

INFORMATICS INSTITUTE OF TECHNOLOGY

In Collaboration with

UNIVERSITY OF WESTMINSTER

**AudioSangraha - An Approach Transforming Sinhala Audio into Summaries**

A Product Specification, Design and Prototype by

Mr. Aqeel Shafy

Supervised by

Mr. Ragu Sivaraman

Submitted in partial fulfilment of the requirements for the BSc Computer Science degree at the University of Westminster.

**February 2024**

**ABSRTACT**

Language is a unique form of communication between human beans with their environment. In that the most natural way of communicating with others is through the voice. For this nowadays there are many speech technologies are available for range of tasks. But still there is a prominent research area for speech recognition on low resource language. Sinhala language is also one of the low resource languages, as there aren’t enough resources available on the internet. Nowadays, people are not intrusive of listening to lengthy audio contents. Even if they listen to lengthy audios, as a result they skip and try to get the information. Due to this they might get a wrong picture of information.

So, as a solution the author has proposed a system for summarizing the lengthy Sinhala audio contents. Due to this people can save their valuable time while getting the correct information through the audio easily. This system takes an audio file/ user can record an audio file to the system and get the generated summary output.

Using Whisper AI author has trained a new model for Audio to Text process. Throughout author will be correcting the grammar and spelling errors to get a high accuracy. Then the author using a model for extractive summarization and generates the summary accordingly.

**Keywords** -Natural Language Processing, Speech Recognition, Extractive Summarization, Audio Summarization

**Subject Descriptors**:

Computing methodologies → Artificial Intelligence → Natural Language Processing → Speech Recognition

Computing methodologies → Artificial Intelligence → Natural Language Processing → Text Summarization

**TABLE OF CONTENTS**

[CHAPTER 01: INTRODUCTION 1](#_Toc158232813)

[1.1 Chapter Overview 1](#_Toc158232814)

[1.2 Problem Background / Problem Domain 1](#_Toc158232815)

[1.2.1 Natural Language Processing 1](#_Toc158232816)

[1.2.2 Low resource Sinhala language 1](#_Toc158232817)

[1.2.3 Speech to Text 2](#_Toc158232818)

[1.2.4 Text Summarization 2](#_Toc158232819)

[1.3 Problem Definition 2](#_Toc158232820)

[1.3.1. Problem Statement 3](#_Toc158232821)

[1.4 Aims and Objectives 3](#_Toc158232822)

[1.4.1 Research Aim 3](#_Toc158232823)

[1.4.2 Research Objectives 3](#_Toc158232824)

[1.6 Research Gap 6](#_Toc158232825)

[1.7 Contribution to the Body of Knowledge 7](#_Toc158232826)

[1.7.1 Contribution to the problem domain 7](#_Toc158232827)

[1.7.2 Contribution to the research domain 7](#_Toc158232828)

[1.8 Research Challenge 7](#_Toc158232829)

[1.9 Chapter Summary 7](#_Toc158232830)

[CHAPTER 02: SOFTWARE REQUIREMENT SPECIFICATION 8](#_Toc158232831)

[2.1 Chapter Overview 8](#_Toc158232832)

[2.2 Rich Picture Diagram 8](#_Toc158232833)

[2.3 Stakeholder Analysis 9](#_Toc158232834)

[2.3.1 Stakeholder Onion Model 9](#_Toc158232835)

[2.3.2 Stakeholder Viewpoints 9](#_Toc158232836)

[2.4 Selection of Requirement Elicitation Methods 10](#_Toc158232837)

[2.5 Discussion of Findings 11](#_Toc158232838)

[2.5.1 Findings from Literature Review 11](#_Toc158232839)

[2.5.2 Findings from Survey 12](#_Toc158232840)

[2.5.3 Findings from Interview 16](#_Toc158232841)

[2.5.4 Summary of Findings 17](#_Toc158232842)

[2.6 Context Diagram 18](#_Toc158232843)

[2.8 Use Case Specification 19](#_Toc158232844)

[2.9 Requirements with prioritization 20](#_Toc158232845)

[2.9.1 Functional Requirement 21](#_Toc158232846)

[2.9.2 Non-Functional Requirement 21](#_Toc158232847)

[2.10 Chapter Summary 22](#_Toc158232848)

[CHAPTER 03: DESIGN 23](#_Toc158232849)

[3.1 Chapter Overview 23](#_Toc158232850)

[3.2 Design Goals 23](#_Toc158232851)

[3.3 High level Design 24](#_Toc158232852)

[3.3.1 Architecture Diagram 24](#_Toc158232853)

[3.3.2 Discussion of tiers/ layers of the Architecture 25](#_Toc158232854)

[3.4 System Design 25](#_Toc158232855)

[3.4.1 Choice of design paradigm 25](#_Toc158232856)

[3.5 Detailed Design Diagrams 26](#_Toc158232857)

[3.5.1 Data Flow Diagram 26](#_Toc158232858)

[3.5.2 System Process Flowchart 27](#_Toc158232859)

[3.5.3 User Interface Design 28](#_Toc158232860)

[3.6 Chapter summary 28](#_Toc158232861)

[CHAPTER 04: IMPLEMENTATION 29](#_Toc158232862)

[4.1 Chapter Overview 29](#_Toc158232863)

[4.2 Technology Selection 29](#_Toc158232864)

[4.2.1 Technology Stack 29](#_Toc158232865)

[4.2.2 Dataset Selection 30](#_Toc158232866)

[4.2.3 Development Frameworks 30](#_Toc158232867)

[4.2.4 Programming Languages 31](#_Toc158232868)

[4.2.5 Libraries 31](#_Toc158232869)

[4.2.6 IDE 31](#_Toc158232870)

[4.2.7 Summary of Technology Selection 32](#_Toc158232871)

[4.3 Implementation of the Core Functionality 32](#_Toc158232872)

[4.3.1 Audio to Text model implementation 33](#_Toc158232873)

[4.4 User Interface 34](#_Toc158232874)

[4.5 Chapter Summary 34](#_Toc158232875)

[CHAPTER 05: CONCLUSION 35](#_Toc158232876)

[5.1 Chapter Overview 35](#_Toc158232877)

[5.2 Deviations 35](#_Toc158232878)

[5.2.1 Scope related deviations 35](#_Toc158232879)

[5.2.2 Schedule related deviations 35](#_Toc158232880)

[5.3 Initial Test Results 35](#_Toc158232881)

[5.4 Required Improvements 35](#_Toc158232882)

[5.5 Demo of the Prototype 36](#_Toc158232883)

[5.6 Chapter Summary 36](#_Toc158232884)

[REFERENCES 37](#_Toc158232885)

[APPENDIX 41](#_Toc158232886)

**LIST OF FIGURES**

[Figure 1: Rich Picture Diagram 8](#_Toc158225533)

[Figure 2: Stakeholder Onion Model 9](#_Toc158225534)

[Figure 3: Context Diagram 18](#_Toc158225535)

[Figure 4: Use Case Diagram 18](#_Toc158225536)

[Figure 5: High Level Design 24](#_Toc158225537)

[Figure 6: Data Flow Diagram 1 26](#_Toc158225538)

[Figure 7: Data Flow Diagram 2 26](#_Toc158225539)

[Figure 8: Flowchart 27](#_Toc158225540)

[Figure 9: Split into Training and Testing Sets 33](#_Toc158225541)

[Figure 10: Read the Audio Files 33](#_Toc158225542)

[Figure 11: Pre-trained Whisper Model 33](#_Toc158225543)

[Figure 12: Setting Training Arguments 34](#_Toc158225544)

[Figure 13: Train model 34](#_Toc158225545)

**LIST OF TABLES**

[Table 1: Research Objectives 6](#_Toc158222922)

[Table 2: Stakeholder Analysis 10](#_Toc158222923)

[Table 3: Requirement Elicitation Methods 11](#_Toc158222924)

[Table 4: Literature Review Findings 11](#_Toc158222925)

[Table 5: Survey Findings 15](#_Toc158222926)

[Table 6: Interview Findings 17](#_Toc158222927)

[Table 7: Summary of Findings 17](#_Toc158222928)

[Table 8: UC1 Description 19](#_Toc158222929)

[Table 9: UC2 Description 20](#_Toc158222930)

[Table 10: MoSCoW Principal 20](#_Toc158222931)

[Table 11: Functional Requirements 21](#_Toc158222932)

[Table 12: Non-functional Requirements 22](#_Toc158222933)

[Table 13: Design Goals 24](#_Toc158222934)

[Table 14: Technology Stack 30](#_Toc158222935)

[Table 15: Development Frameworks 30](#_Toc158222936)

[Table 16: Libraries used for The System Implementation 31](#_Toc158222937)

[Table 17: IDEs used for The System Implementation 32](#_Toc158222938)

[Table 18: Summary of Technology Selection 32](#_Toc158222939)

[Table 19: Required Improvements 36](#_Toc158222940)

**LIST OF ABREVIATIONS**

|  |  |
| --- | --- |
| NLP | Natural Language Processing |
| STT | Speech to Text |
| LRL | Low Resource Language |
| BERT | Bidirectional Encoder Representations from Transformers |
| E2E | End to End |
| GPU | Graphics Processing Unit |
| OS | Operational System |
| NLTK | Natural Language Toolkit |
| TDNN | Time Delay Neural Network |
| SSADM | Structured System Analysis and Design Method |
| OOAD | Object Oriented Analysis and Design |
| UI | User Interface |
| LSTM | Long Short-Term Memory |
| ASR | Active Speech Recognition |
| WER | Word Error Rate |
| DNN | Deep Neural Network |
| HMM | Hidden Markov Model |
| GUI | Graphical User Interface |

# CHAPTER 01: INTRODUCTION

# 1.1 Chapter Overview

The Sinhala language holds an important position as a language. Mostly Sinhala language is used in Sri Lanka, and it is the prominent language. but it is also used within the Sinhala speaking communities across the globe. There is a rich cultural and historical significance in the language. But it is considered as a low resource language as there is a lack of dataset available on the internet. Speech and text are the most valuable things to communicate with others. So, in this research the author has suggested a solution for the Sinhala language users on an audio summarization system. Moreover, the author will be describing the problem background, research gap, contribution to the project (both problem domain and the research domain), aim of the project, challenges, motivation and comparing the existing systems with the limitations in detail.

## 1.2 Problem Background

### 1.2.1 Natural Language Processing

Natural Language Processing is a language that interacts with the computer and human language (Millstein, 2020). And it focuses on how computers are programmed to analyze large amounts of data. Around us there are billions of text data generated. Social media, WhatsApp, Facebook, Instagram etc and other blogs, news channels, google platforms etc are beneficial from NLP. And it can be used for text understanding, speech recognition, analytics tasks, such as classifying documents and analyzing sentiment text, as well as more advanced tasks, such as answering questions, translating documents and summarizing reports (Gruetzemacher, 2022). And these benefits are that the computer works like humans do.

### 1.2.2 Low resource Sinhala language

Language is the unique form of communication between humans with their environment. There are more than 7000 languages used in the world. Sinhala is one of them. The Sinhala language belongs to the Indo-Aryan language category in the small island of Sri Lanka. And there are two types of grammar in the language, one is written and other is spoken. And there are 25 types of structures in the language (de Silva, 2019). The median age of the population in Sri Lanka is 34.5 and there are more than 21 billion people living in Sri Lanka. They use Sinhala as their primary language in the country. And 52.6% of people use the internet (Kemp, 2022). But still the Sinhala language is considered as a low resource language, because there aren't enough resources available on the internet (Deshpande & Jahirabadkar, 2021).

### 1.2.3 Speech to Text

The most natural and friendly communication through human beings is the voice to interact with others (Khandare et al., 2019). Nowadays speech technologies are commonly available for a range of tasks. Speech to text is the process to convert the speech to written text. These advanced technologies empower the machines to effectively respond to a human voice. The use of human voice with the machines proves that it's faster than the traditional keyboard input (Das & Prasad, 2015). And it will be an advantage for those who are frustrated using machines using the keyboard. Even these days the speech recognition has a prominent research area for the low resource language (Weerasinghe et al., 2020). Unfortunately, the Sinhala speech to text has been carried out, and fewer have been successful due to a lack of economic interest (Kasthuri Arachchige & Weerasinghe, 2023).

### 1.2.4 Text Summarization

Text summarization has the ability to generate long text documents into shorter and accurate summaries (Singh, 2020). Single and multi-document are the two types of summarization input types. And there are two types of output it generates. Abstractive summarization and extractive summarization are the two types (Singh, 2020). Based on the input of the document the extractive summarization collects the important sentence and forms a summary. While abstractive summarization forms its own sentences to generate the summary like the human do. Generic, domain specific and query-based types are based on the purpose of the text summarization. Due to the large number of internet users in recent days, the text summarization race is high (Prudhvi et al., 2020). But it's challenging when it comes to a low resource language like Sinhala, as there is a lack of resources available (Deshpande & Jahirabadkar, 2021).

## 1.3 Problem Definition

Speech recognition has the ability to convert human voice audio to machine readable format. Audio is considered as the most effective mode of communication between human-being (Khandare et al., 2019). So, it is the easier way to get or transfer the information between others. While audio is converted to text there might be grammatical and spelling mistakes. As the Sinhala letters, spellings and the pronunciation of the words are complex. So, more mistakes may occur when the speech is recognized. And when its summarized text too may have been mistaken.

Nowadays with the technological improvement, listening to hours of lengthy audios like broadcasting of news on radio channels, speeches, lecture audios etc are time consuming. As some of them are not intrusive to listening to lengthy audios.

### 1.3.1. Problem Statement

The Sinhala language users are unable to summarize the Sinhala lengthy audio files like broadcasting news on radio, speeches, interviews, audio books and lecture audios due to a lack of resources available.

## 1.4 Aims and Objectives

### 1.4.1 Research Aim

*The aim of the research is to design, develop and evaluate* a summarization system for the low resource of Sinhala language audio data using natural language processing.

As further elaboration on the aim the author will create a system for summarizing the lengthy audios with the extractive summarization. And before the audio is summarized the author will improve the accuracy of the spelling and the grammar of Sinhala audio. And finally, the user will be able to listen to a summarized short audio unless listening to a lengthy audio.

### 1.4.2 Research Objectives

|  |  |  |  |
| --- | --- | --- | --- |
| **Research Objectives** | **Description** | **Learning Outcomes** | **Research Questions** |
| Research Problem | RO1 - To explore the challenges in Low resourced languages  RO2 - To identify the specific user needs when it comes to summarize  RO3 - To identify the research gap for summarizers for low resource languages. | LO1, LO3, LO6 | RQ3 |
| Literature Review | RO4 - To review the existing approaches and limitations on low resource languages  RO5 - To identify the techniques, methodologies, algorithms and models related to the audio summarizer  RO6 - To identify the specific challenges when it comes to Sinhala summarizer  RO7 - To identify what are the datasets available for the research | L01, L04, LO5, LO8 | RQ2, RQ1 |
| Requirement Elicitation | R08 - To gather the requirements and feedbacks by the industrial experts  R09 - To collect the user reviews from the existing systems to improve   RO10 - To identify the requirements related to build the system | LO1, LO3, LO5, LO6 | RQ2, RQ3 |
| Design | RO11 - To create a user-friendly interface   RO12 - To create the required data flows, diagrams to design the architecture | LO2, LO5, LO7 | RQ2 |
| Implementation | RO13 - To develop the model to convert the audio to text  RO14 - To develop the text summarization model  RO15 - To manage the data storage systems   RO16 - To develop the User Interface | LO2, LO4, LO5, LO7, LO8 | RQ1, RQ2, RQ3 |
| Testing and Evaluation | RO17 - To provide a high performance and accuracy in the system  RO18 - To create the test plan related to the system  RO19 - To perform the testing of unit testing, functional testing and usability testing | LO1, LO5, LO7, | RQ2, RQ3 |
| Documentation | RO20 - To organize a structured document related to the research | LO8 | RQ2, RQ3 |

Table 1: Research Objectives

## 1.6 Research Gap

Most of the research has contributed to the automatic summarization system, and Speak recognition system for the English language. And there is a noticeable research gap when it comes to the low resource languages. When compared to the existing work the author has elaborated to implement a system for the target audience of Sinhala language.

* Summarizing the low resource language of Sinhala audio files with extractive summarization (Warnasooriya et al., 2020)
* Correcting the Sinhala language grammar, spelling of speech when it converts in to audio to text (Weerasinghe et al., 2020)
* Improving the accuracy of the poor audio quality data (Weerasinghe et al., 2020)

## 1.7 Contribution to the Body of Knowledge

### 1.7.1 Contribution to the problem domain

Natural Language Process has the ability to give high impact for low resourced Sinhala language audio summarization when it comes to the contribution to the problem domain. These days NLP has been a highly contributed area between the researches, when it comes to the low resourced languages (Gruetzemacher, 2022). And Sinhala audio data summarization system will be a great deal between the Sinhala language users and Sinhala-communities utilizing people’s valuable time listening to lengthy audios, as it will be summarized.

### 1.7.2 Contribution to the research domain

After considering the existing works the author concluded by implementing a system to summarize the Sinhala audio data with Extractive summarization. As contributions to the research domain the author will be addressing the accuracy of the spelling and grammatical issues in Sinhala Language (Weerasinghe et al., 2020) as it is known as a low resourced language, because there aren't enough data resources available on the internet (Deshpande & Jahirabadkar, 2021).

## 1.8 Research Challenge

In this research there are three main tasks. They are generating the audio to text, correcting the Sinhala spelling and the grammar of the text of the audio when it converts into text, summarizing the text with extractive summarization. It is challenging when it comes to low resource languages that have a lack of resources available on the internet. So, there might be some challenges when it comes to finding data, techniques and tools to improve the accuracy of the summarizer.

## 1.9 Chapter Summary

As a summary of this chapter the author has discussed the problem domain, challenges, research aim and limitations of the existing works. And finally, after considering the existing works the author has come up with the research gap and the contribution for the project.

# CHAPTER 03: SOFTWARE REQUIREMENT SPECIFICATION

## 2.1 Chapter Overview

In this chapter it provides a rich picture diagram and an onion model identifying the stakeholders of the system. And the author will be exploring the requirement elicitation including literature review, surveys and interviews. Moreover, it will discuss the use case diagram, functional and nonfunctional requirements of the system.

## 2.2 Rich Picture Diagram

The given rich picture diagram below provides a helicopter view of the wider environment of the system. And it clearly states the stakeholders interacting with the system and others. It also highlights the negative and positive aspects of the system.

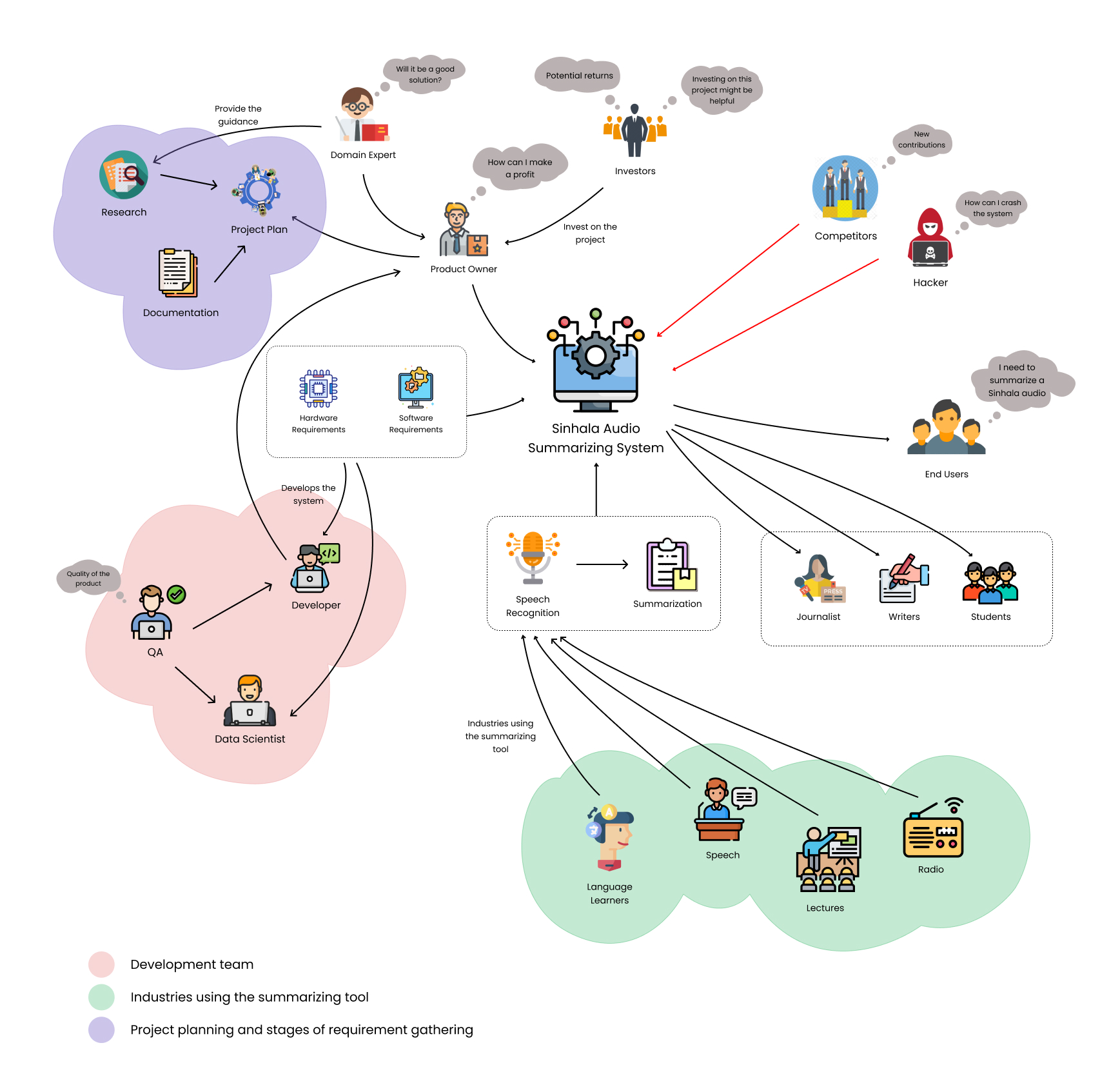


Figure 1: Rich Picture Diagram

## 2.3 Stakeholder Analysis

### 2.3.1 Stakeholder Onion Model

The stakeholder onion model below provides each stakeholder in the system which is in different environments. This helps the author to identify the stakeholders with positive and negative structure and an organizing part of the project.

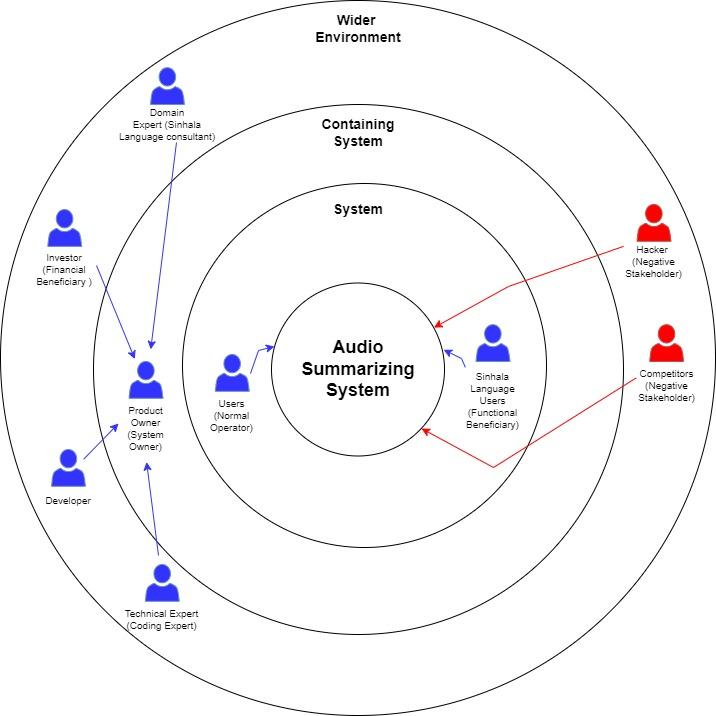


Figure 2: Stakeholder Onion Model

### 2.3.2 Stakeholder Viewpoints

|  |  |  |
| --- | --- | --- |
| **Stakeholder** | **Stakeholder type** | **Description** |
| Users | Normal operator | Who will be using the system to summarize Sinhala audio files |
| Sinhala language users | Functional beneficiary | They are the ones who will be benefited from the system |
| Product owner | System owner | The product owner is who will be handling the system |
| Technical expert | Consultant/ Coding expert | Who will be providing/guiding on the coding requirements |
| Developer | Operational Maintainer | Develops the system using the gathered requirements |
| Investor | Financial Beneficiary | Who will be financially investing on the project and improve the system to get |
| Domain expert | Consultant | Will guide on the project with necessary requirements |
| Competitors | Negative Stakeholder | Will be implementing similar systems |
| Hackers | Negative Stakeholder | Tries to crash the system |

Table 2: Stakeholder Analysis

## 2.4 Selection of Requirement Elicitation Methods

Requirement elicitation is the process to gather requirements from stakeholders what are the expectations. There are several methods to carry out to gather the requirements. Here the author has selected the literature review, survey and interviews to gather the requirements.

|  |
| --- |
| **Literature review** |
| LR was selected, as it has the ability to identify the research gap of the existing work and make a contribution to the field. Using the gathered requirements (techniques used), it helps the author to improve the system with a better result. |
| **Survey** |
| Distributing surveys or questionnaires will help the author to understand the user needs, experience of the existing systems and what should be improved. This will be a suitable method to gather requirements on a larger number of populations. |
| **Interviews** |
| Interviews will help to gather requirements in detail. The author focuses on having interviews with the domain and technical experts. This will help the author in gathering the requirements to fulfill the system on the technical and domain wise, identifying and clarifying the specific needs to the project. |

Table 3: Requirement Elicitation Methods

## 2.5 Discussion of Findings

### 2.5.1 Findings from Literature Review

|  |  |
| --- | --- |
| **Findings** | **Citation** |
| The HMM performs a far better accuracy than the other traditional approaches. And to get a high accuracy the dataset should be with more vocabulary | (Weerasinghe et al., 2020) |
| For a better summarization result semantic features can be used. | (Shah et al., 2019) |
| Compared to TDNN+LSTM and DNNs, TDNN+LSTM shows a lower WER. But still in speech recognition tasks TDNNs perform much better. | (Karunathilaka, 2020) |
| The ASR system provides a lower accuracy in sentence recognition compared to IVR | (Dinushika et al., 2020) |
| Figuring out the relationship between the words will give an accurate summary. | (Jing et al., 2021) |

Table 4: Literature Review Findings

### 2.5.2 Findings from Survey

A questionnaire was shared publicly, as it is hard to get the response by the target audience for the particular application. The author was able to collect 133+ responses. The responses and the aim of the questions are stated below. In the APPENDIX the survey can be found.

|  |  |
| --- | --- |
| **Question** | **Are you a person who listens to Sinhala audio content?** |
| **Aim** | To find out if the participant is a person who listens to Sinhala audio contents. |
| **Findings** | With the above result, it can be said there is a majority of users who listen to Sinhala audio contents. So there is the potential of users who are the target ordinance for this system. |
| **Question** | **What type of Sinhala audio do you listen to?** |
| **Aim** | To find out what the Sinhala audio content they used to listen to. |
| **Findings** | It states that there is a huge amount saying that they listen to Sinhala speeches, next podcast and news. So, there is a noticeable number of users who listen to this type of lengthy audios, which means that from the proposed system they could be benefited. |
| **Question** | **When it comes to a lengthy audio, how much will you complete listening?** |
| **Aim** | From the above question it identifies how much they will complete listening lengthy audio contents. |
| **Findings** | It states that a large number of users doesn’t 100% complete listening to the audios. Because of this they might miss the useful information in the audio. |
| **Question** | **What are the challenges you face while listening to long audio contents?** |
| **Aim** | This to find out what are the challenges they encountered while listening to lengthy audio files |
| **Findings** | Most of the participants say that multitasking and listening to an audio file is the hardest, next difficult to retain information, difficult to maintain focus. So, this application will focus on these particular challenges. |
| **Question** | **Would you like to get a summarized text version of your lengthy audio?** |
| **Aim** | To find out the importance of implementing this system |
| **Findings** | There is a considerable number of users who are saying they need a summarized version of audio files. This means this system would benefit a large number of users. |
| **Question** | **Have you use any platforms to summarize a Sinhala lengthy audio file** |
| **Aim** | This is to find out if the user has used any existing systems, if yes how was the experience |
| **Findings** | In the above pie chart, it clearly states that there is a large number of users saying that they haven't used any other existing applications, and there is a considerable amount saying who used the existing applications also does not provide an accurate result. |
| **Question** | **What are the features you would want in this type of application?** |
| **Aim** | This is to find out what are the futures there are expecting through this system |
| **Findings** | There is a huge amount saying that they need a combined approach (Audio recognition and Text summarization). So, it clearly states implementing a combined approach will be benefited through a larger amount. |
| **Question** | **How useful will this application be for you?** |
| **Aim** | To find out the users who will be benefited through this application |
| **Findings** | There is a larger number of users that might be benefited through this application. |

Table 5: Survey Findings

### 2.5.3 Findings from Interview

The interviews were conducted within the domain related and technical experts.

|  |  |  |
| --- | --- | --- |
| **Codes** | **Theme** | **Conclusion** |
| ‘Existing datasets or Audio to Text’  ‘Model implementation’  ‘User-friendly UI’ | Dataset Collection and Speech recognition model | The experts mentioned that to look for publicly available Sinhala ASR datasets. So, through that dataset they said look out of the speakers, the accent and the recording conditions.  And they mentioned that implementing a model for Sinhala speech recognition will be an advantage.  When it comes to the UI, they mentioned to make it simple, so the user can easily summarize the audio based on their input. |
| ‘Existing applications does not have summarization based on audio’ | Research gap and scope of the project | There isn't a summarization system for audio for Sinhala language. So, they mentioned the research gap is valid and will be good to address. Throughout the audio recognition correcting the grammar and the spelling of the sentence will be highly recommended. |
| ‘Background noise of an audio’ | Background noise removal | There are approaches like denoising techniques, spectral subtraction, Wiener filtering they mentioned for filtering out the background noises. |
| ‘Summarizing techniques’ | Text summarizer | They mentioned that there are two ways to summarize a text. Extractive and abstractive. When it comes to this system, they recommended to have an extractive summarization approach for the summarization purpose. |

Table 6: Interview Findings

### 2.5.4 Summary of Findings

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Id** | **Findings** | **Literature Review** | **Survey** | **Interview** |
| 1 | Expresses a need of Sinhala audio summarization system | ✓ | ✓ | ✓ |
| 2 | Implementing a model for audio recognition | ✓ |  | ✓ |
| 3 | The relationship within the words will give an accurate summary |  |  | ✓ |
| 4 | Generate summary with correct grammar and spellings | ✓ |  | ✓ |
| 5 | Identify the suitable dataset |  | ✓ | ✓ |
| 6 | Use pretrained models to get high accuracy | ✓ |  | ✓ |
| 7 | User friendly and simple interface for the system |  | ✓ | ✓ |

Table 7: Summary of Findings

## 2.6 Context Diagram

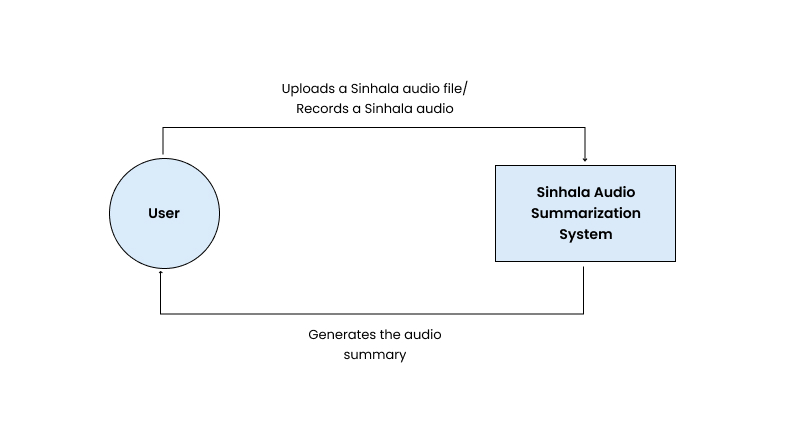
The context diagram provides the system boundaries and the interaction between the users. In the below diagram it shows the user has to upload an audio file or record an audio file to the system. And the system will generate the summary of the audio to the user.   


Figure 3: Context Diagram

**2.7 Use Case Diagram**  
The below use case diagram describes the functionalities of the system, including the actors and other related components.

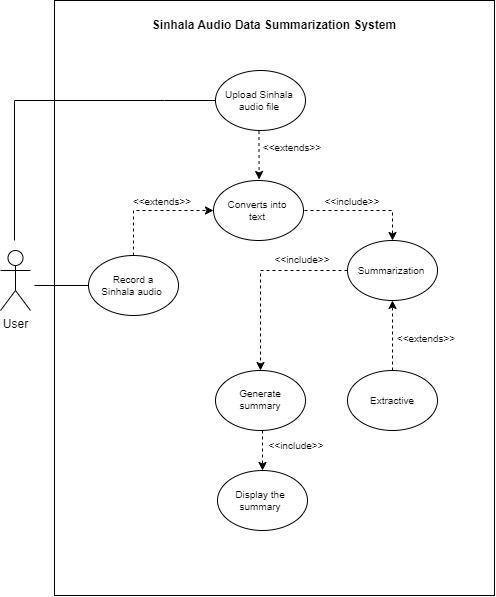


Figure 4: Use Case Diagram

## 2.8 Use Case Specification

|  |  |
| --- | --- |
| Use Case Name | Upload Sinhala audio file |
| Use case ID | UC1 |
| Description | The user needs to upload a Sinhala audio mp3 file |
| Priority | High |
| Actors | User |
| Pre-conditions | None |
| Post-condition | User is able to see the generated text, by given audio |
| Extended use case | Recognize the audio file |
| Included use case | None |
| Main flow | 1. User uploads a Sinhala audio file 2. System recognizes the audio and converts the audio into text with correct spellings and grammar. |
| Alternative flow | None |
| Exceptional flow | File format errors will be displayed |

Table 8: UC1 Description

|  |  |
| --- | --- |
| Use Case Name | Summarization |
| Use case ID | UC2 |
| Description | The user needs to click the summary to get the summarized version of the audio file |
| Priority | High |
| Actors | User |
| Pre-conditions | None |
| Post-condition | User has to get a summary of the uploaded audio file |
| Extended use case | Extractive summarization |
| Included use case | None |
| Main flow | 1. User clicks the summary button 2. System generated the extractive summarization of the uploaded audio file |
| Alternative flow | None |
| Exceptional flow | Error while summarizing, or will generates the same results as the converted text input |

Table 9: UC2 Description

## 2.9 Requirements with prioritization

The MoSCoW principle used to manage priorities of the requirements in the project effectively.

|  |  |
| --- | --- |
| Must have(M) | The feature requirement which are mandatory to be implement the system |
| Should have(S) | Requirements or features which are important, but not necessary for the prototype. |
| Could have(C) | These are nice to have. Can be considered as future works to the system. |
| Will not have(W) | The functionalities are out of the scope, which will not be implemented on the system. |

Table 10: MoSCoW Principal

### 2.9.1 Functional Requirement

|  |  |  |  |
| --- | --- | --- | --- |
| S. No | Functional Requirement | Priority | Use case |
| FR1 | Should be able to upload an audio file to the system/ record an audio at real time | M | Upload a Sinhala audio file/ Record a Sinhala audio file |
| FR2 | The system must not support other than audio file | M | Upload a Sinhala audio file/ Record a Sinhala audio file |
| FR3 | The system should generate the Sinhala text accordingly | M | Converts into text |
| FR4 | The system should correct the spellings and the grammar of the generated text | M | Converts into text |
| FR5 | User should be able to copy the generated text | S | Converts into text |
| FR6 | The generated text should be summarized extractive | M | Summarization |
| FR7 | User should be able to copy the summary | S | Summarization |
| FR8 | The user should be able to upload other language audios | W | Upload a Sinhala audio file/ Record a Sinhala audio file |
| FR9 | The user should be able to upload videos | W | Upload a Sinhala audio file/ Record a Sinhala audio file |
| FR10 | The system generates summary of other languages | W | Summarization |
| FR7  The user should be able to upload other language audios |  |  |  |  |
| FR8  The user should be able to upload videos |  |  |  |  |
| FR9  The system generates summary of other languages |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |

Table 11: Functional Requirements

### 2.9.2 Non-Functional Requirement

|  |  |  |
| --- | --- | --- |
| **S. No** | **Non-Functional Requirement** | **Priority** |
| NFR1 | Usability The system should be user-friendly and easy to operate the system to the user. | M |
| NFR2 | Fluency The generated text should be written without and grammatical errors or any spelling mistakes | M |
| NFR3 | Performance The system should be able to work smoothly | S |
| NFR4 | Security  The system should be protecting the user data while preventing unauthorized access | M |

Table 12: Non-functional Requirements

## 2.10 Chapter Summary

This chapter discussed about the Rich picture diagram, identified stakeholder for the system, the onion model. And it has been discussed what are the findings from the literature review, Survey and conducted interviews. At the end it has stated the context diagram, use case diagrams, functional and non-functional requirements of the system.

# CHAPTER 06: DESIGN

## 6.1 Chapter Overview

This chapter discusses the designs and architectures related to the system. There is the system architecture design, component diagrams, data flow diagrams and user interface designs and flow charts.

## 6.2 Design Goals

|  |  |
| --- | --- |
| **Design goal** | **Description** |
| Performance | As the system take multiple audio inputs, the system should run smoothly without any failure and a delay while the system should provide a high-quality and efficient summarized output. |
| Usability | The UI of the system should be more simple, clean, straight forward and allow the users to easily navigate through the system functionalities to upload audio files and get the summarized output. |
| Scalability | The scalability of the system should be capable of performing with less time to recognize the audio file and generate the text. And the system should be able to upload multiple audios and generate the summaries. |
| Reusability | This project related codes and other relevant components should be able to be reused for another project. |
| Correctness | The system generates the text from recognizing the audio first and generate the text accordingly. And the multiple audio outputs should be combined. Else the grabbed information will be misled. |

Table 13: Design Goals

## 6.3 High level Design

### 6.3.1 Architecture Diagram

The following high-level diagram consists of three tier architecture which has the presentation tier, logic tier and data tier.

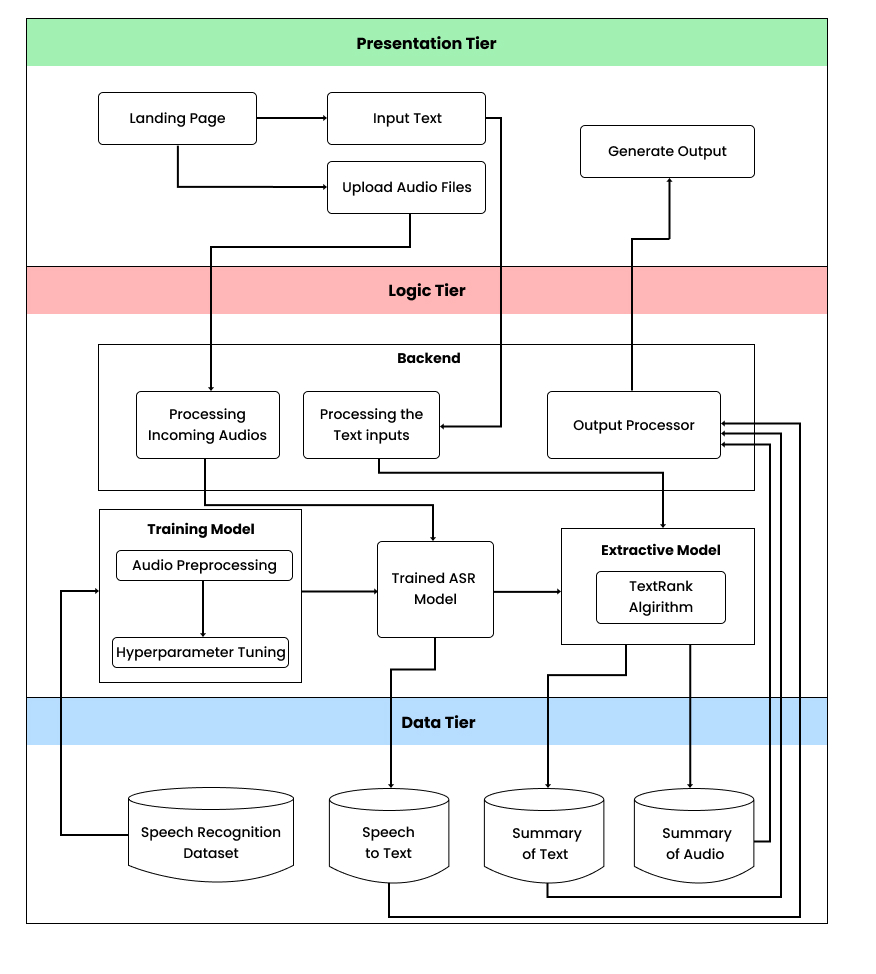


Figure 5: High Level Design

### 6.3.2 Discussion of tiers/ layers of the Architecture

**Data Tier**

* Speech recognition dataset **-** This will be used to convert the input audio files into text format
* Speech to text – This will generate the text from the input audio files to the user
* Summary of text – This will generate the summarized version of text to the user
* Summary of audio – This will generate the summarized version of audio files to the user

**Logic Tier**

* Dataset preprocessing - Before training the model the dataset should be preprocessed. For the audio summarizing, the audio preprocessing is used.
* Model training – The preprocessed data fed to the model, for speech recognition it learns to make predictions and gives effective outcomes
* Processing audio files - This is where the audio is converted into text. One by one audio will be fed to the model and generate the text combining as a paragraph
* Processing text input - This is where the text is getting summarized.
* Extractive Model – Using Text Rank algorithm it generates the summaries according to sentence score
* Output Processor – This will used to get the audio to text, text summarization or the audio summarization output and send back to user

**Presentation Tier**

* Landing page - This provides a user friendly and understandable user interface for the user to navigate through the system functionalities.
* Upload audio files - The system allows the user to upload audio files to the system.
* Input text - This displays the generated summary version of the provided text by the user.
* Generate summary - This displays the generated summary version of the provided audio files or the text input by the user.

## 6.4 System Design

### 6.4.1 Choice of design paradigm

After a clear understanding of the design paradigm, SSADM was chosen by the author over OOAD. SSADM is more suitable for this project, as it is systematic, perfectly structured and easy for prototyping. There are several factors for rejecting OOAD. One of those is that an object-oriented approach doesn't benefit this project, as it is based on a data science component. SSADM has the ability to improve the accuracy, efficiency and documentation of information systems.

## 6.5 Detailed Design Diagrams

### 6.5.1 Data Flow Diagram

The level 01 DFD provides a basic understanding of the system. And the level 02 DFD provides a more detailed version of how the system function elaborates.

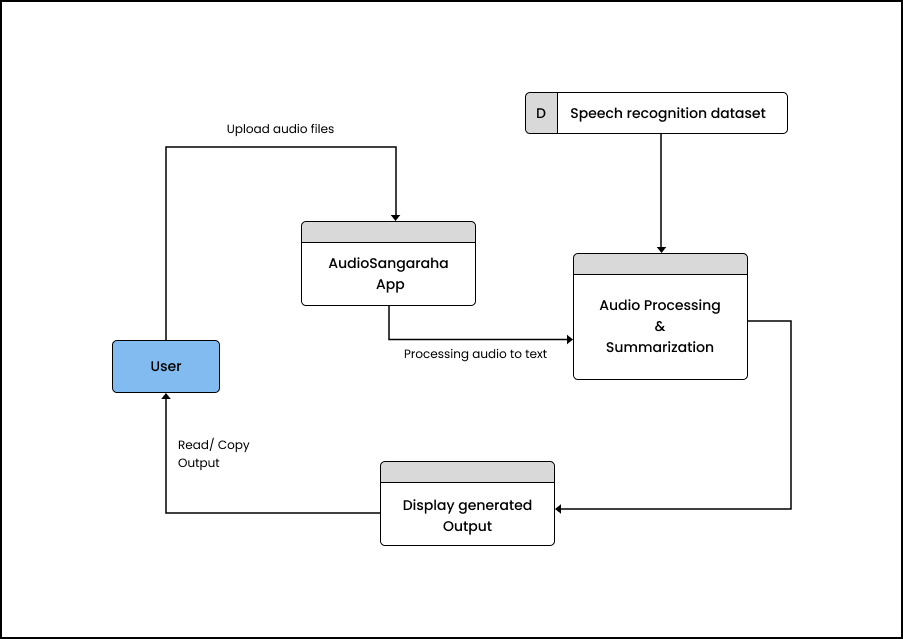


Figure 6: Data Flow Diagram 1

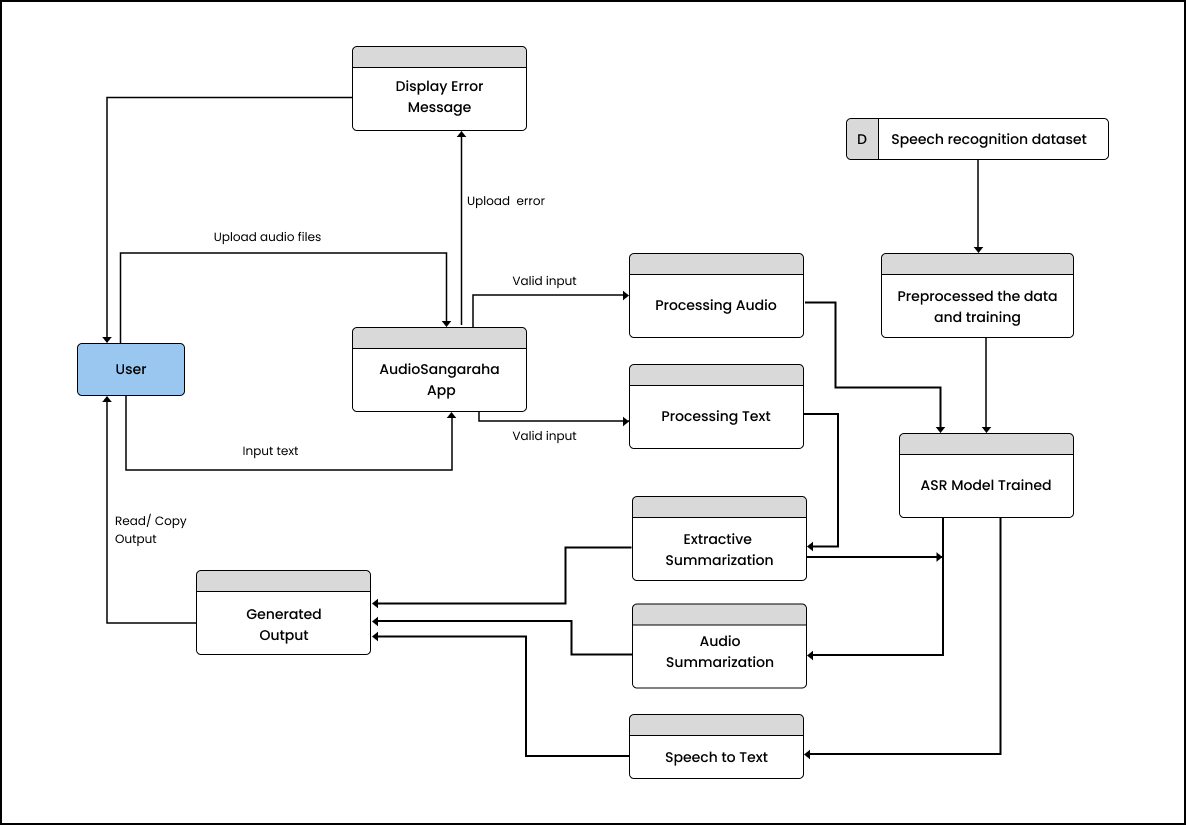


Figure 7: Data Flow Diagram 2

### 6.5.2 System Process Flowchart

The following flow chart describes the key steps involved in the audio data summarization system.

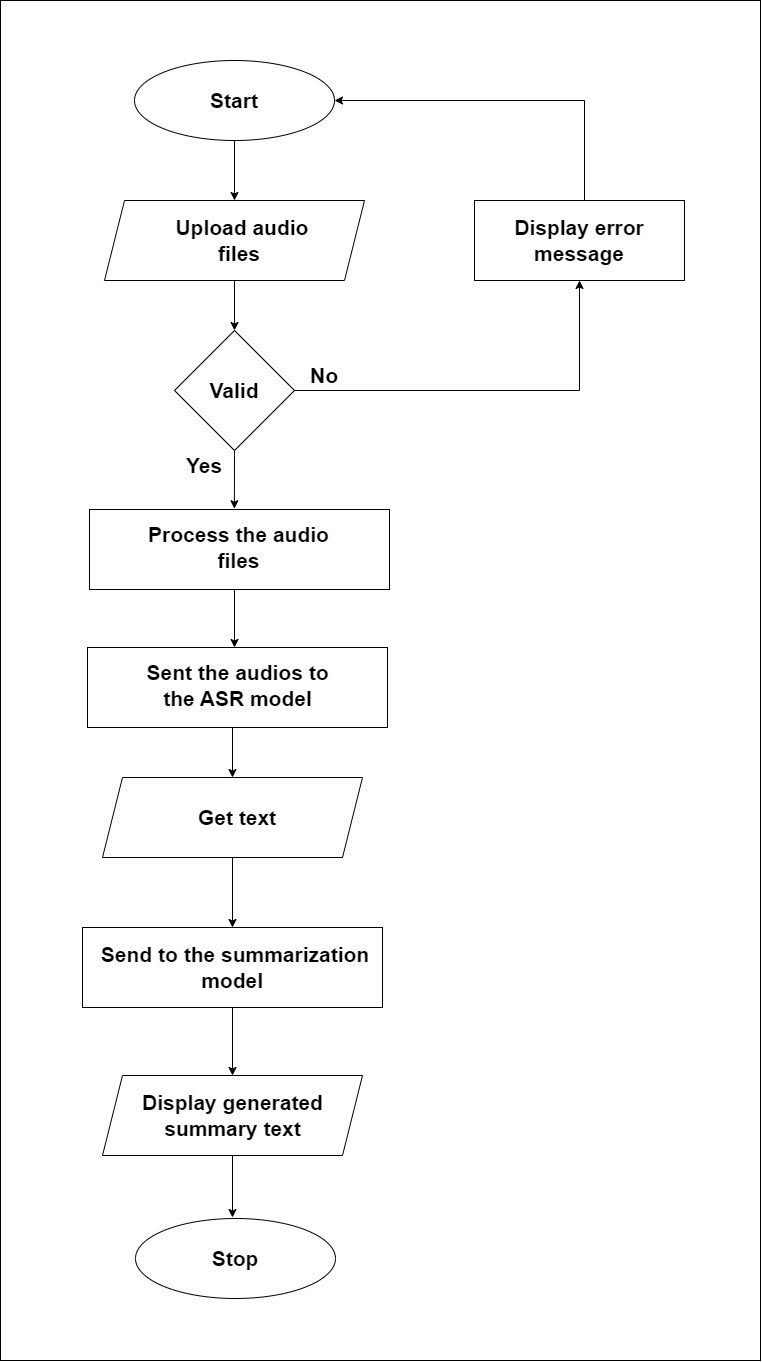


Figure 8: Flowchart

### 6.5.3 User Interface Design

The design of the UI is more important, as it helps the users to easily navigate through the system and understand the functionalities easily. So, for the proposed system the prototype is a web-based application. Also, the system should be responsive for mobile users. The following provides the Wireframe for the proposed audio summarization system. It has a simple and user-friendly interface. Other related UI low-fidelity and the high-fidelity designs of the system are attached in the APPENDIX

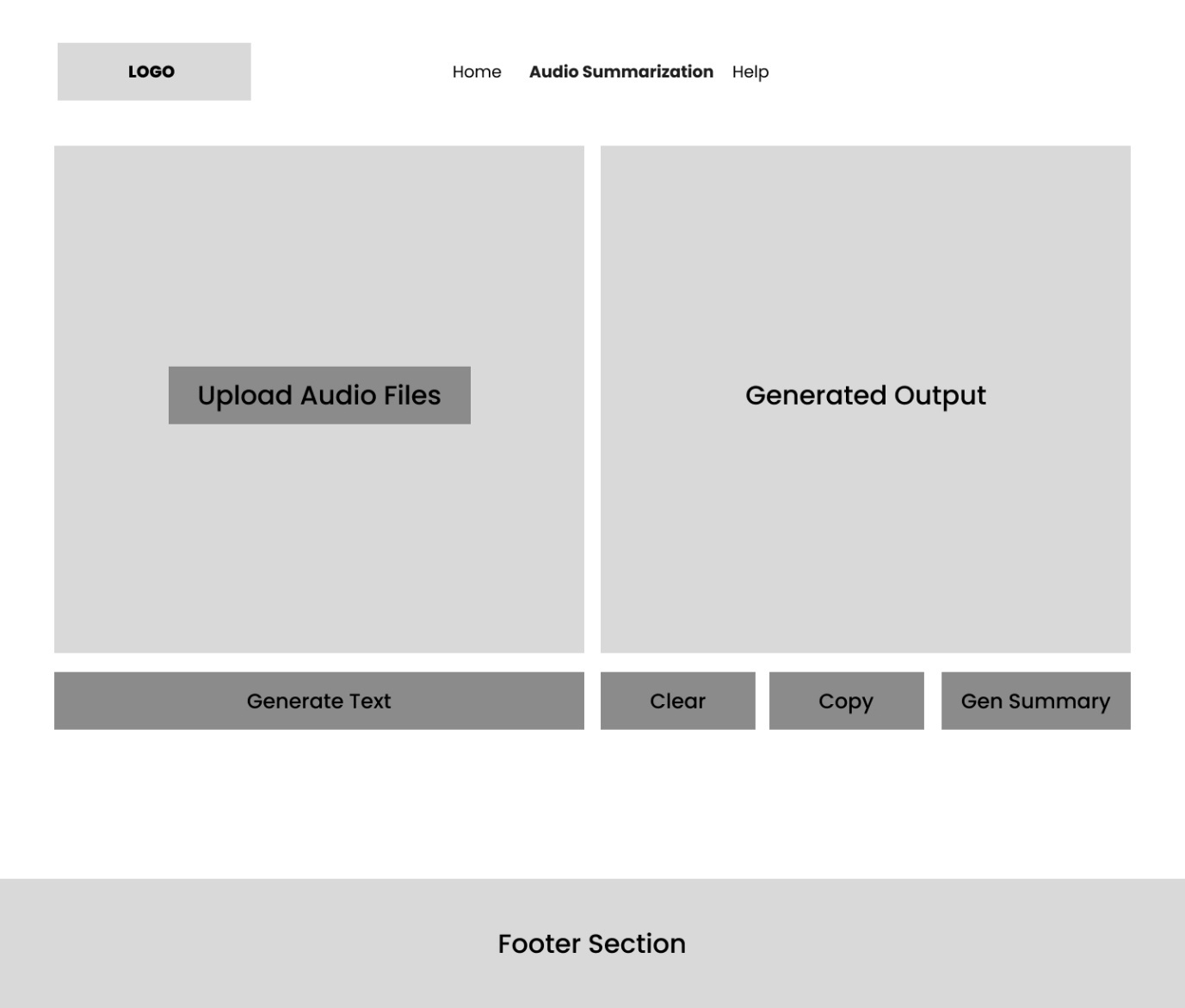


Figure 9: Wireframe of the system

## 6.6 Chapter summary

In this chapter it discussed about the design goals of this system. And for the design methodology author has selected the SSAD. Moreover, the high-level designs, data flow diagrams and the flowchart for the system are discussed. At the end of the chapter the UI for proposed system also included.

# CHAPTER 07: IMPLEMENTATION

## 7.1 Chapter Overview

In this chapter it is going to discuss the implementation of the system, like what technology is going to be used, the dataset for the model training, the frameworks, libraries and IDEs with a clear justification. At the end the implementation of the core functionality is discussed.

## 7.2 Technology Selection

### 7.2.1 Technology Stack

The following presents what are the technologies used to build the system, presenting in the presentation tier, logic tier and in the data tier

|  |
| --- |
| **Presentation Tier** |
|  |
| **Logic Tier** |
|  |
| **Data Tier** |
|  |

Table 14: Technology Stack

### 7.2.2 Dataset Selection

The dataset for this project is the main requirement. A dataset with high quality audio files which gives a clear speech, minimizing the background noises, and containing more data will give a high accuracy for this type of project. So, the first point of a dataset is needed for the audio to text process. Author was able to find an ASR dataset from Kaggle which contains more than 50000+ data. So, the author has chosen the dataset taken from Kaggle and created a subset on 5000 data. Also, the author used the Audacity software to record audios and created a custom dataset which contains 500 data and combine with that dataset. Here the dataset is taken to a .csv file which contains two columns namely the labeled sentences and the related file path. For the text summarization a dataset wasn’t required as it uses only an algorithm.

### 7.2.3 Development Frameworks

For the development of audio summarization there are various frameworks available. The below frameworks are used for this project as this is a web application.

|  |  |
| --- | --- |
| **Framework** | **Justification** |
| Flask | For the backend deployment of this project Flask framework is used. For python it will be a great choice as it is a lightweight and flexible framework. |
| Bootstrap | To develop a responsive and visually appealing web application bootstrap will be a great framework. This will make the author to build the application. |

Table 15: Development Frameworks

### 7.2.4 Programming Languages

Python is specifically suited for data science related projects. And the Python language is easy to learn, use, understand and it has the capability of handling multiple libraries and frameworks. For the existing systems like for text summarization, audio recognition, the researchers have used Python for the implementation. So, the author has chosen Python as the programming language.

### 7.2.5 Libraries

|  |  |
| --- | --- |
| **Library** | **Justification** |
| Librosa | This will be used for audio analysis tasks |
| NLTK | NLTK is widely used library for NLP tasks, this provides for tokenization, stemming, tagging and text preprocessing tasks |
| Tensorflow | This library is used for tasks like audio processing and text summarization process |
| Pytorch | This will be used for summarization tasks |
| Transformers | This is used for training ASR, and it provides access for the pre-trained models |
| Pandas | Pandas is mainly used to manipulate data and analysis structured data |
| NumPy | This is used for working with the arrays |

Table 16: Libraries used for The System Implementation

### 7.2.6 IDE

|  |  |
| --- | --- |
| Google Colab Pro | Google Colab Pro version performs well for the project related model training and testing. As in the Pro version it provides the compute unites and high ram for the training the model. And it allows to run Python codes and easily imports the libraries related to the project |
| VS Code | This is valuable IDE for the project implementation, the frontend and the backend. |

Table 17: IDEs used for The System Implementation

### 7.2.7 Summary of Technology Selection

|  |  |
| --- | --- |
| **Component** | **Tools** |
| Programming Languages | Python |
| Frameworks | Flask, Bootstrap |
| Libraries | Pytorch, NLTK, Tensorflow, Librosa, Transformers, Pandas |
| IDE | Google Colab Pro, VS Code |
| Version Control | Github, Huggingface |

Table 18: Summary of Technology Selection

## 7.3 Implementation of the Core Functionality

This system involves several key steps to implement the Sinhala audio summarization system. As the first step the user needs to upload audio files to the system. After it has been processed it should generate the text according to that. For the specific task the author has used the ASR dataset as mentioned above. The dataset was preprocessed and using a transfer learning approach the dataset was fine-tuned with a pre trained whisper AI model. After that that author create a model for the summarization purpose which is an extractive summarization approach. Using word frequency and sentence scoring algorithm it selects the most important sentence and generate the summary output. Once the audios are fed to the ASR model it generates the sentences combing as a paragraph. Then it is passed to the summarization model and generates the summary.

### 7.3.1 Audio Preprocessing

### A screen shot of a computer program Description automatically generated

Before training the ASR model as the first part the dataset is cleaned. The punctuation marks are removed, the duplicated rows are removed, English words and sentences are removed, and also the null values are also removed from the dataset.

### 7.3.2 Spilt the Dataset to Train and Test

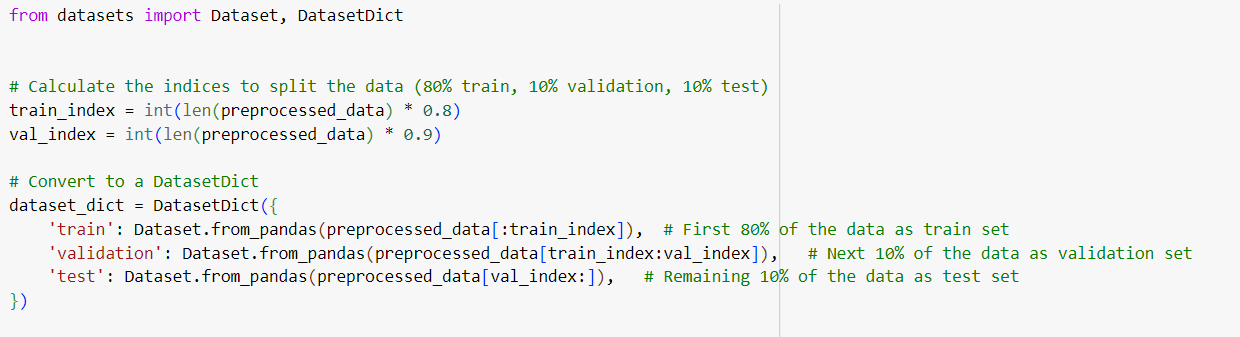


Figure 10: Split into Training, Testing and Validation Sets

Using the necessary libraries the audio data is split into training, testing and validation sets. The first 80% of data will be for training and 10% for testing and remaining 10% data will be split for validation sets. This helps to organize the sets of data for training, testing and validation.

A white background with colorful lines

Description automatically generated

Figure 11: Map the Audio Files

Here the ‘librosa’ library helps to read audio files from the provided function. And it maps the function to each row in the dataset. For each row it adds an audio column which contains the audio path, array and the sampling rate. Below image states it creates a new dataset dictionary which include updated dataset contain training, testing and validation.

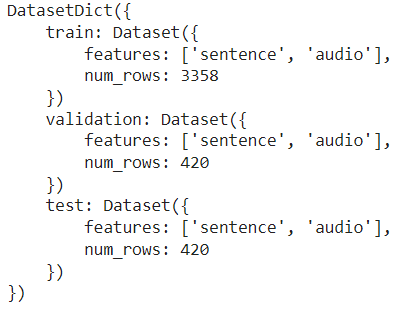


Figure 12: Mapped Output

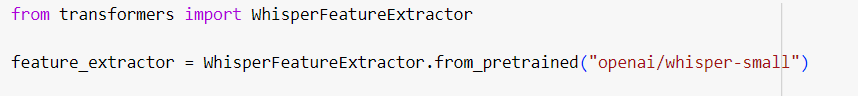




Figure 13: Loading Whisper Model

Here a pre-trained whisper model will be loaded using the ‘transformers’ library. A tokenizer and a whisper processor are initialized to the whisper model.

### 7.3.3 Setup the Training arguments and Train the ASR Model

Then the training arguments will be set and passed to the Seq2Seq model trainer, it uses the Hugging Face transformers library. These include the training process, how it learns from the data, performance and save the checkpoints. And after the arguments are set, the model will be trained. And after the model has been trained it has been pushed to the Hugging Face



Figure 14: Setting Training Arguments

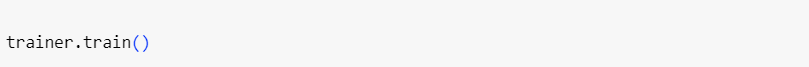


Figure 15: Train model

### 7.3.4 Text Summarization Model

Here the necessary libraries are imported. And then the imported stop word text file is used for stop-word removal.

A screen shot of a computer program

Description automatically generated

Figure 16: Stop-word Removal

A screenshot of a computer program

Description automatically generated

Figure 17: Generate frequency and Scoring sentence

Then the author uses to generate frequency for the words, and then using word frequency it scores the sentences and find the average score of the sentence. And it will look out of the sentences with scores greater and produce as the summary.

## A computer screen shot of a code Description automatically generated

Figure 18: Generate Summary

## 7.4 User Interface

For the UI implementation the author has used HTML, CSS, JavaScript and the Bootstrap framework. The below presents the UI of Home page and Audio Summarization. Other pages in the system are placed in the APPENDIX.



Figure 19: Home Page UI

## A screenshot of a computer Description automatically generated

Figure 20: Audio Summarization UI

## 7.5 Chapter Summary

This chapter clearly explains what the languages, technologies and tools are used for the implementation. Moreover, it has been discussed the implementation core functionality in detail with the necessary code snippets.

# CHAPTER 8: TESTING

## 8.1 Chapter Overview

In this chapter it will discussing about the testing methods used for the system. It will discuss the testing criteria, functional and non-functional testing, testing of the model, integration of the module and what was the limitation of testing faced by the author.

## 8.2 Objectives and Goals of Testing

The objective and the main purpose of testing is to verify that the developed system functionalities work as the expected, without any errors. To achieve this the priorities, the objectives of testing are stated above.

* To verify the implementation of the system works fine without any errors.
* Verify the ASR model in AudioSangaraha system operates as expected and has gone through the testing process.
* Also to verify the models produce the results as expected.
* To verify that the system has fulfill functional requirement which is the “Must have” and “Should have” in the MoSCoW technique.
* Also to ensure that the system fulfills the non-functional requirement.
* To state the potential area of improvements in the system

## 8.3 Testing Criteria

For the testing criteria the author uses to access the system in two methods. The two methods are stated above.

* Functional Testing – In this method it uses to test the functional requirements to ensure that all the features are determine well.
* Structural Testing – In this method it uses to test the non-functional requirements of the system. Also checks the system compliance with the performance of function requirements.

## 8.4 ASR Model Testing

For an ASR model there are several metrics to test the model as mentioned it literature review. Here the author will be using MER, WER and CER for the Whisper speech recognition model testing.

### 8.4.1 Match Error Rate (MER)

The testing metrics of MER for the model shows 0.4, which means 0.6 of words are recognized correctly. This value is high as the author has train the model with a limited dataset.

A screenshot of a computer program

Description automatically generated

### 8.4.2 Character Error Rate (CER)

CER measures the percentage of incorrectly recognized characters. For the testing metrics of CER for the model shows 0.3, which means 0.7 of characters are recognized correctly.

A screen shot of a computer code

Description automatically generated

### 8.4.3 Word Error Rate (WER)

WER measures the percentage of incorrectly recognized words. For the testing data, testing metrics of WER for the model shows 0.7. Which means it shows a high rate of WER score. One reason for this is the trained dataset it was a small amount of data. And the other reason is the Whisper model use for speech recognition only has the ability to recognize the first 30 seconds of audio. So, this is the reason for getting high WER score.

## 

## 8.5 Functional Testing

The functionalities in the system which are mentioned in the CHAPTER 04 are tested and stated below in the table.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| FR ID | Use case | Expected result | Actual result | Status |
| FR1  The system should be able upload multiple audio files to the system | Upload multiple audio files to the system | The system allows to upload multiple audio files | The system allows to upload multiple audio files | Pass |
| FR2  The system must not support other than audio file | Upload other than audio files to the system | The system restricts or gives a prompt saying doesn’t support other than audio file | The system gives a prompt saying not defined | Pass |
| FR3  The system should generate the text accordingly | Generate the text from the audio accordingly | The audio files output text should be combined as paragraph | The system produces the generated text from the audio files combining as a paragraph | Pass |
| FR4  User should be able to copy the generated text | Generated text should be able to copy | Should be able to copy the text | Able to copy the text | Pass |
| FR5  User should be able to summarize the generated text | Generated text should be able to summarize | System should be able to summarize the text | System summarizes the text | Pass |
| FR6  User should be able to copy the generated summary | Generated summary should be able to copy | Should be able to copy the generated summary | Able to copy the summary text | Pass |

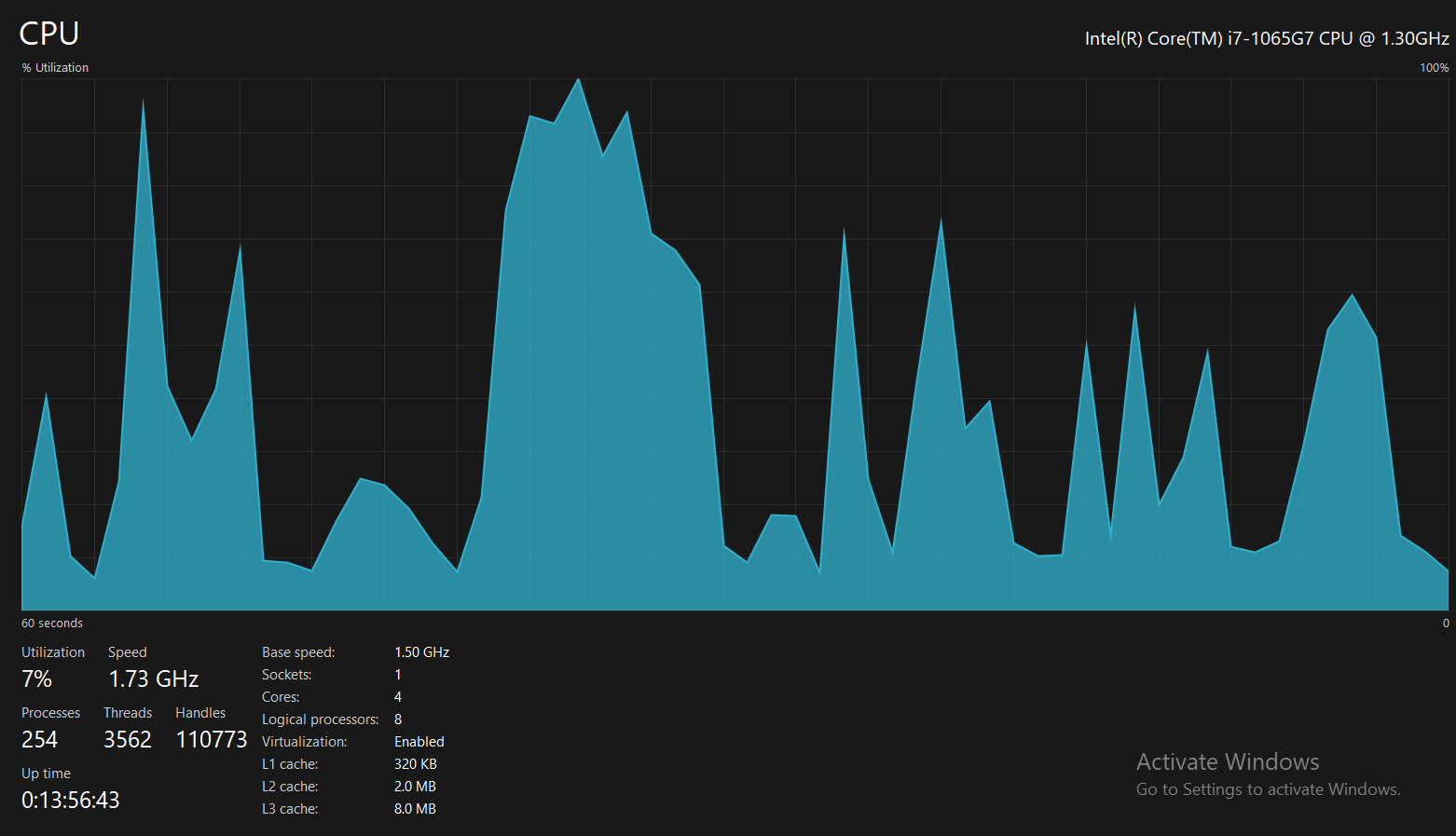
## 8.6 Module Integration Testing

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Module | Input | Expected Result | Actual Result | Status |
| Input of audio files | Upload audio files | Upload multiple audio files in the audio input section | Can upload multiple audio files | Pass |
| Input of audio files | Verify the audio files | Popup a message saying if other than the audio files selected/ Or any other audio formats are added | Message popups saying not defined | Pass |
| Input Text | Enter a Sinhala paragraph/ Copy and paste a Sinhala paragraph | Paste Sinhala paragraph in the text area | Able to paste Sinhala paragraph in the text area | Pass |
| Input Text | Verify the sentence size | Popup a message saying add more than one sentence | Message popups asking for more sentences | Pass |
| Generate audio summary | Once the audio generated text displays, should be able form the summary | Click the summary button and should be to get the summary | Able to produce the summary accordingly | Pass |

## 8.7 Non-Functional Testing

### 8.7.1 Performance Testing

The performance testing for the web application is crucial on producing the user experience. This system shows that with a minimum resource this web application can be run on a local environment. The performance on a CPU screenshot is placed below.



A screenshot of a computer

Description automatically generated

### 8.7.2 Security Testing

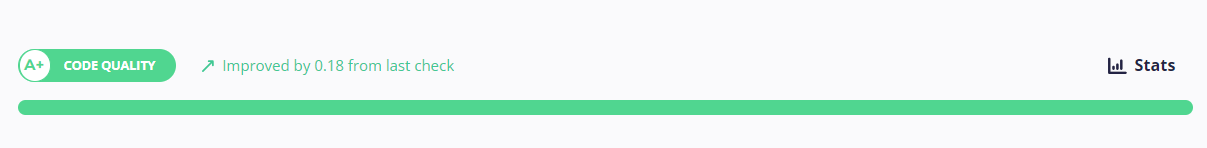
The security testing for a system involves in application to protect data. In this system it doesn’t store or collect any of user information’s and data, or any harmful contents. Also, it doesn’t include any other third-party activities involved.

### 8.7.3 Usability Testing

The author has thought of the usability and developed this web application. This has a simple UI which helps the user to navigate through the pages, and functionalities in the application. This was tested within the end users.

### 8.7.4 Maintainability Testing

The implementation code of the system is available in the author GitHub repository. To check the maintainable testing author used a tool to address the code quality (Codefactor.io). The below image shows that the system code has a A+ code quality.



## 8.8 Limitations of the Testing Process

The author has faced of various limitation while the testing process due to the limitation of resource in Sinhala language. One of the major limitations of the testing is that the Whisper model which is used for the speech recognition has a limitation of predicting the correct text within a time limit (It is able recognize the first 30 seconds of audio). Also, with the limitation of the computational power and resources a limited of dataset was used to train the model. Because this the model WER was high and was difficult to get a lower WER. For the text summarizing a testing wasn’t conducted as the author was unable and there wasn’t publicly available dataset for the extractive summarization.

## 8.9 Chapter Summary

First of all, in this chapter it has stated the goals of the objectives in testing and then it has been discussed about the testing metrics used to test the speech recognition model, the status of the functional and non-functional testing in the system. Also, at the end of the chapter it has been discussed what are the limitations of the testing the author faced.

# CHAPTER 05: CONCLUSION

## 5.1 Chapter Overview

In this chapter, it has been what is the progress level of the project, what are the future steps going to be taken. And using a small presentation the author has been demonstrating the prototype.

## 5.2 Deviations

### 5.2.1 Scope related deviations

The author needs to build a model for the audio to text process. So, he has used a transfer learning approach for a whisper AI pretrained model and fine tune it according to the dataset gathered.

### 5.2.2 Schedule related deviations

Within the scheduled time period the author has completed the documentation related to the project and partially completed the prototype. The author is confident on completing the prototype within the given time period.

## 5.3 Initial Test Results

The dataset gathered was used to train, as there is a large amount of data the author has created a subset from the dataset. But unfortunately, the Colab notebook was getting crashed due to the GPU level. So, the author will be using the Colab pro version and training the model in the upcoming days.

## 5.4 Required Improvements

|  |  |
| --- | --- |
| **Required improvements** | **How you intend to do it** |
| Audio to Text model | Author will be using the Colab pro version for the training purposes. As the audio data is at a high level. And the author will improve grammatical and the spelling errors accordingly. |
| Text summarization | For text summarization the author will be implementing a model. |

Table 19: Required Improvements

## 5.5 Demo of the Prototype

The following link provides the code for the prototype

<https://colab.research.google.com/drive/1c010yHPWUTGNWVhHmbtU7vzSv9uqIrtg?usp=sharing>

The following YouTube link provides the video for the prototype

<https://youtu.be/3rNK2r6RJTw>

## 5.6 Chapter Summary

In this chapter it is been explained about the deviations from the scope and schedule of the project. Then it discusses about the improvement of proposed system. And at the end author gives a demo of the prototype.

## REFERENCES

Das , P. and Prasad, V. (2015) VOICE RECOGNITION SYSTEM: SPEECH-TO-TEXT. Available at: https://www.researchgate.net/publication/304651244\_VOICE\_RECOGNITION\_SYST EM\_SPEECH-TO-TEXT (Accessed: 01 September 2023).

de Silva, N. (2019) Survey on Publicly Available Sinhala NaturalLanguage Processing Tools and Research. Available at: https://www.researchgate.net/publication/333649787\_Survey\_on\_Publicly\_Available\_ Sinhala\_Natural\_Language\_Processing\_Tools\_and\_Research (Accessed: 05 September 2023).

Deshpande, P. and Jahirabadkar, S. (2021) A survey on statistical approaches for abstractive summarization of low resource language documents, SpringerLink. Available at: https://link.springer.com/chapter/10.1007/978-981-16-4016-2\_69 (Accessed: 01 September 2023).

Dhananjaya, V. et al. (2022) Bertifying Sinhala -- a comprehensive analysis of pre-trained language models for Sinhala Text Classification, arXiv.org. Available at: https://arxiv.org/abs/2208.07864 (Accessed: 12 September 2023).

Dhananjaya, V. et al. (2022) Bertifying Sinhala -- a comprehensive analysis of pre-trained language models for Sinhala Text Classification, arXiv.org. Available at: https://arxiv.org/abs/2208.07864 (Accessed: 25 September 2023).

Dinushika, T. et al. (2020) Speech Command Classification System for Sinhala Language based on Automatic Speech Recognition. Available at: https://ieeexplore.ieee.org/document/9037648/ (Accessed: 15 January 2024).

Gamage, B. et al. (2021) Improving Sinhala Speech Recognition Through e2e LF-MMI Model, https://aclanthology.org. Available at: https://aclanthology.org/2021.iconmain.26.pdf (Accessed: 24 September 2023).

Gruetzemacher, R. (2022) The power of Natural Language Processing, Harvard Business Review. Available at: https://hbr.org/2022/04/the-power-of-natural-language-processing (Accessed: 31 August 2023).

Jing, B. et al. (2021) Multiplex Graph Neural Network for extractive text summarization, ACL Anthology. Available at: https://aclanthology.org/2021.emnlp-main.11/ (Accessed: 15 January 2024).

Karunathilaka, N.A.K.H.S. (2020) Low-resource Sinhala Speech Recognition using Deep Learning, Digital Library of University of Colombo School of Computing: Home. Available at: https://dl.ucsc.cmb.ac.lk/jspui/ (Accessed: 14 January 2024).

Kasthuri Arachchige , T. and Weerasinghe, R. (2023) Tacosi: A Sinhala text to speech system with Neural Networks | IEEE ..., TacoSi: A Sinhala Text to Speech System with Neural Networks. Available at: https://ieeexplore.ieee.org/abstract/document/10145749 (Accessed: 05 September 2023).

Kemp, S. (2022) Digital 2022: Sri Lanka - datareportal – global digital insights, DataReportal. Available at: https://datareportal.com/reports/digital-2022-sri-lanka (Accessed: 05 September 2023).

Khandare, P. et al. (2019) AUDIO DATA SUMMARIZATION SYSTEM USING NATURAL LANGUAGE PROCESSING, 06(09). doi:https://www.irjet.net/archives/V6/i9/IRJETV6I957.pdf.

Lewis, Mike, et al. “BART: Denoising Sequence-To-Sequence Pre-Training for Natural Language Generation, Translation, and Comprehension.” Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, 2020, <https://doi.org/10.18653/v1/2020.acl-main.703>.

Millstein, F. (2020) Natural language processing with python: natural language processing using NLTK, https://scholar.google.com/. Available at: https://books.google.lk/books?hl=en&lr=&id=vXzvDwAAQBAJ&oi=fnd&pg=PA4&d q=Frank+Millstein.+(2020).+Natural+Language+Processing+Using+NLTK.+Frank+M illstein.&ots=02SOrlVUaE&sig=i1bsvq75ZnWN9HC2lhcD0F3dIc&redir\_esc=y#v=onepage&q=Frank%20Millstein.%20(20 20).%20Natural%20Language%20Processing%20Using%20NLTK.%20Frank%20Mill stein.&f=false (Accessed: 31 August 2023).

Phair, David, and Kerryn Warren. “Saunders’ Research Onion: Explained Simply.” Grad Coach, Jan. 2021, gradcoach.com/saunders-research-onion/. Accessed 28 Sept. 2023.

Prudhvi, K. et al. (2020) Text summarization using Natural Language Processing, SpringerLink. Available at: https://link.springer.com/chapter/10.1007/978-981-15- 5400-1\_54 (Accessed: 04 September 2023).

Rathnayake, B.R.M.S.R.B., Manathunga, K. and Kasthurirathna, D. (2023) ‘Talking Books’ : A Sinhala Abstractive Text Summarization Approach for Sinhala Textbooks. Available at: https://ieeexplore.ieee.org/search/advanced (Accessed: 08 September 2023).

S. Yu, Philip , et al. “Understanding Pre-Trained BERT for Aspect-Based Sentiment Analysis.” Aclanthology, Dec. 2020, aclanthology.org/2020.coling-main.21.pdf.

Shah , M., Jan , R. and Mohd, M. (2019) Text document summarization using word embedding, Expert Systems with Applications. Available at: https://www.sciencedirect.com/science/article/abs/pii/S0957417419306761?via%3Dihub (Accessed: 14 January 2024).

Singh, A. (2020) Text summarization using NLP, Medium. Available at: https://medium.com/analytics-vidhya/text-summarization-using-nlp-3e85ad0c6349 (Accessed: 04 September 2023).

Warnasooriya , W.M.P.N. et al. (2020) Sinhala speech recognition system for journalists in Srilanka, SINHALA SPEECH RECOGNITION SYSTEM FOR JOURNALISTS IN SRILANKA. Available at: https://www.researchgate.net/publication/346624775\_SINHALA\_SPEECH\_RECOGN ITION\_SYSTEM\_FOR\_JOURNALISTS\_IN\_SRILANKA (Accessed: 11 September 2023).

Weerasinghe, R. et al. (2020) Low-resource sinhala speech recognition using Deep Learning | IEEE ... Available at: https://ieeexplore.ieee.org/document/9325468 (Accessed: 12 September 2023).

Weerasinghe, R. et al. (2020) Low-resource sinhala speech recognition using Deep Learning | IEEE ... Available at: https://ieeexplore.ieee.org/document/9325468 (Accessed: 12 September 2023).

## APPENDIX

