A

Lab-I: Project Phase I Report

On

"Developing software to automatically translate resource materials between English to Indian regional languages"

SUBMITTED TO THE PUNYASHLOK AHILYADEVI HOLKAR SOLAPUR UNIVERSITY, SOLAPUR IN THE PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF THE DEGREE OF

BACHELOR OF TECHNOLOGY(CSE)

SUBMITTED BY

Mr. Mustafa Masuldar	**Roll No. 16**
Ms. Sakshi Deshmukh	**Roll No. 17**
Ms. Pooja Pawar	**Roll No. 18**
Ms. Nandini Madre	**Roll No. 19**
Ms.Maleka Iram Muchale	**Roll No.20**

UNDER THE GUIDANCE OF

Prof. S. A. Dhanwe Sir

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERINGN B
NAVALE SINHGAD COLLEGE OF ENGINEERING
PAH SOLAPUR UNIVERSITY, SOLAPUR

2023-2024

CERTIFICATE

This is to certify that the Lab-I: Project Phase I Report entitled

"Developing software to automatically translate resource materials between English to Indian regional languages"

submitted by

Mr. Mustafa Masuldar	**Roll No. 16**
Ms. Sakshi Deshmukh	**Roll No. 17**
Ms. Pooja Pawar	**Roll No. 18**
Ms. Nandini Madre	**Roll No. 19**
Ms.Maleka Iram Muchale	**Roll No.20**

is a bonafide work carried out by above students under the supervision of **Prof. S. A. Dhanwe** and it is submitted towards the fulfillment of the requirement of PAH Solapur University, Solapur, for the award of the degree of Bachelor of Technology (Computer Science and Engineering) during academic year 2023-24.

Prof. S. A. Dhanwe
Guide

Dr. D. P. Gandhmal
HOD-CSE

Dr. S. D. Nawale
Principal
N B Navale Sinhgad College of Engineering, Solapur

Place:
Date:

DECLARATION

We hereby declare that the work embodied in this Lab-I: Project Phase I Report Entitled "Developing software to automatically translate resource materials between English to Indian regional languages" is carried out by us in partial fulfillment of the degree B.Tech. Computer Science and Engineering from N. B. Navale Sinhgad College of Engineering, solapur and we have not submitted the same to any other University/Institute for the award of any other degree.

Mr. Mustafa Masuldar

Ms. Sakshi Deshmukh

Ms. Pooja Pawar

Ms. Nandini Madre

Ms.Maleka Iram Muchale

ACKNOWLEDGEMENT

We would like to thank our guide **Prof. S. A. Dhanwe** for helping me out in selecting the topic and contents, giving valuable suggestions in preparation of Lab-I: Project Phase I report and presentation.

We are grateful to Dr. D. P. Gandhmal, Head of CSE Department, for providing a healthy environment and facilities in the department. He allowed us to raise our concern and worked to solve it by extending his cooperation from time to time.

Goal makes us do work. Vision is more important than a goal which makes us do work in the best way to make work equally the best. Thanks to the Principal, Dr. S. D. Nawale for his support and vision.

We are also Thankful to all the CSE department staff members for their extended support and valuable guidance.

Mr. Mustafa Masuldar

Ms. Sakshi Deshmukh

Ms. Pooja Pawar

Ms. Nandini Madre

Ms.Maleka Iram Muchale

ABSTRACT

CIPAM, dedicated to advancing Intellectual Property Rights (IPR) awareness, commercialization, and enforcement, has created extensive educational materials. The software aims to bridge linguistic gaps by translating these materials from English to key Indian regional languages: Hindi, Marathi, Bengali, Gujarati, Tamil, and Telugu.

- The software's core features include its ability to translate various formats, such as Word documents, PDFs, and text within images. Notably, it focuses on maintaining the contextual integrity and
- Professionalism of the original content, ensuring that translations are not only accurate but also accessible to the general public.
- This abstract outlines a comprehensive approach involving Natural Language Processing (NLP) techniques, machine learning algorithms, and collaboration with experts to ensure the software's accuracy and user-friendliness.

Contents

List of Figures

Chapter 1

INTRODUCTION

CIPAM, dedicated to advancing Intellectual Property Rights (IPR) awareness, commercialization, and enforcement, has created extensive educational materials. The software aims to bridge linguistic gaps by translating these materials from English to key Indian regional languages: Hindi, Marathi, Bengali, Gujarati, Tamil, and Telugu.

- The software's core features include its ability to translate various formats, such as Word documents, PDFs, and text within images. Notably, it focuses on maintaining the contextual integrity and professionalism of the original content, ensuring that translations are not only accurate but also accessible to the general public.

- This abstract outlines a comprehensive approach involving Natural Language Processing (NLP) techniques, machine learning algorithms, and collaboration with experts to ensure the software's accuracy and user-friendliness.

1.1 Problem Definition

Developing software to automatically translate resource materials between English to Indian regional languages

1.1.1 Motivation

1. Enhancing Accessibility: Creating software for automatic translation between English and Indian regional languages expands access to valuable resource materials. This technology breaks language barriers, ensuring that information, education, and resources are available and understandable to a wider audience, thus promoting inclusivity and equity.

2. Preserving Cultural Diversity: India is a linguistically diverse country with numerous regional languages. By developing software for automatic translation, it promotes the preservation and propagation of these languages. It encourages the exchange of ideas, literature, and cultural heritage between English and various Indian languages, fostering a deeper appreciation for linguistic diversity.

3. Facilitating Communication: Such software simplifies communication across different linguistic communities. It encourages collaboration, knowledge sharing, and interaction among individuals and organizations, fostering stronger relationships and understanding between English-speaking populations and those who primarily use Indian regional languages.

4. Boosting Efficiency and Productivity: Automating the translation process saves time and resources. It allows for the swift conversion of materials, which can be crucial in educational settings, government services, businesses, and other sectors where quick access to translated information is essential. This efficiency boosts productivity and streamlines operations across various domains.

1.2 Scope and Objective

1.2.1 Scope

- The software covers translation needs for Word documents, PDFs, and text within images.
- The project includes developing software for seamless translation from English to various Indian regional languages.
- The software aims to create a user-friendly interface, preserve the original meaning, and offer customization features.
- Emphasis is on speed, productivity, quality assurance, user feedback, and offline functionality.

1.2.2 Objective

- Develop software for accurate translation of IPR resource materials from English to Indian regional languages (Hindi, Marathi, Bengali, Gujarati, Tamil, Telugu) across various formats, ensuring the preservation of the text's intended meaning.
- Key objectives include accuracy, multilingual support, user-friendly interface, scalability, contextual understanding, customization, quality control, performance optimization, privacy and security, integration, machine learning and AI, cross-platform compatibility, cultural sensitivity, and documentation and support.

1.3 Applications

1. Education and E-learning Platforms: Implementing automatic translation software can significantly benefit educational institutions and e-learning platforms. It enables the translation of educational materials, textbooks, and online courses from English to various Indian regional languages, facilitating easier comprehension for students whose primary language may not be English. This application promotes inclusive learning and improves access to quality education.

2. Government Services and Public Information: Government agencies can utilize this technology to translate crucial information, public service announcements, forms, and documents into multiple Indian regional languages. This helps in disseminating important information to a broader audience, ensuring that all citizens can access and understand government services and policies effectively.

3. Business and E-commerce: E-commerce platforms and businesses aiming to expand their reach in India can benefit from automatic translation. Translating product descriptions, customer reviews, and marketing content into regional languages can attract and engage a more diverse customer base. This approach enhances user experience and increases sales by catering to non-English speaking populations.

4. Healthcare Sector: In the healthcare domain, translating medical resources, prescriptions, health guidelines, and patient information into regional languages can significantly improve healthcare accessibility. It assists healthcare providers in effectively communicating with patients who might not be proficient in English, thus ensuring better healthcare outcomes and understanding.

5. Content Creation and Media: Content creators, publishers, and media houses can use automatic translation to reach a wider audience. Translating articles, news, blogs, and multimedia content into regional languages enhances audience engagement and expands the content's impact, allowing for broader dissemination and increased readership/viewership.

Chapter 2

LITERATURE SURVEY

- ***Jayanthi, N., Lakshmi, A., Raju, C. S. K., & Swathi, B. (2020). Dual Translation of International and Indian Regional Language using Recent Machine Translation. 2020 3rd International Conference on Intelligent Sustainable Systems (ICISS).***

  Title: Dual Translation of International and Indian Regional Language using Recent Machine Translation

  Neural Machine Translation (NMT) is a modern and powerful approach that has resulted in major improvements compared to traditional techniques of machine translation for translating one language into another. In the world, India is a very multicultural and multilingual country. People of India from various regions use their own regional communication languages, making India stand at the second position in the world to have maximum languages. In India, English is provided as the second extra official language. But the usage of English in India is very less forming a communication gap. To minimize this gap by translating one language into another language is almost impossible for humans. It can be achieved by a machine translation. This paper focuses on translating Indic languages using the translation technique of Neural technology. A Sequence to Sequence model with encoder-decoder attention mechanism of neural machine translation is proposed for Telugu language conversion into English and vice versa.

- ***" Nadeem Jadoon Khan, Waqas Anwar, Nadir Durrani ,Machine Translation Approaches and Survey for Indian Languages "***

  Title: Machine Translation Approaches and Survey for Indian Languages

  In this study, we present an analysis regarding the performance of the state-of-art Phrase based Statistical Machine Translation (SMT) on multiple Indian languages. We report baseline systems on several language pairs. The motivation of this study is to promote the

development of SMT and linguistic resources for these language pairs, as the current state-of-the-art is quite bleak due to sparse data resources. The success of an SMT system is contingent on the availability of a large parallel corpus. Such data is necessary to reliably estimate translation probabilities. We report the performance of baseline systems translating from Indian languages (Bengali, Guajarati, Hindi, Malayalam, Punjabi, Tamil, Telugu and Urdu) into English with average 10% accurate results for all the language pairs.

" Pulipaka, S. K., Kasaraneni, C. K., Sandeep Vemulapalli, V. N., & Mourya Kosaraju, S. S. (2019). *Machine Translation of English Videos to Indian Regional Languages using Open Innovation. 2019 IEEE International Symposium on Technology and Society (ISTAS).* "

Title: Machine Translation of English Videos to Indian Regional Languages using Open Innovation

In spite of many languages being spoken in India, it is difficult for the people to understand foreign languages like English, Spanish, Italian, etc. The recognition and synthesis of speech are prominent emerging technologies in natural language processing and communication domains. This paper aims to leverage the open source applications of these technologies, machine translation, text-to-speech system (TTS), and speech-to text system (STT) to convert available online resources to Indian languages. This application takes an English language video as an input and separates the audio from video. It then divides the audio file into several smaller chunks based on the timestamps. These audio chunks are then individually converted into text using IBM Watson's speech-to-text (STT) module.

- **" RAJANI S, Translation Across Cultures: From The Regional To The Universal "**

Title: Translation Across Cultures: From The Regional To The Universal

India is a nation of many regional languages. The advent of the Europeans brought in a necessity to understand the regional languages through a foreign language. The Europeans felt it necessary to learn and understand our languages so that their religion and religious teachings would reach the indigenous people better. In the process, many Christian missionaries who came to India learnt different dialects of the regional languages along with Sanskrit and made way for translation works. Reverend Ferdinand Kittle who came to India as a missionary and Ideologist, worked on similar lines and gave the monumental work; the kannada-English Dictionary. Even to this day this dictionary is considered as the authentic document of reference. This work helped many European readers to easily understand Kannada and hence connect themselves to the Kannada literary culture. In the long run numerous Kannada literary works of different genres were translated into English, giving it wider and broader readership. Here is a humble effort to peep into the translations of different genres of Kannada literature that have gained universal fame and applaud. Girish Karnad's Hayavadana, The Vachanas of Basavanna, Parva by S. L Bhyrappa, and few poems of K S Narasimha Swamy are a few works that are considered here.

- **" Sindhu, D. V., & Sagar, B. M. (2016). *Study on machine translation approaches for Indian languages and their challenges. 2016 International Conference on Electrical, Electronics,* Communication, *Computer and Optimization Techniques (ICEECCOT). "***

Title: Study on Machine Translation Approaches for Indian Languages and Their Challenges

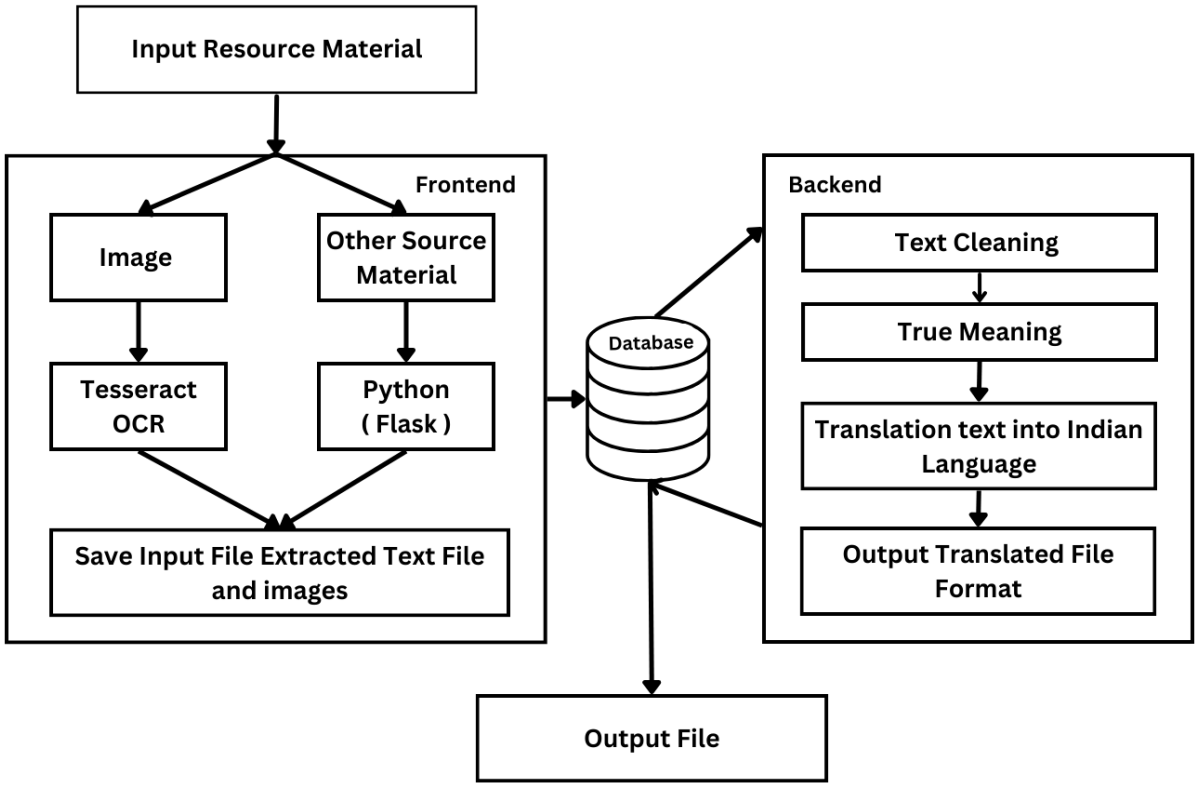
This survey mainly focuses on the developments of machine translation for the Indian languages. The survey throws a light on rule-based approach, empirical based approach and hybrid based approaches for machine translation. Every approach has its own advantages and disadvantages. Machine Translation (MT) is a process which translates from one language to another language. Due to rapid globalization there is an increased data over the web machine translation plays a very important role to reduce the language barrier between different regions. In a country like India with 22 official languages shows a high attention for the translation. This paper focuses on the different MT systems for Indian languages and also their challenges.

- **Research papers :**https://drive.google.com/drive/folders/1ZwulcEIcZVzt9p8-eFw194DCzRvXS0v9?usp=sharing

Chapter 3
SYSTEM DESIGN

3.1 System Architecture

- Input resource material (text or image)
- Detect language of input resource material
- Extract text from image
- Translate text to desired Indian regional language using machine translation
- Output translated text in desired format



The system architecture for extracting text from images, as shown in the diagram, can be divided into two main parts: the frontend and the backend.

Frontend:

The frontend is responsible for receiving the input resource material, which can be an image or other source material containing text. The frontend also performs some pre-processing on the input resource material, such as noise reduction and image enhancement.

Backend :

The backend is responsible for the actual text extraction and recognition process. It uses an optical character recognition (OCR) engine to extract the text from the input resource material. The OCR engine may also perform additional tasks, such as language detection and text cleaning.

The backend also includes a database to store the extracted text and other information, such as the source of the input resource material and the OCR engine used.

Process

The following is a step-by-step overview of the text extraction and recognition process:

The input resource material is sent to the frontend.

The frontend performs some pre-processing on the input resource material, such as noise reduction and image enhancement.

The pre-processed input resource material is sent to the backend.

The backend uses an OCR engine to extract the text from the input resource material.

The extracted text is cleaned and normalized.

The cleaned and normalized text is stored in the database.

The extracted text can now be used for various purposes, such as translation, indexing, and search.

Additional features

The system architecture also includes some additional features, such as:

Translation: The system can translate the extracted text into different languages. This is useful for extracting text from images and other source materials that are in a foreign language.

Output format: The system can output the extracted text in different formats, such as plain text, HTML, and PDF. This allows users to choose the format that is most convenient for them.

Example use case

A typical example of how this system architecture could be used is in a document digitization application. In this case, the input resource material would be a scanned document image. The system would then extract the text from the image and store it in a database. The extracted text could then be used to index the document and make it searchable.

3.2 Analysis Model

3.2.1 DFD level 0

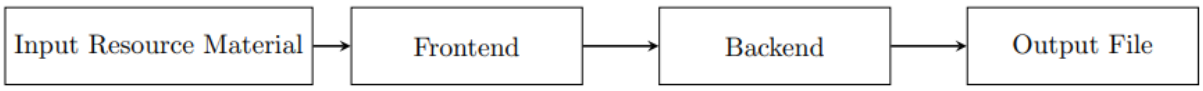


Fig : DFD level 0

The diagram shows the system at the highest level of abstraction, with a single input (image sent) and a single output (translated text).

The system works by first extracting the text from the input image. This can be done using an optical character recognition (OCR) algorithm. Once the text has been extracted, it is cleaned to remove noise and errors. The cleaned text is then translated to the output language using a machine translation API. Finally, the translated text is saved to a file.

The DFD Level 0 diagram provides a high-level overview of the system and its components. It is a useful tool for understanding the basic flow of data through the system.

3.2.2 DFD level 1

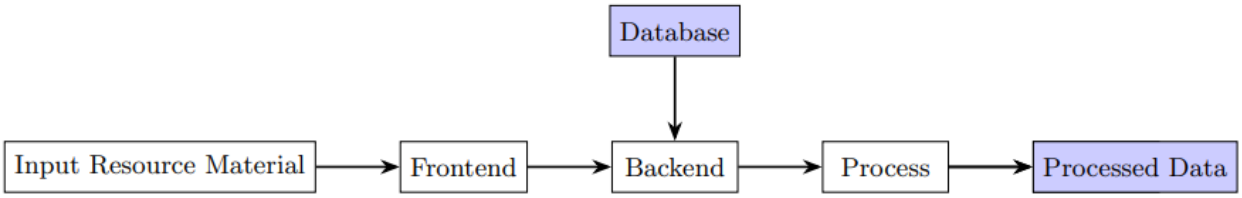


Fig : DFD level 1

The image you sent is a flow chart for extracting text from a source material, such as a PDF, image, or video. The process can be divided into the following steps:

Input: The first step is to input the source material into the text extractor. This can be done by manually uploading the file or by using a URL to the file.

Text Extraction: The text extractor will then use various techniques to extract the text from the source material. This may involve OCR (optical character recognition) to extract text from images, or PDF parsing to extract text from PDF files.

Text Processing: Once the text has been extracted, it may need to be processed to clean it up and make it more machine-readable. This may involve removing non-textual elements, such as images and tables, or correcting spelling and grammar errors.

Output: The final step is to output the extracted text to a desired format, such as a text file, PDF file, or HTML file. This may involve additional processing, such as translation or summarization.

The flowchart also shows two additional steps:

Other Source Material: The text extractor may also be able to extract text from other types of source material, such as websites, emails, and social media posts.

Translator: The extracted text can also be translated into another language using a translator.

The flowchart can be adapted to extract text from other types of source material, such as images and videos, by using different text extraction techniques. The flowchart can also be adapted to include additional steps, such as translation and summarization.

3.2.3 DFD level 2

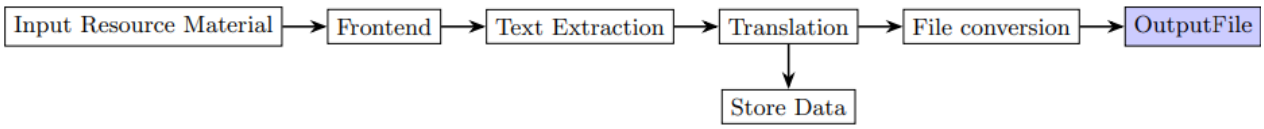


Fig : **DFD level 2**

The DFD2 diagram you sent is a more detailed version of the Level 1 DFD for a Text Extraction System. It breaks down the Text Extractor, Text Cleaner, and Translator subsystems into their components and data flows.

Text Extractor: The Text Extractor subsystem is responsible for extracting text from a source material. It has the following components:

OCR (Optical Character Recognition) component: Extracts text from images.

PDF Parsing component: Extracts text from PDF files.

Other Text Extractors: Extracts text from other types of source material, such as websites, emails, and social media posts.

The data flows between the Text Extractor components are as follows:

Source Material to OCR component: This data flow contains the source material that the user wants to extract text from.

OCR Output to PDF Parsing component (optional): This data flow contains the extracted text from the OCR component. It is optional because the PDF Parsing component can also receive text from other components.

Other Text Extractors output to PDF Parsing component (optional): This data flow contains the extracted text from the other text extraction components. It is optional because the PDF Parsing component can also receive text from other components.

PDF Parsing output to Text Extractor output: This data flow contains the extracted text from the PDF Parsing component.

Text Cleaner : The Text Cleaner subsystem is responsible for cleaning up the extracted text to make it more machine-readable. It has the following components:

It has the following components:

Machine Translation (MT) Engine: Translates the text using a machine translation algorithm.

Human Translator: Reviews and edits the machine translation output.

The data flows between the Translator components are as follows:

Text Cleaner output to MT Engine: This data flow contains the cleaned text from the Text Cleaner subsystem.

The DFD2 diagram shows that the Text Extraction System is a complex system with many components and data flows. However, the diagram also shows that the system is well-organized and that the components and data flows are well-defined. This makes the system easier to understand, design, implement, and maintain.

3.3 UML Diagrams

3.3.1 Class Diagram

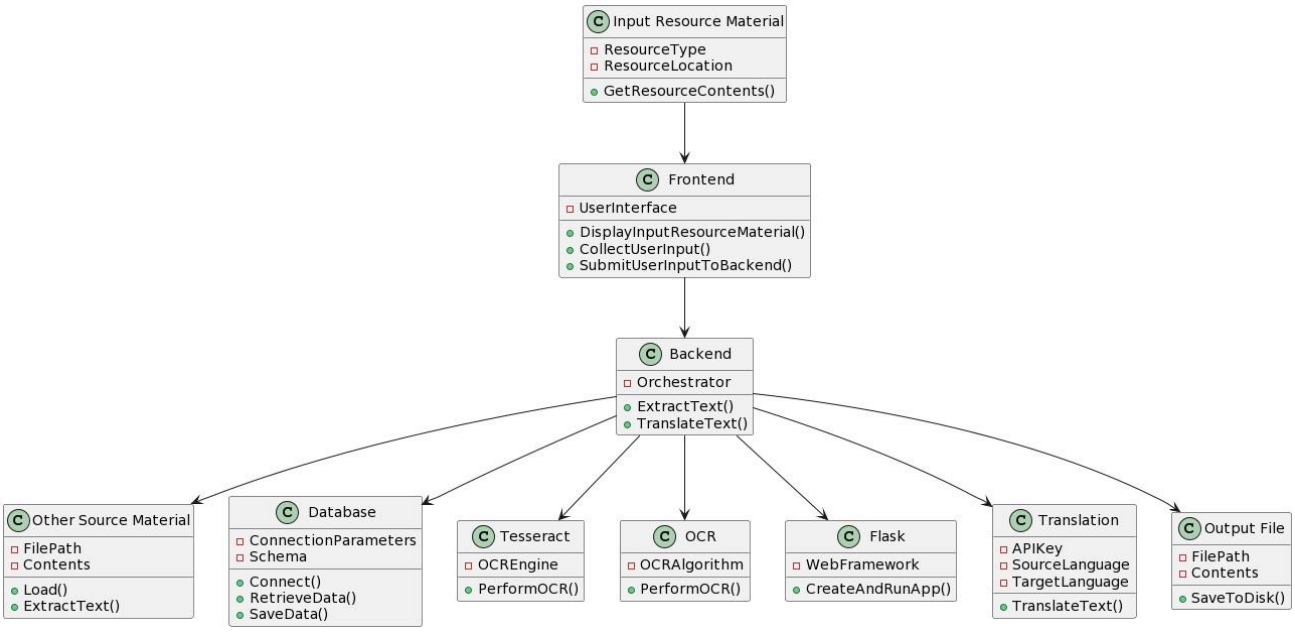


Fig: Class Diagram

This diagram shows that the system has three main components:

Input: The input component is responsible for reading the input resource material, which can be an image or other source material.

Preprocessing: The preprocessing component preprocesses the input resource material to make it easier to extract the text. This may involve noise reduction, image enhancement, and other techniques.

Extraction: The extraction component extracts the text from the input resource material. This may involve using OCR or other techniques.

Cleaning: The cleaning component cleans the extracted text to remove errors and artifacts. This may involve removing punctuation, stop words, and other elements.

Storage: The storage component stores the cleaned text in a database.

Output: The output component generates an output file from the cleaned text. This may involve generating a text file, PDF document, or other format.

The relationships between the classes in the diagram show the flow of data through the system. The input resource material is first read by the input component. The input component then passes the input

resource material to the preprocessing component. The preprocessing component then preprocesses the input resource material and passes it to the extraction component. The extraction component then extracts the text from the input resource material and passes it to the cleaning component. The cleaning component then cleans the extracted text and passes it to the storage component. The storage component then stores the cleaned text in a database. Finally, the output component generates an output file from the cleaned text.

3.3.2 Use Case Diagram :

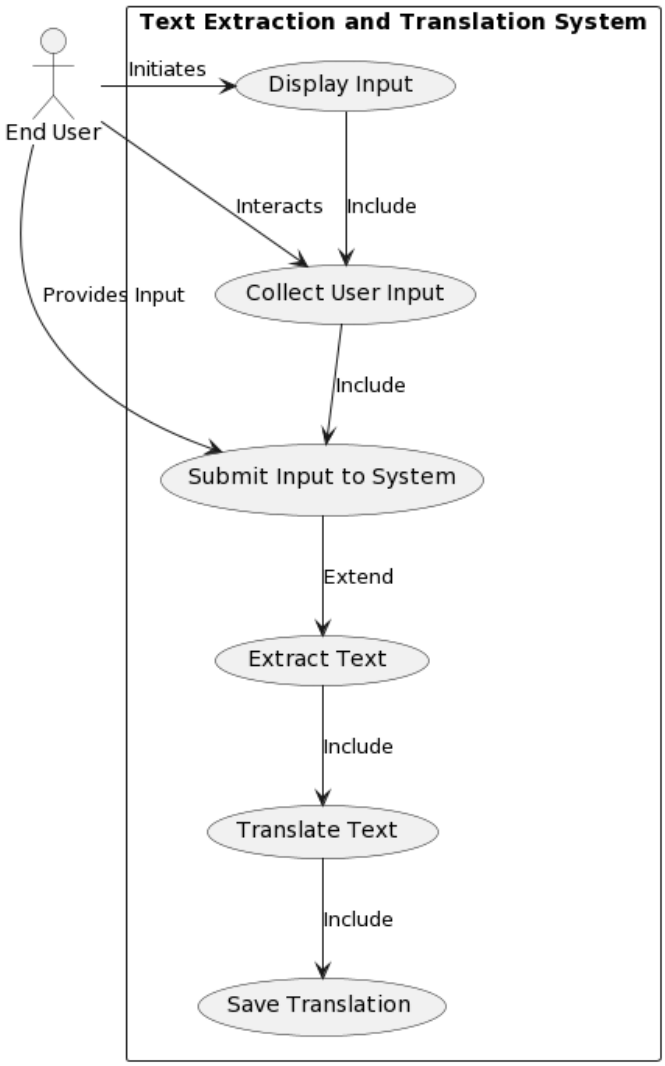


Fig : Use Case Diagram

Provide source material: The end user or resource creator provides the source material to the system. This can be done in a variety of ways, such as uploading a file, providing a link to a web page, or entering text directly into the system.

Submit source material: The end user or resource creator submits the source material to the system. This triggers the translation process.

Validate source material: The system validates the source material to ensure that it is in a format that can be translated. This may involve checking the file type, character encoding, and other factors.

Translate source material: The system translates the source material into the desired language. This is done using a variety of techniques, such as statistical machine translation, rule-based machine translation, and neural machine translation.

Provide translated resource: The system provides the translated resource to the end user or resource creator. This can be done in a variety of ways, such as downloading a file, receiving an email with the translated resource, or viewing the translated resource directly in the system.

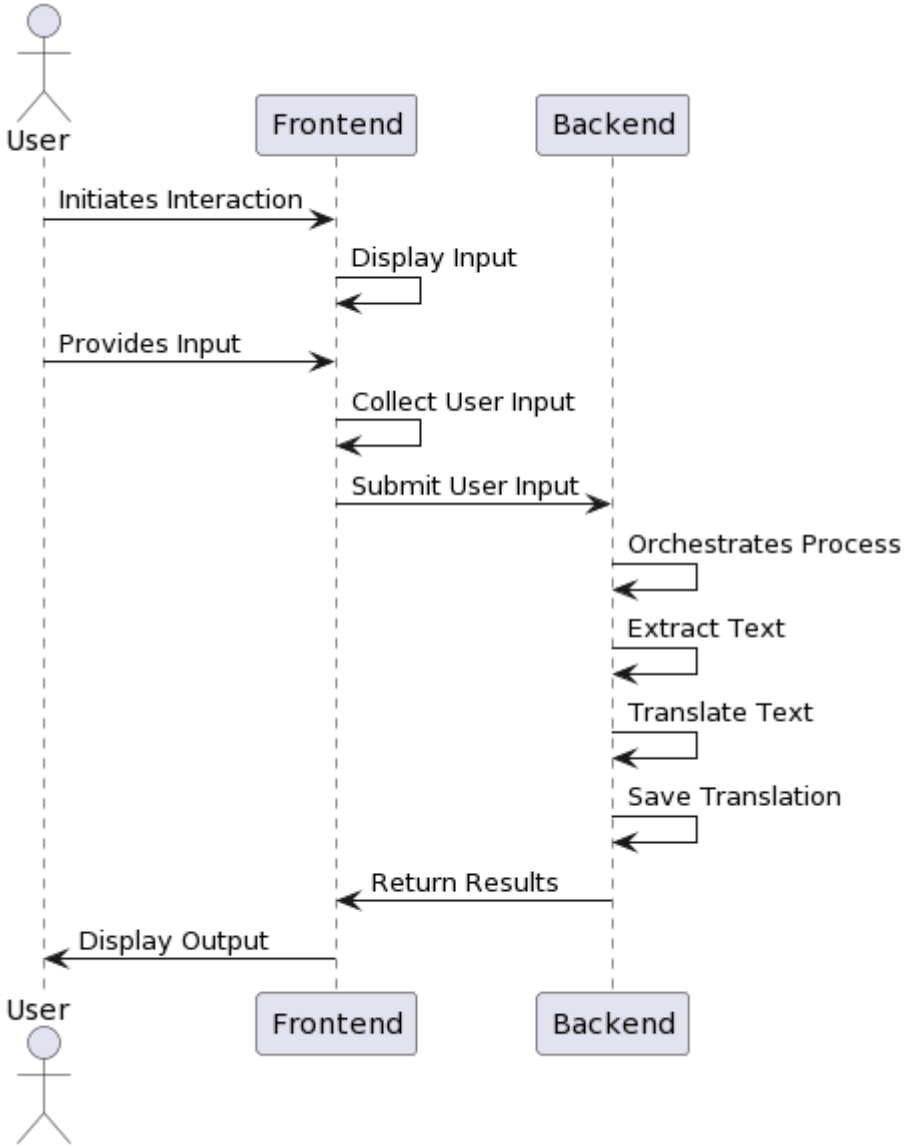
In addition to these core steps, the diagram also shows a few other important aspects of the automated translation process:

The system is able to provide confirmation of upload to the resource creator. This helps to ensure that the resource material was successfully submitted to the system.

The system is able to validate the source material before translating it. This helps to avoid errors in the translation process.

The system is able to provide the translated resource to the end user or resource creator in a variety of ways. This makes it easy for users to access the translated resource in a format that is convenient for them.

3.3.3 Sequence Diagram



The sequence diagram you provided shows the steps involved in extracting text from an image using a distributed system. The system consists of the following components:

User: The user sends an input image to the system.

Frontend: The frontend receives the input image from the user and forwards it to the backend.

Backend: The backend coordinates the text extraction process. It requests the OCR service to extract the text from the image, and then preprocesses and cleans the extracted text. Finally, it saves the cleaned text to the database and requests the Output File Generator component to generate an output file.

OCR service: The OCR service extracts the text from the image.

Text Preprocessor: The Text Preprocessor component preprocesses the extracted text. This may involve removing noise, correcting errors, and normalizing the text.

Text Cleaner: The Text Cleaner component cleans the preprocessed text. This may involve removing stop words, punctuation marks, and other artifacts.

Database: The database stores the cleaned text.

Output File Generator: The Output File Generator component generates an output file from the cleaned text. The output file may be a text file, PDF document, or other format.

The following is a detailed explanation of each step in the sequence diagram:

The user sends an input image to the frontend.

The frontend receives the input image from the user and forwards it to the backend.

The backend sends the input image to the OCR service.

The OCR service extracts the text from the image and returns it to the backend.

The backend preprocesses and cleans the extracted text.

The backend saves the cleaned text to the database.

The backend sends the cleaned text to the Output File Generator component.

The Output File Generator component generates an output file from the cleaned text.

The backend sends the output file to the user.

The sequence diagram shows that the system is distributed, meaning that the different components are located on different machines. This allows the system to scale to handle large workloads. Additionally, the system is fault-tolerant, meaning that it can continue to operate even if one of the components fails.

Here is an example of how the system might be used:

A user takes a picture of a document using their smartphone.

The user uploads the picture to a cloud-based service.

The cloud-based service extracts the text from the image using the distributed system described above.

The cloud-based service returns the extracted text to the user in a convenient format, such as a text file or PDF document.

3.3.4 State Chart Diagram

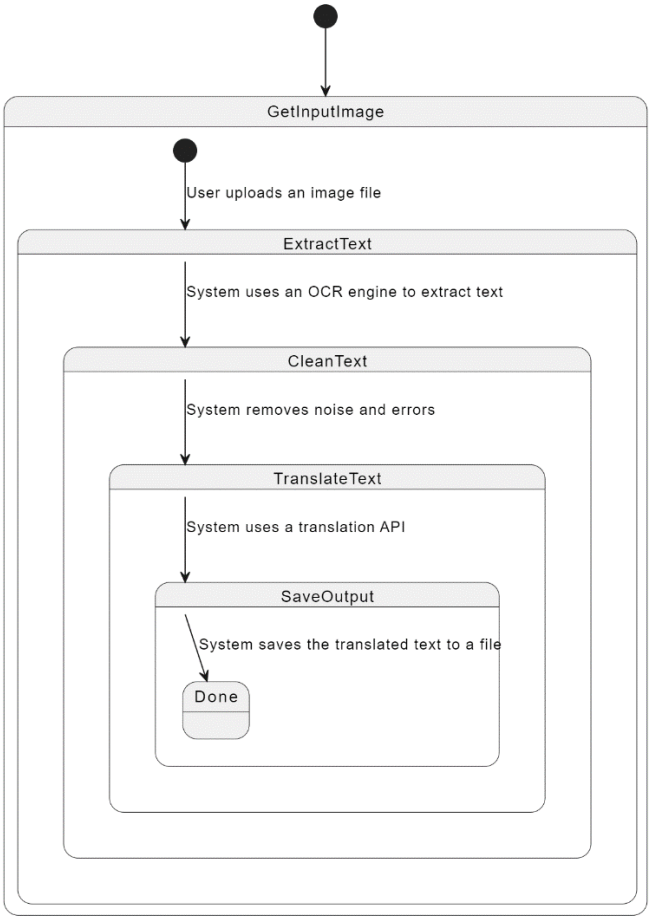


Fig : **State Chart Diagram**

In state chart diagram you sent shows the different states that a text extraction system can be in, as well as the events that trigger transitions between states.

The system starts in the Initialized state. From the Initialized state, the system can transition to the Input File Handling state or the Idle state.

If the system transitions to the Input File Handling state, it will extract the text from the input file. Once the text has been extracted, the system will transition to the Text Processing state.

In the Text Processing state, the system will perform any necessary preprocessing on the text, such as noise reduction or image enhancement. Once the text has been preprocessed, the system will transition to the Optical Character Recognition (OCR) state.

In the Optical Character Recognition (OCR) state, the system will extract the text from the image. Once the text has been extracted, the system will transition to the Text Cleaning state.

In the Text Cleaning state, the system will remove any errors or artifacts from the extracted text. Once the text has been cleaned, the system will transition to the Database state.

In the Database state, the system will save the cleaned text to a database. Once the text has been saved, the system will transition to the Output File Handling state.

In the Output File Handling state, the system will generate an output file from the text in the database. Once the output file has been generated, the system will transition to the Idle state.

The system can transition back to the Input File Handling state from the Idle state.

The state chart diagram also shows the following events:

Input File Received: This event triggers the system to transition from the Initialized state to the Input File Handling state.

Text Extracted: This event triggers the system to transition from the Input File Handling state to the Text Processing state.

Text Preprocessed: This event triggers the system to transition from the Text Processing state to the Optical Character Recognition (OCR) state.

Text Recognized: This event triggers the system to transition from the Optical Character Recognition (OCR) state to the Text Cleaning state.

Text Cleaned: This event triggers the system to transition from the Text Cleaning state to the Database state.

Text Saved: This event triggers the system to transition from the Database state to the Output File Handling state.

Output File Generated: This event triggers the system to transition from the Output File Handling state to the Idle state.

Here is a more detailed explanation of each state and transition in the state chart diagram:

Initialized: This is the initial state of the system. In this state, the system is waiting for an input file to be received.

Input File Handling: In this state, the system is extracting the text from the input file. Once the text has been extracted, the system transitions to the Text Processing state.

Text Processing: In this state, the system is performing any necessary preprocessing on the text, such as noise reduction or image enhancement. Once the text has been preprocessed, the system transitions to the Optical Character Recognition (OCR) state.

Optical Character Recognition (OCR): In this state, the system is extracting the text from the image. Once the text has been extracted, the system transitions to the Text Cleaning state.

Text Cleaning: In this state, the system is removing any errors or artifacts from the extracted text. Once the text has been cleaned, the system transitions to the Database state.

Database: In this state, the system is saving the cleaned text to a database. Once the text has been saved, the system transitions to the Output File Handling state.

Output File Handling: In this state, the system is generating an output file from the text in the database. Once the output file has been generated, the system transitions to the Idle state.

Idle: In this state, the system is waiting for the next input file to be received.

3.3.5 Activity Diagram

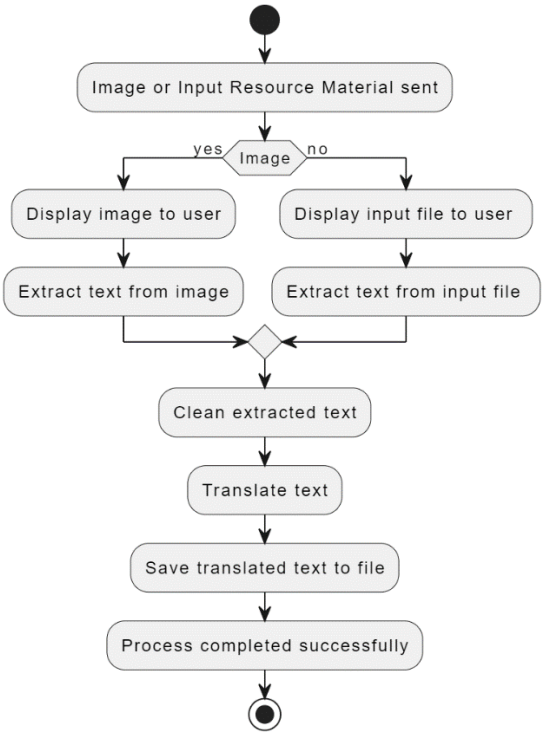


Fig : Activity Diagram

The activity diagram you sent shows the process of extracting text from a file. The process begins with the Input File Handling activity, which extracts the text from the input file. The extracted text is then passed to the Text Processing activity, which performs any necessary preprocessing on the text, such as noise reduction or image enhancement.

The preprocessed text is then passed to the Optical Character Recognition (OCR) activity, which extracts the text from the image. The extracted text is then passed to the Text Cleaning activity, which removes any errors or artifacts from the text.

The cleaned text is then saved to a database in the Database activity. The Output File Handling activity then generates an output file from the text in the database. The output file is then returned to the user in the Output activity.

Here is a more detailed explanation of each activity in the diagram:

Input File Handling: This activity extracts the text from the input file. The input file can be a variety of formats, such as a PDF, Word document, or image file.

Text Processing: This activity performs any necessary preprocessing on the text, such as noise reduction or image enhancement. This is necessary to ensure that the OCR engine can accurately extract the text from the image.

Optical Character Recognition (OCR): This activity extracts the text from the image. The OCR engine uses a variety of techniques to identify the text in the image, such as character recognition and pattern matching.

Text Cleaning: This activity removes any errors or artifacts from the extracted text. This may include removing any extra spaces, punctuation marks, or other characters that are not part of the text.

Database: This activity saves the cleaned text to a database. The database can be used to store the text for later use or to generate output files.

Output File Handling: This activity generates an output file from the text in the database. The output file can be a variety of formats, such as a text file, PDF document, or Word document.

Output: This activity returns the output file to the user.

The activity diagram also shows the following control flows:

The flow from the Input File Handling activity to the Text Processing activity indicates that the extracted text is passed to the text processing activity for preprocessing.

The flow from the Text Processing activity to the Optical Character Recognition (OCR) activity indicates that the preprocessed text is passed to the OCR engine for text extraction.

The flow from the Optical Character Recognition (OCR) activity to the Text Cleaning activity indicates that the extracted text is passed to the text cleaning activity for cleaning.

The flow from the Text Cleaning activity to the Database activity indicates that the cleaned text is saved to the database.

The flow from the Database activity to the Output File Handling activity indicates that the text in the database is used to generate the output file.

The flow from the Output File Handling activity to the Output activity indicates that the output file is returned to the user.

Chapter 4

REQUIREMENT SPECIFICATION

4.1 Software Requirement

- IDE: VS Code ,Tesseract Ocr Engine
- Browser: Chrome, Mozilla Firefox, Microsoft Edge.
- Database Management System: SQLite

4.2 Hardware Requirement

- **Personal Computer** , **Tesseract OCR** , **Translation API**
- **Storage:** Adequate storage space to store the dataset, user-uploaded videos, and processed data.

Chapter 5

SOFTWARE DEVELOPMENT APPROACH

5.1 Software Development Life Cycle Model

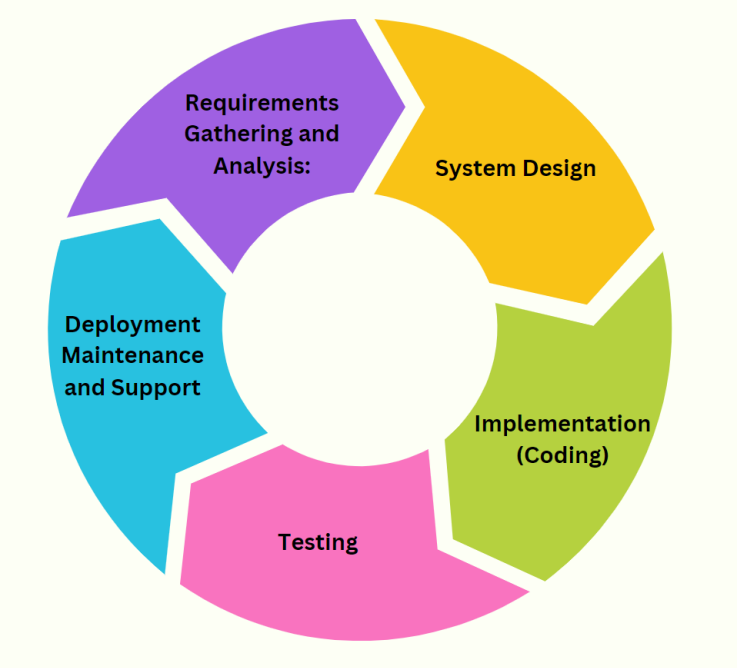


Fig : 5.1 SDLC

1. **Requirements Gathering and Analysis**

In the initial phase, the project kicks off with Requirements Gathering and Analysis. This involves conducting interviews with stakeholders to identify the project's goals and functionalities. The focus is on defining user requirements, system capabilities, and any constraints that may impact the development process.

2. **System Design**

Following the requirements phase, the System Design stage comes into play. Here, the project team plans the overall architecture and design of the mental health assessment system. This includes creating high-level system architecture and developing detailed designs for the user interface, backend, and database components.

3. **Implementation (Coding)**

Once the design is in place, the project moves into the Implementation or Coding phase. During this stage, the development team translates the design into a functional system by writing code for both frontend and backend components. Special attention is given to implementing video processing algorithms, questionnaire logic, and ensuring smooth database connectivity.

4. **Deployment**

The Testing phase is a crucial step to ensure the system works as intended. Various testing levels, including unit testing, integration testing, and system testing, are conducted to verify the functionality and interactions between different modules. User acceptance testing is performed to validate the system with real users.

5. **Maintenance and Support**

Upon successful testing, the Deployment phase is initiated. This involves setting up the server infrastructure for hosting the application, deploying the application and database, and ensuring data security measures are implemented.

The Maintenance and Support phase is ongoing, addressing issues, enhancing features, and providing continuous support. The project team monitors system performance, implements updates and patches, and offers user support while considering user feedback for improvements.

5.2 Technology Stack

- **Programming Languages:** Python, HTML,CSS JS
- **Python Libraries:**
  - Tesseract –Character Recognation
  - Tensorflow – Audio Processing
  - Pandas – Mathematical operations
  - Numpy -data manipulation
- **Frameworks:** Falsk,django

- **Database:** SQLite
- **Algorithms :** NLP ,CNN
- **API :** Translation API

Programming Languages:

- **Python:** A versatile, high-level programming language widely used for web development, machine learning, and data analysis.
- **Javascript:** A scripting language commonly employed for front-end web development, bringing interactivity to websites.
- **CSS (Cascading Style Sheets):** CSS is a style sheet language used for describing the presentation of a document written in HTML. It enables the separation of content from presentation, allowing developers to control the layout, colors, fonts, and other visual aspects of a webpage. CSS is used to enhance the design and appearance of web pages.

- **HTML (Hypertext Markup Language):** HTML is the standard markup language for creating and structuring web pages. It uses a system of elements represented by tags to define the structure of a webpage, such as headings, paragraphs, links, images, forms, and more.

Python Libraries:

- **Tesseract OCR (Optical Character Recognition)** : Tesseract is a powerful OCR (Optical Character Recognition) tool primarily used for extracting text from images and video frames, making it ideal for video processing applications.
- **Tensorflow:** An open-source machine learning library, utilized here for audio processing.
- **Pandas and Numpy:** Essential tools for data analytics, providing data manipulation and visualization capabilities.

Frameworks:

- **Flask :** Flask is a lightweight web framework for Python, suitable for small to medium-sized projects.
- **Django :** Django is a high-level web framework with built-in features, ideal for larger, data-driven applications.

Database:

- **SQLite**: SQLite is a lightweight, self-contained relational database system often used in embedded applications and mobile development due to its simplicity and portability..

Algorithms:

- **CNN (Convolutional Neural Network)**: A deep learning algorithm crucial for image and video analysis, employed here for facial expression recognition.
- **NLP (Natural Language Processing):** Focuses on the interaction between computers and human languages, potentially used for sentiment analysis in textual data.

API:

- **Translation API:** A Translation API is a tool that enables developers to integrate automatic language translation into their applications, facilitating multilingual communication and support.

Chapter 6

FINDINGS AND WORK PLAN FOR NEXT PHASE

By implementing the recommended features and considerations outlined in the previous response, this software can become a valuable resource for students, industries, the general public, police, judiciary, and customs officials. It will facilitate not only the translation of content but also the dissemination of knowledge in a manner that is both accurate and easy to understand.

Based on the image, the current phase of the project has focused on extracting text from images and other source materials into Indian languages. The system has been able to achieve this using a combination of Tesseract OCR and Python (Flask). The system is able to clean the extracted text and save it to a file.

Work plan for next phase

The next phase of the project will focus on translating the extracted text into Indian languages. The following tasks will be completed:

Identify a suitable translation API.

Develop a connector to the translation API.

Integrate the connector with the existing system.

Test the system to ensure that it is able to translate text accurately.

Deploy the system to production.

Timeline

The next phase of the project is expected to take 4 weeks to complete.

The following is a tentative timeline:

Week 1: Identify a suitable translation API and begin developing a connector.

Week 2: Complete the development of the connector and begin integrating it with the existing system.

Week 3: Test the system to ensure that it is able to translate text accurately.

Week 4: Deploy the system to production.

Resources

The following resources will be required for the next phase of the project:

Developer to develop and integrate the translation API connector.

QA engineer to test the system.

Production engineer to deploy the system to production.

Risks

The following risks have been identified for the next phase of the project:

The translation API may not be able to translate text accurately.

The integration of the translation API connector with the existing system may be complex.

There may be unexpected technical challenges during deployment.

Mitigation strategies

The following mitigation strategies will be employed to address the identified risks:

A thorough evaluation of the translation API will be conducted before it is selected.

The integration of the translation API connector with the existing system will be designed to be modular and flexible.

A detailed deployment plan will be developed and tested before the system is deployed to production.

REFERENCES

[1] 1. N. Jayanthi, A. Lakshmi, C. S. K. Raju, and B. Swathi, "Dual translation of international and Indian regional language using recent machine translation," in 2020 3rd International Conference on Intelligent Sustainable Systems (ICISS), December 2020, pp. 682-686.

[2] A. Godase and S. Govilkar, "Machine translation development for Indian languages and its approaches," International Journal on Natural Language Computing, vol. 4, no. 2, pp. 55-74, 2015.

[3] N. J. Khan, W. Anwar, and N. Durrani, "Machine translation approaches and survey for Indian languages," arXiv preprint arXiv:1701.04290, 2017.

[4] A. Kunchukuttan, A. Mishra, R. Chatterjee, R. Shah, and P. Bhattacharyya, "Sata-anuvadak: Tackling multiway translation of Indian languages," pan, vol. 841(54,570), pp. 4-135, 2014.

[5] S. K. Pulipaka, C. K. Kasaraneni, V. N. S. Vemulapalli, and S. S. M. Kosaraju, "Machine translation of English videos to Indian regional languages using open innovation," in 2019 IEEE International Symposium on Technology and Society (ISTAS), November 2019, pp. 1-7.

[6] S. Rajani, "Translation Across Cultures: From The Regional To The Universal," Think India Journal, vol. 22, no. 6, pp. 189-194, 2019.

[7] S. K. Dwivedi and P. P. Sukhadeve, "Machine translation system in Indian perspectives," Journal of Computer Science, vol. 6, no. 10, pp. 1111, 2010.

[8] J. Ray, "A review of terminological work being done in Indian Languages," in Proceedings of Translating and the Computer: Term banks for tomorrow's world, 1982.

[9] D. V. Sindhu and B. M. Sagar, "Study on machine translation approaches for Indian languages and their challenges," in 2016 International Conference on Electrical, Electronics, Communication, Computer and Optimization Techniques (ICEECCOT), December 2016, pp. 262-267.

[10] M. Singh and P. Bhatia, "Automated conversion of English and Hindi text to Braille representation," International Journal of Computer Applications, vol. 4, no. 6, pp. 25-29, 2010.

[11] J. C. Modh and J. Saini, "Study of Machine Translation Systems and Techniques for Indian Languages," The Journey of Indian Languages: Perspectives on Culture and Society, 2019.