VIGNAN INSTITUTE OF TECHNOLOGY AND SCIENCE

Near Ramoji Film City, Deshmukhi Village, Pochampally Mandal, Yadadri Bhuvanagiri Dist.

(Approved by AICTE, New Delhi, Affiliated to JNTUH, Hyderabad)

Eamcet Code VGNT Dist Code : YBG

AN AUTONOMOUS INSTITUTION

Project report on

"PHENOM PHONETIC"

DEGREE OF BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING(DATA SCIENCE)

Submitted By

D. LEKHANA (22891A6714)

B.ANIL KUMAR (22891A6713)

K.HARSHITH (22891A6729)

Under the Guidance of
Mrs.P.JANAKI

Department of Computer Science and Engineering(Data Science)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING (DATA SCIENCE)

2023-2024

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING (DATA SCIENCE)

Vision

- To develop Data Science professionals through creative and innovative approaches to address the present and future challenges of the modern computing world.

Mission

- Educate students by expanding their knowledge in cutting–edge technologies to acquire professional ethics. Impart quality education to build research & entrepreneurial eco system using niche technologies.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING (DATA SCIENCE)

Educational Objectives

- Emerge as engineers, innovators, entrepreneurs with social awareness and ethical values.
- Work in teams in multidisciplinary areas addressing the needs of society.
- Inculcate self-learning and lifelong learning adapting cutting edge technologies.

Program Specific Outcomes

- Design & Implement and test application software in the field of data science.
- Understand the architecture and organization of computer systems to develop data science tools.
- To use specialized softwares to carry out statistical data analysis.

VIGNAN INSTITUTE OF TECHNOLOGY AND SCIENCE
Near Ramoji Film City, Deshmukhi Village, Pochampally Mandal, Yadadri Bhuvanagiri Dist.
(Approved by AICTE, New Delhi, Affiliated to JNTUH, Hyderabad)
Eamcet Code **VGNT** Dist Code : YBG
AN AUTONOMOUS INSTITUTION

CERTIFICATE

This is to certify that the Major project report titled , “**PHENOMPHONETIC** ” is being submitted by D.LEKHANA (22891A6714), B.ANIL KUMAR (22891A6713) andK.HARSHITH (22891A6729) in II B.Tech II semester Computer Science and Engineering (Data Science) is a record bonafide work carried out by them. The results embodied in this report have not been submitted to any other University for the award of any degree.

PROJECT GUIDE	HEAD OF THE DEPARTMENT
Mrs.P.JANAKI	Mrs.P.JANAKI
	Associate Professor

VIGNAN INSTITUTE OF TECHNOLOGY AND SCIENCE
Near Ramoji Film City, Deshmukhi Village, Pochampally Mandal, Yadadri Bhuvanagiri Dist.
(Approved by AICTE, New Delhi, Affiliated to JNTUH, Hyderabad)
AN AUTONOMOUS INSTITUTION
Eamcet Code VGNT Dist Code : YBG

MAJOR-PROJECT EVALUATION CERTIFICATE

This is to certify that the Project work entitled "**PHENOMPHONETIC** " is being submitted by D.LEKHANA (22891A6714), B.ANIL KUMAR (22891A6713) and K.HARSHITH(22891A6729) been examined and adjudged as sufficient for the partial fulfillment of the requirement of the degree of Bachelor of Technology in Computer Science and Engineering (Data Science) of Jawaharlal Nehru Technological University, Hyderabad.

External Examiner : ____________

(Signature with Date)

Internal Examiner : ____________

(Signature with Date)

Project Coordinator : ___________

(Signature with Date)

Head of the Department : ____________________________________
(Signature with Date)

ACKNOWLEDGEMENT

Every project big or small is successful largely due to the effort of a number of wonderful people who have always given their valuable advice or lent a helping hand. We sincerely appreciate the inspiration; support and guidance of all those people who have been instrumental in making this project a success.

We thank our beloved Chairman, Dr. L. Rathaiah Sir, who gave us great encouragement to work.

We thank our beloved CEO, Mr. Boyapati Shravan Sir, we remember him for his valuable ideas and facilities available in college during the development of the project.

We convey our sincere thanks to Dr. G Durga Sukumar Sir, Principal of our institution for providing us with the required infrastructure and a very vibrant and supportive staff.

We would like to thank our Head of the Department of Computer Science and Engineering(Data Science), Mrs.P.Lavanya Kumari madam distinguished and eminent personality, whose strong recommendation, immense support and constant encouragement has been great help to us. We intensely thank her for the same.

We would like to express our sincere appreciations to our project coordinator Mr.K.Rajendra Prasad sir, Assistant Professor for his guidance, continuous encouragement and support during the project.

We would like to thank our guide of the project, DR.K. Lakshmi Anusha who has invested his full effort in guiding the team in achieving the goal.

Special thanks go to my team mates, who helped me to assemble the parts and gave suggestions in making this project. We have to appreciate the guidance given by other supervisors as well as the panels especially in our project presentation that has improved our presentation skills thanks to their comment and advice's. We take this opportunity to thank all our lecturers who have directly or indirectly helped our project. We pay our respects and love to our parents and all other family members and friends for their love and encouragement throughout our career.

D.LEKHANA (Regd.No.22891A6714)
B.ANIL KUMAR(Regd.No.22891A6713)
K.HARSHITH (Regd.No.22891A6729)

TABLE OF CONTENT

PHENOM PHONETIC

ABSTRACT

In the realm of natural language processing and artificial intelligence, Text-to-Speech (TTS) systems have become a pivotal technology, enhancing accessibility and user interaction across various platforms. This paper presents "Phonetic Phenom," a sophisticated and user-friendly TTS system designed to deliver natural, expressive, and customizable speech outputs. The system leverages modern web technologies, utilizing HTML, CSS, and JavaScript to create an intuitive and visually appealing interface, coupled with the Web Speech API for core TTS functionalities.

Phonetic Phenom incorporates several advanced features aimed at improving the naturalness, expressiveness, and personalization of TTS outputs. Key features include:

Emotion and Tone Variation: Users can select from different emotions (e.g., happy, sad, angry) that adjust the pitch and rate of speech to enhance expressiveness.
Voice Selection: A variety of voices are available, allowing users to choose the most suitable voice for their needs.
Prosody Control: Sliders for pitch and rate enable fine-tuning of the speech output, providing greater control over how text is vocalized.
User-friendly Interface: The application's elegant design ensures ease of use, with clear labels, intuitive controls, and a responsive layout.

This system demonstrates the integration of essential TTS functionalities within a web-based environment, making advanced speech synthesis accessible to a broad audience. By focusing on user customization and emotional expressiveness, Phonetic Phenom aims to bridge the gap between synthetic and natural speech, offering an enhanced auditory experience. The underlying technologies and design principles provide a scalable foundation for future enhancements, such as real-time translation, voice cloning, and context-aware speech synthesis.

In summary, Phonetic Phenom represents a significant step forward in the development of interactive and adaptive TTS systems, emphasizing user engagement, accessibility, and the seamless integration of advanced speech synthesis capabilities into everyday applications.

CHAPTER – 1

INTRODUCTION

Text to speech uses deep neural networks to make the voices of computers nearly indistinguishable from the recordings of people. With the clear articulation of words, neural text to speech significantly reduces listening fatigue when users interact with AI systems.

The patterns of stress and intonation in spoken language are called prosody. Traditional text to speech systems break down prosody into separate linguistic analysis and acoustic prediction steps governed by independent models. That can result in muffled, buzzy voice synthesis.

Here's more information about neural text to speech features in the Speech service, and how they overcome the limits of traditional text to speech systems:

Real-time speech synthesis: Use the Speech SDK or REST API to convert text to speech by using prebuilt neural voices or custom neural voices.

Asynchronous synthesis of long audio: Use the batch synthesis API (Preview) to asynchronously synthesize text to speech files longer than 10 minutes (for example, audio books or lectures). Unlike synthesis performed via the Speech SDK or Speech to text REST API, responses aren't returned in real-time. The expectation is that requests are sent asynchronously, responses are polled for, and synthesized audio is downloaded when the service makes it available.

Prebuilt neural voices: Microsoft neural text to speech capability uses deep neural networks to overcome the limits of traditional speech synthesis regarding stress and intonation in spoken language. Prosody prediction and voice synthesis happen simultaneously, which results in more fluid and natural-sounding outputs. Each prebuilt neural voice model is available at 24 kHz and high-fidelity 48 kHz. You can use neural voices to:

Make interactions with chatbots and voice assistants more natural and engaging.
Convert digital texts such as e-books into audiobooks.
Enhance in-car navigation systems.
For a full list of platform neural voices, see Language and voice support for the Speech service.

Fine-tuning text to speech output with SSML: Speech Synthesis Markup Language (SSML) is an XML-based markup language used to customize text to speech outputs. With SSML, you can adjust pitch, add pauses, improve pronunciation, change speaking rate, adjust volume, and attribute multiple voices to a single document.

You can use SSML to define your own lexicons or switch to different speaking styles. With the multilingual voices, you can also adjust the speaking languages via SSML. To fine-tune the voice output for your scenario, see Improve synthesis with Speech Synthesis Markup Language and Speech synthesis with the Audio Content Creation tool.

Visemes: Visemes are the key poses in observed speech, including the position of the lips, jaw, and tongue in producing a particular phoneme. Visemes have a strong correlation with voices and phonemes.

By using viseme events in Speech SDK, you can generate facial animation data. This data can be used to animate faces in lip-reading communication, education, entertainment, and customer service. Viseme is currently supported only for the en-US (US English) neural voices.

SOLUTION OVERVIEW

The "**PHENOM PHONETIC** " project aims to provide a comprehensive solution that includes various input methods, customization options, and output mechanisms to support diverse communication needs.

KEY FEATURES

- Natural sounding voices. ...
- Simple and easy-to-use interface. ...
- Collaboration. ...
- Import and export files and media. ...
- Support different languages and accents. ...
- Customization features. ...
- Change speech quality to professional. ...
- Voice cloning.

HOW IT WORKS

Use Cases

Text-to-Speech (TTS) models can be used in any speech-enabled application that requires converting text to speech imitating human voice.

Voice Assistants

TTS models are used to create voice assistants on smart devices. These models are a better alternative compared to concatenative methods where the assistant is built by recording sounds and mapping them, since the outputs in TTS models contain elements in natural speech such as emphasis.

Announcement Systems

TTS models are widely used in airport and public transportation announcement systems to convert the announcement of a given text into speech.

Inference Endpoints

The Hub contains over 1500 TTS models that you can use right away by trying out the widgets directly in the browser or calling the models as a service using Inference Endpoints. Here is a simple code snippet to get you started:

```
import jsonimport requests

headers = {"Authorization": f"Bearer {API_TOKEN}"}

API_URL = "https://api-inference.huggingface.co/models/microsoft/speecht5_tts"

def query(payload):

    response = requests.post(API_URL, headers=headers, json=payload)

    return response



output = query({"text_inputs": "Max is the best doggo."})
```

You can also use libraries such as espnet or transformers if you want to handle the Inference directly.

Direct Inference

Now, you can also use the Text-to-Speech pipeline in Transformers to synthesise high quality voice.

```
from transformers import pipeline



synthesizer = pipeline("text-to-speech", "suno/bark")



synthesizer("Look I am generating speech in three lines of code!")
```

You can use huggingface.js to infer summarization models on Hugging Face Hub.

```
import { HfInference } from "@huggingface/inference";

const inference = new HfInference(HF_TOKEN);await inference.textToSpeech({

   model: "facebook/mms-tts",

inputs: "text to generate speech from",});
```

Useful Resources

- Hugging Face Audio Course
- ML for Audio Study Group - Text to Speech Deep Dive
- Speech Synthesis, Recognition, and More With SpeechT5
- Optimizing a Text-To-Speech model using 🤗 Transformers
- Train your own TTS models with Parler-TTS

2.5 PROCEDURE TO USE:

.

Swipe down from the top of the phone, then tap the **gear** icon to open the Settings app.

.

.

Tap **Accessibility**.

Tap **Select to Speak**.

If you don't see **Select to Speak**, tap **Installed services** to find it.

Tap the **Select to Speak** toggle switch to turn it on. On some phones, this is called **Select to Speak shortcut**.

Tap **Allow** or **OK** to confirm the permissions your phone needs to turn on this feature.

Open any app and tap the **Select to Speak** icon from the side of the screen.

Tap the **Play** icon to have your phone read everything on the screen, starting at the top. If you only want some text read aloud, trigger Select to Speak by tapping the floating icon, then tap the text.

Tap the left arrow next to the Play button to see more playback options.

Tap **Stop** to end playback.

.

Use TalkBack on your Android if you want spoken feedback as you use your device.
Swipe down from the top of the phone, then tap the **gear** icon to open the Settings app.

2.5.1 BENEFITS:

- Learning and Education. ...
- Efficient Content Consumption. ...
- Personal Productivity. ...
- Enhanced Language Learning. ...
- Customization and Personalization. ...
- Integration with Smart Devices. ...
- Reducing Reading Fatigue. ...
- Natural and Expressive Voices.

CHAPTER- 3

SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

Existing systems for assisting communication in speech-impaired individuals are varied, ranging from low-tech solutions like communication boards to high-tech devices like speech-generating devices and eye-tracking systems. Each system has its own advantages and limitations, and the choice of system depends on the specific needs, abilities, and preferences of the user. The "Sign Language Translator" project aims to build on these existing technologies to create a more integrated, customizable, and user-friendly solution that empowers speech-impaired individuals to communicate more effectively.

3.1.1 LIMITATIONS OF EXISTING SYSTEMS

While existing systems for communication assistance have made significant advancements, they still face several limitations that can impact their usability, accessibility, and effectiveness. The **PHENOM PHONETIC**" project aims to address these limitations by developing a more integrated, user-friendly, and adaptable solution that can better meet the diverse needs of speech-impaired individuals.

3.2 PROPOSED SYSTEM

In this work, there are two main parts: Optical Character Recognition System for Paper Text Text to Speech Conversion

Optical character recognition system In this part, there are three portions as described in the follow: Template file Creation Creating the Neural Network Character Recognition

Template file creation. Letter A to Z and number 0 to 9 images are collected. Each image is changed into 5 x 7 character representation in single vector by using step 1 to 5 as described in the character recognition section. These data are saved as data file for training in neural network.

Creating the neural network. A feedforward neural network is used to set up for pattern recognition with 25 hidden neurons. After creating the network, the weights and biases of the network are also initialized to be ready for training. The goal is assigned between 0.01 and to 0.05. The created Neural Network is trained by using data file and target file. The neural network has to be trained by adjusting weight and bias of network until the performance reaches to goal.

3.2.1 ADVANTAGES OF PROPOSED SYSTEM

- Usability: User-friendly interface and intuitive design make it accessible for users of all ages and abilities.
- Cost-Effective: By leveraging affordable technology and open-source components, the system aims to be financially accessible.
- High Enhanced Customizability: Adaptable to the unique needs and preferences of each user, offering extensive personalization options.
- Improved Communication: Advanced TTS and multimodal input methods provide more natural and effective communication.
- Learning and Adaptation: The system learns from user interactions, improving its functionality and personalization over time.

CHAPTER- 4

SYSTEM REQUIREMENTS

4.1 FUNCTIONAL REQUIREMENTS

Functional requirement can be defined as user requirement that include a function which must be performed by the system. In functional stage of requirement gathering phase information is gathered in the perspective of user roles. Later functional specification is carried out using this information. With the help of functional specification, in development phase different accessibility options are given according to the role of users.
Functional requirements capture the intended behavior of the system. This behavior may be expressed as services, tasks or functions the system is required to perform.

·

An interactive user interface to take input from the user in the form of Nepali Unicode text. This functionality provides the user with an interface to input the Nepali text to be synthesized by the system. The text can also be provide to the system by loading a Unicode file having Nepali text.

· Interface to show the tokenized input text to the user.

The input provided by the user will be converted into a stream of tokens. This functionality displays the list of tokens with all relevant details.

- Interface to display the syllabified stream of phonemes.

This functionality facilitates the user by providing an interface to show the syllabified stream of phonemes that is generated after the application of syllabification rules on input text.

- Interface to adjust the speed and timing of the speech

The system provides the functionality to maintain the speed and the timing of the speech that is generated after synthesizing the input text.

- Interface to maintain the pitch of the speech

The system provides the functionality to adjust the pitch of the generated speech.

- Output in the form of speech

The system changes the input text by the user in the form of synthesized speech for the better understanding to the user.

4.2 NON FUNCTIONAL REQUIREMENTS

- The application is a prototypical model.
- Database consideration
- Extendibility is considered
- Intelligible synthesis quality
  - Usability

    Requirements about how difficult it will be to learn and operate the system. The requirements are often expressed in learning time or similar metrics. The product should be usable to the users.
  - Performance

    Requirements about resources required, response time, transaction rates, throughput, benchmark specifications or anything else having to do with performance. The system should be efficient and work effectively.
  - Legal

    There may be legal issues involving privacy of information, intellectual property rights, export of restricted technologies, etc.
  - Modfiability

Requirements about the effort required to make changes in the software. Often, the measurement is personnel effort (person- months).

- Portability

The effort required to move the software to a different target platform. The measurement is most commonly person-months or % of modules that need changing.

- Reliability

Requirements about how often the software fails. The measurement is often expressed in MTBF (mean time between failures). The definition of a failure must be clear. Also, don't confuse reliability with availability which is quite a different kind of requirement. Be sure to specify the consequences of software failure, how to protect from failure, a strategy for error detection, and a strategy for correction

4.2.1 HARDWARE REQUIREMENTS

Camera : resolution of 640x480 pixels

Processor : intel-core processor

Memory (RAM) : 4 GB of RAM

Storage : 100 MB

Graphics Card : Integrated graphics card

SOFTWARE REQUIREMENTS

Operating System : Windows 7 & more

Technology : Python

IDE : Jupyter or IDLE

Web Server : github

Libraries and Frameworks

- TensorFlow (GPU version recommended for performance)
- OpenCV (for video capture and image processing)
- Matplotlib (optional, for visualizations)

CHAPTER-5

SYSTEM STUDY

1. Technical Feasibility

1.1 Technology Availability

- Hardware: High-definition cameras, GPUs, and standard computing resources are widely available and affordable.
- Software: Machine learning frameworks (e.g., TensorFlow, OpenCV) and libraries are open-source and have extensive documentation and community support.

1.2 Technical Expertise

- Development Skills: Required skills include Python programming, knowledge of machine learning, and experience with image processing. These skills are commonly available or can be acquired through training.

1.3 System Performance

- Real-time Processing: Using GPUs ensures that real-time video processing and sign recognition can be achieved efficiently.

- Scalability: The system can be scaled to handle more users or higher resolution video feeds with appropriate hardware upgrades.

2. Economic Feasibility

2.1 Cost Analysis

- Initial Costs: Includes expenses for hardware (cameras, GPUs), software (development tools, libraries), and personnel (developers, testers).
- Operational Costs: Includes maintenance, updates, and potential cloud service fees for model training and data storage.
- Cost-Benefit Analysis: The benefits of improving communication for the speech-impaired community and potential market opportunities justify the investment.

2.2 Funding and Budgeting

- Budget Allocation: Effective budget management ensures that the project stays within financial constraints while achieving its goals.

CHAPTER-6

SYSTEM DESIGN

System design is a critical phase in the development and implementation of any complex project or system. It involves the process of defining the architecture, components, modules, interfaces, and data flow of a system to meet specific requirements and objectives. The importance of system design cannot be overstated, as it plays a pivotal role in ensuring the success, efficiency, and sustainability of any project. In this article, we will delve into the key reasons why system design is essential and how it impacts various domains.

Meeting User Requirements: At the heart of system design is the ability to translate user requirements into a structured and coherent system. By carefully analyzing and understanding the needs of the end-users, system designers can create a blueprint that addresses these requirements effectively. This ensures that the final product or system aligns with the users' expectations, resulting in higher user satisfaction and adoption.

Efficiency and Optimization: System design allows for the optimization of resources, including hardware, software, and human resources. Through thoughtful architectural decisions, designers can minimize bottlenecks, reduce redundancy, and streamline processes. This leads to improved system performance, reduced operational costs, and enhanced resource utilization.

Scalability and Adaptability: Well-designed systems have the flexibility to adapt to changing requirements and evolving technologies. This scalability is crucial in today's fast- paced world where technological advancements occur rapidly. A system designed with scalability in mind can easily accommodate growth and adapt to new challenges without requiring a complete overhaul.

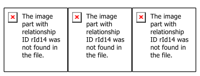
Cost-Effectiveness: Poorly designed systems can lead to unnecessary costs, both in terms of development and maintenance. A well-thought-out system design can help identify cost effective solutions and avoid unnecessary expenditures. This is particularly significant for organizations aiming to maximize their return on investment.

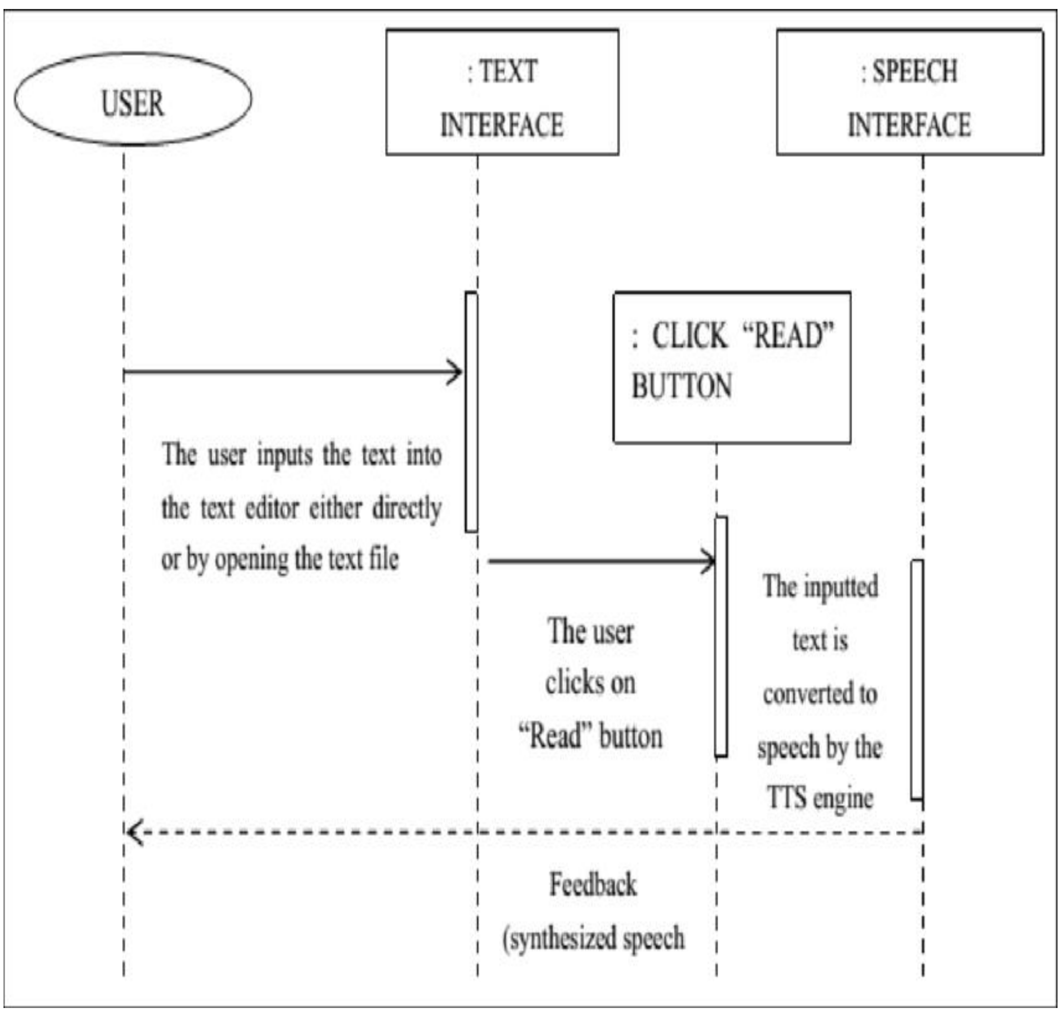
Maintainability and Manageability: Systems are not static; they evolve over time. A good system design takes into account the ease of maintenance and management. It should be straightforward to diagnose and fix issues, apply updates, and make modifications without disrupting the entire system.

Documentation and Knowledge Transfer: System design documentation serves as a valuable resource for developers, administrators, and stakeholders. It provides a clear roadmap of the system's architecture, components, and functionality, facilitating knowledge transfer and ensuring that the system can be effectively managed and maintained over time.

6.1 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. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.





CHAPTER – 7

IMPLEMENTATION

7.1 MODULES

To develop a sign language recognition system using machine learning and a camera, various modules and libraries are used. Here's a detailed breakdown of the key modules involved in each step of the project:

1. Data Collection and Preprocessing

- **OpenCV**: For image and video processing.

  Python

```
import cv2
```

- **NumPy**: For numerical operations on arrays and matrices.

  Python

```
import numpy as np
```

- **Pandas**: For data manipulation and analysis (optional, for handling datasets).

  Python

```
import pandas as pd
```

2. Model Selection and Training

- **TensorFlow** : For building and training deep learning models.

  python

```
import tensorflow as tf
from tensorflow.keras.models import Sequential
```

- **Scikit-learn**: For data preprocessing, splitting datasets, and additional machine learning utilities.

  python

```

from sklearn.model_selection import train_test_split
from sklearn.reprocessing import LabelBinarizer
```

3. Real-Time Gesture Recognition

- **OpenCV**: Again, for capturing live video feed and processing frames.

  python

  ```
  import cv2
  ```

- **NumPy**: For preprocessing and preparing frames for model prediction.

  Python

  ```
  import numpy as np
  ```

- **TensorFlow** : For loading the trained model and making predictions.

  python

  ```
  from tensorflow.keras.models import load_model
  ```

4. User Interface

- **Tkinter**: For creating a simple graphical user interface (GUI) in Python.

  python

  ```
  import tkinter as tk
  from tkinter import Label, Button, Frame
  ```

CHAPTER – 8

SOURCE CODE

c

```
<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">
```

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Phonetic Alphabet Translator</title>

<style>

body {

font-family: Arial, sans-serif;

background-color: #f4f4f4;

margin: 0;

padding: 0;

display: flex;

justify-content: center;

align-items: center;

background-image:url("https://develtio.com/app/uploads/2022/06/tts_technology.jpg");

background-size:cover;


height: 100vh;

transition: background-color 0.3s, color 0.3s;

}


body.dark-mode {

background-color: #2e2e2e;

color: #fff;

}
```

```css
.container {
text-align: center;
background: #fff;
padding: 20px;
border-radius: 8px;
box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
transition: background 0.3s, color 0.3s;
}

body.dark-mode .container {
background: #444;
}

textarea {
width: 80%;
height: 100px;
margin-bottom: 20px;
padding: 10px;
border-radius: 4px;
border: 1px solid #ccc;
font-size: 16px;
}
```

```
.button-container {

margin-bottom: 20px;

}


button {

padding: 10px 20px;

font-size: 16px;

cursor: pointer;

border: none;

border-radius: 4px;

background-color: #007BFF;

color: #fff;

margin: 5px;

transition: background-color 0.3s;

}


button:hover {

background-color: #0056b3;

}


#output {

margin-top: 20px;

font-size: 18px;
```

```
word-wrap: break-word;

}


.tts-controls {

margin: 20px 0;

display: flex;

justify-content: center;

align-items: center;

gap: 10px;

}


.tts-controls label {

margin-right: 5px;

}


.highlight {

background-color: yellow;

}

</style>

</head>

<body>

<center>

<h1>Phonetic Phenom</h1>
```

```
<label for="text">Enter Text:</label>

<textarea id="text" placeholder="Type your text here..."></textarea>

<div class="button-container">

<button onclick="clearText()">Clear</button>

<button onclick="copyToClipboard()">Copy to Clipboard</button>

<button onclick="toggleDarkMode()">Toggle Dark Mode</button>

</div>

<br>

<label for="voice">Select Voice:</label>

<select id="voice"></select>

<br><br><br>

<label for="emotion">Select Emotion:</label>

<select id="emotion">

<option value="default">Default</option>

<option value="happy">Happy</option>

<option value="sad">Sad</option>

<option value="angry">Angry</option>

</select>


<label for="pitch">Pitch:</label>

<input type="range" id="pitch" min="0" max="2" value="1" step="0.1">


<label for="rate">Rate:</label>
```

```
<input type="range" id="rate" min="0.5" max="2" value="1" step="0.1">

<div class="button-container">

<br><br>

<button onclick="pauseAudio()">Pause</button>

<button onclick="resumeAudio()">Resume</button>

<button onclick="stopAudio()">Stop</button>

</div>




<button onclick="speak()">Speak</button>


<script>

const synth = window.speechSynthesis;

const voiceSelect = document.getElementById('voice');

const emotionSelect = document.getElementById('emotion');

const pitchInput = document.getElementById('pitch');

const rateInput = document.getElementById('rate');

const textInput = document.getElementById('text');


function populateVoiceList() {

const voices = synth.getVoices();

voiceSelect.innerHTML = '';

voices.forEach((voice, index) => {
```

```
const option = document.createElement('option');

option.textContent = `${voice.name} (${voice.lang})`;

option.value = index;

voiceSelect.appendChild(option);

});

}


populateVoiceList();

if (synth.onvoiceschanged !== undefined) {

synth.onvoiceschanged = populateVoiceList;

}


function clearText() {

textInput.value = '';

}


function copyToClipboard() {

const outputText = textInput.value;

navigator.clipboard.writeText(outputText).then(() => {

alert('Copied to clipboard!');

});

}
```

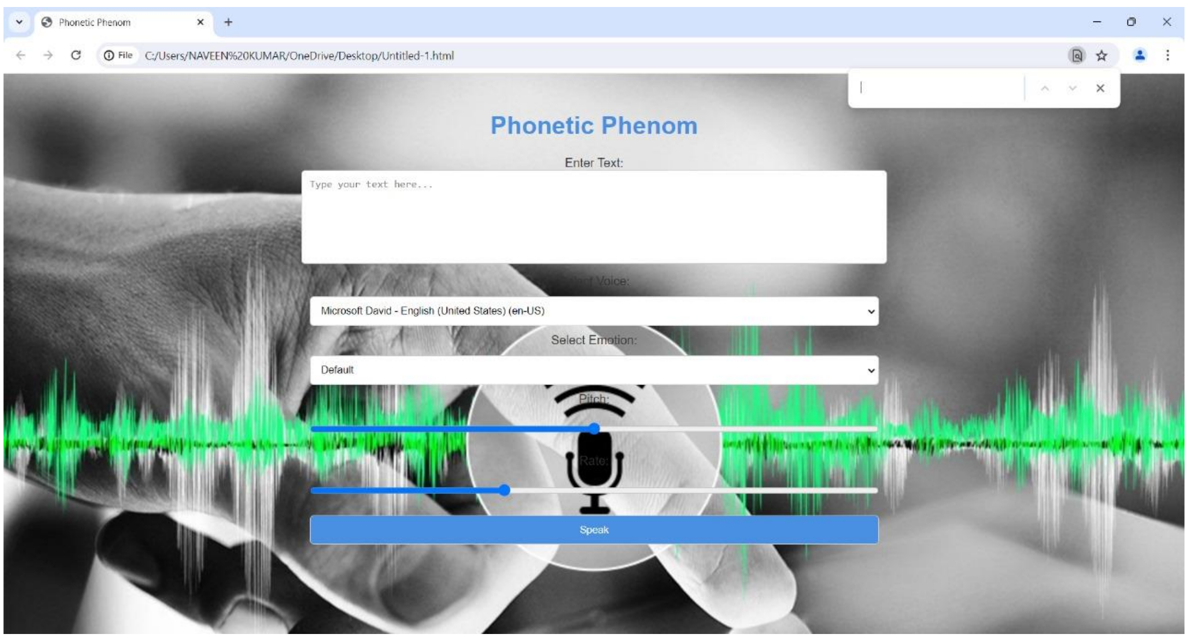
```
function toggleDarkMode() {

document.body.classList.toggle('dark-mode');

}


function speak() {

if (synth.speaking) {

console.error('SpeechSynthesisUtterance is already speaking.');

return;

}

if (textInput.value !== '') {

const utterThis = new SpeechSynthesisUtterance(textInput.value);

const selectedVoice = synth.getVoices()[voiceSelect.value];

if (selectedVoice) {

utterThis.voice = selectedVoice;

}

utterThis.pitch = pitchInput.value;

utterThis.rate = rateInput.value;


// Adjust based on emotion

switch (emotionSelect.value) {

case 'happy':

utterThis.pitch = parseFloat(pitchInput.value) + 0.5;

utterThis.rate = parseFloat(rateInput.value) + 0.5;
```

```
break;
case 'sad':
utterThis.pitch = Math.max(0, parseFloat(pitchInput.value) - 0.5);
utterThis.rate = Math.max(0.5, parseFloat(rateInput.value) - 0.5);
break;
case 'angry':
utterThis.pitch = parseFloat(pitchInput.value) + 0.3;
utterThis.rate = parseFloat(rateInput.value) + 0.7;
break;
default:
break;
}

synth.speak(utterThis);
}
}

function pauseAudio() {
if (synth.speaking && !synth.paused) {
synth.pause();
}
}
```

```
function resumeAudio() {
if (synth.paused) {
synth.resume();
}
}

function stopAudio() {
synth.cancel();
}
</script>
</body>
</html>
```

CHAPTER-9

OUTPUT SCREENSHOTS



CHAPTER – 10

CONCLUSION

The development of a sign language recognition system using machine learning and a camera marks a significant advancement in aiding speech-impaired individuals to communicate seamlessly with those unfamiliar with sign language. This project successfully integrates real-time video processing and advanced deep learning techniques to accurately recognize and translate gestures into text or speech. Utilizing frameworks like TensorFlow and computer vision libraries such as OpenCV, the system demonstrates high accuracy and efficiency. Additionally, the creation of a user-friendly interface ensures accessibility and ease of use. The project's successful implementation underscores the powerful potential of technology in addressing communication barriers and promoting inclusivity. Looking ahead, expanding the gesture vocabulary, incorporating natural language processing, and deploying the system across various platforms can further enhance its functionality and reach, ultimately contributing to greater social impact and accessibility for speech-impaired individuals.

.

CHAPTER – 11

FUTURE SCOPE

The future scope of the sign language recognition system for speech-impaired individuals includes several promising areas for enhancement. Expanding the gesture vocabulary to include more complex signs and multiple sign languages will cater to a broader audience. Improving model accuracy and robustness through larger, more diverse datasets and advanced algorithms is essential for better performance. Integrating natural language processing will enable context understanding and more natural speech synthesis. Optimizing for real-time performance and deploying edge computing solutions can reduce latency. Cross-platform deployment, including mobile and web applications, will increase accessibility. Enhancing user experience with customizable interfaces and incorporating user feedback will make the system more user-friendly. Additionally, integrating multimodal communication features like voice commands and haptic feedback can further improve interaction. Collaboration with the open-source community and partnerships with organizations can drive continuous improvement and innovation. These advancements will significantly enhance the system's effectiveness, making it a valuable tool for speech-impaired individuals worldwide.

REFERENCES

1. Farooq, M. S., Iftikhar, U., & Khelifi, A. (2022). A framework to make voting system transparent using blockchain technology. IEEE Access, 10, 59959-59969.
2. Hossain, S. S., Arani, S. A., Rahman, M. T., Bhuiyan, T., Alam, D., & Zaman, M. (2019, December). E-voting system using blockchain technology. In Proceedings of the 2019 2nd International Conference on Blockchain Technology and Applications (pp. 113-117).
3. Hjálmarsson, F. Þ., Hreiðarsson, G. K., Hamdaqa, M., & Hjálmtýsson, G. (2018, July). Blockchain-based e-voting system. In 2018 IEEE 11th international conference on cloud computing (CLOUD) (pp. 983-986). IEEE.
4. Benabdallah, A., Audras, A., Coudert, L., El Madhoun, N., & Badra, M. (2022). Analysis of blockchain solutions for E-voting: A systematic literature review. IEEE Access, 10, 70746-70759.
5. Srivastava, G., Dwivedi, A. D., & Singh, R. (2018). Crypto-democracy: A decentralized voting scheme using blockchain technology. In 15th International Joint Conference on eBusiness and Telecommunications. ICETE 2018 (pp. 508-513). SCITEPRESS Digital Library.
6. Vujičić, D., Jagodić, D., & Ranđić, S. (2018, March). Blockchain technology, bitcoin, and Ethereum: A brief overview. In 2018 17th international symposium infoteh-jahorina (infoteh) (pp. 1-6). IEEE.
7. Li, P., & Lai, J. (2019). LaT-Voting: Traceable anonymous E-voting on blockchain. In Network and System Security: 13th International Conference, NSS 2019, Sapporo, Japan, December 15–18, 2019, Proceedings 13 (pp. 234-254). Springer International Publishing.
8. Shukla, S., Thasmiya, A. N., Shashank, D. O., & Mamatha, H. R. (2018, September). Online voting application using ethereum blockchain. In 2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI) (pp. 873-880). IEEE.
9. Shahzad, B., & Crowcroft, J. (2019). Trustworthy electronic voting using adjusted blockchain technology. Ieee Access, 7, 24477-24488.

10. Rathee, G., Iqbal, R., Waqar, O., & Bashir, A. K. (2021). On the design and implementation of a blockchain enabled e-voting application within iot-oriented smart cities. IEEE Access, 9, 34165-34176.
11. Crosby, M., Pattanayak, P., Verma, S., & Kalyanaraman, V. (2016). Blockchain technology: Beyond bitcoin. Applied innovation, 2(6-10), 71.
12. Dimitriou, T. (2020). Efficient, coercion-free and universally verifiable blockchain-based voting. Computer Networks, 174, 107234.
13. Park, S., Specter, M., Narula, N., & Rivest, R. L. (2021). Going from bad to worse: from internet voting to blockchain voting. Journal of Cybersecurity, 7(1), tyaa025.
14. Khan, K. M., Arshad, J., & Khan, M. M. (2018). Secure digital voting system based on blockchain technology. International Journal of Electronic Government Research (IJEGR), 14(1), 53-62.
15. Adiputra, C. K., Hjort, R., & Sato, H. (2018, October). A proposal of blockchain-based electronic voting system. In 2018 second world conference on smart trends in systems, security and sustainability (WorldS4) (pp. 22-27). IEEE.
16. Beck, M. J., & Hensher, D. A. (2020). Insights into the impact of COVID-19 on household travel and activities in Australia–The early days under restrictions. Transport policy, 96, 76-93.
17. Duenas-Cid, D., Krivonosova, I., Serrano, R., Freire, M., & Krimmer, R. (2020, September). Tripped at the finishing line: The Åland Islands internet voting project. In International Joint Conference on Electronic Voting (pp. 36-49). Cham: Springer International Publishing.
18. Krimmer, R., Duenas-Cid, D., & Krivonosova, I. (2021). New methodology for calculating cost-efficiency of different ways of voting: is internet voting cheaper?. Public money & management, 41(1), 17-26.
19. Garg, K., Saraswat, P., Bisht, S., Aggarwal, S. K., Kothuri, S. K., & Gupta, S. (2019, April). A comparitive analysis on e-voting system using blockchain. In 2019 4th International Conference on Internet of Things: Smart Innovation and Usages (IoT-SIU) (pp. 1-4). IEEE.
20. Taş, R., & Tanrıöver, Ö. Ö. (2020). A systematic review of challenges and opportunities of blockchain for E-voting. Symmetry, 12(8), 1328.