A

# Project Report

on

# LANGUAGE TRANSLATOR

Submitted in Partial Fulfillment of
the Requirements for the Degree
of

**Bachelor of Engineering**

in

**Information Technology**

to

**Kavayitri Bahinabai Chaudhari
North Maharashtra University, Jalgaon**

Submitted by
**Shaikh Rehan Farooque Maniyar**
**Pravin Digamber Patil**
**Sayyed Sahil Shakil**
**Yogesh Suresh Patil**

Under the Guidance of
**Prof. Pravin K. Patil**

**DEPARTMENT OF COMPUTER ENGINEERING**
SSBT's COLLEGE OF ENGINEERING AND TECHNOLOGY,
BAMBHORI, JALGAON - 425 001 (MS)
2021 - 2022

**SSBT's COLLEGE OF ENGINEERING AND TECHNOLOGY, BAMBHORI, JALGAON - 425 001 (MS)**

**DEPARTMENT OF COMPUTER ENGINEERING**

# CERTIFICATE

This is to certify that the project entitled *Language Translator*, submitted by

**Shaikh Rehan Farooque Maniyar**
**Pravin Digamber Patil**
**Sayyed Sahil Shakil**
**Yogesh Suresh Patil**

in partial fulfillment of the degree of *Bachelor of Engineering* in *Information Technology* has been satisfactorily carried out under my guidance as per the requirement of Kavayitri Bahinabai Chaudhari North Maharashtra University, Jalgaon.

**Place:** Jalgaon

Prof. Pravin K. Patil
**Guide**

Prof. Dr.Manoj E. Patil
**Head**

Prof. Dr. G. K. Patnaik
**Principal**

# Acknowledgements

We would like to express deep gratitude and sincere thanks to all who helps to complete the Project successfully. Also thanks Prof. Dr. G. K. Patnaik, Principal, for having provided the facilities to complete the Project work. A deep gratitude goes to Prof. Dr. Manoj E. Patil, Head of the Department, for granting us opportunity to conduct Project work. Also sincere thanks to Prof. Pravin K. Patil , Project Guide, for the valuable suggestions
and guidance at the time of need. Sincere thanks to Mr. Akash D. Waghmare, Asst. Prof., Project Incharge and Great thanks to our friends, Project associates and all those who helped directly or indirectly for completion of the Project. Last but not least thanks to our parents.

Shaikh Rehan Farooque Maniyar
Pravin Digamber Patil
Sayyed Sahil Shakil
Yogesh Suresh Patil

# Contents

# List of Figures

# Abstract

Machine Translation (MT) software today provides adequate conversion of foreign languages to one's native tongue; however, dialects, slang, and character conversion errors result in partially successful translations. For an accurate translation, a native speaker is often required to correct the translation by using sentence structure and word use cues to capture the true meaning. MT character conversion from Cyrillic, Asian, and Arabic languages to western characters induce errors in the translated text which can change the meaning or result in characters being associated together that do not form words. The authors present a solution using open source MT and the International Organization for Standardization (ISO) character mapping. The solution provides proper character conversion to achieve greater translation accuracy for web-based content.

# Chapter 1

# Introduction

The chapter describes all the details regarding introduction of a Language Translator. The introduction part of the report includes all the basic information required to elaborate the overall idea of the Language Translator. It includes the basic concept of a Language Translator as well as Machine Translation. The chapter also includes the different techniques which are used while developing the web application. The functionality which is provided in the proposed system is intended to translate one Language to another native or foreign language.

Section 1.1 describes the Background of the topic. The purpose of Section 1.2 is to explain Motivation behind the project. Section 1.3 contains the Problem Definition of project. Scope of the project is included Section 1.4. Section 1.5 contains Objective of project. Section1.6 describes identification of software development Process Model. Section 1.5 contains Organization of Report. Section 1.6 includes the summary of project.

## 1.1 Background

The word ‘translation’ comes from a Latin term which means “to bring or carry across”. Another relevant term comes from the Ancient Greek word of ‘metaphrasis’ which means “to speak across” and from this, the term ‘metaphrase’ was born, which means a “word-for-word translation”. These terms have been at the heart of theories relating to translation throughout history and have given insight into when and where translation have been used throughout the ages. It is argued that the knowledge and findings of Greek academics was developed and understood so widely thanks to the translation work of Arabic scholars. When the Greeks were conquered, their works were taken in by Arabic scholars who translated them and created their own versions of the scientific, entertainment and philosophical understandings. These Arabic versions were later translated into Latin, during the middle Ages, mostly throughout Spain and the resulting works provided the foundations of Renaissance academics.

The origins of machine translation can be traced back to the work of Al-Kindi, a 9th-century Arabic cryptographer who developed techniques for systemic language translation, including cryptanalysis, frequency analysis, and probability and statistics, which are used in modern machine translation

## 1.2 Motivation

Nowadays, speaking multiple languages is considered very helpful thing. So an idea of developing such mobile application which will help to do so that to in a handy way is proposed. Also it will help to blend in to different religion, culture, and countries

## 1.3 Problem Definition

Machine generated translation is much more accurate than Human translation. Travelers often rely on Human translators that result in increase in expense when the person is trying to communicate in a foreign language. Some seek help from their friends or relatives in these situations. This issue will be addressed using social media. A social network brings people together to help each other and exchange experiences. The ultimate goal is to eliminate language barriers for travelers by connecting them to interpreters through Text Boxes and Voice Assistant. This will help the user get the trusted translation from various grades of translators, basic to fluent, affecting the price rate of the translated information. Also, providing a way for the app makers to make some extra money.

## 1.4 Scope

India is the 5th most leading business country. We have trade links with Germany, Russia and United States chiefly. Hence, these languages are gaining ground within the country. Also, most prestigious institutions abroad require a third language skill to admit students and the trend shows that students are attracted more to German and French. Translators in India make a lot of money too. Hindi is the market and every literature across the world has to be translated to reach to the common layman of India.

## 1.5 Objective

The main objective of the project is to develop an application that will provide a Platform to translate one language (Source Language) to another language (Target Language). What is the objective of translation?

The goal of translation practice for non-specialists is to find the language skills of the learner, to refine their thematic and cultural knowledge and to encourage them to think and to react.

## 1.6 Identification of software development Process Model

The four basic process activities of specification, development, validation and evolution are organized differently in different development processes. In the waterfall model, they are organized in sequence, while in incremental development they are interleaved.

Software Development Life Cycle(SDLC) is a process used by the software industry to design, develop and test high quality software. The SDLC aims to produce a high quality software that meets or exceeds customer expectations, reaches completion within times and cost estimates.

### 1.6.1 WaterFall Model- Design

Figure 1.1 shows “The Waterfall Model”.Waterfall approach was the first Model to be used widely in Software Engineering to ensure success of the project. In The Waterfall approach, the whole process of software development is divided into separate phases. In the Waterfall model, typically the outcome of one phase acts as the input for the next phase sequentially. The following illustration is a representation of the different phases of the Waterfall Model

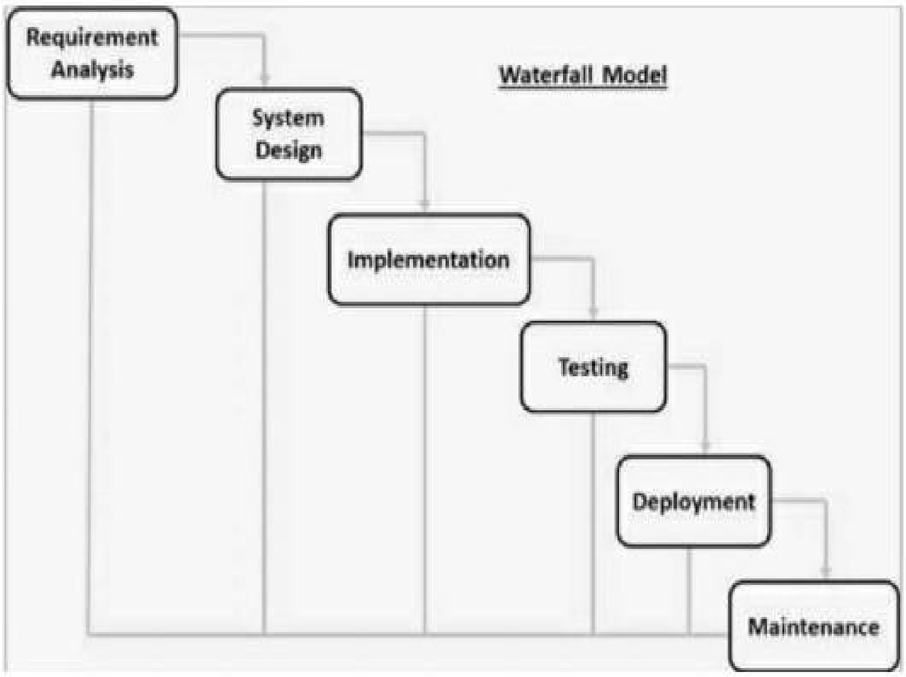


Figure 1.1: WaterFall Model

### 1.6.2 Requirement Analysis

All possible requirements of the system to be developed are captured in the phase and documented in a requirement specification document

### 1.6.3 System Design

The requirement specifications from first phase are studied in the phase and the system design is prepared. The system design helps in specifying hardware and system requirements and helps in defining the overall system architecture

### 1.6.4 Implementation

With inputs from the system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.

### 1.6.5 Integration and Testing

All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures. Deployment: Once the functional and non-functional testing is done, the product is deployed in the customer environment or released into the market.

### 1.6.6 Maintenance

There are some issues which come up in the client environment. To fix those issues, patches are released. Also to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment. All these phases are cascaded to each other in which progress is seen as flowing steadily downwards (like a waterfall) through the phases. The next phase is started only after the defined set of goals are achieved for previous phase and it is signed off, so the name Waterfall Model. In the model, phases do not overlap. The project under consideration is being developed using the above discussed model .

## 1.7 Organization of Report

The report is described in following way,

- **CHAPTER 1**, entitled as Introduction describes the details about Background, Problem Definition, Scope and Objective of the project, Identification of Software Development Process Model and Organization of report.
- **CHAPTER 2**,entitled as Project Planning and Management consists of details about the Feasibility Study, Risk Analysis, Project Scheduling, Effort Allocation and Cost Estimation of the project.
- **CHAPTER 3**, entitled as Analysis describes in detail, the Requirement Collection and Iden-tification, H/w and S/w Requirements, Functional and Non-Functional Requirements and a Software Requirements Specification(SRS).
- **CHAPTER 4**, includes design about System Architecture, Data Flow Diagram and various UML Diagrams.
- **CHAPTER 5**,provides the conclusion and future work which can be done on the topic.

## 1.8 Summary

In the chapter all the details about problem definition, motivation, scope, objective of the project and selection of software development model are included. In the next chapter, description about the Project Planning and Management is provided.

# Chapter 2

# Project Planning and Management

The chapter includes details regarding Feasibility study, Risk Analysis, Project Scheduling, Effort Allocation and Cost Estimation as well as a short summary.

Section 2.1 includes details regarding feasibility that includes Operating, Economical and Technical feasibility. Details regarding Risk Analysis like Commercial risks, Design risks and other risks as well are discussed in Section 2.2. Section 2.3 provides explanation about Project Scheduling while section 2.4 describes the effort allocation i.e effort taken by each group member. Section 2.5 provides information about cost estimation of the project.

## 2.1 Feasibility Study

Once scope has been identified, it is reasonable to ask: Can the software be built to meet the scope? Is the Project feasible? All too often, software engineers rush past the questions (or are pushed past them by impatient managers or customers), only to become mired in a project that is doomed from the onset .

Feasibility is the analysis of risks, costs and benefits relating to economics, technology and user operation. There are several types of feasibility depending on the aspects they cover. Some important feasibility is as follows:

- Operational Feasibility
- Economical Feasibility
- Technical Feasibility

### 2.1.1 Economic Feasibility

More commonly known as cost/benefit analysis the procedure is to determine the benefits and savings that are expected from system and compare them with costs, decisions is made.

To design and implement the system. The part of feasibility study gives the top management the economic justification for the new system. The is an important input to the management the management, because very often the top management does not like to get confounded by the various technicalities that bound to be associated with a project of the kind. A simple economic analysis that gives the actual comparison of costs and benefits is much more meaningful in such cases. In the system, the organization is most satisfied by economic feasibility. Because, if the organization implements the system, it need not require any additional hardware resources as well as it will be saving lot of time.

The proposed system i.e language translator uses Android platform for development as well as a separate point of interest and approaches that requires less amount of money and produces efficient results at a low price. When the project will be a complete software functionality, people will have to pay lesser money, though it will be a proprietary one, still the money paid in comparison to the yields will be negligible. Therefore, the project will prove to be economically feasible.

### 2.1.2 Operational Feasibility

People are inherently resistant to change and computer has been known to facilitate changes. An estimate should be made of how strong the user is likely to move towards the development of computerized system. These are various levels of users in order to ensure proper authentication and authorization and security of sensitive data of the organization.

Here, the proposed system for a language translator can be justified as operationally feasible based on the following:

- The methods of processing and presentation are completely acceptable by the users as they meet all the user requirements.
- The users have been involved during the preparation of requirement analysis and design process.
- The system will precisely identify the points of interest and display the relevant and useful information according to the specified format.
- The system will prove to be useful to the people who are totally unknown of the country and their language.

### 2.1.3 Technical Feasibility

Technical feasibility centres on the existing manual system of the test management process and to what extend it can support the system. According to feasibility analysis procedure the technical feasibility of the system is analyzed and the technical requirement such as software facilities, procedure, inputs are identified. It is also one of the important phases of the system development activities. System covers greater level of a user friendliness combined with greater processing speed. Therefore the cost of maintenance can be reduced. Since processing speed is very high and the work is reduced in maintenance point of view management convince that the project is operationally feasible.

Standards are something of a universal language for a software application. Without standards, (language translator) machine translation-based project is a thing of its own barely compatible with the others. That complicates the process of unifying solutions to the greater whole which makes the overall development of the technology much slower than it could have been if everyone had been on the same page.

## 2.2 Risk Analysis

The aim of the work is to adapt, combine, implement and evaluate engineering techniques in a new application aiming at analyzing the risks that are present. The risks evaluated in the production process are directly associated to the quality and safety of the environment, as well as to the procedures involved. The risks are analyzed. The risks evaluated in the production process are directly associated to the quality and safety of the environment, as well as to the procedures involved.

### 2.2.1 Commercial Risks

Following are some commercial risks identified in the project-

Certain words found in one language may not be available in the language it is being translated into. Sometimes, a single word may have two meanings in a particular language. In such as case, the translator is expected to be skilled to know the difference and render the word in the right meaning. A word may have an innocuous meaning in one culture but might take on a derogatory meaning in another.

### 2.2.2 Design/Engineering Risks

Risk: Errors and design assumptions. Action to be taken: Use the expertise by project co-odinators, developers.

### 2.2.3 Other Risks

There are some other risks other than the above discussed they are as follows-

Technology risks are slightly different and it is the translator who assumes these risks. They must invest in the tools of their trade and take steps to protect their work and run their business efficiently. Sometimes it may seem like a time consuming chore, but technology and data management are a key part of a modern translator's life, so we have to invest in them.

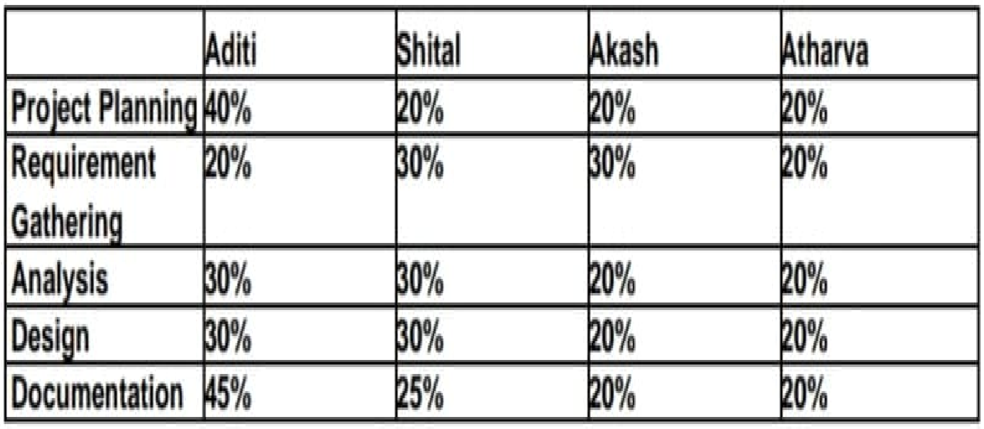
## 2.3 Project Scheduling

The section specifies the project scheduling of the project. Software project scheduling is an activity that distributes estimated effort across the planned project duration by allocating the effort to specific software engineering task. In the phase, we are identifying all the major software engineering activities and the product function to which they are applicable. As the linear sequential model have been selected for developing the project, the work is divided according to the phases of the model. As a group of four members working on the project, the project is scheduled accordingly. If the project development goes as per the planned schedule, the project schedule defines the task and that must be tracked and controlled as progress.

## 2.4 Effort Allocation

Figure 2.1 shows Effort Allocation.Each of the software project estimation techniques leads to an estimate of work units required for completion of the software development. The characteristics of each project task dictate the distribution of efforts. There are four members in the project and there are four phases such as Project Planning, Requirement Gathering, Analysis, Design, so the below table shows efforts of each member in the project.

Technology risks are slightly different and it is the translator who assumes these risks. They must invest in the tools of their trade and take steps to protect their work and run their business efficiently. Sometimes it may seem like a time consuming chore, but technology and data management are a key part of a modern translator's life, so we have to invest in them.



| | Aditi | Shital | Akash | Atharva |
|---|---|---|---|---|
| Project Planning | 40% | 20% | 20% | 20% |
| Requirement Gathering | 20% | 30% | 30% | 20% |
| Analysis | 30% | 30% | 20% | 20% |
| Design | 30% | 30% | 20% | 20% |
| Documentation | 45% | 25% | 20% | 20% |

Figure 2.1: Effort Allocation

## 2.5 Cost Estimation

Cost estimation is a set of techniques and procedures used to arrive at a cost estimate. These techniques are utilized by the process of cost estimation to compute the output from the given set of inputs. (Constructive Cost Model) is a regression model based on LOC, i.e number of Lines of Code. It is a procedural cost estimate model for software projects and often used as a process of reliably predicting the various parameters associated with making a project such as size, effort, cost, time and quality. It was proposed by Barry Boehm in 1970 and is based on the study of 63 projects, which makes it one of the best documented models.

The key parameters which define the quality of any software products, which are also an outcome of the Cocomo are primarily Effort and Schedule. Effort: Amount of labor that will be required to complete a task. It is measured in person-months unit. Schedule: Simply means the amount of time required for the completion of the job, which is, of course, proportional to the effort put. It is measured in the units of time such as weeks, months.

## 2.6 Summary

In the chapter, all the details related to Project Planning and Management are mentioned. In the next chapter, all the details regarding the requirements gathering and analysis are presented.

# Chapter 3

# Analysis

The chapter describes everything related to the requirement gathering and further analysis such as hardware, software, functional and non-functional requirements. Also it has the software requirements specification that provides complete description of the requirements of the system.

Section 3.1 describes requirement collection and identification. All the hardware and software requirements are discussed in Section 3.2. Section 3.3 describes the functional and non-functional requirements of the system to be developed. Section 3.4 describes the Software Requirements Specification (SRS)

## 3.1 Requirements Collection and Identification

In system engineering and software engineering, requirements analysis focuses on the tasks that determine the needs or conditions to meet the new or altered product or project, taking account of the possibly conflicting requirements of the various stakeholders, analyzing, documenting, validating and managing software or system requirements. The main objective here is to understand the existing system thoroughly and check the feasibility requirements. Requirements Analysis includes three types of activities:

- **Eliciting Requirements** Eliciting Requirements :( E.g. the project charter or definition), Business process documentation, and stake holder interviews. Sometimes it is also called requirements gathering or requirements discovery.

- **Analyzing Requirements** Analyzing Requirements: Determining whether the stated requirements are clear, complete, consistent and unambiguous, and resolving any apparent conflicts.

- **Recoding Requirements** Recoding Requirements: Requirements may be documented in various forms, usually including a summary list and may include natural-language documents, use cases, user stories, process specifications and a variety of s models including data models.

## 3.2 H/W And S/W Requirements

Hardware and software requirements of the projects such as type of processor, memory, storage, graphics and OS are:

### 3.2.1 Hardware Requirements

The following things are required:

- Qualcomm Snapdragon 625 for Mobile and Intel CORE i5 8th Gen for Computer
- Android Phone or Laptop
- 4 GB RAM

### 3.2.2 Software Requirements

Software Requirement deal with denting software resource requirement and prerequisites that need to be installed on a computer to provide optimal functioning of an application. These Requirements or pre-requisites are generally included in the software installation pack-age and need to be installed separately before the software is installed.

- **Operating System:** Operating System: Android 9.0(Pie) Or Windows 10
- **Front End** Front End: Android Application/Web Application
- **Back End** Back End: Java , Firebase

## 3.3 Functional and non-functional Requirements

In the following chapter functional and non functional requirements are mentioned.

### 3.3.1 Functional Requirement

It deals with the functionalities required from the system which are as follows. All the functionalities that are required to help satisfy the main goal of the system are included within the functional requirements.

Automatic speech recognition (ASR), MT, and voice synthesis (or text to speech; TTS). The ASR component processes the voice in its original language, creating a text version of what the speaker said. This text in the original language goes through the MT component, which translates it to the target language.

### 3.3.2 Non-Functional Requirement

There are Quality Requirements that stimulate how well a software does what it has to do. Performance of the number of terminals to be supported is dependent on the server that will be used at the time of deployment.

The proposed system will thereby provide service and it can be monitored on a basis where a server dependent performance will be mapped on time, when the software will be deployed successfully.

## 3.4 Software Requirement's Specification (SRS)

A software requirements specification (SRS) is a document that captures complete description about how the system is expected to perform. It is usually signed off at the end of requirements engineering phase. The quality characteristics of SRS must be according to the given guidelines:

- Correctness
- Completeness
- Consistency
- Unambiguousness
- Ranking for importance and stability
- Modifiability
- Verifiability
- Traceability

## 3.5 Summary

In the chapter, Analysis was presented which included the hardware and software requirements, functional and non-functional requirements and the software requirements specification (SRS) as well. In the next chapter, Design is described along with various UML diagrams.

# Chapter 4

# Design

System design provides the understanding and procedural details necessary for implementing the system.

Section 4.1 describes the System Architecture which includes. Description about the Data Flow Diagram is provided in Section 4.2. Section 4.3 describes various UML diagrams related to the architecture. Section 4.4 provides a short summary.

## 4.1 System Architecture

A System Architecture is the conceptual model that defines the structure, behaviour, and more views of a system and architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures and Assistant architecture can comprise system components, the extremely visible properties of those components, the relationship i.e the behavior between them. It can provide a platform in which system can be procured, and systems developed, that will work together to implement the overall system.

Rule-based approach is yet another approach for machine translation. This approach gives grammatical correct translation by using set of rules. Basically, the rule-based machine translation system contains a source language morphological analyzer, a source language parser, translator, target language morphological analyzer, target language parser and several lexicon dictionaries. Source language morphological analyzer analyzes a source language word and provides the morphological information. Source language parser is a syntax analyzer that analyzes source language sentences. Translator is used to translate a source language word into target language. Target language morphological analyzer works as a generator and it generates appropriate target language words for the given grammatical information. Also target language parser works as a composer and it composes a suitable target language sentence. Furthermore, this type of machine translation system needs minimum of three dictionaries namely the source language dictionary, the bilingual dictionary and the target language dictionary. Source language morphological analyzer needs a source language dictionary for morphological analysis. Bilingual dictionary is used by the translator for translating source language into target language; and the target language morphological generator uses the target language dictionary to generate target language words.

Figure 4.1 can present general architecture of the rule-based machine translation system. 13 A number of machine translation systems have been designed through the rule based approach. Among others Apertium is a rule-based Machine Translation system, which translates related languages. This is an open–source system that can be used to translate any related two languages. The Apertium engine follows a shallow transfer approach and consists of the eight pipelined modules, such as deformatter, A morphological analyzer, A parts-of-speech (PoS) tagger, A lexical transfer module, A structural transfer module, A morphological generator, A postgenerator, and A re-formatter.

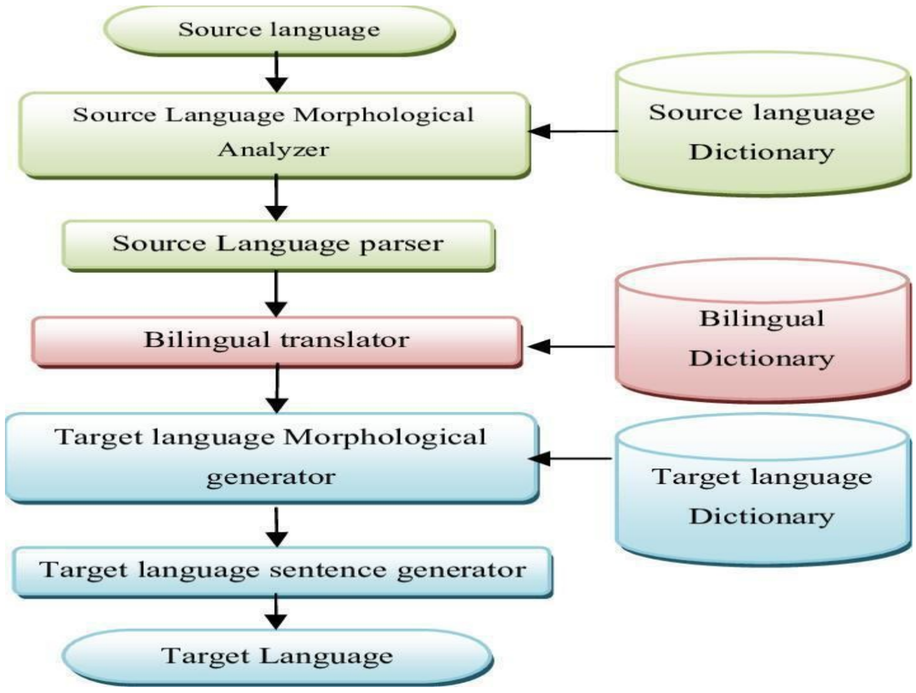

Source language
Source Language Morphological Analyzer
Source language Dictionary
Source Language parser
Bilingual translator
Bilingual Dictionary
Target language Morphological generator
Target language Dictionary
Target language sentence generator
Target Language

Figure 4.1: System Architecture

## 4.2 Data Flow Diagram

A data flow diagram is a flowchart can help to visualize the data pipeline of a system user can trace happens to the data as it moves between components. It is a great to find redundancies and optimize the speed and responsiveness of software. A DFD is often used As a preliminary step to create an overview of the system going into great detail, it can later be elaborated. DFDs are used for the visualization of data processing (structured design). A DFD show kind of information input to and output from the system, the data will advance through the system, and it the data will be stored.

It represented information of process timing processes will operate in sequence or in parallel, unlike a traditional structured flowchart which focuses on control flow, or a UML activity workflow diagram, which presents both control and data, flows as a unified model. A data flow diagram can dive into progressively more detail by using levels and layers, zeroing in on a particular piece. DFD levels are numbered 0 or 1, and occasionally go to even Level 3 or beyond. The necessary level of detail depends on the scope of what you are trying to accomplish

## DFD Level-0 Diagram

DFD Level 0 is also called a Context Diagram. It is a basic overview of the whole system or process being analyzed or modeled. It is designed to be an at a glance view, showing the system as a single high-level process, with its relationship to external entities. It easily understood a wide audience, including stakeholders, business analysts, data analysts and developers. In the Figure 4.2 the data flow diagram level 0 is described.

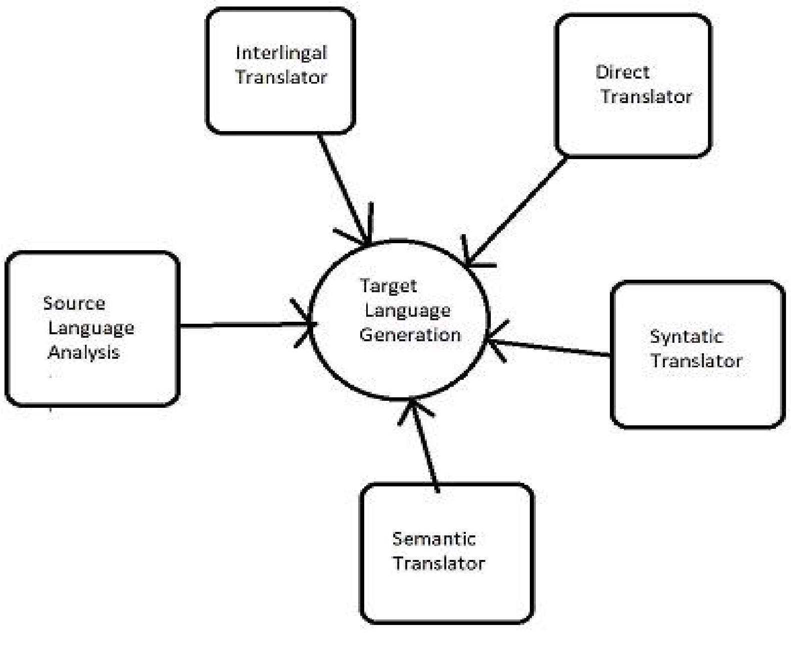


Figure 4.2: DFD Level-0 Diagram

## DFD Level-1 Diagram

DFD Level 1 provides a more detailed breakout of pieces of the Context Level Diagram. It highlight the main functions carried out by the system, it break down the high - level process of the Context Diagram into its sub processes. In the Figure 4.3 the data flow diagram level

1 is described

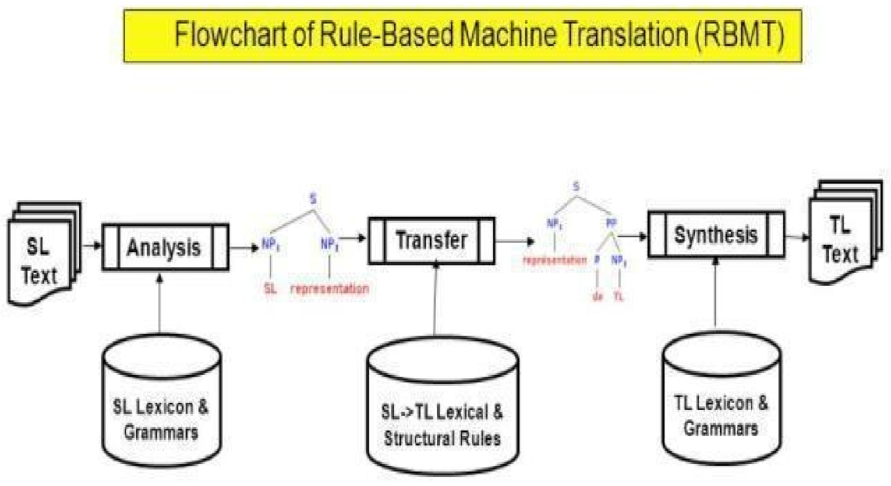


Figure 4.3: DFD Level-1 Diagram

## DFD Level-2 Diagram

DFD Level 2 then goes one step deeper into parts of Level 1. It may require more text to reach the necessary level of detail the system is functioning. In the Figure 4.4 and Figure 4.5 deep information of level 1 function.

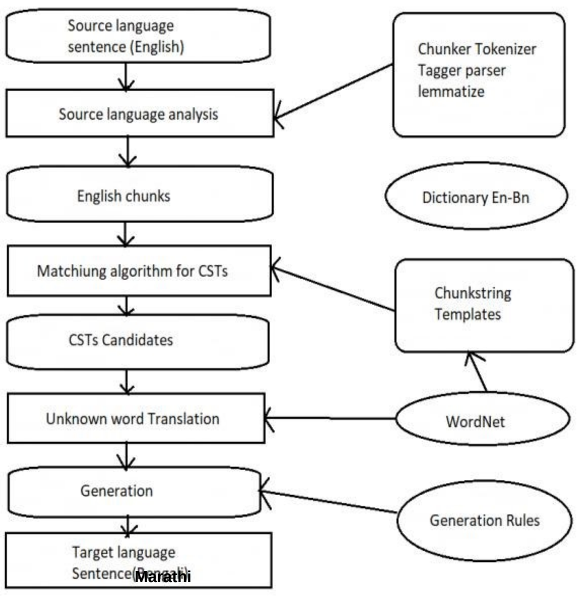


Figure 4.4: DFD Level-2 Diagram

## UML Diagrams

Unified Modeling language (UML) is a standardized modeling language enabling developers to specify, visualize, construct and document artifacts of a software system. Thus, UML makes these artifacts scalable, secure and robust in execution. UML is an important aspect involved in object-oriented software development. It uses graphic notation to create visual models of software systems.

### UseCase Diagram

A Use Case diagram shows the interaction between the system and entities external to the system. These entities are called actors which have specific role in the system. The figure shows the use case diagram for proposed system. Purpose of Use Case Diagram is to know or show functionality of the system.

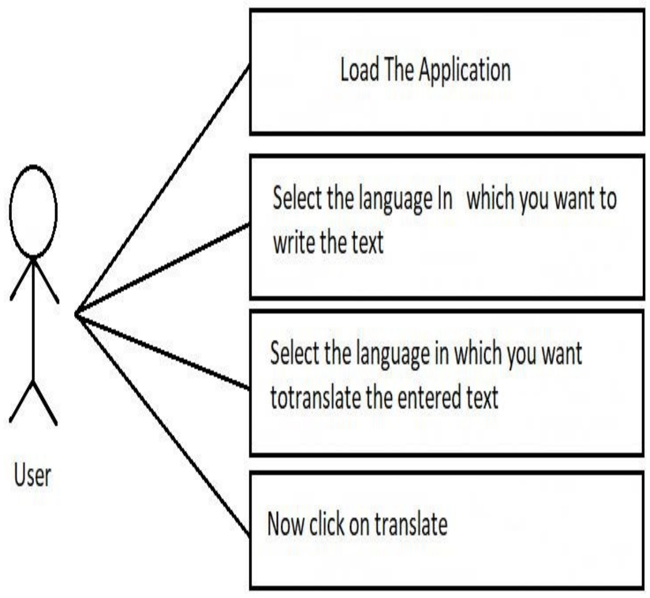

Load The Application
Select the language In which you want to write the text
Select the language in which you want totranslate the entered text
Now click on translate
User

Figure 4.5: UseCase Diagram

### Sequence Diagram

A sequence diagram simply depicts interaction between objects in a sequential manner. Purpose of Sequence Diagram is to show flow of functionality. A sequence diagram generally consists of objects, messages exchanged between objects lifeline of each object and focus of control for each message.

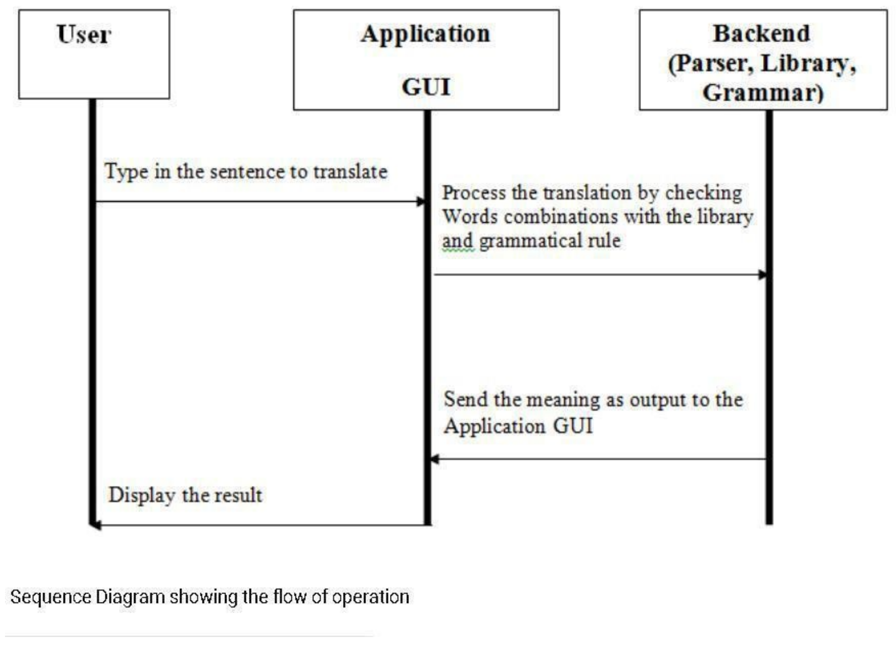

User
Application GUI
Backend (Parser, Library, Grammar)
Type in the sentence to translate
Process the translation by checking Words combinations with the library and grammatical rule
Send the meaning as output to the Application GUI
Display the result
Sequence Diagram showing the flow of operation

Figure 4.6: Sequence Diagram

### 4.2.1 Class Diagram

A Class diagram is used to represent the static view of the system. It mainly use classes, interfaces and their relationships. The Figure shows the class diagram for proposed system. Purpose of Class Diagram is to show structural aspects of the system

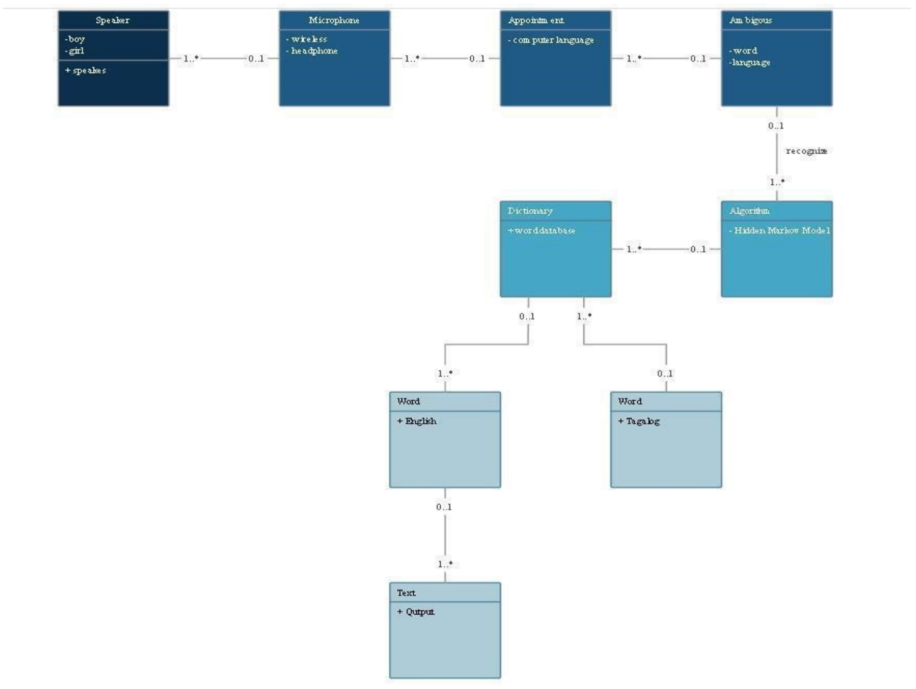


Figure 4.7: Class Diagram

### 4.2.2 State Diagram

A state diagram is used to represent the behavioral details of a particular software system. The behavior specifies the sequence of states an object within the system goes through during its lifetime. Purpose of state diagram is to show the flow of control from state to state.

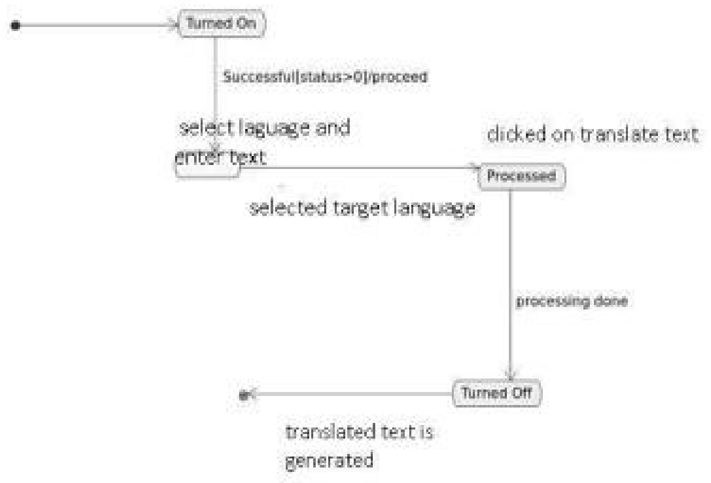

Turned On
Successful[status>0]/proceed
select laguage and enter text
clicked on translate text
Processed
selected target language
processing done
Turned Off
translated text is generated

Figure 4.8: State Diagram

### 4.2.3 Component diagram

A component diagram is generally used to show a set of components and their relationships. Graphically a component diagram is a collection of vertices and arcs where components are the vertices and the relationships form the arcs. Particularly components are connected with the help of dependency relationship that shows the dependencies among various components.

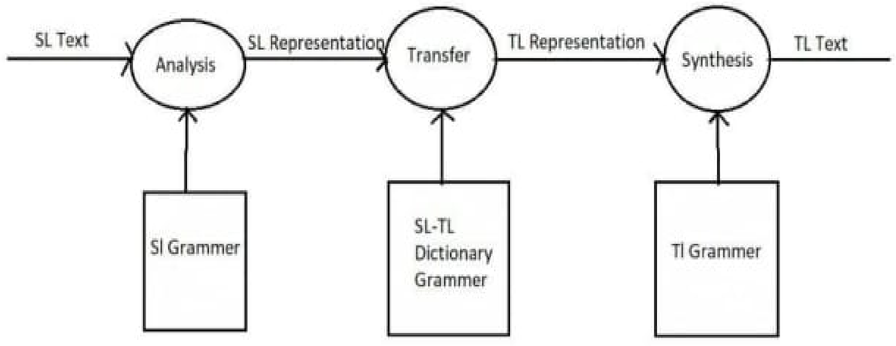


Figure 4.9: Component Diagram

### 4.2.4 Deployment Diagram

A deployment diagram shows the configuration of the run-time processing nodes and the components that reside on them. Deployment diagram addresses the static deployment view of a system. It is related to component diagram where a node typically encloses one or more components.

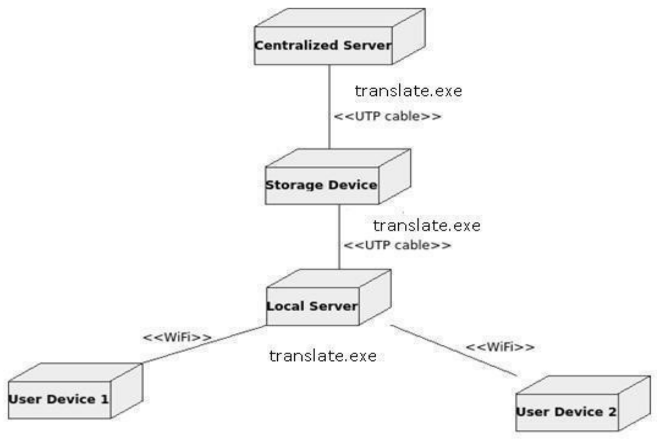


Figure 4.10: Deployement Diagram

### 4.2.5 Activity Diagram

An activity diagram shows the basic activities between two intermediate states of a state chart diagram. Activity diagram shows the flow from activity to activity. An activity is an ongoing non-atomic execution with a state machine. Activity ultimately results in some action which is made up of executable atomic computation that results in a change in state of system or the return of value. The activity diagram is a collection of vertices and arcs and forking, joining operations.

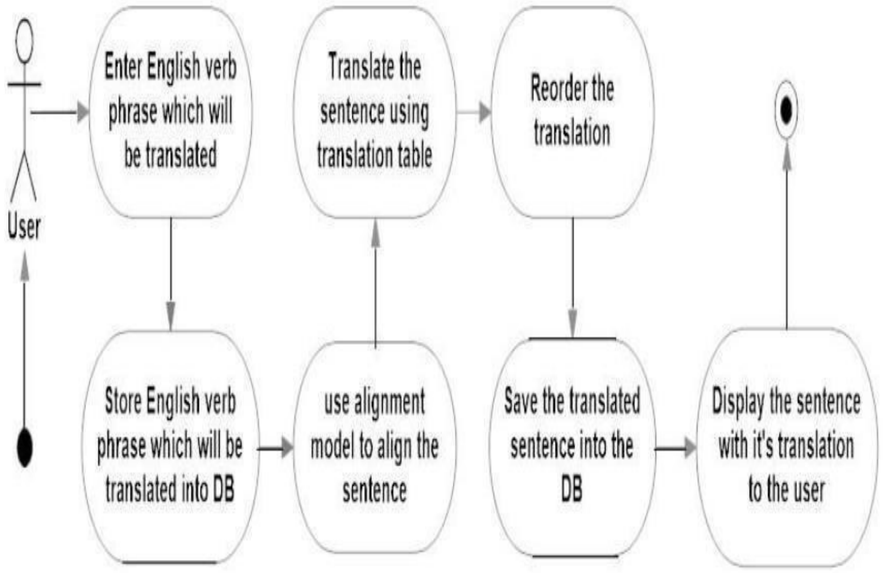

User
Enter English verb phrase which will be translated
Translate the sentence using translation table
Reorder the translation
Store English verb phrase which will be translated into DB
use alignment model to align the sentence
Save the translated sentence into the DB
Display the sentence with it's translation to the user

Figure 4.11: Activity Diagram

## 4.3 Summary

In the chapter, design and architecture of Rule-Based approach for Language Translation Application is described. In next chapter conclusion of report is mentioned.

# Chapter 5

# Coding And Implementation

The chapter includes coding and implementation of spam words detection using machine learning algorithms. Provides information about coding and implementation part of system.

In section 5.1 describes Algorithm/Steps. Requirements Collection and Identification are described in section 5.2. In section 5.3 describes the Hardware and Software Requirements. Modules in Project are described in Section 5.4. In section 5.5 Summary is described

## 5.1 Algorithms/Steps

The Steps section of the project describe the algorithms that are used in the project for classification and the steps required for the overall implementation of the design.

### 5.1.1 Algorithms Used

In the following chapter the algorithm used for development of project are discussed. Neural Machine Translation Algorithm

The algorithm can be given as:

**Input**Input: Text, Voice, Source Language.

**Output**Output: Text, Sound, Target Language.

**For One Language to another Language:**For One Language to another Language: Source language morphological analyzer detects the source language.

- Text is broken down into small units using source language parser.
- The text is translated by a bilingual translator into a simple form.
- Target Language is generated using Target language morphological .
- The is formed into a proper sentence using Target language generator.

### 5.1.2 Steps of Implementation

Steps are as follows:

- First select the source language in which user is entering the text.
- Enter the text which is to be translated. The user can also use the voice assistance for input.
- Select the language in which the user wants to translate the text i.e. Target Language.
- Click on the" Translate" button.
- The translated text will be displayed on the screen and the user can also hear the pronunciation of the translated text.

## 5.2 Required Software and Hardware for development

The section deals with the type of hardware that is used and the software to support the same. Selection and identification of suitable software is also taken into account.

### 5.2.1 Software Requirements

Software Requirements defines a requirement is requires for development of the proposed system. The description of software is required for developing the system. It is written for developers and users both because developer need the requirement for develop the system and user need it requirement for use on the client side. By using software requirement knows the nature of system in environment it run or executed. Software Requirements for system is given as follows:

1. **Operating System**Operating System: Android 7.0 Nougat or above.
2. **Front End** Front End: Android Application.
3. **Back End** Back End: Java, Firebase.

### 5.2.2 Hardware Requirements

Hardware Requirements defines a requirement is requires for development of the proposed system. The requirement of hardware require for computer system for execution. It is use for speed, efficiency, quality, time and storage. The hardware requirement is minimum for the system. Hardware requirements for system is given as follows:

1. Qualcomm Snapdragon 625 or later or Exynos 7904 or later
2. GPU @ Mali G72 or later
3. 4 GB RAM

## 5.3 Modules in Project

In the proposed work, the overall functionality of the system can be divided into steps and overview of the workflow of approach.

**Creating a textual interface for input as well as output:**The interface helps to provide input to the system and shows output in the textual form.

**Voice Assistance:**The voice assistance helps to provide the input to the system without using the textual interface.

**Text to Speech:**This Module is used to convert the output text into speech form which helps the pronunciation of the target language.

## 5.4 Summary

In this chapter, the implementation details, implementation environment are described. The next chapter, System Testing, is discussed.

# Chapter 6

# Testing

Testing is the process to prove that the software works correctly. It is used to test whether a particular software satisfies most of the possible conditions. Test cases are designed for the purpose that include certain conditions based on boundary values, module functions, etc. that are unfavorable for the software and it is observed whether all the test cases are passed by the software or not.

Section 6.1 describes white box testing. While the details about black box testing are given in the section 6.2. The section 6.3 of the chapter tells about manual testing. The details about automated testing are provided in section 6.4.

## 6.1 White Box Testing

White Box Testing

White box testing, sometimes called as glass-box testing is a test case design method that uses the control structure of the procedural design to derive test cases. Using white box testing methods, the software engineers can derive test cases that:

1. Guarantee all independent paths within a module have been exercised at least once.

2. Exercise all logical decisions on their true and false sides.

3. Exercise all loops at their boundaries and within their operational bounds.

4. Exercise internal data structures to ensure their validity.

## 6.2 Black Box Testing

Black Box Testing Black box testing also called as behavioral testing focuses on the functional requirements of the software. That is, black box testing enables the software engineer to derive sets of Input conditions that will fully exercise all functional requirements for a program. Black box testing is not an alternative to white box techniques. Rather, it is a complementary approach that is to uncover a different class of errors than white box methods. Black box testing attempts to find errors in the following categories:

1. Incorrect or missing functions.
2. Interface errors.
3. Errors in data structures or external data base access.
4. Behavior or performance errors.
5. Initialization and termination errors.

## 6.3 Manual Testing

Manual Testing Manual testing is testing of the software where tests are executed manually by a QA analyst. It is performed to discover bugs in software under development. In manual testing, the tester checks all the essential features of the given application or software. In the process, the software testers execute the test cases and generate the test reports without the help of any automation software testing tools. It is a classical method of all testing types and helps find bugs in software systems. It is generally conducted by an experienced tester to accomplish the software testing process.

1. The initial investment in the manual testing is comparatively lower.
2. Manual testing is not as accurate because of the possibility of the human errors.
3. No need for programming in Manual Testing.
4. Manual testing proves useful when the test case only needs to run once Or twice.
5. While testing a small change, an automation test would require coding which could be time-consuming. While you could test manually on the fly.

## 6.4 Automated Testing

Automated Testing In Automated Software Testing, testers write code/test scripts to automate test execution. Testers use appropriate automation tools to develop the test scripts and validate the soft-ware. The goal is to complete test execution in a less amount of time. Automated testing entirely relies on the pre-scripted test which runs automatically to compare actual result

With the expected results. The helps the tester to determine whether or not an application performs as expected. Automated testing allows you to execute repetitive task and regres-sion test without the intervention of manual tester. Even though all processes are performed automatically, automation requires some manual effort to create initial testing scripts.

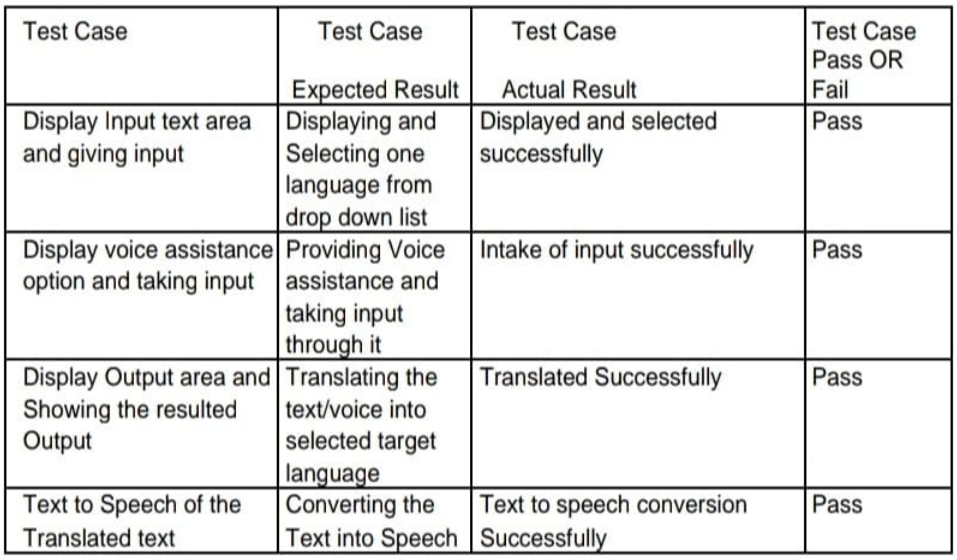
1. The initial investment in the automated testing is higher
2. Automated testing is a reliable method, as it is performed by tools and scripts There is no testing fatigue.
3. Programming knowledge is a must in automation testing.
4. Automation testing is useful when frequently executing the same set of test cases.
5. Testing coverage can be increased because automation testing tool never forgets to check even the smallest unit.

## 6.5 Test Cases Identification and Execution

Test Cases Identification and Execution

Test case is the set of inputs along with the output and some additional information like

1. Test Case ID Number identifying the test cases.
2. Test Case Name of the test case.
3. Test Case Description Details/purpose of the test case.
4. Steps Sequence of steps that the tester must follow to perform test cases.
5. Test Data Input given to the function.
6. Expected Results what the tester should see when test case is executed.
7. Actual result what the tester actually witnesses when test case is executed.
8. Test Result (P/F) indicates whether the test case passed, failed, etc. (Figure6.1)



| Test Case | Test Case<br>Expected Result | Test Case<br>Actual Result | Test Case Pass OR Fail |
|---|---|---|---|
| Display Input text area and giving input | Displaying and Selecting one language from drop down list | Displayed and selected successfully | Pass |
| Display voice assistance option and taking input | Providing Voice assistance and taking input through it | Intake of input successfully | Pass |
| Display Output area and Showing the resulted Output | Translating the text/voice into selected target language | Translated Successfully | Pass |
| Text to Speech of the Translated text | Converting the Text into Speech | Text to speech conversion Successfully | Pass |

Figure 6.1: Test Cases

# Chapter 7

# Result And Discussion

The chapter includes the Result and Discussion about Location based augmented reality system using SLAM and Point of Interest algorithms.

In section 7.1 describes Result. Discussion is described in section 7.2. In section 7.3 Summary is described

## 7.1 Result

The section shows the result in the form of screenshots (step by step working of system).

- Step 1: Initial Layout of the System :

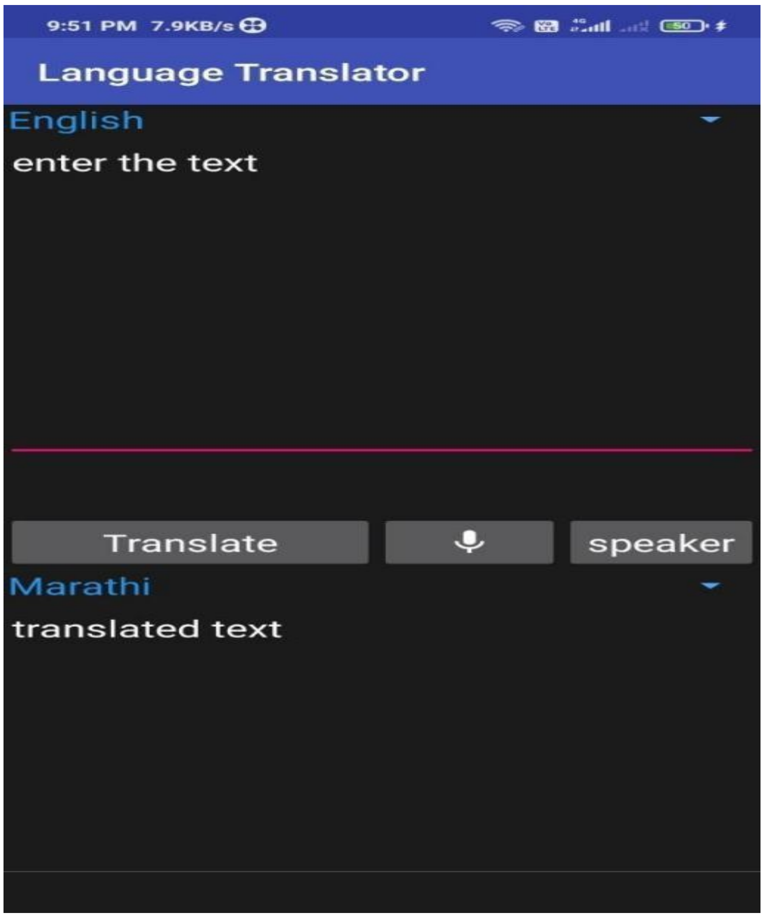


Figure 7.1: Layout of the System

- Step 2: Entering the Text in the first box after selecting both Source and Target Languages :

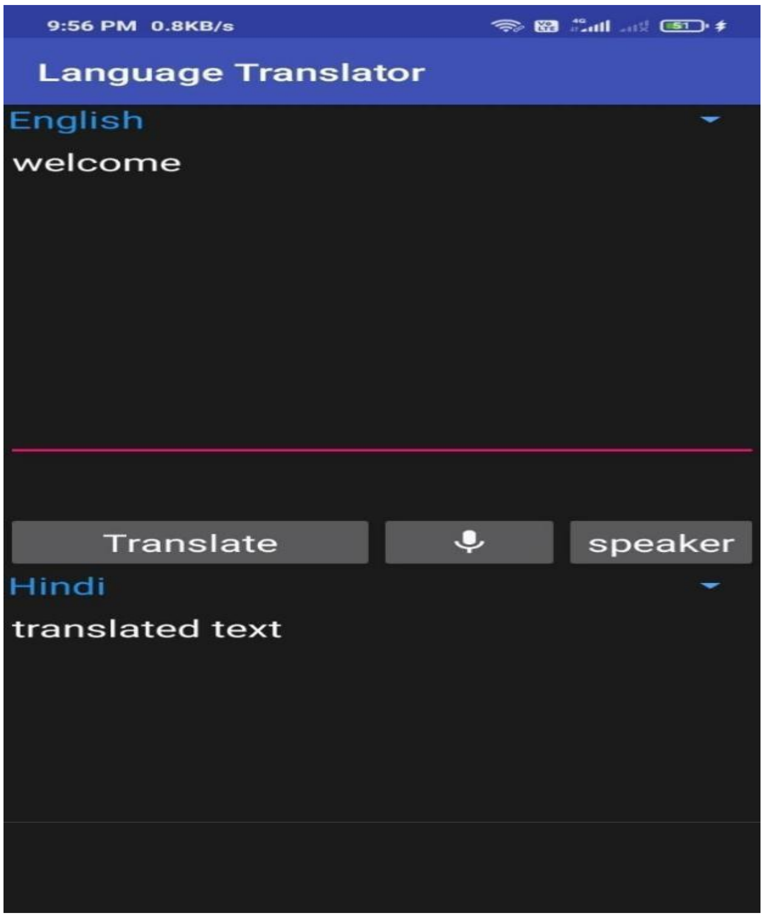
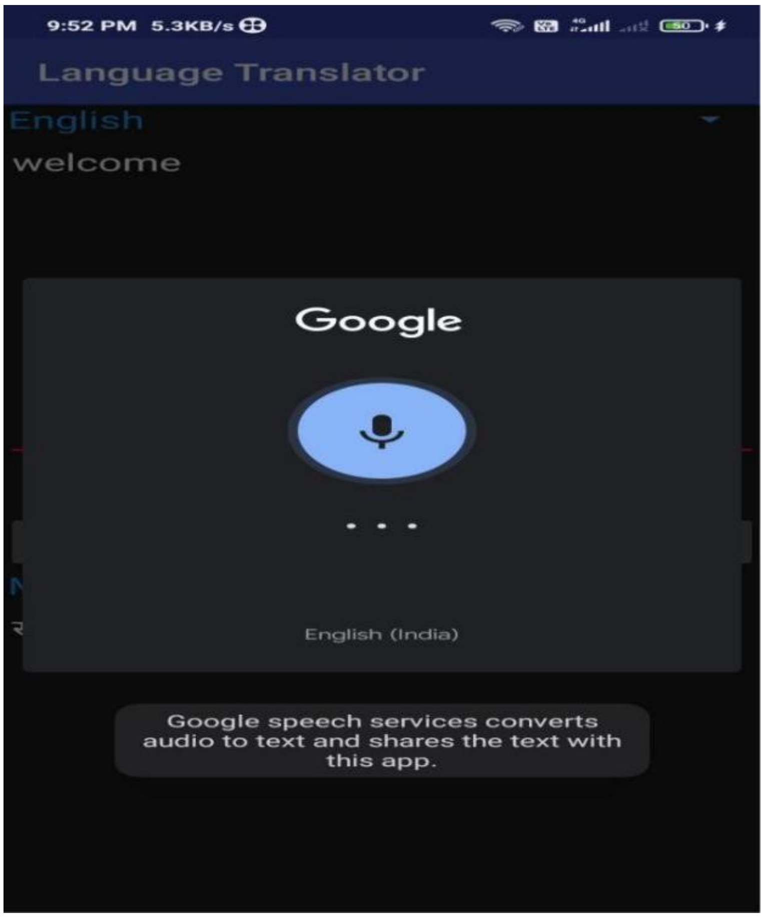


Figure 7.2: Selecting languages

OR

- Step 2:Taking the input through Voice Assistance:



Google speech services converts audio to text and shares the text with this app.

Figure 7.3: Voice Assistance

- Step 3: Translation Takes Place:

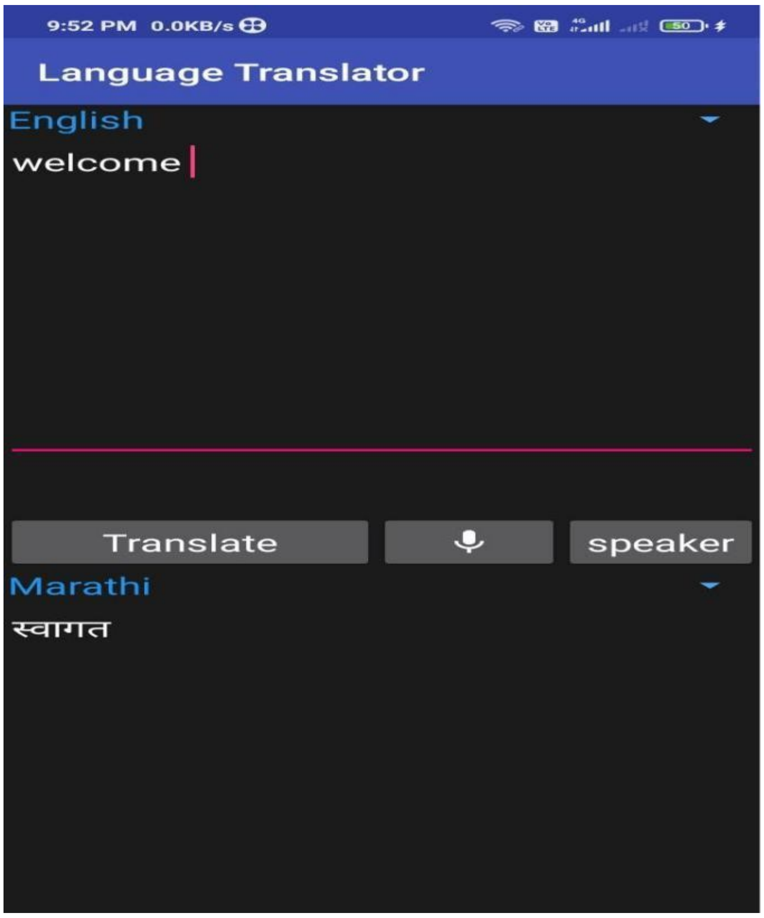


Figure 7.4: Translated Text

## 7.2 Discussion

In the following section discussion about the project is given.

1. Advantages

   (a) Bridging the Linguistic Barriers between Countries.

   (b) Improved Communication and Exchange of Ideas.

   (c) Building Strong Relationships.

2. Disadvantages

   (a) Inconsistency in the Level of Accuracy

   (b) Costly Mistakes

3. Application

   (a) Traveling to other countries

(b) Communicating with friends.

(c) Talking with prospective clients.

(d) IT Industry

(e) Tourism

(f) Participating in global meetings.

(g) Education.

## 7.3 Summary

The chapter explains Result and Discussion of the system. And discussed the Advantages and Disadvantages. The next chapter describes the Conclusion and Future Scope.

# Chapter 8

# Conclusion And Future Work

## 8.1 Conclusion

Each machine translation approach has its advantages and disadvantages. What one approach possesses, the other one seems to be lacking and vice versa. Rule-based methods focus on trying to understand the grammar rules while the statistical approach pays very minimal or no attention to the grammar of a particular language. The rule-based machine translation approach has been implemented in computational linguistics since its very early days. Human involvement in this approach is significant as it is the human agent who creates the rules. In other words, the humans use their knowledge and experience to prepare the rules.

## 8.2 Future Work

Machine Translation can increase translators' capabilities by 3-5 times in some cases, allowing for more content to be localized in a shorter amount of time. With increased productivity and reduced costs, companies will be able to translate more content into more languages.

Machine translation technology and products have been used in many application scenarios, such as business travel, tourism, cross-language information retrieval and so on. In terms of the object of translation, there are written language-oriented text translation and spoken language-oriented phonetic translation.

# Bibliography

1. Koehn, Philipp (2010). Statistical Machine Translation. Cambridge: Cambridge University Press. P.15. ISBN 9780521874151.

2. Nirenburg, Sergei (1989). "Knowledge-Based Machine Translation". Machine Trandation 4(1989), 5 - 24. Kluwer Academic Publishers. 4 (1): 5–24. JSTOR 40008396.

3. Hettige, B.; Karunananda, A.S. (2011). "Computational Model of Grammar for English to Sinhala Machine Translation". 2011 International Conference on Advances in ICT for Emerging Regions (ICTer). The International Conference on Advances in ICT for Emerging Regions - ICTer20 11 : 02631. pp. 26–31. doi:10.1109/ICTer.2011.6075022. ISBN 978-1-4577-1114-5. S2CID 45871137.

4. Lonsdale, Deryle; Mitamura, Teruko; Nyberg, Eric (1995). "Acquisition of Large Lexicons for Practical Knowledge-Based MT". Machine Translation 9: 251-283. Kluwer Academic Publishers. 9 (3–4): 251–283. doi:10.1007/BF00980580. S2CID 1106335.

5. . Lagarda, A.-L.; Alabau, V.; Casacuberta, F.; Silva, R.; Díaz-de-Liaño, E. (2009). "Statistical Post-Editing of a Rule-Based Machine Translation System" (PDF). Proceedings of NAACL HLT 2009: Short Papers, pages 217–220, Boulder, Colorado. Association for Computational Linguistics. Retrieved 20 June 012.