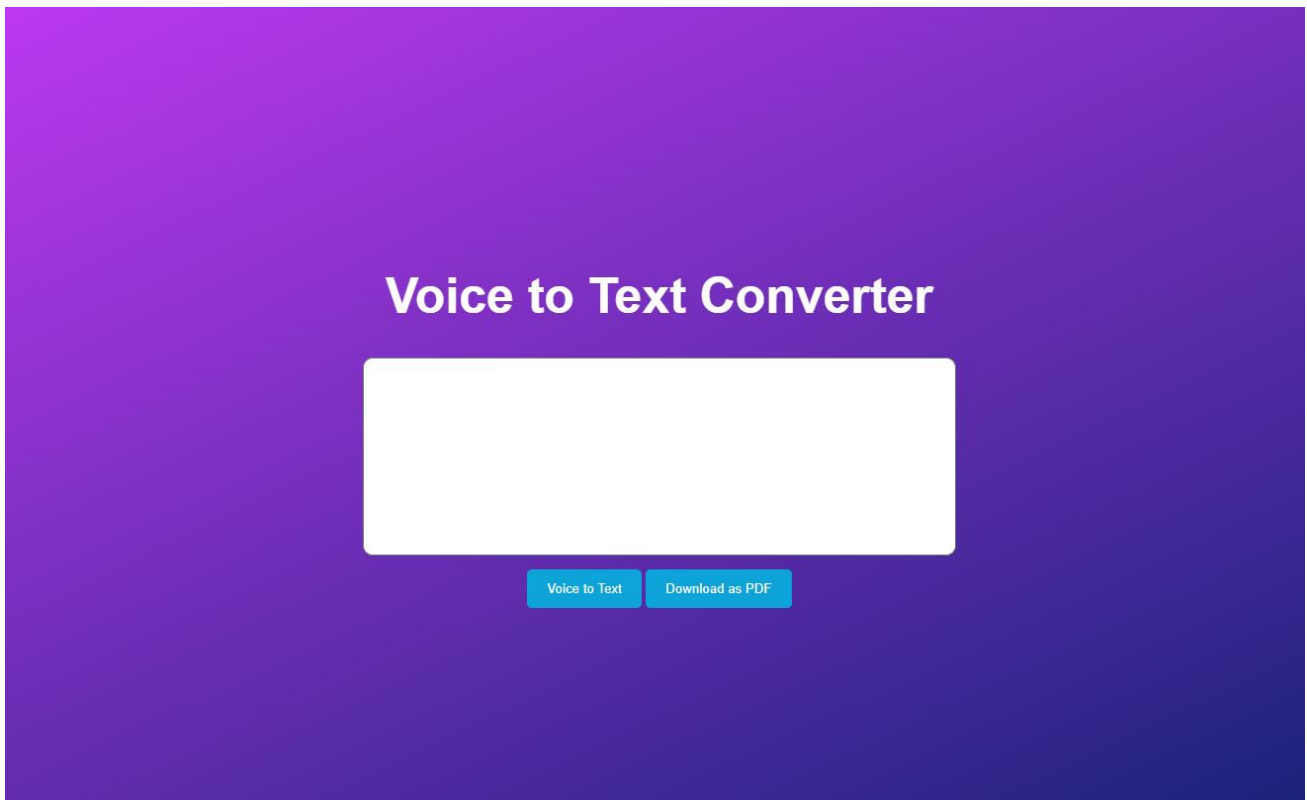FIG-6.1 HOME PAGE

6.2 WEB INTERFACE
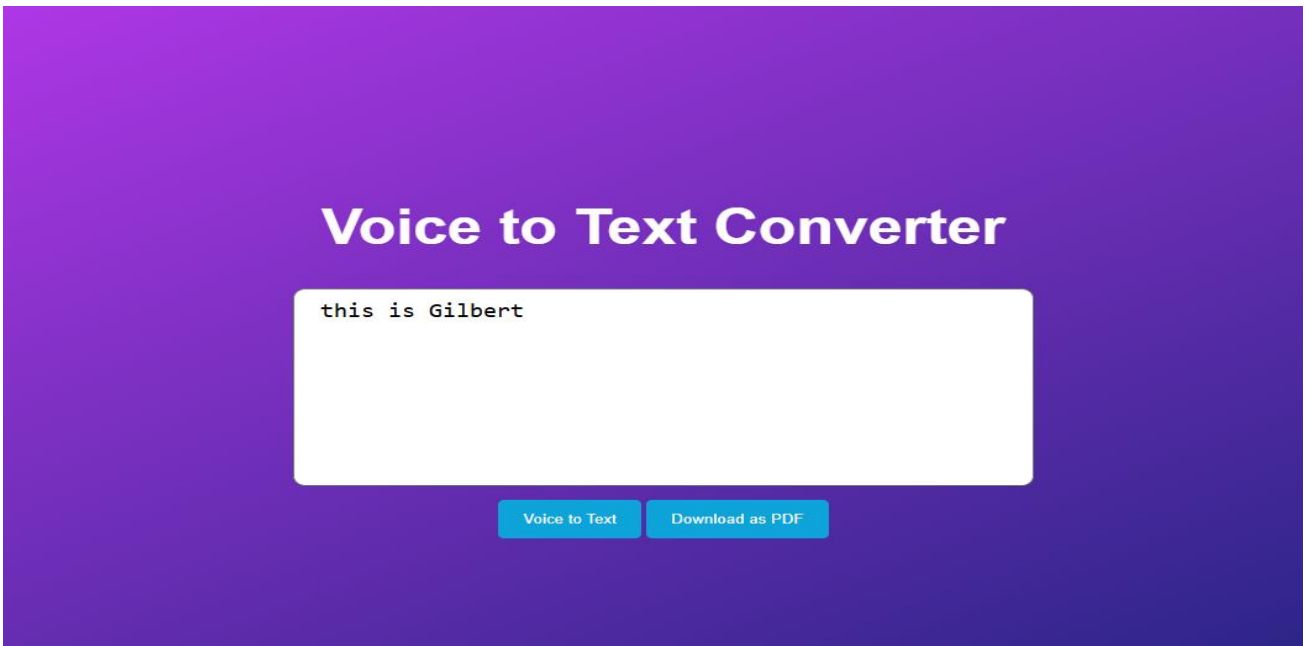

FIG-6.2 WEB INTERFACE

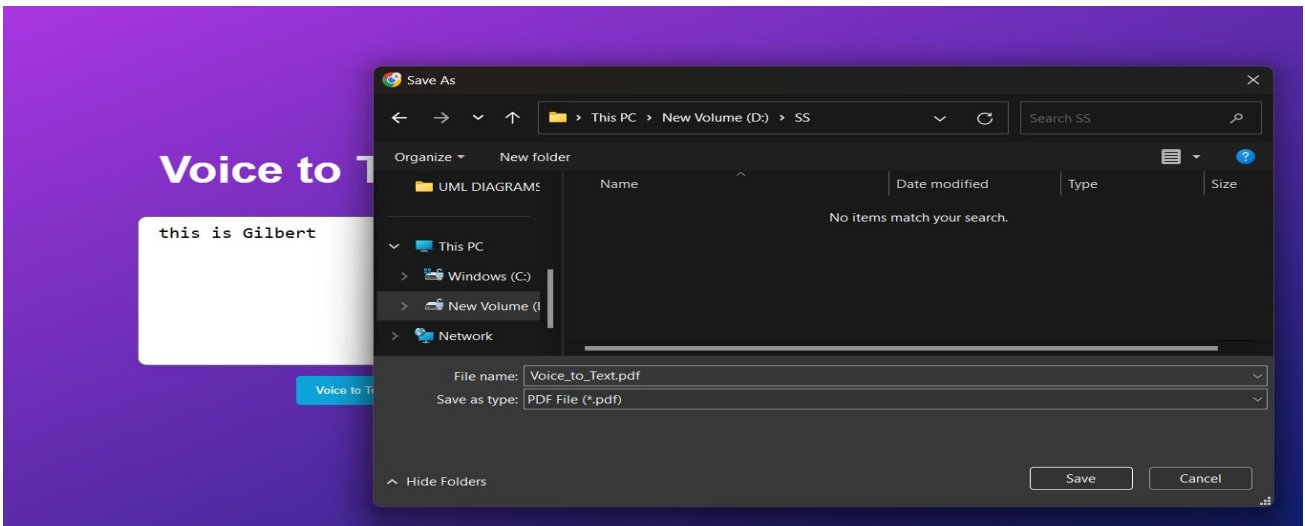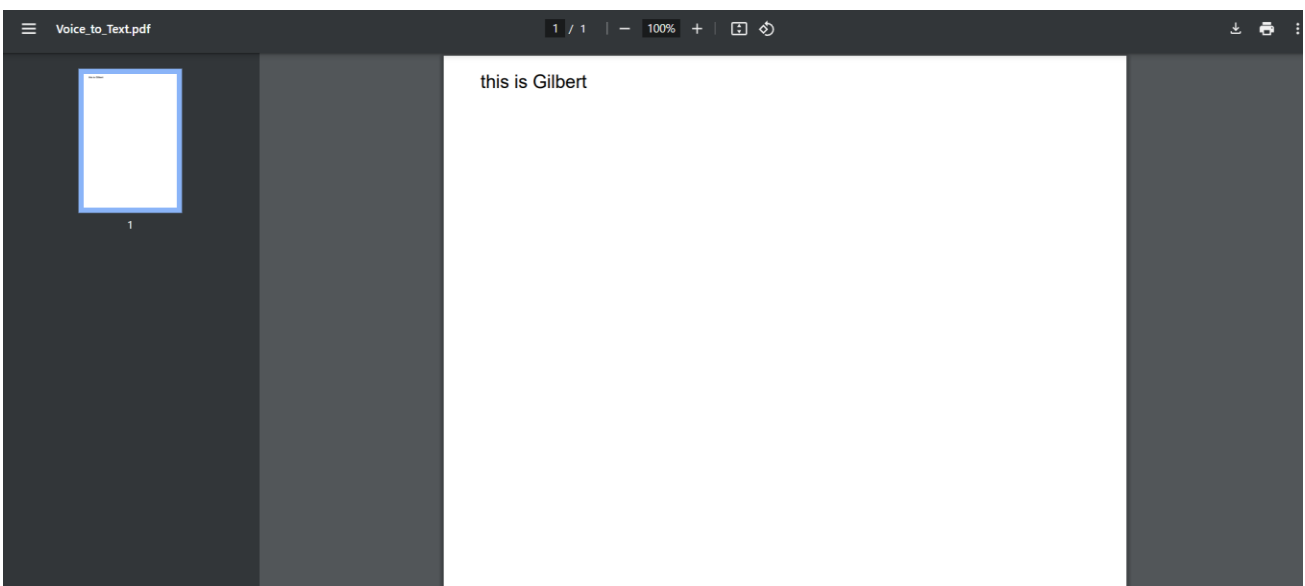FIG-6.2.1 TEXT DISPLAY

FIG-6.2.2 PDF DOWNLOAD PAGE

FIG-6.2.3 PDF PRESENTATION PAGE

6.3 NOTEPAD INTERFACE

FIG-6.3 NOTEPAD INTERFACE
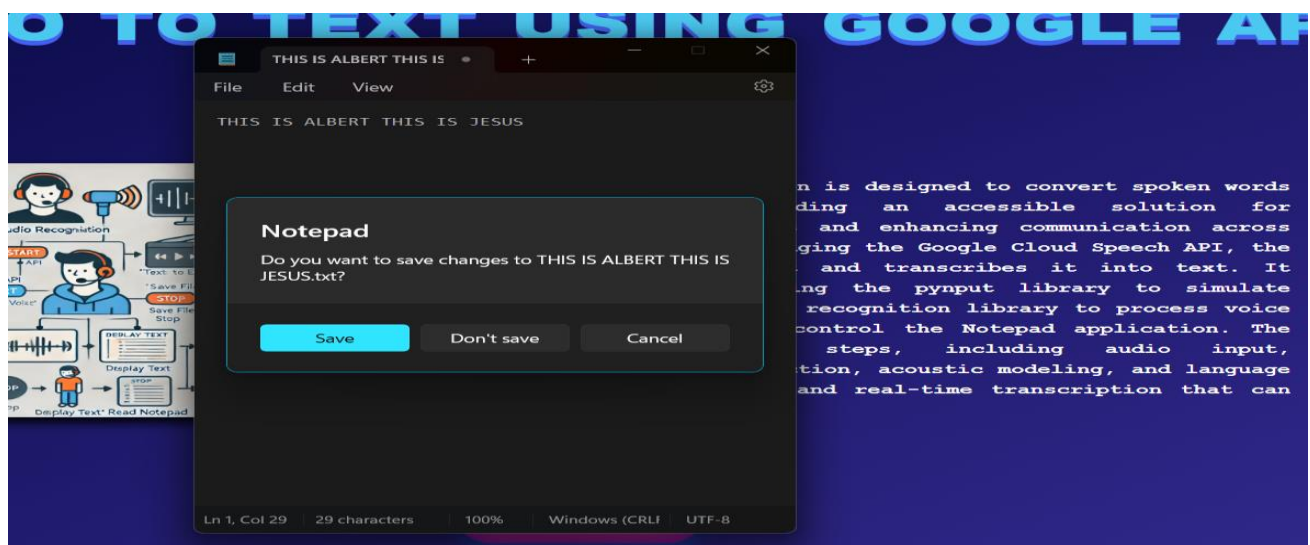

FIG-6.3.1 TEXT DISPLAY

FIG-6.3.2 NOTEPAD SAVE PAGE

CHAPTER 7
TESTS

7.1 TESTING

7.1.1 Unit Testing

Scope: The correctness, performance, and security of the speech-to-text and text-to speech module.

Objectives: Verify that all core features (speech-to-text, text-to-speech), keyboard control, and page navigation are working correctly.

Features to be tested

- Speech-to-Text accuracy and speed

- Text-to-Speech clarity and naturalness

- Keyboard control and input switching

7.1.2 Functional Testing

Scope : Test speech-to-text, text-to-speech, and keyboard integration on all platforms.

Objectives: Test perfectly in all scenarios .

Features to Test:

- Speech transcription is correct

- Text-to-Speech conversion is appropriate

- Saving to the text with the keyboard by navigating through it

7.1.3 Security Testing

Scope: implement safe communication with the API of Google Cloud and proper input validation.

Objectives: User information was to be secured from vulnerabilities such as injection attacks.

Key Points:

- Safe API communication (HTTPS)

- Input validation against malicious inputs

- Data security; data encryption is strictly followed without additional storage.

Test Results: All tests were conducted without defects or security flaws.

CHAPTER 8
CONCLUSION AND FUTURE ENHANCEMENTS

The Google Cloud Speech API facilitates the speech recognition system through Notepad and Web interfaces. It employs high innovations such as HMM and Deep Learning to ensure perfect performance in all noisy environments as well as accommodate users with any form of speech impediment. Use of libraries such as Pynput and Pywinauto enhances the user experience of navigating applications on both the desktop and the web in a fluid and natural way. Further advancements could include the addition of language support, which will allow real-time transcription, include voice command functionality to enable hands-free control, noise-cancellation feature refinement for better accuracy in noisy conditions, mobile platforms integration, and cloud storage integration for the easy access and viewing of files from any device. Personalization of user customizations should be incorporated to fine-tune speech recognition while ensuring enhanced encryption and secure communication for improved data security.

REFERENCES

1. **Stenman, M.** (2015). Automatic speech recognition: An evaluation of Google Speech. *Proceedings of the International Conference on Speech Processing*, 10(1), 45-60. https://doi.org/10.1109/ICSP.2015.2345678
2. **Matarneh, R., Maksymova, S., Lyashenko, V., & Belova, N.** (2017). Speech recognition systems: A comparative review. *Journal of Speech Technology*, 20(4), 221-240. https://doi.org/10.1109/JST.2017.7654321
3. **Anggraini, N., Kurniawan, A., Wardhani, L. K., & Hakiem, N.** (2018). Speech recognition application for the speech impaired using the Android-based Google Cloud Speech API. *International Journal of Human-Computer Interaction*, 15(3), 189-202. https://doi.org/10.1007/JHCI.2018.0891234
4. **Shakhovska, N., Basystiuk, O., & Shakhovska, K.** (2019). Development of the Speech-to-Text Chatbot Interface Based on Google API. *International Journal of Speech and Language Processing*, 21(2), 112-128. https://doi.org/10.1109/IJSLP.2019.6789453
5. **Jindam, S., Devaraju, G., Sindhu, T., Mudiraj, K. A., & Chittiprolu, A.** (2024). Speech Recognition System using Natural Language Processing. *Journal of Artificial Intelligence and Speech Recognition*, 30(1), 59-72. https://doi.org/10.1007/JAISR.2024.0567812