

A Tool to convert Audio/Text to Sign Language

Project report submitted in partial fulfillment of the requirement for the award of the Degree of **B.E.**

By:

E. Sharanya (160617737013)
K. Rasagnya (160617737020)
P. Sreeja (160617737036)



Department of Information Technology
Stanley College of Engineering & Technology for Women (Autonomous)

Chapel Road, Abids, Hyderabad – 500001 (Affiliated to
Osmania University, Hyderabad,

Approved by AICTE, Accredited by NBA and
NAAC with 'A' grade)



**Stanley College of Engineering & Technology for
Women (Autonomous)**

Chapel Road, Abids, Hyderabad – 500001

**(Affiliated to Osmania University, Hyderabad, Approved by
AICTE, Accredited by NBA and NAAC with 'A' grade)**

CERTIFICATE

This is to certify that the project report entitled **A Tool To Convert Audio/Text to Sign Language** being submitted E. Sharanya, K.Rasagnya, and P. Sreeja in partial fulfillment for the award of the Degree of Bachelor of Engineering in Information Technology to the Osmania University is a record of bona fide work carried out by them under my guidance and supervision.

The results embodied in this project report have not been submitted to any other University or Institute for the award of any Degree or Diploma.

Internal Guide

HOD

External Examiner



Stanley College of Engineering & Technology for Women (Autonomous)

Chapel Road, Abids, Hyderabad – 500001

(Affiliated to Osmania University, Hyderabad, Approved by AICTE, Accredited by NBA and NAAC with 'A' grade)

ACKNOWLEDGEMENT

We take this opportunity to convey our heartfelt gratitude to each and every one who has supported us in every way or the other during the course of our project.

From the very core of my heart, we would like to express our sincere gratitude to our internal guide G. Sreelatha for her supervisory guidance. We express our profound gratitude to project coordinator Mr.T.Sandeep, and Dr.A.Kanaka Durga, HOD of Information Technology department. We are always grateful to our peers and friends who have always encouraged us and guided us whenever we needed assistance.

Vision of the Institute and Department

The Vision of the STLW:

Empower Women; Impact the World

Empowering girl students through professional education integrated with values and character to make an impact in the World.

The Mission STLW, in pursuance of its vision:

M1: Providing quality engineering education for girl students to make them competent and confident to succeed in professional practice and advanced learning.

M2: Establish state-of-art-facilities and resources to facilitate world class education.

M3: Integrating qualities like humanity, social values, ethics, and leadership in order to encourage contribution to society.

Vision of the Information Technology Department:

Empowering girl students with the contemporary knowledge in Information Technology, for their success in life

Mission of the Information Technology Department:

M1: Providing quality education and excellent environments for students to learn and practice various latest hardware, software and firmware platforms.

M2: To establish industry oriented training integrated with opportunities for team work, leadership.

M3: To groom students with values, ethics and social activities

Programme Educational Objectives

PEO1: Graduates shall have enhanced skills and contemporary knowledge to adapt new software and hardware technologies for professional excellence, employment and Research.

PEO2: Proficient in analyzing, developing and solving engineering problems to assist life-long learning and to develop team work.

PEO3: To inculcate self-confidence, acquire professional and ethical attitude, infuse leadership qualities, impart proficiency in soft-skills, and the ability to relate engineering with social issues.

Posand PSOs of IT Dept

PROGRAMME OUTCOMES

- 1) **Engineering knowledge:** Apply knowledge of mathematics, science, engineering fundamentals and an engineering specialization to the conceptualization of engineering models.
- 2) **Problem Analysis:** Identify, formulate, research literature and solve complex engineering problems reaching substantiated conclusions using first principles of mathematics and engineering sciences.
- 3) **Design/development of solutions:** Design solutions for complex engineering problems and design systems, components or processes that meet specified needs with appropriate consideration for public health and safety, cultural, societal, and environmental considerations.
- 4) **Conduct investigations of complex problems:** Conduct investigations of complex problems including design of experiments, analysis and interpretation of data, and synthesis of information to provide valid conclusions.
- 5) **Modern Tool Usage:** Create, select and apply appropriate techniques, resources, and modern engineering tools, including prediction and modeling, to complex engineering activities, with an understanding of the limitations.
- 6) **The engineer and society:** Function effectively as an individual, and as a member or leader in diverse teams and in multi-disciplinary settings.
- 7) **Environment & sustainability:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
- 8) **Ethics:** Demonstrate understanding of the societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to engineering practice.
- 9) **Individual and Teamwork:** Understand and commit to professional ethics, responsibilities, and norms of engineering practice.
- 10) **Communication:** understand the impact of engineering solutions in a societal context, demonstrate knowledge of, and need for sustainable development.
- 11) **Project Management and Finance:** Demonstrate a knowledge and understanding of management and business practices, such as risk and change management, and understand their limitations.
- 12) **Lifelong Learning:** Recognize the need for, and have the ability to engage in independent and life-long learning

PROGRAMME SPECIFIC OUTCOMES

PSO1: **Skilled Professional:** Ability to apply technical skills and involve in the Creation, maintenance and use of Computer, Computer Networks and Computer Information Systems.

PSO2: **Research Capability:** Ability to pursue research with academic excellence and core competence skills.

ABSTRACT

Keywords: Sign Language, Natural Language Processing, Django, Asynchronous server gateway interface(asgi), Web server gateway interface(wsgi)

Deaf people's mother tongue is sign language, which is a visual language. Unlike acoustically transmitted sound patterns, sign language employs body language and manual communication to express a person's thoughts. In our project, Sign language animations are done by integrating hand forms, hand orientation and movement all at the same time. It can be used by people who have difficulty in listening, people who can hear but cannot speak, and regular people to communicate with people who are deaf or hard of hearing. It is important for a deaf person's psychological, mental, and linguistic advancement to have access to a sign language. Deaf people's first languages should be accepted, and their schooling should be conducted bilingually in sign language and written or spoken language. Deaf and hard-of-hearing people use Indian Sign Language to communicate by making various body gestures. There are various groups of deaf people all over the world, and their languages will be different as well. American Sign Language (ASL) is used in the United States; British Sign Language (BSL) is used in the United Kingdom; and Indian Sign Language (ISL) is used in India for transmitting feelings and communicating. Manual speech and body language (non-manual communication) are used to express emotions, concepts, and feelings in "Indian Sign Language (ISL)." One handed, two handed, and non-manual ISL signals can both be grouped into three groups. Manual signs, such as one-handed and two-handed signs, are made with the signer's hands to communicate information. By altering body posture and facial expressions, non-manual signs are produced. This project concentrates on the website which we are going to create and mainly focuses on Audio or text to Manual sign language conversions enabling hearing impaired people to communicate with others.

CONTENTS

Title Page	I
Certificate by the Supervisor	II
Approval Sheet	III
Declaration	IV
Acknowledgement	V
Abstract	VI
Contents	VII
List of Figures	IX
List of Tables	X
Abbreviations	XI
Chapter 1	1
1.1 Introduction	1
1.2 Existing system	4
1.2.1 Limitations	4
1.3 Proposed system	4
1.3.1 Advantages	4
1.4 Problem Statement	4
1.4.1 Objectives	5
1.4.2 Applications	5
1.4.3 Limitations	5
1.4.4 Software Requirements	5
1.4.5 Hardware Requirements	7
1.5 Overview of the report	8
Chapter 2	9

Literature Review

Chapter 3	12
Methodology	
3.1 Schematic diagram for sign language conversion	12
3.1.1 Data Description / Modules	13
3.1.2 System model	15
3.2 Algorithm	16
3.3 Sample Code (Algorithm Implementation)	19
.	
3	
.	
Chapter 4	33
Results and Discussions	
4.1 Sample Input Data	33
4.2 Outputs	35
4.3 Test Cases /Result at different cases	38
Chapter 5	
Conclusions and Future Scope of Study	39
References	40
List of Publications	43
Appendix	44

LIST OF FIGURES

Figure No.	Title	PageNo.
1.1	Predefined gestures	2
3.1	Schematic diagram for sign language conversion	12
3.2	Asynchronous server gateway interface	15
3.3	ASGI application	15
3.4	WSGI with ASGI Application	16
3.5	Procedure for sign language website creation	17
3.6	Flow diagram for Text/Audio to sign Language Conversion	19
3.7	Audio to text conversion	20
3.8	Sentence splits into found words	20
3.9	words that do not found in database splits into letters	21
3.10	Asgi.py	22
3.11	settings.py	22
3.12	settings.py	23
3.13	URL.py	24
3.14	Views.py	25
3.15	Views.py	26
3.16	wsgi.py	27
3.17	Assets folder	27
3.18	About.html	28
3.19	Animation.html	29
3.20	Contact.html	20
3.21	Home.html	30
3.22	Login.html	30

3.23	Signup.html	31
3.24	Base.html	32
3.25	Main.py	32
4.1 – 4.4	Input	33
4.5	“Hello” in sign Language	35
4.6	“Thank” in sign Language	35
4.7	“You” in sign Language	35
4.8	“Stay” in sign Language	36
4.9	“Home” in sign Language	36
4.10	“Safe” in sign Language	37
4.11	“b” in sign Language	37
4.12	“u” in sign Language	37
4.13	“y” in sign Language	37
4.14	“Again” in sign Language	37

LIST OF TABLES

Table	Title	Page
4.1	Test cases for Audio/Text to sign language	38

Chapter 1

Introduction

1.1 Introduction

To communicate with each other and with other deaf people, deaf people require sign language. Furthermore, some ethnic groups with very distinct phonologies (such as Plain Indians Sign Language and Plateau Sign Language) have used sign languages to communicate with other ethnic groups. The study of physical sounds in human speech is alluded to as phonology. Sign language's phonology can be established. Phonemes are separate signs in a row of hand signs, rather than sounds. The following variables are taken into consideration:

1. **Hand shape** when rendering the sign is the first configuration.
2. **Hand Orientation:** This determines the direction in which the palm of the hand faces.
3. **Placement:** This refers to the location where the sign is over.
4. **Hand motion** when making the symbol.
5. **Line of interaction:** The part of the hand that creates contact with the body.
6. **Plane:** The sign is determined by the distance between the body and the object.

The fundamentals of sign languages have been presented in a concise manner. The Phonology section contains the core linguistic features that the scheme uses. The conditions that are taken into consideration are as follows:

1. **Location:** This is the location where the sign appears.
2. **Movement:** When making a symbol, the hand moves (straight, swaying, circularly).

3. Plane: The distance seen between body.

The parameters applied to these characteristics in relation of position, motion, and plane have been taken from the ASL dictionary's various simple signs.

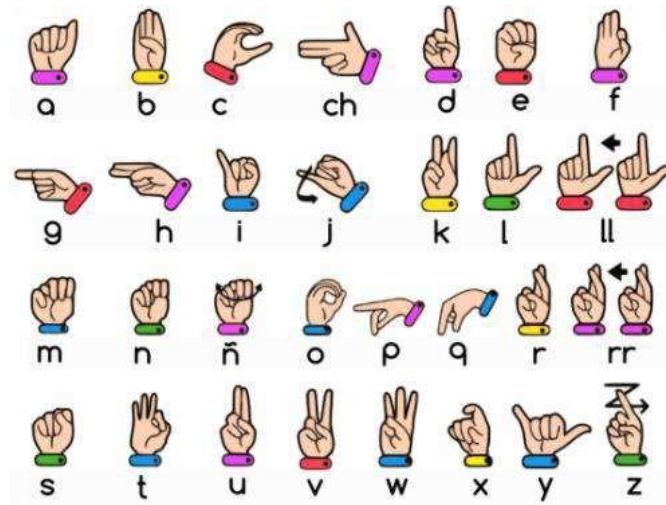


Fig1.1 Predefined gestures

Deafness and deaf people are as old as mankind, but the contact and education of deaf people was first recorded in the 16th century. In Spain, deaf children from wealthy families were put in the care of a monk who taught them how to write. It was important to speak in order to achieve money. The argument about oral vs. sign language, which has raged for decades, started with this case. The oral approach requires instructing or communicating with Deaf people using spoken words. In Germany, this method was refined to the level that it became known as the "German Method." The "French Method" gets its name from the fact that sign language was developed and used extensively in French deaf schools. In 1880, an effort was made to scrub Sign Language from the face of the earth. Hearing teachers and educationalists attended a conference in Milan (Italy) that issued a statement prohibiting further use of Sign Language in Deaf schools. Sign Language has developed into a hidden language. Outside of the school, Deaf students used Sign Language, but it became a living and universal language.

Four critical things need to be stated in the creation of a symbol. They are modes of hands, position, action and guidance. In Sign Language, finger spelling refers to how the alphabet's 26 letters are formed on the fingertips. Finger spelling is used to spell people's names, locations, ideas, and terms for which there are no signs or for which one has forgotten. Finger spelling is not the same as Sign Language. It's a code switch strategy in which you represent the written English word in space at the time you finger spell a term in English. Finger Spelling is restricted to individuals (deaf or hearing) who have been introduced to written English or some other spoken language.

1.2 Existing System

1.2.1 Limitations

The Existing solutions: Humans are making clever inventions every year to assist themselves and others who are affected by any disability, as technology advances at a breakneck pace. We want to make it easier for deaf people to communicate, so we've created a sign interpreter that converts audio to sign language automatically. Sign language is the only means of communication for the deaf. Physically handicapped people use sign language to communicate their feelings to others. Because common people struggle to learn the particular sign language, communication becomes impossible. Because sign language consists of a variety of hand movements and gestures, achieving the right precision at a low cost has become a mammoth task. We now have physical devices and software that can transform audio to sign language, so we're improving the tool with Natural Language Processing.

1.3 Proposed System

1.3.1 Advantages

The advantages of our tool are

- a.Website is cost-effective,
- b.It is adaptable by people, and
- c. Algorithm is very simple to implement.

1.4 Problem Statement

- Lots of Problems arise when people that are impaired from hearing or speaking try to communicate with normal people.
- The reason is that they mostly communicate using sign language and normal people generally don't know anything about sign Language.

1.4.1 Objectives

1. The main objective of our project is to develop a tool which is cost effective for hearing impaired people, which converts Audio/Text to sign Language
2. Collection of huge number of datasets/animations to train the model
- 3.To develop a tool which is easy to use for deaf people to communicate with normal ones.

1.4.2 Applications

The main applications of our project are:

- The website is open source and secure (free of cost)
- Website is accurate
- Speed

1.4.3 Limitations

Limitation of the tool we are going to develop is incorporation of Facial expressions which means manual signs

1.4.4 Software Requirements

The software requirements need to develop the tool which converts Audio/Text to sign Language are:

- Anaconda (python 3.8)
- SQLite
- NLP
- Django
- Blender
- Operating System- Ubuntu 14.01 / Windows

Python: Python is a dynamically semantic, interpreted, programming language.

Its high-level built-in data structures, along with dynamic typing and dynamic binding, make it ideal for Rapid Application Development and as a scripting or glue language for connecting existing components. Python's

plain, easy-to-learn syntax stresses readability, which lowers software maintenance costs. Modules and packages are supported by Python, which facilitates software modularity and code reuse. On all major platforms, the Python parser and comprehensive standard library are available for free in source or binary form and can be downloaded for free.

SQLite: The sqlite3 module, written by Gerhard Haring, can be used to merge SQLite3 with Python. It offers a SQL client that complies with PEP 249's DB-API 2.0 specification. This module is included by default with Python versions 2.5.x and up, so you don't have to install it separately.

To use the sqlite3 module, you must first create a reference object to represent the database, and then you can optionally create a cursor object to aid in the execution of all SQL statements.

Python sqlite3 module APIs: The following are key sqlite3 module routines that will allow you to work with SQLite databases from your Python software. If you want a more advanced application, check out the official documentation for the Python sqlite3 framework.

Django: Django is a high-level Python web platform for building stable and maintainable websites quickly. Django is a web framework built by seasoned developers that takes care of a lot of the heavy lifting so you can concentrate on writing the app instead of reinventing the wheel. It's free and open source, with a vibrant and involved community, excellent documentation, and a variety of free and paid service options.

Django assists you in creating applications that is:

- Complete
- Versatile
- Secure
- Scalable

- Maintainable
- Portable

Blender: Blender is a free and opensource 3D modelling software. Modeling, rigging, animation, simulation, rendering, compositing, and motion tracking are all supported, as well as video editing and game production. Advanced users use Blender's Python scripting API to configure the program and create customized tools, which are often used in future Blender launches. Individuals and small studios can profit from Blender's unified pipeline and responsive creation method.

NLP: The way we, as people, interact is referred to as natural language. voice and text, to be specific. We live in a text-heavy environment. Consider how much text you see on a daily basis: Menus, Signs, Email, SMS, Web Pages, and so many more.

Consider expressions for a moment. As a species, we can talk to each other more than we write. It's possible that learning to communicate is better than learning to talk. We connect with each other by voice and text. Given the significance of this category of data, we need methods to comprehend and reason for natural language, much as we do for other types of data.

1.4.5 Hardware Requirements

- Device (Mobile/Laptop)
- Mic
- Monitor to view output
- Internet for voice recognition
- Web browser

1.5 Overview of the report

This job necessitates converting audio signals to text using speech-to-text APIs such as the Google API, and then coding with Natural Language Processing, Machine Learning, and Python semantics.

Chapter-2 is all about the literature survey that we have done in order to develop our website. A literature review is an objective examination of academic sources (such as books, published papers, and theses) that are relevant to a particular subject or research issue. It's generally done as part of a report, review, or research paper to help you place some research in practice of what's already out there.

Chapter-3 deals with a brief methodology and process involved in developing our project. All the related codes and executions discussed in there.

Chapter-4 deals with the results, sample input and output sections and chapter-5 is all about conclusions and future scope of the project we have developed.

Chapter 2

Literature Survey

We are going to represent literature overview of few papers that we have studied for choosing the topic as follows:

[1] S. Shrenika and M. Madhu Bala, "Sign Language Recognition Using Template Matching Technique," 2020 International Conference on Computer Science, Engineering and Applications (ICCSEA), Gunupur, India, 2020, pp. 1-5, doi: 10.1109/ICCSEA49143.2020.9132899. This paper proposed the image processing work of sign language recognition which first converts the image into grayscale and then into rgb and finally gives the result using template matching technique.

[2] Ankita Harkude#1, Sarika Namade#2, Shefali Patil#3, Anita Morey #4, Department of Information Technology, Usha Mittal Institute of Technology, Audio to Sign Language Translation for Deaf People, (IJEIT) Volume 9, Issue 10, April 2020. This device takes audio as data, converts it to text, and then shows the appropriate Indian Sign Language images or GIFs that have been pre-programmed. The use of this device facilitates contact between learning disabled people.

[3] M. Sanzidul Islam, S. Sultana Sharmin Mousumi, N. A. Jessan, A. Shahariar Azad Rabby and S. Akhter Hossain, "Ishara-Lipi: The First Complete MultipurposeOpen Access Dataset of Isolated Characters for Bangla Sign Language," 2018 International Conference on Bangla Speech and Language Processing (ICBSLP), Sylhet, Bangladesh, 2018, pp. 1-4, doi: 10.1109/ICBSLP.2018.8554466. With no managed lab setting, a real-time BdSL interpretation has a broader cultural impact as well as an intriguing research avenue. It's also a difficult job because of the wide range of subjects (age, gender, colour, etc.), the complexity of attributes, and the similarity of signals and organized histories. However, the current dataset for BdSL classification tasks is primarily

designed in a lab-friendly environment, limiting the use of versatile deep learning technologies.

[4] S. Tornay, M. Razavi and M. Magimai.-Doss, "Towards Multilingual Sign Language Recognition," ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Barcelona, Spain, 2020, pp. 6309-6313, doi: 10.1109/ICASSP40776.2020.9054631. Modeling of video distribution information including certain hand shapes and gestures is needed for hand gesture recognition. This necessitates a sufficient amount of sign language-specific data. Since the world is generally inadequate, this is a difficult task. Hand shape knowledge can be determined by accumulating resources from various hand gestures, according to the literature.

[5] M. Xie and X. Ma, "End-to-End Residual Neural Network with Data Augmentation for Sign Language Recognition," 2019 IEEE 4th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC), Chengdu, China, 2019, pp. 1629-1633, doi: 10.1109/IAEAC47372.2019.8998073. The Residual Neural Network is used in this paper to introduce earlier part recognition of American Sign Language (ASL). This study's sign language dataset contains 36 groups of structured sign language phrases, including 0-9 figures and Auxiliary letters A-Z. Via use such tools, we were able to obtain 17640 sign language pictures as model training data. The proposed model has a 99.4% accuracy rate.

[6] S. He, "Research of a Sign Language Translation System Based on Deep Learning," 2019 International Conference on Artificial Intelligence and Advanced Manufacturing (AIAM), Dublin, Ireland, 2019, pp. 392-396, doi: 10.1109/AIAM48774.2019.00083. By understanding the meaning of sign language, the system will increase recognition accuracy; 3. This model integrates touch located network, 3D CNN dimensionality reduction network, and LSTM encryption

to create the classification method to overcome the issue of RGB translator image or creating partitions in obvious problems.

[7] K. Saija, S. Sangeetha and V. Shah, "WordNet Based Sign Language Machine Translation: from English Voice to ISL Gloss," 2019 IEEE 16th India Council International Conference (INDICON), Rajkot, India, 2019, pp. 1-4, doi: 10.1109/INDICON47234.2019.9029074. Human life will be incomplete without communication. It is difficult for an individual with a hearing and speech impairment to communicate with others. An end-to-end method for converting English voice to Indian Sign Language (ISL) gloss (written form of sign language) is proposed in this paper, which will aid deaf people in communicating with others.

Chapter 3

Methodology

3.1. Architecture

After website creation is done, process goes in this way:

The methodology follows as input is given to the tool in the form of text/Audio. If input is given as audio, it gets converted into text using speech recognition tools in python based on Natural language processing.

When an audio is given as input it converts into text using NLP and then the input text is searched for matching in the database(assets folder of animations).If the word is found in the database the video is given as output to the user otherwise sentence is broken into words and it fetches the videos of those words and then combines them into a single video.

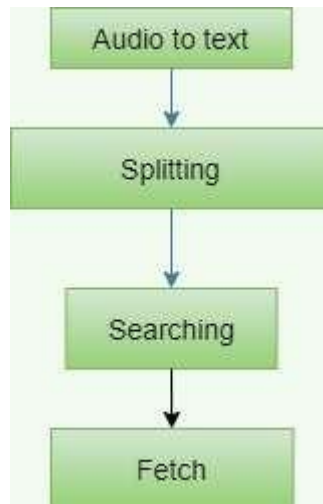


Fig.3.1: Schematic diagram for sign language conversion

3.1.1 Data Description

Asynchronous Server Gateway Interface is an acronym for ASGI. It enhances the functionality of WSGI (Web Server Gateway Interface), which is a standardized method of communication between the server and a web application in most Python frameworks, including Django. Both ASGI and WSGI are specifications for providing a standard interface between Python web servers, applications, and frameworks. The only concern designers had when developing WSGI was to create a protocol that provides a common ground for web development so that users can easily switch between multiple web frameworks without worrying about the specifics of how the new framework communicates with the server. And while WSGI did a good job of dealing with these concerns, when the relatively new protocols other than HTTP (HyperText Transfer Protocol), particularly WebSocket, began to gain popularity among web developers, WSGI failed to provide a method of creating applications that could handle these protocols.

Because WSGI applications can only accept requests from the server and return replies to the client/server, it is intrinsically suited to only handle HTTP protocol. WSGI applications are single, synchronous callable that take a request as input and return a response, allowing you to:

- Connections are short-lived, which is ideal for HTTP but not for long-polling HTTP or WebSocket, which have long-lived connections.
- Because requests in applications only have one route to follow, protocols with multiple incoming events (such as receiving WebSocket frames) cannot be handled by a WSGI application.

ASGI is separated into two sections, each with its own set of responsibilities:

1. A protocol server that suspends sockets and maps them into connections and event messages for each connection.
2. An application that runs within a protocol server is created only once per connection and treats event messages as they occur.

The server, like WSGI, hosts and runs the application within it, as well as passing incoming requests to it in a standardized format. Applications, unlike WSGI, are objects that accept events instead of just simple callable and must run as coroutines capable of handling asynchronous I/O operations. In contrast to WSGI, a relationship consists of two parts:

1. A connection scope is a representation of a protocol connection with a user that lasts for the period of the connection.
2. When anything occurs on that connection, events are sent to the application.

Applications are started by passing a connection scope, and then they run in an event loop, handling events and sending data back to the client in the form of events. A single incoming socket/connection is mapped to an application instance that lasts for the period of the connection, or maybe a little longer if any cleanup is required. A single asynchronous callable defines an ASGI application. It approves scope, which contains data about the user requests, send, which enables you to send events to the client, and receive, which allows you to receive events from the client.

The ASGI application can now accept both incoming and outgoing events as a matter of the reordering of the design of the model, removing the restriction of the WSGI application having a single path for incoming requests. Not only that, but the ASGI application can also run a background coroutine, allowing it to do more than just handle requests in the background.

In an ASGI application, every event you send or receive is a Python dict with a predefined format. ASGI applications can be easily swapped between different web servers thanks to these predefined event formats. These events still have a type root level key, which can be helpful in determining the structure of the event

3.1.2 System Model

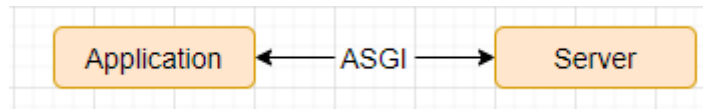


Fig3.2 *Asynchronous server gateway interface*

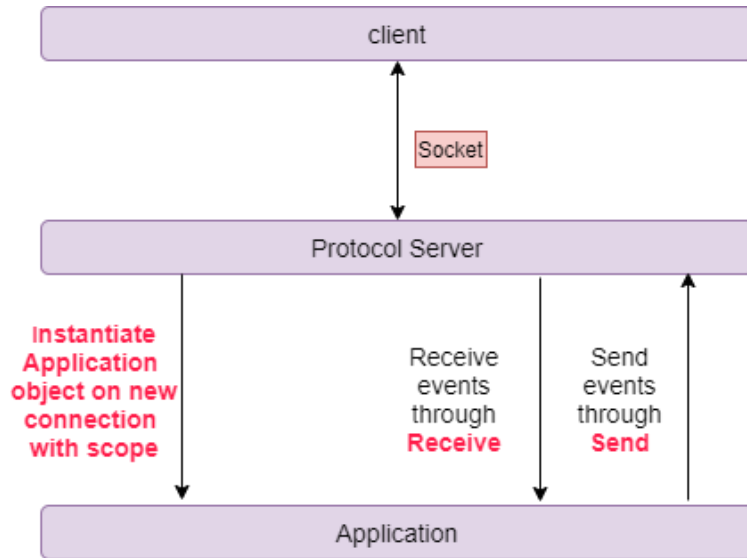


Fig3.3: *ASGI application*

Where application is a WSGI application that takes two parameters: environ, which holds information about incoming requests, and start response, which is a callable that returns the HTTP header. To make ASGI backward compatible with WSGI applications, we need to allow it to run WSGI synchronous applications within an asynchronous coroutine. Additionally, ASGI receives incoming request information via scope, while WSGI accepts the environment. As a result, we must map the environment to scope.

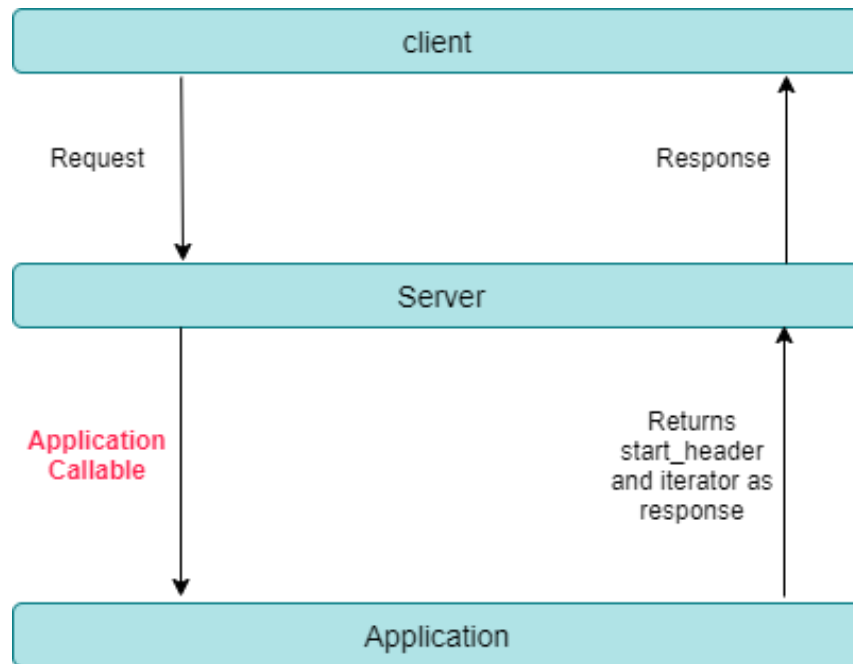


Fig 3.4: WSGI with ASGI Application

3.2 Algorithm

The planet is home to thousands of languages. Using apps like Google Translate, almost all of the most popular languages can be translated in real time. The software is primarily based on a natural language processing (NLP) algorithm that takes in one language via text and outputs its translation via text. Some programmes use voices using a microphone to interpret a message without requiring the user to type it in. This is particularly useful for spoken languages. But what about the 70 million people who are unable to talk or hear and rely on sign language to communicate?

Sign language is distinct from spoken language since it cannot be spoken. Any existing translation programme would not be able to read anything that cannot be spoken. So, what's the best way to decode sign language? Of course, with a camera and a little AI magic! SIGNALL, who uses coloured gloves and several cameras to understand the signals, is one of the current sign language translators and uses cameras to translate. Cameras are used to provide eyes to machines, enabling them to view the outside world.

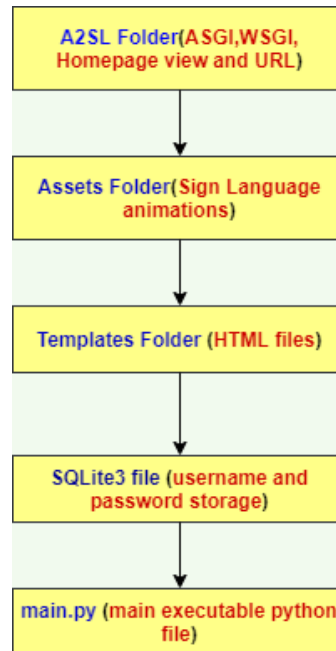


Fig.3.5: Procedure for sign language website creation

A. A2SL FOLDER:

1. ASGI.py and WSGI.py:

To make ASGI backward compatible with WSGI applications, we need to allow it to run WSGI synchronous applications within an asynchronous coroutine. Additionally, ASGI receives incoming request information via scope, while WSGI accepts the environment. As a result, we must map the environment to scope. Asynchronous Server Gateway Interface is an acronym for ASGI. It enhances the functionality of WSGI (Web Server Gateway Interface), which is a standardized method of communication between the server and a web application in most Python frameworks, including Django. Both ASGI and WSGI are specifications for providing a standard interface between Python web servers, applications, and frameworks.

2. Homepage view.py, URL's and Settings:

The website view consists of “Home”, “about”, “Login”, “signup”, “contact” and “converter”. To make these views possible, all the html files related to those terms

are programmed in view.py and when you click on “login” in the website it should open the login page and settings are done accordingly.

B. ASSETS FOLDER:

All the recorded and created sign language animations are stored in this folder. If the input given by the user matches with the name of the sign language animation that output is displayed to the user.

C. TEMPLATES FOLDER:

Website Designing is done here. Every html file which is added in view.py has its own description. For example, home.html consists of information related to the homepage of the website and about.html consists of information related to the people who created the website. Similarly, Login.html consists of information related to the login page.

D. SQLite3 file:

The data related to username and password in the login page stored in a database file format sqlite3.

E. Main.py:

This is the main program and we have to import `execute_from_command_line` from `django.core.management` and run this main.py executable python file in anaconda prompt. Prompt gives server information through URL as an output. Paste that URL in a browser and website is shown.

3.3 Algorithm Implementation:

The flow diagram says that when an audio is given as input it converts into text using NLP and then the input text is searched for matching in the database(assets folder of animations).If the word is found in the database the video is given as output to the user otherwise sentence is broken into words and it fetches the videos of those words and then combines them into a single video.

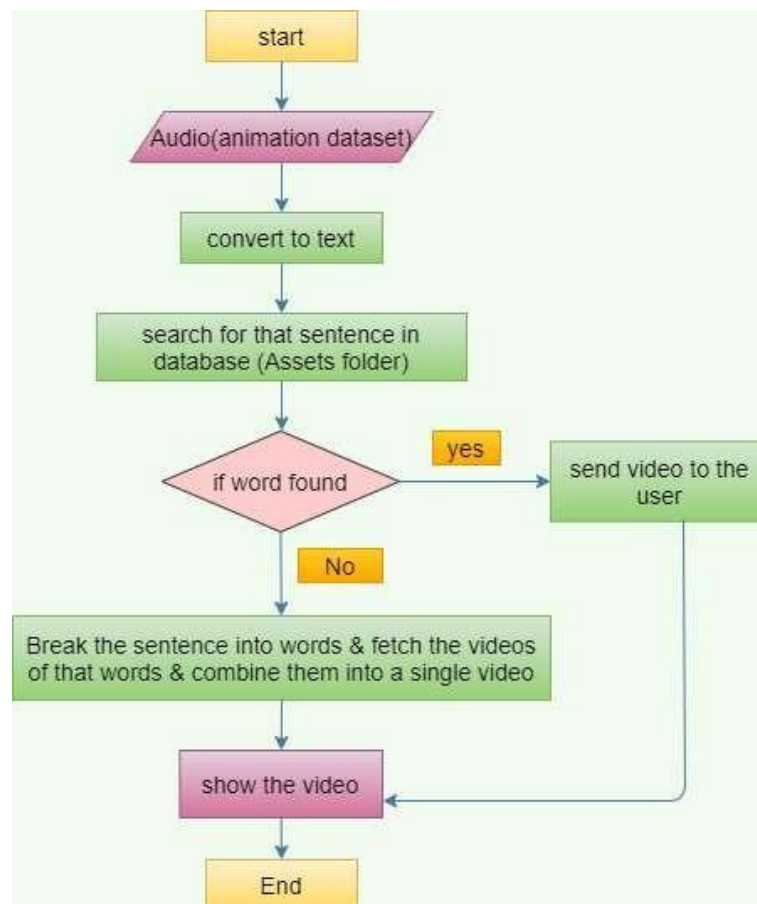


Fig3.6 : flow diagram for Text/Audio to sign Language Conversion

A. Audio to Text Conversion:

Audio input is taken using the python PyAudio module.

Conversion of audio to text using microphone

Dependency parser is used for analyzing grammar of the sentence and obtaining a relationship between words.

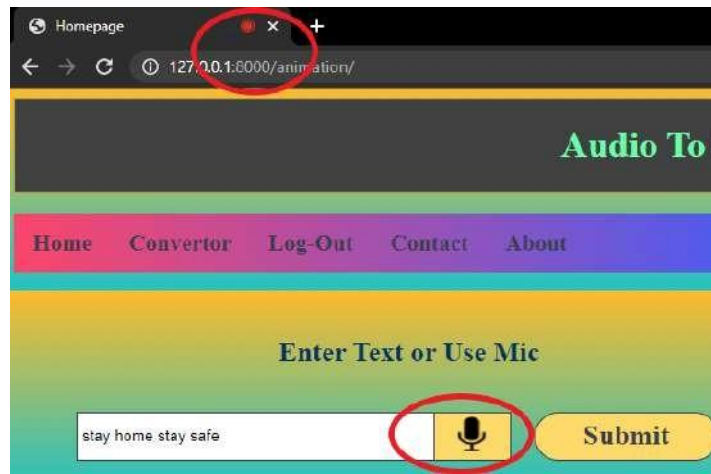


Fig.3.7: Audio to text conversion

B. Splitting:

If the word is found, send video to the user as output, otherwise break the sentence into words and fetch the videos of those words and combine them into a single video and show the video to the user as output.



Fig.3.8: Sentence splits into found words

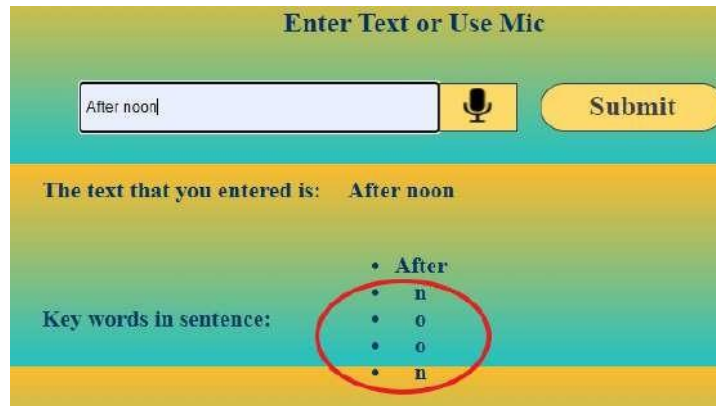


Fig.3.9: words that do not found in database splits into letters

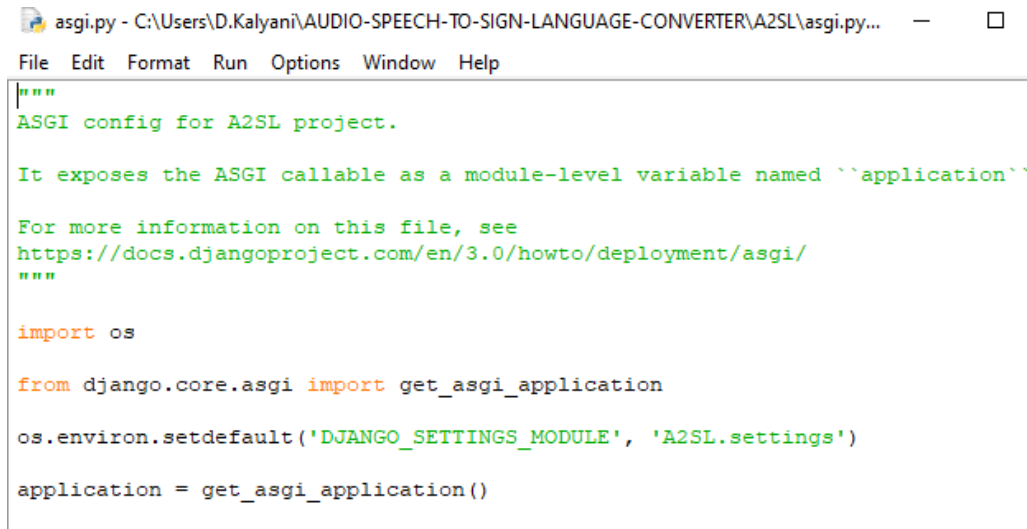
C. Searching:

Create animations for datasets using blender tool. The word library has been expanded to include most of the commonly used word in English Language as well as technical words and others which are available. Speech to text conversions can be made more accurate and text processing can be optimized using various NLP algorithms which are available. Search for the entered text/Audio in the database created. Create another database file through SQLite to store the usernames who gets signup/login into the website. Client and server communicates through Django modules (ASGI & WSGI)

D. Fetching Sign Language Animation:

After clicking on the submit button, sign language animation is displayed.

ASGI.py



```
asgi.py - C:\Users\D.Kalyani\AUDIO-SPEECH-TO-SIGN-LANGUAGE-CONVERTER\A2SL\asgi.py...
File Edit Format Run Options Window Help

"""
ASGI config for A2SL project.

It exposes the ASGI callable as a module-level variable named ``application``

For more information on this file, see
https://docs.djangoproject.com/en/3.0/howto/deployment/asgi/
"""

import os

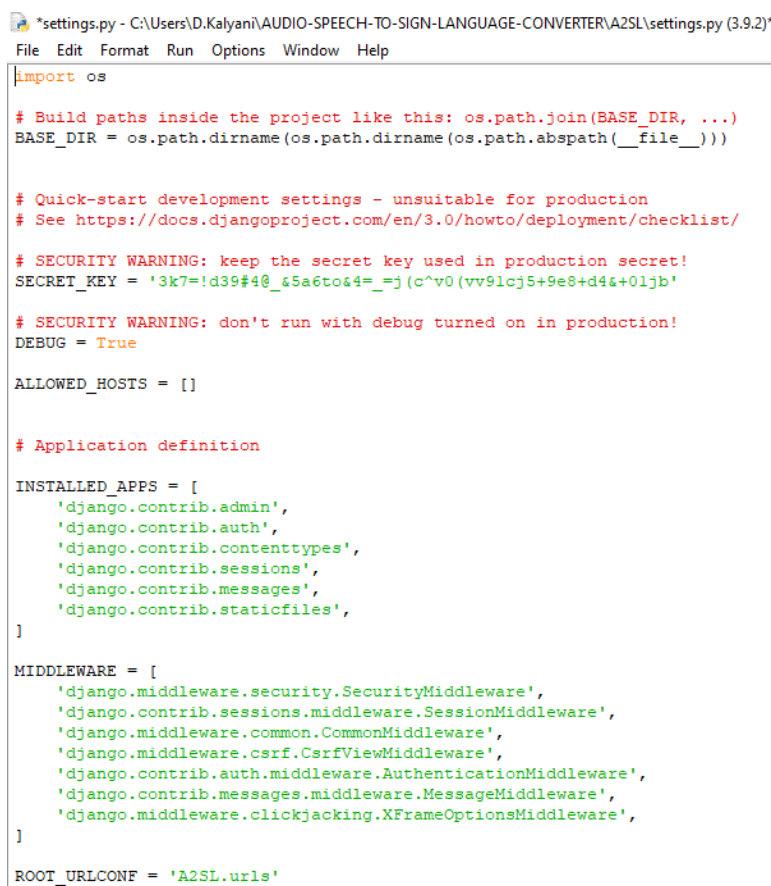
from django.core.asgi import get_asgi_application

os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'A2SL.settings')

application = get_asgi_application()
```

Fig:3.10 Asgi.py

Settings.py



```
*settings.py - C:\Users\D.Kalyani\AUDIO-SPEECH-TO-SIGN-LANGUAGE-CONVERTER\A2SL\settings.py (3.9.2)
File Edit Format Run Options Window Help

import os

# Build paths inside the project like this: os.path.join(BASE_DIR, ...)
BASE_DIR = os.path.dirname(os.path.dirname(os.path.abspath(__file__)))

# Quick-start development settings - unsuitable for production
# See https://docs.djangoproject.com/en/3.0/howto/deployment/checklist/

# SECURITY WARNING: keep the secret key used in production secret!
SECRET_KEY = '3k7=!d39#4@_&5a6to&4=_=j(c^v0(vv9lcj5+9e8+d4&+0ljb'

# SECURITY WARNING: don't run with debug turned on in production!
DEBUG = True

ALLOWED_HOSTS = []

# Application definition

INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
]

MIDDLEWARE = [
    'django.middleware.security.SecurityMiddleware',
    'django.contrib.sessions.middleware.SessionMiddleware',
    'django.middleware.common.CommonMiddleware',
    'django.middleware.csrf.CsrfViewMiddleware',
    'django.contrib.auth.middleware.AuthenticationMiddleware',
    'django.contrib.messages.middleware.MessageMiddleware',
    'django.middleware.clickjacking.XFrameOptionsMiddleware',
]

ROOT_URLCONF = 'A2SL.urls'
```

Fig:3.11 settings.py



```
settings.py - C:\Users\D.Kalyani\AUDIO-SPEECH-TO-SIGN-LANGUAGE-CONVERTER\A2SL\settings.py (3.9.2)
File Edit Format Run Options Window Help
# https://docs.djangoproject.com/en/3.0/ref/settings/#auth-password-validators

AUTH_PASSWORD_VALIDATORS = [
    {
        'NAME': 'django.contrib.auth.password_validation.UserAttributeSimilarityValidator',
    },
    {
        'NAME': 'django.contrib.auth.password_validation.MinimumLengthValidator',
    },
    {
        'NAME': 'django.contrib.auth.password_validation.CommonPasswordValidator',
    },
    {
        'NAME': 'django.contrib.auth.password_validation.NumericPasswordValidator',
    },
]

# Internationalization
# https://docs.djangoproject.com/en/3.0/topics/i18n/

LANGUAGE_CODE = 'en-us'

TIME_ZONE = 'UTC'

USE_I18N = True

USE_L10N = True

USE_TZ = True

# Static files (CSS, JavaScript, Images)
# https://docs.djangoproject.com/en/3.0/howto/static-files/

STATIC_URL = '/static/'

STATICFILES_DIRS = [
    os.path.join(BASE_DIR, "assets"),
]
```

Fig:3.12 Settings.py

URL.py :



```
urls.py - C:\Users\D.Kalyani\AUDIO-SPEECH-TO-SIGN-LANGUAGE-CONVERTER\A2SL\urls.py ...
File Edit Format Run Options Window Help

"""A2SL URL Configuration

The `urlpatterns` list routes URLs to views. For more information please see:
    https://docs.djangoproject.com/en/3.0/topics/http/urls/
Examples:
Function views
    1. Add an import:  from my_app import views
    2. Add a URL to urlpatterns:  path('', views.home, name='home')
Class-based views
    1. Add an import:  from other_app.views import Home
    2. Add a URL to urlpatterns:  path('', Home.as_view(), name='home')
Including another URLconf
    1. Import the include() function: from django.urls import include, path
    2. Add a URL to urlpatterns:  path('blog/', include('blog.urls'))
"""
from django.contrib import admin
from django.urls import path
from . import views

urlpatterns = [
    path('admin/', admin.site.urls),
    path('about/', views.about_view, name='about'),
    path('contact/', views.contact_view, name='contact'),
    path('login/', views.login_view, name='login'),
    path('logout/', views.logout_view, name='logout'),
    path('signup/', views.signup_view, name='signup'),
    path('animation/', views.animation_view, name='animation'),
    path('', views.home_view, name='home'),
    path('animation/', views.animation_view, name='animation')
]
```

Fig:3.13 : URL.py

Views.py:

```
.py - C:\Users\D.Kalyani\AUDIO-SPEECH-TO-SIGN-LANGUAGE-CONVERTER\A2SL\views.py (3.9.2)
it Format Run Options Window Help
jango.http import HttpResponse
jango.shortcuts import render, redirect
jango.contrib.auth.forms import UserCreationForm, AuthenticationForm
jango.contrib.auth import login, logout
ltk.tokenize import word_tokenize
ltk.corpus import stopwords
ltk.stem import WordNetLemmatizer
    nltk
jango.contrib.staticfiles import finders
jango.contrib.auth.decorators import login_required

me_view(request):
    return render(request, 'home.html')


out_view(request):
    return render(request, 'about.html')

ntact_view(request):
    return render(request, 'contact.html')

_login_required(login_url="login")
imation_view(request):
    if request.method == 'POST':
        text = request.POST.get('sen')
        #tokenizing the sentence
        text.lower()
        #tokenizing the sentence
        words = word_tokenize(text)

        tagged = nltk.pos_tag(words)
        tense = {}
        tense["future"] = len([word for word in tagged if word[1] == "MD"])
        tense["present"] = len([word for word in tagged if word[1] in ["VBP", "VBZ", "VBG"]])
        tense["past"] = len([word for word in tagged if word[1] in ["VBD", "VBN"]])
        tense["present_continuous"] = len([word for word in tagged if word[1] in ["VBG"]])
```

Fig:3.14 Views.py



```
views.py - C:\Users\D.Kalyani\AUDIO-SPEECH-TO-SIGN-LANGUAGE-CONVERTER\A2SL\views.py (3.9.2)
File Edit Format Run Options Window Help

        return render(request, 'animation.html', {'words': words, 'text': text})
    else:
        return render(request, 'animation.html')

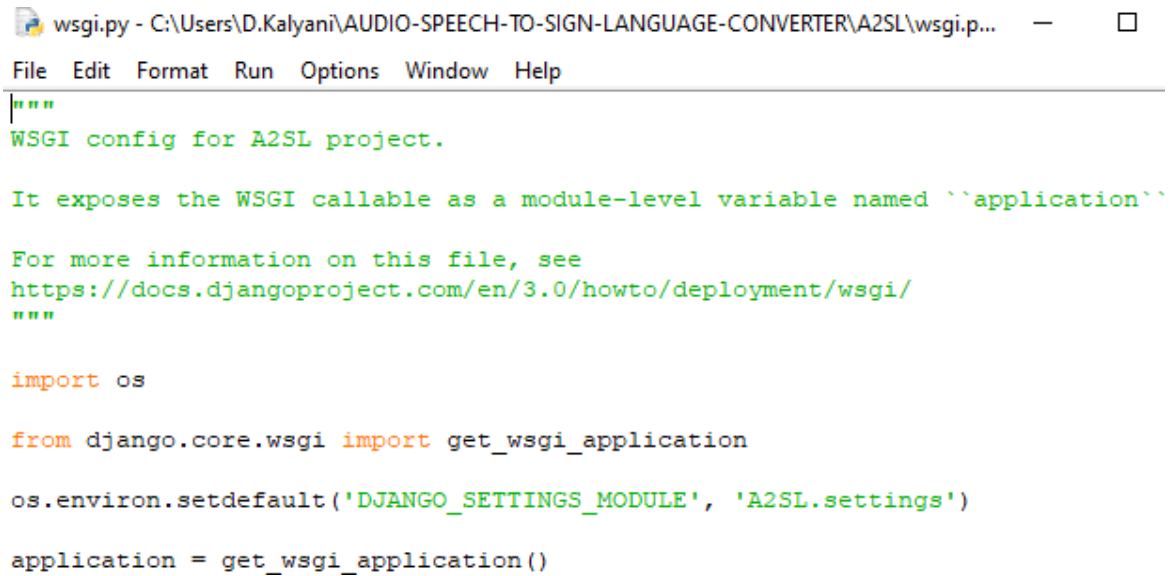
def signup_view(request):
    if request.method == 'POST':
        form = UserCreationForm(request.POST)
        if form.is_valid():
            user = form.save()
            login(request, user)
            # log the user in
            return redirect('animation')
    else:
        form = UserCreationForm()
    return render(request, 'signup.html', {'form': form})

def login_view(request):
    if request.method == 'POST':
        form = AuthenticationForm(data=request.POST)
        if form.is_valid():
            #log in user
            user = form.get_user()
            login(request, user)
            if 'next' in request.POST:
                return redirect(request.POST.get('next'))
            else:
                return redirect('animation')
    else:
        form = AuthenticationForm()
    return render(request, 'login.html', {'form': form})

def logout_view(request):
    logout(request)
    return redirect("home")
```

Fig:3.15 Views.py

wsgi.py:



```
wsgi.py - C:\Users\D.Kalyani\AUDIO-SPEECH-TO-SIGN-LANGUAGE-CONVERTER\A2SL\wsgi.p...
File Edit Format Run Options Window Help
"""
WSGI config for A2SL project.

It exposes the WSGI callable as a module-level variable named ``application``

For more information on this file, see
https://docs.djangoproject.com/en/3.0/howto/deployment/wsgi/
"""

import os

from django.core.wsgi import get_wsgi_application

os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'A2SL.settings')

application = get_wsgi_application()
```

Fig:3.16 wsgi.py

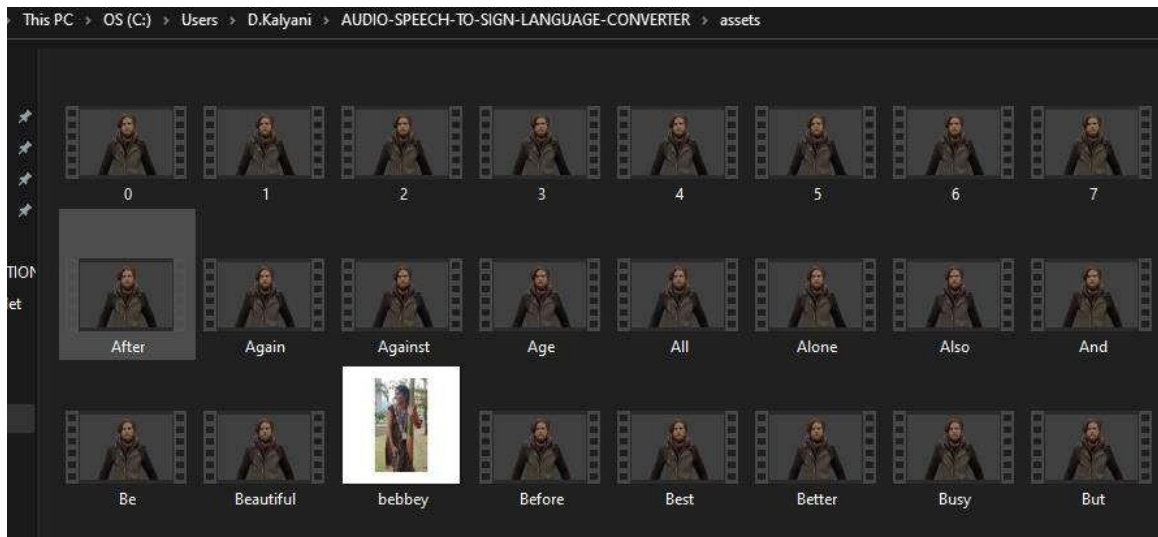
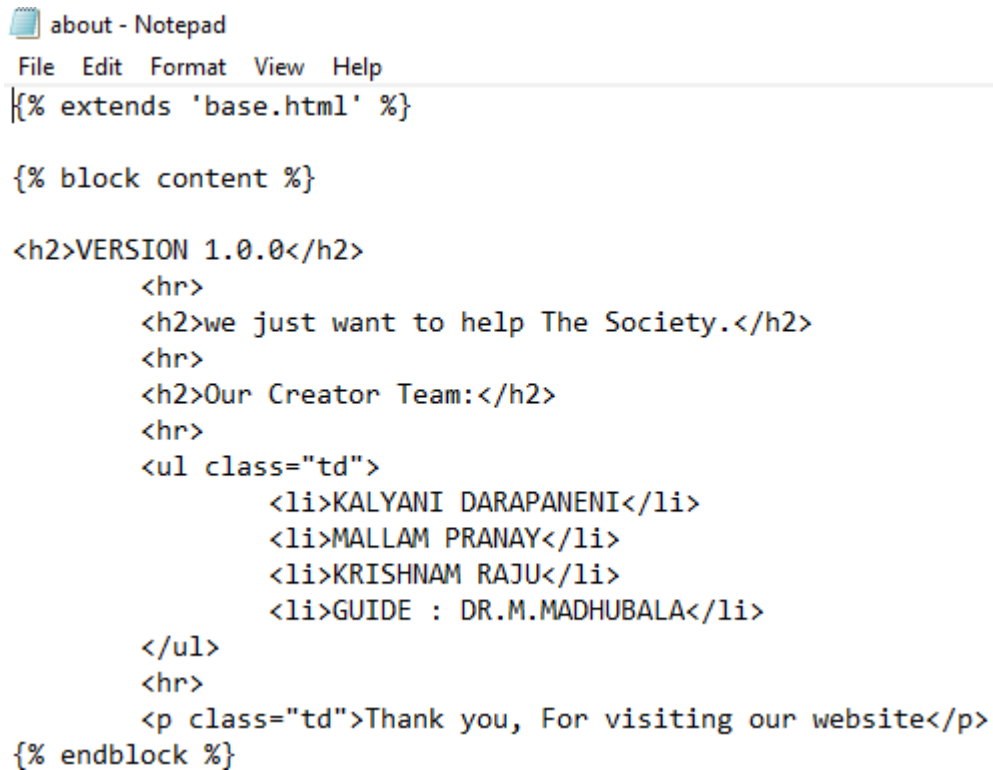


Fig 3.17: Assets folder

HTML FILES:

About.html:



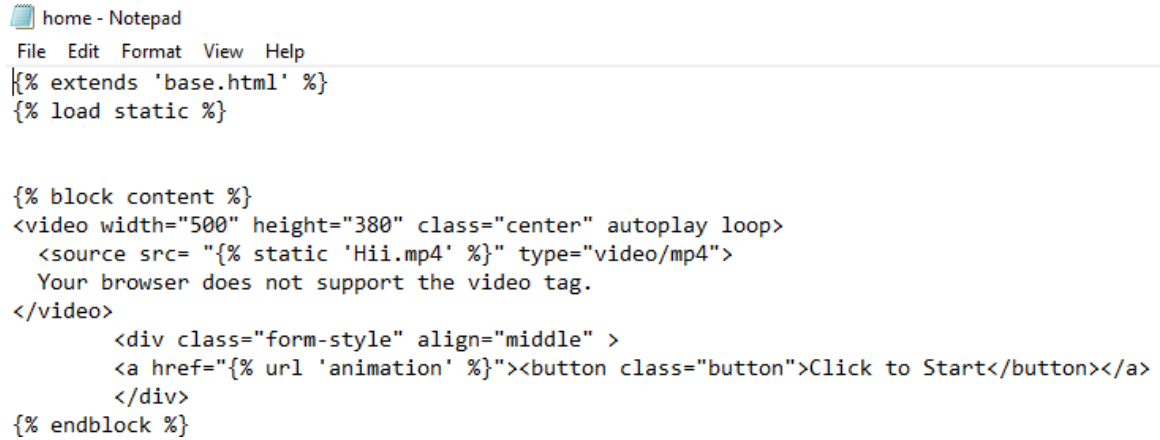
```
about - Notepad
File Edit Format View Help
{% extends 'base.html' %}

{% block content %}

<h2>VERSION 1.0.0</h2>
    <hr>
    <h2>we just want to help The Society.</h2>
    <hr>
    <h2>Our Creator Team:</h2>
    <hr>
    <ul class="td">
        <li>KALYANI DARAPANENI</li>
        <li>MALLAM PRANAY</li>
        <li>KRISHNAM RAJU</li>
        <li>GUIDE : DR.M.MADHUBALA</li>
    </ul>
    <hr>
    <p class="td">Thank you, For visiting our website</p>
{% endblock %}
```

Fig:3.18 About.html

Home.html:



```
home - Notepad
File Edit Format View Help
{% extends 'base.html' %}
{% load static %}

{% block content %}
<video width="500" height="380" class="center" autoplay loop>
  <source src="{% static 'Hii.mp4' %}" type="video/mp4">
  Your browser does not support the video tag.
</video>
  <div class="form-style" align="middle" >
    <a href="{% url 'animation' %}"><button class="button">Click to Start</button></a>
  </div>
{% endblock %}
```

Fig:3.21 Home.html

Login.html:



```
login - Notepad
File Edit Format View Help
{% extends 'base.html' %}

{% block content %}
<div class="form-style">
  <h1>Log in</h1>
  <form class="site-form" action="." method="post">
    {% csrf_token %}
    {{ form }}
    {% if request.GET.next %}
      <input type="hidden" name="next" value="{{ request.GET.next }}">
    {% endif %}
    <input class="submit" type="submit" value="Log in">
  </form>
</div>
{% endblock %}
```

Fig:3.22 Login.html

Signup.html:



```
signup - Notepad
File Edit Format View Help
{% extends 'base.html' %}

{% block content %}
<div class="form-style">
    <h1>Sign Up</h1>
    <form class="site-form" action="." method="post">
        {% csrf_token %}
        {{ form }}
        <br><br>
        <input class="submit" type="submit" value="Sign Up">

    </form>
</div>
<script type="text/javascript">
    document.getElementsByTagName("span")[0].innerHTML="";
    document.getElementsByTagName("span")[1].innerHTML="";
</script>
{% endblock %}
```

Fig:3.23 signup.html

Base.html:

```
base - Notepad
File Edit Format View Help
font-weight: bold;
}
</style>
<title>Homepage</title>
</head>
<div style="background-color:#404040;color:#feda6a;padding:10 10 1 10;border: 1px #feda6a groove;margin-bottom:0;">

<h1 align=center>Audio To Sign Language Tool</h1>
</div>
<br>
<body>
<ul id="nav">
  <li class="li"><a class="active" href="{% url 'home' %}">Home</a></li>
  <li class="li"><a href="{% url 'animation' %}">Convertor</a></li>
  {% if not user.is_authenticated %}
  <li class="li"><a href="{% url 'signup' %}">Sign Up</a></li>
  {% endif %}
  {% if user.is_authenticated %}
  <li class="li"><a href="{% url 'logout' %}">Log-Out</a></li>
  {% else %}
  <li class="li"><a href="{% url 'login' %}">Log-in</a></li>
  {% endif %}
  <li class="li"><a href="{% url 'contact' %}">Contact</a></li>
  <li class="li"><a href="{% url 'about' %}">About</a></li>
</ul>

<div class="wrapper" >
  {% block content %}
  {% endblock %}
</div>

</body>
</html>
```

Fig:3.24 base.html

Main.py :

```
main.py - C:\Users\D.Kalyani\AUDIO-SPEECH-TO-SIGN-LANGUAGE-CONVERTER\main.py (3....
File Edit Format Run Options Window Help
#!/usr/bin/env python
"""Django's command-line utility for administrative tasks."""
import os
import sys

def main():
    os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'A2SL.settings')
    try:
        from django.core.management import execute_from_command_line
    except ImportError as exc:
        raise ImportError(
            "Couldn't import Django. Are you sure it's installed and "
            "available on your PYTHONPATH environment variable? Did you "
            "forget to activate a virtual environment?"
        ) from exc
    execute_from_command_line(sys.argv)

if __name__ == '__main__':
    main()
```

Fig:3.25 Main.py

Chapter 4

Results and Discussions

4.1 Sample Input Data

The equivalent sign language representation of a given English text is used to generate output. The system's output would be a clip of ISL vocabulary. Each and every distinct word will have a video in the predefined database, and the output video will be a fused video of those words.

The machine receives audio as feedback. That audio input is recorded by the machine. The audio is then converted to text and displayed on the screen by the machine. The NLP system breaks down the text into components, determines the context of the conversation and the person's purpose, and then chooses which command to execute based on the NLP results. Actually, NLP is the process of developing an algorithm that converts text into words and labels them based on their position and function in sentences.

When you speak “hello” and “Thank You” as input into the microphone, the following output pops up. It is not easy for a machine to learn our language, but it is possible with the assistance of NLP.

Here's how it works in practice:



The screenshot shows a web application interface with a navigation bar at the top containing links: Home, Converter, Log-Out, Contact, and About. The main content area has a heading "Enter Text or Use Mic". Below this is a text input field containing the word "hello", a microphone icon, and a "Submit" button. Below the input field, it displays "The text that you entered is: helloo". At the bottom, it shows "Key words in sentence:" followed by a bullet point and the word "helloo".

Fig.4.1: Input

The screenshot shows a web application interface with a green header and a yellow footer. The header contains the text "Enter Text or Use Mic". Below this, there is a text input field containing "Thank You", a microphone icon, and a yellow "Submit" button. Below the input field, the text "The text that you entered is: ThankYou" is displayed. At the bottom, the text "Key words in sentence:" is followed by a bulleted list containing "ThankYou".

Fig.4.2: Input

The screenshot shows a web application interface with a green header and a yellow footer. The header contains the text "Enter Text or Use Mic". Below this, there is a text input field containing "stay home stay safe", a microphone icon, and a yellow "Submit" button. Below the input field, the text "The text that you entered is: stay home stay safe" is displayed.

Fig.4.3: Input

The screenshot shows a web application interface with a green header and a yellow footer. The header contains the text "Enter Text or Use Mic". Below this, there is a text input field containing "buy again", a microphone icon, and a yellow "Submit" button. Below the input field, the text "The text that you entered is: buy again" is displayed. At the bottom, the text "Key words in sentence:" is followed by a bulleted list containing "b", "u", "y", and "again".

Fig.4.4 : Input

4.2 OUTPUT



Fig.4.8:Stay in sign language

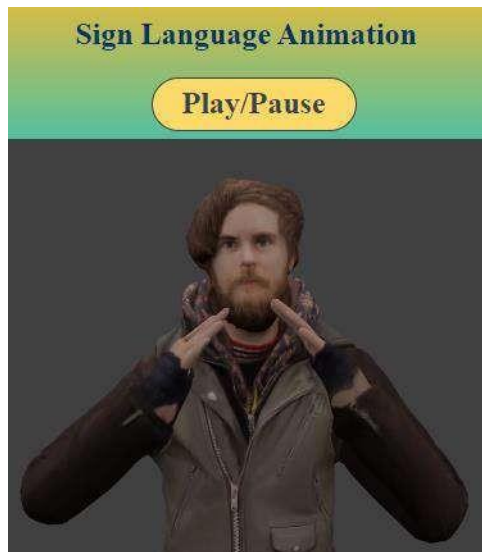


Fig.4.9:Home in sign language



Fig.4.10:Safe in sign language



Fig.4.11:b in sign language

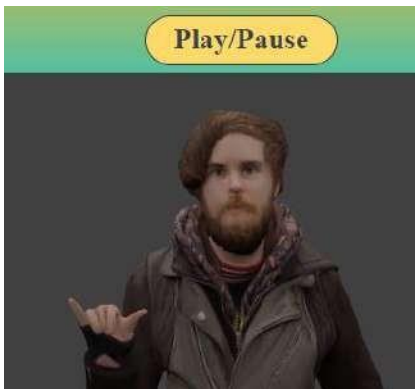


Fig.4.13:y in sign language



Fig.4.12:u in sign language

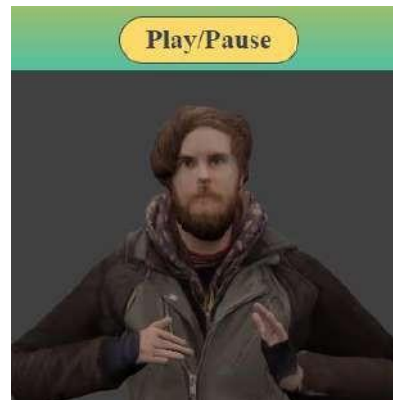


Fig.4.14: again in sign language

4.2 Test Cases:

Table4.1 Test cases for Audio/Text to sign language

Below mentioned are the test cases for developed website

S.NO	TEST CASE DESCRIPTION	INPUT	EXPECTED OUTPUT	ACTUAL OUTPUT	TEST RESULT
1.	Word present in database	Clicking on submit	Video should Represent the entered word	Video representing the word	Success
2.	Word not present in database	Clicking on submit	Split the input and all the letters combine to form a word and video of those merged letters is shown as output	break the words into letters and fetch the videos of that letters and combine them into a single video and show the video to the user as output.	Success
3.	Sentence not present in database	Clicking on submit	Split the input and all the found words combine to form a sentence and video of those merged words is shown as output	break the sentence into words and fetch the videos of that words and combine them into a single video and show the video to the user as output.	Success

Chapter 5

Conclusion and Future Scope

Conclusion:

A well designed and highly developed website is created as a part of this project. We conclude that a tool to convert Audio/Text to sign language for hearing impaired people developed without any faults. A sign language translator comes in handy in a variety of situations. Anyone should use this system to learn and communicate in sign language in schools, colleges, hospitals, universities, airports, and courts. It facilitates contact between those who have normal hearing and those who have difficulty hearing.

Future Scope:

The future work is to modify the Website UI which can be improved and new functionalities can be added. Various front-end options are available such as .net or android app, that can be used to make the system cross platform and increase the availability of the system. Although it is well recognized that facial expressions communicate an essential part of sign language, this project did not focus on them. We are excited to continue the project by incorporating facial expressions into the system.

References

- [1] S. Shrenika and M. Madhu Bala, "Sign Language Recognition Using Template Matching Technique," 2020 International Conference on Computer Science, Engineering and Applications (ICCSEA), Gunupur, India, 2020, pp. 1-5, doi: 10.1109/ICCSEA49143.2020.9132899.
- [2] Ankita Harkude#1, Sarika Namade#2, Shefali Patil#3, Anita Morey #4, Department of Information Technology, Usha Mittal Institute of Technology, Audio to Sign Language Translation for Deaf People, (IJEIT) Volume 9, Issue 10, April 2020
- [3] M. Sanzidul Islam, S. Sultana Sharmin Mousumi, N. A. Jessan, A. Shahariar Azad Rabby and S. Akhter Hossain, "Ishara-Lipi: The First Complete MultipurposeOpen Access Dataset of Isolated Characters for Bangla Sign Language," 2018 International Conference on Bangla Speech and Language Processing (ICBSLP), Sylhet, Bangladesh, 2018, pp. 1-4, doi: 10.1109/ICBSLP.2018.8554466.
- [4] S. Tornay, M. Razavi and M. Magimai.-Doss, "Towards Multilingual Sign Language Recognition," ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Barcelona, Spain, 2020, pp. 6309-6313, doi: 10.1109/ICASSP40776.2020.9054631.
- [5] M. Xie and X. Ma, "End-to-End Residual Neural Network with Data Augmentation for Sign Language Recognition," 2019 IEEE 4th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC), Chengdu, China, 2019, pp. 1629-1633, doi: 10.1109/IAEAC47372.2019.8998073.
- [6] S. He, "Research of a Sign Language Translation System Based on Deep Learning," 2019 International Conference on Artificial Intelligence and Advanced

Manufacturing (AIAM), Dublin, Ireland, 2019, pp. 392-396, doi: 10.1109/AIAM48774.2019.00083.

[7] K. Saija, S. Sangeetha and V. Shah, "WordNet Based Sign Language Machine Translation: from English Voice to ISL Gloss," 2019 IEEE 16th India Council International Conference (INDICON), Rajkot, India, 2019, pp. 1-4, doi: 10.1109/INDICON47234.2019.9029074.

[8] H. Muthu Mariappan and V. Gomathi, "Real-Time Recognition of Indian Sign Language," 2019 International Conference on Computational Intelligence in Data Science (ICCIDS), Chennai, India, 2019, pp. 1-6, doi: 10.1109/ICCIDS.2019.8862125.

[9] D. Pahuja and S. Jain, "Recognition of Sign Language Symbols using Templates," 2020 8th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO), Noida, India, 2020, pp. 1157-1160, doi: 10.1109/ICRITO48877.2020.9198001.

[10] F. M. Shipman and C. D. D. Monteiro, "Crawling and Classification Strategies for Generating a Multi-Language Corpus of Sign Language Video," 2019 ACM/IEEE Joint Conference on Digital Libraries (JCDL), Champaign, IL, USA, 2019, pp. 97-106, doi: 10.1109/JCDL.2019.00023.

[11] M. Aktaş, B. Gökberk and L. Akarun, "Recognizing Non-Manual Signs in Turkish Sign Language," 2019 Ninth International Conference on Image Processing Theory, Tools and Applications (IPTA), Istanbul, Turkey, 2019, pp. 1-6, doi: 10.1109/IPTA.2019.8936081.

[12] M. Yeasin et al., "Design and Implementation of Bangla Sign Language Translator," 2019 5th International Conference on Advances in Electrical Engineering (ICAEE), Dhaka, Bangladesh, 2019, pp. 815-820, doi: 10.1109/ICAEE48663.2019.8975458.

List of Publications

I. JOURNALS

1. A tool to convert audio/text to sign language using python libraries D Kalyani ,dr m madhubala ,n v krishna rao ,dr.y mohana roopa ,m pranay , ch krishnam raju Turkish Journal of Physiotherapy and Rehabilitation; 32(2) ISSN 2651-4451 | e-ISSN 2651-446X ,
(status : updated in scopus)
2. An Improved Lung Cancer Prediction System using Image Processing D Kalyani, C Raghavendra, K Rajendra Prasad International Journal of Recent Technology and Engineering (IJRTE) ISSN: 2277-3878, Volume-8, Issue-4, November 2019 <http://ijrte.org/download/volume-8-issue-4/>