



*Mini Project Report On*

## **Malayalam Parser for Dataset Creation**

*Submitted in partial fulfillment of the requirements for the  
award of the degree of*

## **Bachelor of Technology**

*in*

## ***Computer Science & Engineering***

**By**

**Fathima Jennath N.K (U2103089)**

**Gautham C Sudheer (U2103092)**

**Godwin Gino (U2103096)**

**Mohammed Basil (U2103139)**

**Under the guidance of**

**Dr.Mary Priya Sebastian**

**Department of Computer Science & Engineering  
Rajagiri School of Engineering & Technology (Autonomous)  
(Affiliated to APJ Abdul Kalam Technological University)**

**Rajagiri Valley, Kakkanad, Kochi, 682039**

**May 2024**

# CERTIFICATE

*This is to certify that the mini project report entitled "**Malayalam Parser for Dataset Creation**" is a bonafide record of the work done by **Fathima Jennath N.K (U2103089)**, **Gautham C Sudheer (U2103092)**, **Godwin Gino (U2103096)**, **Mohammed Basil (U2103139)**, submitted to the APJ Abdul Kalam Technological University in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology (B. Tech.) in Computer Science and Engineering during the academic year 2023-2024.*

Dr.Mary Priya Sebastian  
Associate Professor  
Dept. of CSE  
RSET

Dr.Saritha S  
Professor  
Dept. of CSE  
RSET

Dr.Preetha K.G  
Head of the Department  
Dept. of CSE  
RSET

# ACKNOWLEDGEMENTS

I wish to express my sincere gratitude towards Dr P. S. Sreejith, Principal of RSET, and Dr. Preetha K.G., Head of the Department of Computer Science and Engineering for providing me with the opportunity to undertake my mini project, "Malayalam Parser for Dataset Creation".

I am highly indebted to my project coordinator, **Dr.Saritha S**, Professor, Department of Computer Science and Engineering for their valuable support.

It is indeed my pleasure and a moment of satisfaction for me to express my sincere gratitude to my project guide **Dr.Mary Priya Sebastian** for her patience and all the priceless advice and wisdom she has shared with me.

Last but not the least, I would like to express my sincere gratitude towards all other teachers and friends for their continuous support and constructive ideas.

**Fathima Jennath NK**

**Gautham C Sudheer**

**Godwin Gino**

**Mohammed Basil**

# Abstract

The “Malayalam Parser for Dataset Creation” project aims to address the scarcity of annotated datasets in the Malayalam language for Natural Language Processing (NLP) applications. The primary objective is to develop a robust Malayalam parser capable of analyzing the syntactic and semantic structures of Malayalam sentences. The creation of this parser involves several key steps, including data collection from diverse sources, preprocessing to ensure data quality, and manual annotation of a representative subset of the data with grammatical and syntactic information. The parser development process encompasses the selection of an appropriate parsing approach, whether rule-based, statistical, or machine learning-based. The model is trained using the annotated Malayalam dataset, focusing on capturing the unique linguistic nuances of the Malayalam language. Evaluation metrics are employed to assess the parser’s performance on a separate test set, guiding iterative refinement and enhancement. The resulting Malayalam parser serves as a valuable tool for the analysis of grammatical structures in new Malayalam text data. Its application contributes to the creation of high-quality Malayalam datasets, crucial for advancing NLP research and applications in the Malayalam language. This project encourages collaboration with linguists, researchers, and the Malayalam-speaking community to ensure linguistic accuracy and relevance in the development of the parser. The “Malayalam Parser for Dataset Creation” project aligns with the broader goal of promoting linguistic diversity in NLP, addressing the challenges posed by the scarcity of resources for underrepresented languages. Through the development of this parser, the project aims to facilitate further research and innovation in Malayalam NLP, opening avenues for the exploration of various language-related tasks and applications.

# Contents

<b>Acknowledgements</b>	<b>i</b>
<b>Abstract</b>	<b>ii</b>
<b>List of Figures</b>	<b>v</b>
<b>List of Tables</b>	<b>vi</b>
<b>List of Abbreviations</b>	<b>vii</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Background . . . . .	1
1.2 Problem Definition . . . . .	1
1.3 Scope and Motivation . . . . .	1
1.4 Objectives . . . . .	2
1.5 Challenges . . . . .	3
1.6 Assumptions . . . . .	4
1.7 Societal / Industrial Relevance . . . . .	4
1.8 Organization of the Report . . . . .	5
<b>2 Software Requirements Specification</b>	<b>6</b>
2.1 Introduction . . . . .	6
2.1.1 Purpose . . . . .	6
2.1.2 Product Scope . . . . .	7
2.2 Overall Description . . . . .	7
2.2.1 Product Perspective . . . . .	7
2.2.2 Product Functions . . . . .	8
2.2.3 Operating Environment . . . . .	8
2.2.4 Design and Implementation Constraints . . . . .	9

2.2.5	Assumptions and Dependencies . . . . .	10
2.3	External Interface Requirements . . . . .	11
2.3.1	User Interfaces . . . . .	11
2.3.2	Hardware Interfaces . . . . .	11
2.3.3	Software Interfaces . . . . .	12
2.3.4	Communication Interfaces . . . . .	12
2.4	System Features . . . . .	12
2.5	Other Nonfunctional Requirements . . . . .	13
2.5.1	Performance Requirements . . . . .	13
2.5.2	Safety Requirements . . . . .	14
2.5.3	Security Requirements . . . . .	15
2.5.4	Software Quality Attributes . . . . .	15
<b>3</b>	<b>System Architecture and Design</b>	<b>16</b>
3.1	System Overview . . . . .	16
3.2	Dataset identified . . . . .	17
3.3	Proposed Methodology/Algorithms . . . . .	17
3.4	User Interface Design . . . . .	17
3.5	Database Design . . . . .	18
3.6	Description of Implementation Strategies . . . . .	18
3.7	Module Division . . . . .	19
3.8	Work Schedule - Gantt Chart . . . . .	20

## List of Figures

3.1	Architecture diagram . . . . .	17
3.2	Work Schedule-Gantt chart . . . . .	20

# List of Tables

1.1	Description of parsing tasks . . . . .	2
-----	--	---



## List of Abbreviations

- NLP - Natural Language Processing
- NER - Named Entity Recognition
- POS - Part-of-Speech

# Chapter 1

## Introduction

### 1.1 Background

Malayalam is spoken widely in Kerala and neighboring areas but has not received as much attention in the tech world as bigger languages like English. This lack of attention has led to a scarcity of tools for analyzing Malayalam text, despite its complex grammar and word forms.

Our project aims to address this issue by creating a specialized system for Malayalam that can understand and process Malayalam text more effectively than current tools. This system will facilitate tasks such as sentiment analysis, translation, and summarization, benefiting areas such as education and business.

We are designing our system to be adaptable and scalable, ensuring that it can evolve to meet diverse needs. Our ultimate goal is to establish a strong foundation for Malayalam language technology, paving the way for future improvements and innovations.

In short, our project focuses on using technology to make working with Malayalam text more efficient, enabling individuals to achieve more in various fields.

### 1.2 Problem Definition

The aim of the project is to create a Malayalam Parser for Dataset Creation, involving data collection, preprocessing, manual annotation, and training using various parsing approaches to address the scarcity of annotated datasets in Malayalam for NLP applications.

### 1.3 Scope and Motivation

#### **Scope:**

The Malayalam Parser project aims to develop an advanced tool capable of understanding

and processing Malayalam text efficiently. It encompasses essential tasks such as tokenization, part-of-speech tagging, parsing, and semantic analysis, providing a comprehensive breakdown of Malayalam sentences. The system's scope extends to facilitating tasks such as identifying different parts of speech and extracting meaningful insights from text. Additionally, the parser's design includes provisions for future expansion, allowing for the incorporation of domain-specific or specialized parsing tasks. This flexibility ensures that the parser can adapt to evolving needs and requirements, making it applicable across various domains and applications.

### **Motivation:**

The motivation behind the Malayalam Parser project stems from the necessity for effective tools to process Malayalam text, essential for informed decision-making and strategic planning. By implementing parsing tasks such as tokenization, part-of-speech tagging, parsing, and semantic analysis, the system enables data-driven decision-making processes, enhancing strategic planning and execution. Furthermore, the project's innovation lies in the creation of annotated datasets, crucial for training and evaluating models for sentiment analysis, machine translation, question answering, and domain-specific parsing. Through these efforts, the project aims to advance natural language processing capabilities in Malayalam, contributing to the improvement of Malayalam language processing technologies and fostering innovation in linguistic research and technology development.

Parsing Tasks	Description
Tokenization	Breaking down sentences into individual words or tokens.
Part-of-Speech Tagging	Assigning grammatical tags to each word in a sentence.
Parsing	Analyzing the structure of sentences to understand their grammatical relationships.
Semantic Analysis	Extracting the meaning and context from sentences.

Table 1.1: Description of parsing tasks

## **1.4 Objectives**

- Develop parsing algorithms to accurately identify linguistic components such as words, phrases, and sentences in Malayalam text, laying the foundation for compre-

hensive analysis.

- Implement functionalities to determine the grammatical structure, syntax, and semantics of Malayalam sentences, facilitating precise linguistic analysis and comprehension.
- Incorporate part-of-speech tagging, syntactic parsing, and semantic analysis capabilities tailored specifically for the Malayalam language, enabling detailed linguistic processing.
- Enhance the Malayalam Parser system to effectively handle compound words, inflections, and variations in word forms commonly encountered in Malayalam text, ensuring robust parsing capabilities.
- Ensure compatibility and interoperability with existing linguistic analysis frameworks, facilitating seamless integration and utilization of the Malayalam Parser within broader NLP applications.
- Continuously refine and optimize the Malayalam Parser system to improve efficiency, accuracy, and adaptability in analyzing and processing Malayalam text data.

## 1.5 Challenges

1. **Morphological Complexity:** Malayalam words can change a lot by adding suffixes, making it hard for the parser to figure out their basic forms and parts of speech. This makes it tough to understand the meaning and grammar of sentences.
2. **Limited Resources:** The scarcity of Malayalam-specific NLP tools and datasets complicates the development and training process. Adapting existing tools from other languages or creating new ones becomes necessary, potentially slowing down progress and hindering the effectiveness of the parser.
3. **Syntactic Freedom:** Malayalam's relatively flexible sentence structure allows for varied word orders, challenging parsing algorithms in determining precise word relationships. This freedom introduces complexity in identifying grammatical elements like subjects, objects, and verbs, especially when word order isn't a definitive indicator.

4. **Data Annotation:** The process of manually annotating data for parts of speech (POS), named entities, and intents is meticulous and time-consuming, requiring expertise in Malayalam grammar. It's crucial to ensure the quality and comprehensiveness of annotated training data for the parser's success, although this can be resource-intensive.
5. **Model Selection and Training:** Optimal performance of POS tagging, named entity recognition, and intent recognition tasks relies on choosing suitable algorithms and training them effectively. However, training on potentially limited or imbalanced datasets necessitates careful optimization, such as data augmentation and hyperparameter tuning, to mitigate any shortcomings.

## **1.6 Assumptions**

1. **Availability of Training Data:** The project assumes a certain amount of high-quality Malayalam text data will be available for annotation and training the parser for POS tagging, NER, and intent recognition.
2. **Effectiveness of Algorithms:** The project assumes that the selected algorithms for POS tagging, NER, and intent recognition will be capable of accurately handling the linguistic complexities inherent in the Malayalam language.
3. **Computational Resources:** Adequate computational resources are assumed to be accessible for training the parser models, as this process can be computationally demanding.
4. **Annotation Quality:** The project assumes the quality of the annotations in the training data, including accuracy and consistency, as these factors greatly influence the performance of the parser models.

## **1.7 Societal / Industrial Relevance**

The project aims to preserve and promote Malayalam, enhancing understanding and analysis of Malayalam text, providing datasets for parts-of-speech tagging, named entity recognition, and intent recognition. It supports research and learning in NLP and

Malayalam linguistics, improving access to information for Malayalam speakers online. The project also supports the development of local language technologies for industries in Malayalam-speaking regions, expanding market opportunities in e-commerce and social media.

## 1.8 Organization of the Report

The organization of the report are as follows:

- **Chapter 1-Introduction:**The introduction covers the background of the project, the problem definition, the scope and motivation, the objectives,the societal and industrial relevance, the assumptions and the challenges faced by the project.
- **Chapter 2-Software Requirements Specification:** This chapter outlines the functional and nonfunctional requirements of the NLP tools for Malayalam. It defines the overall description of the software, external interface requirements, system features, and other nonfunctional requirements necessary for the development and deployment of the tools.
- **Chapter 3-System Architecture and Design:** The system architecture and design chapter provides an overview of the project’s technical framework. It includes discussions on the system overview, architectural design, identified datasets, proposed algorithms, implementation strategies, module division, and a work schedule presented as a Gantt chart for project planning and management.

## Chapter 2

# Software Requirements Specification

### 2.1 Introduction

#### 2.1.1 Purpose

The system aims to enhance the understanding and analysis of Malayalam text data, enabling more accurate and efficient processing of information, including but not limited to retrieval, sentiment analysis, machine translation, and text summarization. By focusing on the unique linguistic characteristics of Malayalam, the project seeks to overcome the challenges posed by the language's morphology, syntax, and semantics. Through the development of innovative algorithms, data structures, and linguistic resources, the project aims to contribute to the advancement of NLP research and technology in the context of the Malayalam language.

- The primary focus of the Malayalam Parser system is to analyze and parse written text in the Malayalam language. This involves tasks such as tokenization, part-of-speech tagging, parsing, and semantic analysis.
- Initially, the system will target general-purpose parsing tasks, which are applicable to a wide range of text data. These tasks form the foundation of the parsing capabilities of the system.
- The system is designed to be extensible, allowing for the inclusion of domain-specific or specialized parsing tasks in the future. This flexibility ensures that the system can adapt to different needs and requirements over time.
- In addition to algorithms and data structures, the project also involves the development of linguistic resources that are crucial for parsing Malayalam text. These resources may include lexicons, grammars, and annotated corpora.

### 2.1.2 Product Scope

The Malayalam Parser is a smart tool that understands Malayalam text. It breaks down sentences to figure out their grammar, structure, and meaning using advanced language processing techniques. It helps with tasks like identifying different parts of speech, understanding how sentences are put together, and analyzing the overall meaning of text.

1. **Informed Decision-Making:** Implementation of parsing tasks including tokenization, part-of-speech tagging, parsing, and semantic analysis for Malayalam text. Extracting insights from Malayalam text aids in data-driven decision-making, enhancing strategic planning and execution.
2. **Innovation:** Developing a Malayalam parser for dataset creation is significant as it encompasses crucial linguistic tasks such as tokenization, part-of-speech tagging, and named entity recognition. This effort is essential for advancing natural language processing capabilities in Malayalam, enabling the development of robust models for sentiment analysis, machine translation, question answering, and domain-specific parsing. The creation of annotated datasets is vital to train and evaluate these models, contributing to the overall improvement of Malayalam language processing technologies.

## 2.2 Overall Description

### 2.2.1 Product Perspective

This software represents a new, standalone product developed to fulfill the crucial task of parsing and analyzing Malayalam language text. Unlike existing systems or components, this software is designed from the ground up to specialize in the intricacies of Malayalam language parsing, catering to the specific needs of users requiring accurate linguistic analysis in Malayalam. The Malayalam Parser software does not belong to a product family but serves as an independent solution, tailored specifically for parsing Malayalam text. It is not intended as a replacement for existing systems but rather as a complementary tool to enhance language processing capabilities in Malayalam. While it operates autonomously, it may interface with other systems or applications, such as text



input sources or linguistic databases, to facilitate seamless integration within broader linguistic analysis frameworks.

### **2.2.2 Product Functions**

Major Functions of the Malayalam Parser:

- Parse and analyze Malayalam language text to identify linguistic components such as words, phrases, and sentences.
- Determine grammatical structure, syntax, and semantics of Malayalam sentences to facilitate accurate linguistic analysis.
- Provide functionality for part-of-speech tagging, syntactic parsing, and semantic analysis tailored for the Malayalam language.
- Support for handling compound words, inflections, and variations in word forms commonly found in Malayalam text.
- Generation of a part-of-speech tagged dataset, named entity dataset, and sentimental tagged dataset, contributing to the advancement of language processing technologies in Malayalam

### **2.2.3 Operating Environment**

For creating the Malayalam Parser, the operating environment encompasses the following specifications:

#### **Hardware Platform**

- The software should be compatible with standard computing hardware commonly used for software development and deployment.
- Minimum hardware requirements include a modern processor (e.g., Intel Core i5 or equivalent), sufficient RAM (at least 4GB), and available storage space for software installation and data processing.

#### **Operating System and Versions**

- The software should be compatible with popular operating systems used for software development and deployment, including Windows, MacOS, and Linux distributions.
- Specific operating system versions supported include:  
Windows 10 or later  
MacOS 10.13 High Sierra or later  
Ubuntu 18.04 LTS or later

## Software Components and Applications

- The Malayalam Parser may require integration with various software components and applications for development, testing, and deployment purposes.
- Development tools such as Python (version 3.6 or later), Java Development Kit (JDK), or other programming languages and frameworks suitable for NLP development may be necessary.
- Libraries and frameworks for natural language processing, such as NLTK (Natural Language Toolkit), spacey or TensorFlow, may be utilized.
- Database systems such as SQLite, PostgreSQL, or MongoDB may be employed for data storage and retrieval if necessary.

### 2.2.4 Design and Implementation Constraints

The development of the Malayalam Parser is subject to various constraints and limitations, including:

- **Hardware Limitations:** The software must be optimized to operate within hardware limitations, including timing and memory constraints. Performance optimizations are crucial to ensure efficient operation across different hardware configuration.
- **Integration with External Systems:** The parser must support standard interfaces and data formats for interoperability with other language processing tools or databases. Ensuring smooth integration is a technical requirement.
- **Language Requirements:** The Malayalam Parser must be designed and implemented to meet the linguistic complexities of the Malayalam language. This includes

handling unique grammar rules, character encoding, and text processing requirements inherent to Malayalam.

- **Character Encoding and Localization:** Proper handling of character encoding specific to Malayalam and localization considerations are crucial. The parser should accurately interpret and process Malayalam characters.
- **Security Considerations:** Security measures such as encryption, access control, and data integrity must be incorporated into the design and implementation of the Malayalam Parser to safeguard against security threats and vulnerabilities.
- **Design Conventions and Programming Standards:** Adherence to design conventions, programming standards, and coding guidelines is paramount. Developers must follow established best practices and organizational standards to ensure maintainability, readability, and consistency of the codebase.

#### 2.2.5 Assumptions and Dependencies

1. **Third-Party or Commercial Components:** It is assumed that the development of the Malayalam Parser will rely on third-party or commercial components. The project is intended to be developed from existing models, without significant dependencies on pre-existing software solutions or proprietary libraries.
2. **Development Environment:** The assumption is made that the development environment for the Malayalam Parser will be properly configured and equipped with necessary tools, including programming languages, development frameworks, and version control systems.
3. **Operating Environment:** It is assumed that the operating environment for testing and deployment of the Malayalam Parser will meet the specified requirements, including compatibility with supported operating systems and hardware configurations.
4. **Constraints and Limitations:** The project assumes that constraints and limitations identified in the SRS, such as corporate policies, regulatory requirements,

and hardware limitations, will be properly addressed and accounted for during the development process.

5. **External Dependencies:** The project does anticipate reliance on external dependencies or reused software components from other projects. Any dependencies arising during development will be clearly documented and managed accordingly.
6. **Availability of Resources:** It is assumed that the necessary resources, including human resources, budget allocation, and development timelines, will be available and allocated appropriately to support the successful development and completion of the Malayalam Parser project.

## 2.3 External Interface Requirements

### 2.3.1 User Interfaces

The project has the potential to be utilized in various future applications, particularly in educational tools designed for teaching purposes. Additionally, it can serve as a foundation for developing apps focused on language learning, text analysis, and speech recognition. This versatility opens up opportunities for creating interactive and innovative tools that cater to different learning styles and linguistic analysis needs. By leveraging the project's capabilities, we can create user-friendly applications that enhance the learning and analysis experience for users. These applications can play a significant role in promoting the Malayalam language and its usage in various contexts.

### 2.3.2 Hardware Interfaces

The software interacts with the hardware components of the system through various interfaces, each with specific communication protocols. It communicates with the CPU to execute algorithms, process data, and manage computational tasks, utilizing the CPU's instruction set architecture. The software uses RAM for temporary storage of linguistic data, parsing results, and intermediate processing steps, managing memory efficiently for complex linguistic analyses. For long-term storage, the software reads and writes data to the storage device, employing file systems for dataset management and data integrity. Optionally, the software can access external datasets, online resources, or collaboration

tools through a network interface, using standard networking protocols such as HTTP or FTP for data exchange. User interaction and data visualization are facilitated through input/output devices such as keyboards, mouse, and monitors, utilizing standard interfaces for user input and output display.

### **2.3.3 Software Interfaces**

The Malayalam parser integrates with various software components, including databases such as SQLite, PostgreSQL, and MongoDB for data storage and retrieval. It is designed to run on popular operating systems like Windows 10 or later, MacOS 10.13 High Sierra or later, and Ubuntu 18.04 LTS or later. Development and NLP tasks are facilitated by Python (version 3.6 or later) along with libraries like NLTK, Spacy, and TensorFlow. These tools aid in implementing NLP algorithms and processing linguistic data. The system may also include integrated commercial components for specific functionalities, each with its own data formats, configurations, and communication protocols. To ensure consistent data sharing, the system may implement global data areas or standardized data formats, depending on the software components involved. Detailed application programming interface (API) protocols should be referenced for precise communication details.

### **2.3.4 Communication Interfaces**

The software requires communication functions for fetching inputs from websites, email, and network server communications. It needs to support standard email protocols like SMTP for sending emails, following MIME standards for message formatting. For web scraping, it should interact with web servers using HTTP or HTTPS for secure communication, and handle HTML parsing for extracting data from web pages. Network server communication may require protocols like FTP or SFTP for file transfers, adhering to communication standards for compatibility and security. Secure protocols such as SMTPS, SFTP, or TLS should be used for data transmission, with defined data transfer rates and synchronization mechanisms for efficiency.

## **2.4 System Features**

## **2.5 Other Nonfunctional Requirements**

### **2.5.1 Performance Requirements**

#### **Parsing Malayalam Text**

Rationale: Parsing text should be fast to provide timely analysis for user interactions or batch processing.

Requirement: The system should be able to parse Malayalam text at a rate of at least 1000 characters per second.

#### **Lemmatization and Morphological Analysis**

Rationale: Lemmatization and morphological analysis are core linguistic processes that should not introduce significant delays.

Requirement: The system should be able to perform lemmatization and morphological analysis for at least 100 words per second.

#### **Dependency Parsing**

Rationale: Dependency parsing is computationally intensive but crucial for understanding sentence structures.

Requirement: The system should be able to perform dependency parsing for at least 50 sentences per second.

#### **Syntax Tree Generation**

Rationale: Syntax tree generation requires complex computations and should be efficient for real-time applications.

Requirement: The system should be able to generate syntax trees for at least 20 sentences per second.

#### **Semantic Analysis**

Rationale: Semantic analysis involves processing text to understand its meaning, requiring significant computational resources. Efficient performance is crucial for real-time or near real-time applications, such as chatbots or text summarization systems.

Requirement: The system should be able to perform semantic analysis for a minimum of 1000 words per second, ensuring timely and responsive processing for various use cases.

### **Part-of-Speech tagging**

Rationale: Part-of-speech tagging is a fundamental task in natural language processing, providing essential grammatical information about words in a sentence. Efficient tagging is necessary for various downstream tasks such as parsing, machine translation, and information extraction.

Requirement: The system should be able to perform part-of-speech tagging for a minimum of 1000 words per second, ensuring fast and responsive tagging for different text lengths and complexities.

### **Performance Optimization**

Rationale: Performance optimization ensures that the system meets its performance requirements under varying loads and conditions.

Requirement: The system should implement optimization strategies to maintain performance levels even under heavy loads, ensuring that response times do not exceed specified thresholds.

#### **2.5.2 Safety Requirements**

Loss, Damage, or Harm Concerns: The product must ensure user and property safety, preventing physical harm or damage. It must also safeguard sensitive data from unauthorized access.

Actions to be Taken: The product should provide clear instructions and warnings to prevent misuse or accidental damage. Regular maintenance and updates are crucial for safe operation.

Actions to be Prevented: Unauthorized modifications and operations that could lead to data loss or corruption should be prevented.

External Policies or Regulations: Compliance with safety standards and regulations is essential, including guidelines from regulatory bodies and industry standards organizations.

Safety Certifications: Obtaining safety certifications demonstrates compliance with stan-

dards and builds user trust in the product’s safety.

### **2.5.3 Security Requirements**

The system must prioritize the security and privacy of the data used or created by the system. It should implement robust measures to prevent unauthorized access, disclosure, or modification of sensitive information, such as annotated dataset files. User identity authentication should be incorporated to verify the identity of users accessing the system, ensuring that only authorized users can access or modify the dataset files. The system must comply with external policies and regulations regarding security and privacy issues, adhering to guidelines set by regulatory bodies or industry standards organizations. Obtaining security and privacy certifications is essential to demonstrate compliance with these standards, reassuring users of the parser’s commitment to data security and privacy protection.

### **2.5.4 Software Quality Attributes**

The Malayalam parser for dataset creation must prioritize correctness and reliability, ensuring accurate parsing and consistent results. Maintainability is crucial for easy updates, while usability requires an intuitive interface. Portability enables the parser to run on different platforms, and interoperability allows integration with other NLP components. Testability ensures thorough testing, and adaptability allows for evolving requirements. These characteristics collectively ensure the parser’s efficiency, effectiveness, and adaptability.



## Chapter 3

### System Architecture and Design

#### 3.1 System Overview

The aim of the project to develop a Malayalam parser is to create a system that can effectively process Malayalam text for tasks such as sentiment analysis, named entity recognition, and part-of-speech (POS) tagging. The process begins with data collection, where Malayalam text is gathered from online sources using web scraping techniques. This collected text is then cleaned to remove any irrelevant information, errors, or special characters.

Language filtering is applied in the cleaning phase to ensure the text is in Malayalam. Following data cleaning, the text undergoes preprocessing, which includes tokenization to break the text into individual words or tokens. Stemming is then applied to reduce inflected words to their base form, simplifying the text for analysis. A crucial step is building a training corpus of preprocessed Malayalam text labeled with the desired information, such as sentiment labels, named entities, and part-of-speech tags. Features are extracted from the preprocessed text and serve as inputs to the machine learning model. A suitable machine learning model is selected and trained on the extracted features and labeled training data. The model's performance is evaluated on a separate set of data to assess its accuracy and effectiveness on new, unseen data. Finally, the trained model is used to process new Malayalam text, tagging it with sentiment labels, named entities, part-of-speech tags, or other relevant information. Linguists can add custom rules to refine the model's output for better accuracy.

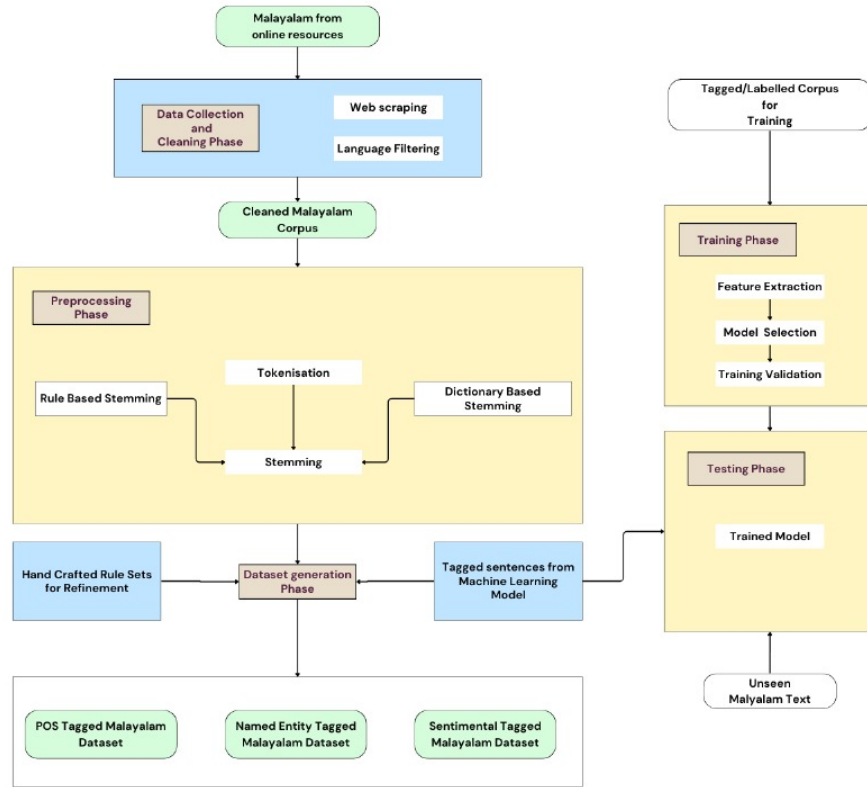


Figure 3.1: Architecture diagram

### 3.2 Dataset identified

This section describes the data source used in the project. Brief its properties and refer it to the appropriate location. Sample subsets of the dataset can be highlighted.

### 3.3 Proposed Methodology/Algorithms

This section describes in detail the methodologies or algorithms associated with your work. Algorithms should be written in appropriate format.

### 3.4 User Interface Design

The user interface design (wireframe designs) can be highlighted in this section. The figures titles should be in a chronological order and self explanatory.

### 3.5 Database Design

The detailed database design and its schema is expected in this section. The database used in the work can be mentioned here. The reason for choosing the database can be substantiated in this section.

### 3.6 Description of Implementation Strategies

The implementation strategies for the project are as follows:

- Data Acquisition and Cleaning:

1. Web Scraping:

- Utilize libraries like BeautifulSoup or Scrapy (Python) to scrape relevant Malayalam websites.
- Define target URLs and develop scraping logic for text extraction.
- Implement techniques to handle pagination or dynamic content loading (if applicable).

2. Language Filtering:

- Implement language detection using libraries like langdetect or textblob (Python) to identify non-Malayalam text.
- Set a threshold or confidence score to filter out content below a certain level of Malayalam probability.

- Data Preprocessing:

1. Tokenization: Choose a suitable tokenization method (word-based, sentence-based, etc.) using libraries like NLTK or spaCy (Python).
2. Handling Non-Textual Elements: Develop logic for managing emojis or other non-textual elements (e.g., removal, special token representation).
3. Stop Word Removal: Implement stop word removal based on a created or located Malayalam stop word list.
4. Stemming or Lemmatization: Use libraries like NLTK or spaCy for stemming (reducing words to root forms) or lemmatization (finding dictionary base

forms). Explore specific stemming/lemmatization algorithms if necessary for Malayalam.

- Sentiment Analysis:

1. Model Selection (if applicable): Consider factors like data size, desired accuracy, and computational resources when choosing a model (e.g., Logistic Regression, Naive Bayes, SVM, RNNs).
2. Data Splitting (if applicable): Split the preprocessed data into training, validation, and testing sets for model development.
3. Model Training and Tuning (if applicable): Implement hyperparameter tuning to optimize model performance during training.
4. Evaluation (if applicable): Evaluate the trained sentiment analysis model's performance on the unseen testing data set. Analyze the results and identify areas for improvement in data quality or model architecture.

- Additional Considerations:

1. Data Storage and Management: Consider implementing data loading/saving functions for various formats (text files, CSV, etc.) if needed for further analysis.
2. Version Control: Utilize a version control system (e.g., Git) to track code changes and facilitate collaboration.
3. Testing: Implement unit tests for critical functions and integration tests to ensure the entire NLP pipeline functions as expected.
4. Logging: Implement logging functionalities to track code execution progress and identify potential issues.

### **3.7 Module Division**

This section describes the different modules involved in this project and a small description of the same is expected. This section ends with the information of which module is assigned to each project member.

### 3.8 Work Schedule - Gantt Chart

#### Gantt chart

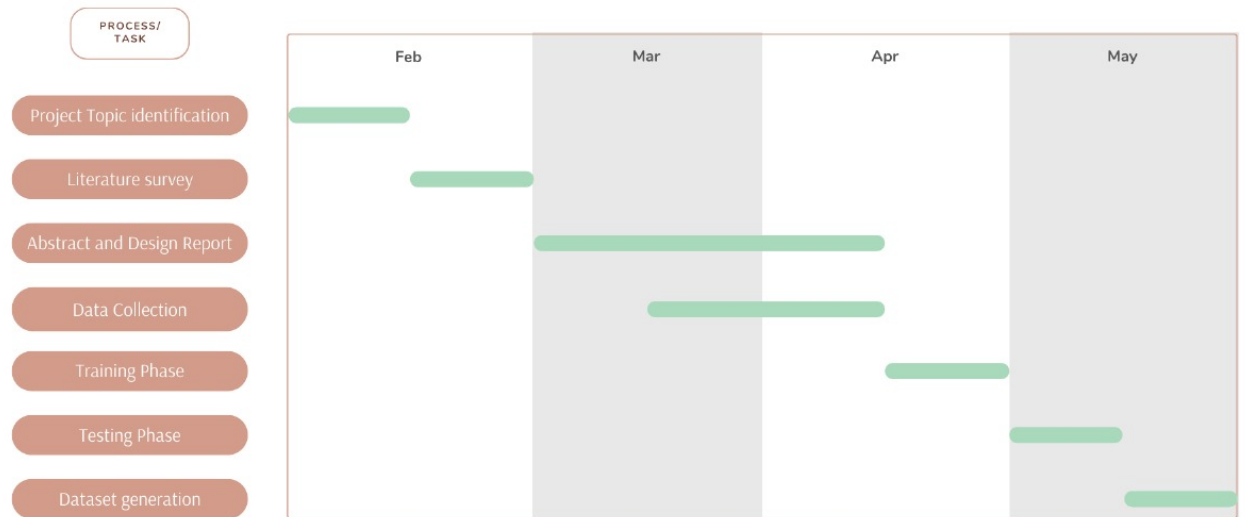


Figure 3.2: Work Schedule-Gantt chart

## Bibliography

- [1] Asopa, S., and Sharma, N. (2021) A Hybrid Parser Model for Hindi Language. Indian Journal of Computer Science and Engineering (IJCSE), Vol. 12(1).
- [2] Chen, D., and Manning, C. D. (2014). A Fast and Accurate Dependency Parser using Neural Networks. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP).
- [3] Nair, L. R. (2013). Language Parsing and Syntax of Malayalam Language. 2nd International Symposium on Computer, Communication, Control and Automation (3CA 2013).
- [4] Berger, A. L., Della Pietra, V. J., and Della Pietra, S. A. (1996). A Maximum Entropy Approach to Natural Language Processing. Association for Computational Linguistics, Vol 22(1).
- [5] Mestry, A., Shende, S., Mahadik, A., and Virnodkar, S. (2014). A Parser: Simple English Sentence Detector and Correction. International Journal of Engineering Research and Technology (IJERT).
- [6] Sethi, N., Agrawal, P., Madaan, V., and Singh, S. K. (2016). A Novel Approach to Paraphrase Hindi Sentences using Natural Language Processing. Indian Journal of Science and Technology, Vol 9(28).
- [7] Smith, D. A., and Eisner, J. (2008). Dependency Parsing by Belief Propagation. Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing, Page 145- 156.
- [8] Bharati, A., Kulkarni, A., and Chaudhury, S. (2007). English Parsers: Some Information- based Observations.

- [9] Jayan, J. P., and R, R. (2009). A Morphological Analyzer for Malayalam - A Comparison of Different Approaches. *International Journal of Computer Science and Information Technology*. Vol 2(2), Page 155-160.
- [10] Vaidya, A., Choi, J. D., Palmer, M., and Narasimhan, B. (2011). Analysis of the Hindi Proposition Bank using Dependency Structure. *Proceedings of the Fifth Law Workshop (LAW V)*, Page 21-29.
- [11] Rajan, M., T.S, R., and Bhojane, V. (2014). Information Retrieval in Malayalam Using Natural Language Processing. *International Journal of Scientific and Engineering Research*, Vol 5(6)
- [12] Rajan, M., Thirumalai, R., and Kumar, V. (2006). Development of a Tamil Parser using Natural Language Processing Techniques. A survey of the state of the art in tamil language technology Vol 6(10).
- [13] Venkatesh, R., Kumar, S., and Arumugam, P. (2014). Building a Lexical Analyzer for Tamil Texts using NLP Approaches. *2014 International Conference on Advances in ICT for Emerging Regions (ICTer)*.
- [14] Thavareesan, S., and Mahesan, S. (2019). Sentiment Analysis in Tamil Texts: A Study on Machine Learning Techniques and Feature Representation. *2019 IEEE 14th Conference on Industrial and Information Systems (ICIIS)*.
- [15] Pai, T. V., Devi, J. A., and Aithal, P. S. (2020). A Systematic Literature Review of Lexical Analyzer Implementation Techniques in Compiler Design. *International Journal of Applied Engineering and Management Letters (IJAEML)*, Vol 4(2), Page 285-301.
- [16] Simmons, R. F., and Burger, J. F. (1968). A Semantic Analyzer for English Sentences. *Mechanical Translation and Computational Linguistics*, Vol 11.