Proposal for C4GT 2023

Name	Naveen Kumar
Contact Information *(Email, Phone Number & Github)*	navstr10@gmail.com +918292835299 https://github.com/stupiddint
Current occupation	Student
Education Details	Institute of Technical Education and Research, Bhubaneswar Pursuing Btech with CSIT
Technical skills with level *(level - Novice/Intermediate/Expert)*	Javascript - Intermediate React - Intermediate Node js - Intermediate Mongodb - Intermediate Git - Intermediate Java (DSA) - intermediate Linux - intermediate Python - novice + ML, DL, and NLP - Novice

TITLE: Developing an NPM module for Translation, Text Extraction, and Transliteration

Summary

This framework-agnostic library integrates with the Anuvaad and Bhashini APIs, offering seamless integration with popular web frontends like Angular and React, as well as content editors. The key functionalities include translation, text extraction, and transliteration, with a specific focus on PDF support. By leveraging the capabilities of Anuvaad and Bhashini, this empowers developers to effortlessly handle document translation, text extraction, and transliteration tasks.

The development approach involves setting up the project directory, integrating the Anuvaad and Bhashini APIs, and implementing core functionalities in separate modules. The library's functions are accessed by the application, which triggers API requests to Anuvaad and Bhashini. The API responses are processed by TransText and returned to the application for further use. Robust error-handling mechanisms and secure authentication are implemented to ensure a smooth and secure experience.

To ensure compatibility and ease of use, this npm module is designed to seamlessly integrate with Angular, React, and content editors. Adapters or plugins specific to each framework or content editor are created, along with clear instructions on how to integrate this as an NPM module. Continuous integration and deployment pipelines are set up for efficient development, and version control is managed using Git for collaboration and code reviews.

Project Detail

1. **Project Overview:**
   The project aims to develop a framework-agnostic library that exposes JavaScript functions to leverage the document text extraction, translation, and transliteration capabilities provided by Anuvaad and Bhashini. The library will be packaged as an NPM module that can be integrated with content editors and web frontends requiring Indian language support. The key goals include providing translation and transliteration capabilities for multiple file formats, with a focus on PDF support.

   a) **Understanding of the project**

The project aims to develop a framework-agnostic library that can be utilized as an NPM module, providing JavaScript functions for seamless integration with web frontends like Angular and React, as well as content editors. The library encompasses the features of translation and text extraction from document files, with a particular focus on prioritizing PDF files. These functionalities are achieved by utilizing the API endpoints provided by Anuvaad. Moreover, the library is designed to possess transliteration capabilities for Indic languages, which can be implemented by leveraging the API endpoints offered by Bhashini.

Goals:

- Framework Agnostic Library: Develop a JavaScript library that provides a standardized interface for translation and text extraction functionalities, ensuring compatibility with popular web frontends like Angular and React.
- Translation Capabilities: Utilize the API endpoints of Anuvaad to enable the seamless translation of document files, with a focus on supporting a variety of file formats, including PDF files.
- Text Extraction: Implement robust text extraction capabilities using Anuvaad's API endpoints, allowing developers to extract text from documents for further processing or analysis.
- Transliteration Support: Integrate Bhashini's API endpoints to enable transliteration capabilities for Indic languages, enhancing the versatility of the library and catering to a wider user base.
- Error Handling and Authentication: Ensure the library handles authentication and error handling effectively, providing informative error messages and securing API interactions.

Entities Involved

Before jumping straight into the implementation of the proposed library and its features, it is quintessential to first and foremost have a good understanding of the API being used in this project.

Anuvaad

Anuvaad is a document translation tool that leverages AI and machine learning technologies to accomplish end-to-end document translation. Project Anuvaad is REST API supports various functionalities related to user management, authentication, password management, file handling, workflow management, and document conversion. Here we are going to use its mainly two functionalities sentence translation and text extraction.

API documentation link

The API endpoints such as

POST   /anuvaad-etl/document-converter/v0/document-converter

API for creating digitized TXT and XLSX files format for translation flow. By utilizing this API, the process of creating digital files that contain the text to be translated can be automated. The generated files can then be used by translation tools or services to import the content for translation, facilitating the localization process. This API streamlines the generation and preparation of files for translation, making it easier to manage and automate translation workflows.

POST   /anuvaad-etl/document-converter/v0/document-exporter

This endpoint is related to the Document Converter in the Anuvaad ETL (Extract, Transform, Load) module. It allows to export documents in a specific format.

Bhashini

Bhashini uses ULCA API which is the largest repository of datasets of Indian languages developed by the Ministry of Electronics and Information Technology (MeiTY). Bhashini captures all data and model contributions through ULCA. Bhashini captures data and model contributions through ULCA and offers API endpoints for transliteration capabilities.

Anuvaad translator

The Anuvaad Translator is a service that translates documents sentence by sentence using a Neural Machine Translation (NMT) module. It receives tokenized sentences, sends them to NMT for translation, and appends the translations back to the document. It integrates with Translation Memory eXchange (TMX) for user-specific translations and User Translation Memory (UTM) for personalized translation caching.

Workflow of Anuvaad translator

The translator pushes sentences to OpenNMT, which are translated and pushed back during the document translation flow.

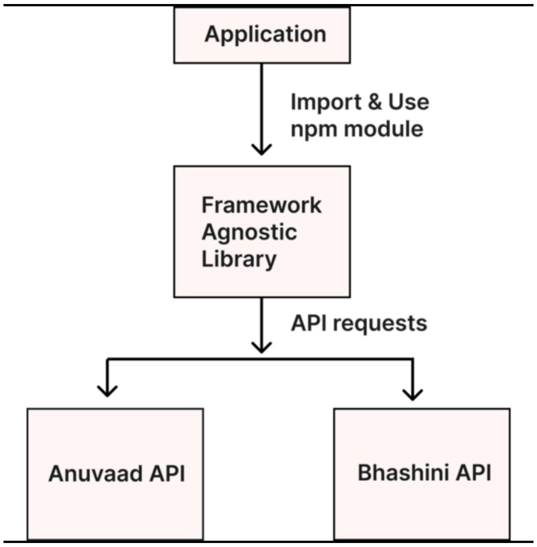
1. translator receives input from tokenizer module [input is a JSON file]
2. then extracted from JSON file
3. then sent to NMT over Kafka for translation

   [NMT expects a batch of 'n' sentences in one request, Translator created 'm' no of batches of 'n' sentences each]
4. then pushes to NPM input topic

   [In parallel it also listens to the NMT's output topic to receive the translation of the batches sent]
5. Once, all the 'm' no of batches are received back from the NMT, the translation of the document is marked complete.
6. Next, appends this translation back to the JSON file
7. then pushed to the content handler via API
8. content handler then stores this translation

2. Proposed work flow

The application interacts with the framework-agnostic library by calling its provided functions. The library, in turn, makes API requests to the Anuvaad and Bhashini APIs to perform translation, text extraction, and transliteration tasks. The API responses are then processed by the library and returned to the application.



The core functionalities of the library are translation, text extraction, and transliteration. These functionalities will be developed in different files to facilitate code readability and reusability. Creating different engines for different functionalities, and each functionality will be tested

using JTest or Cypress Test.
Using Axios or Fetch api, the required API endpoints will be called, and then the functions will be written with required parameters such as - sourceLanguage, targetLanguage, text .
For more flexibility in language choice, we will create a language parser.
For the fast response, different responses will be cached to avoid resending requests.

Writing the NPM module in JavaScript will not cause any problems when using it with Angular, React, or content editors. JavaScript is the primary language used for web development, and both Angular and React are JavaScript frameworks. Content editors also often support JavaScript-based plugins and extensions. However, to ensure compatibility with these frameworks and editors, tests will be done before publishing it.

3. Implementation

Prior to the coding period:

- Denotes the time between the announcement of selected contributors and before the commencement of the coding period.
- Interact with mentors and polish timelines and milestones further.
- Get to know and understand project deliverables better, decide upon regular meetings, and have a great one-on-one interaction while asking for feedback.

a) **Milestone 1: Core Functionality Implementation (Week 1-2)**

- Develop the translation engine using the Anuvaad API endpoints.
- Implement the text extraction engine using the Anuvaad API endpoints.
- Integrate the transliteration engine using the Bhashini API endpoints.
- Test each functionality with different scenarios and sample data to ensure accuracy.

b) **Milestone 2: Error Handling and Documentation (Week 3-4)**

- Implement error handling mechanisms to handle exceptions and provide meaningful error messages.
- Create comprehensive documentation for the library, including usage guidelines, API references, and code examples.
- Conduct thorough testing to ensure the reliability and stability of the library.

c) **Milestone 3: Compatibility and Integration (Week 5-6)**

- Implement error handling mechanisms to handle exceptions and provide meaningful error messages.
- Create comprehensive documentation for the library, including usage guidelines, API reference, and code examples.
- Conduct thorough testing to ensure the reliability and stability of the library.

d) **Milestone 4: Performance Optimization and Final Testing (Week 7-8)**

- Optimize the library's performance by implementing caching mechanisms and improving response times.
- Conduct final testing to validate the library's functionality, compatibility, and performance.
- Fine-tune any remaining issues or bugs to ensure a stable and reliable library.

4. Deliverables

- ❖ A framework-agnostic JavaScript library capable of seamless translation, text extraction, and transliteration using the Anuvaad and Bhashini APIs.
- ❖ Compatibility with Angular, React, and content editors, with clear instructions for integration.
- ❖ Thoroughly tested library with unit tests, integration tests, and performance testing.
- ❖ Optimized performance through caching mechanisms and response time improvements.

- Fully documented library with comprehensive usage guidelines and API reference.

Availability

The duration of the coding period is from July till August.

Number of hours available to dedicate to this project per week	8 hrs per day
	8X6 = 48 hrs per week
Do you have any other engagements during this period? (projects/internships)	No

When do your classes and exams finish?
I will have no class between July to August. As there is my 2nd year break from July to August, I can easily contribute and learn together if anything is necessary. And i can also give more time if needed for the project completion.

Personal Information

About Me:

I am Naveen Kumar, a second-year undergraduate student studying at the Institute of Technical Education and Research, Bhubaneswar, and pursuing Computer Science Engineering as my major. I am an official member of the official coding group ‘CODEX’ at my college. I have a keen interest in development and working on open-source projects. I have participated in two hackathons, winning third place in one of them in my first year. I have also participated in a symposium organized at my college.

What is your motivation for applying to this project?

At first glance, the project impressed me as a developer and individual. As a developer, this project provides me with a unique opportunity to contribute to language accessibility and empower users with multilingual support. It aligns perfectly with my passion for leveraging technology to bridge language barriers and make information more accessible to diverse audiences. The project's real-world impact, learning opportunities, and potential to make a difference inspired me as an individual, I have always wanted to work on a project like this.
I believe that this NPM package would greatly help developers, and I possess enough Javascript knowledge to take up this issue.

Do you plan to continue working on Project Anuvaad in the future?

Yes, I plan to keep working on Project Anuvaad in the future. I want to contribute to the project on different topics and with various languages because it's the kind of project I've always wanted to be a part of.

What if you are not chosen for this project at C4GT?

This is the only project I am writing a proposal for, it may not be up to par as I knew about it only 4 days before submission deadline. I liked this whole project, and I will love to still contribute under the guidance of the mentors, fellow contributors, and the project team for free.

Previous experience/open source projects :

I have been doing web development for the past 20+ months, and I am a full-stack web developer with the MERN stack. I am also a beginner in ML and NLP. I have written a research paper for Healthacre Associated Infections (HAI) in my college. As I already have a good amount of exposure to related technologies, I believe this project would be a great chance to use my skills to make a good impact. I have a very good knowledge of Javascript from beginning to advance and working on it for last 1.5 years. And I have also contributed in opensource projects.

Project Name	Project Description	Links (if any)
LinksHub	It's open-source, i contributed the api for icons. (issue #258)	https://github.com/rupali-codes/LinksHub
Blog app	A full stack MERN project	https://github.com/stupiddint/Blog-app
HAI Identification	Ml project, identifies infections on given inputs	https://github.com/stupiddint/Symposium-source-code