MALLA REDDY COLLEGE OF ENGINEERING AND TECHNOLOGY

SOFTWARE REQUIREMENTS SPECIFICATION (SRS)

for

Chatnslate: The Language-Free Chat Experience

Prepared by

S. No	Roll Number	Student Name
1.	22N31A6627	Bheemanapally Abhaya Sri
2.	22N31A6629	Bochkar Nikhith
3.	22N31A6666	Jakkireddy Sri Charan Reddy

Supervisor:	**Mr. S. Venkateswara Raju**
Designation:	**Assistant Professor**
Department:	**Computational Intelligence**
Batch ID:	
Date:	
Supervisor Sign. & Date	

Department of Computational Intelligence

Title of the Project: Chatnslate : The Language - Free Chat Experience

Content

Revisions

Version	Primary Author(s)	Description of Version	Date Completed
1.0	B. Abhaya sri	Primary Revision giving an overall view of the project and document.	

1 Introduction

The Live Language Translator WebApp is an innovative communication tool designed to bridge language gaps and enable seamless conversations between people from different linguistic backgrounds. Built using Python and advanced Natural Language Processing (NLP) technologies, this web-based application offers real-time message translation, allowing users to chat effortlessly in their native languages. Whether for travel, international collaboration, language learning, or making new global connections, the webapp provides a smooth, chat-like experience similar to platforms like WhatsApp or Messenger. With instant translation and integrated messaging in one platform, it removes the need for switching between apps or manual translation, making cross-language communication natural, efficient, and accessible to everyone.

1.1 Document Purpose

This document defines the requirements and specifications of Chatnslate, a real-time multilingual chat web application. It is intended to guide developers, stakeholders, and testers in understanding the project's functionality, design, and scope.

1.2 Project/Product Scope

The system will:

- A web-based chat platform with real-time translation.
- Allows users to chat in their native language, while automatically translating into the recipient's language.
- Provides a **WhatsApp/Messenger-like experience** with smooth, instant messaging.
- Built with **Python, NLP, and web technologies**.
- Target users: Travelers, international teams, students, and individuals seeking cross-language communication.

1.3 Existing System

- Users currently rely on Google Translate or similar tools.
- Requires copy-paste, which disrupts conversation flow.
- Lacks built-in, real-time translation in messaging apps.

1.4 Problems with Existing System

- Breaks chat continuity.
- Leads to slow communication.
- Causes misunderstandings in group/international chats.
- Excludes users who are not fluent in common languages (like English).

1.5 Proposed system

- Chatnslate: a web application with automatic real-time translation.
- Python + NLP powered translation.
- Integrated messaging + translation in one platform.
- WebSocket for real-time chat.
- Smooth interface, no switching apps required.

1.6 Advantages of Proposed Systems

- Seamless multilingual conversations.
- Enhances global communication.
- Saves time by eliminating manual translation.
- Suitable for travel, collaboration, friendships, education.

2 Overall Description

2.1 Feasibility Study

- Technical Feasibility
    - Built with Next.js, WebSocket, REST APIs, and Gemini AI API.
- Economic Feasibility
    - Uses open-source frameworks and scalable cloud deployment.
- Operational Feasibility
    - Easy to use, similar to common messaging apps.

2.2 Product Functionality

- User authentication (Clerk).
- Real-time messaging (WebSocket).
- Instant translation via Gemini AI API.
- Cross-platform support (web, mobile-friendly).

2.3 Design and Implementation Constraints

- Dependent on internet connectivity.
- Limited to supported languages of translation API.
- Requires secure authentication to protect user data.

2.4 Assumptions and Dependencies

- Users have **internet access** and modern browsers.
- Translation API (Gemini AI) provides accurate translations.
- Server uptime and reliability are maintained.

Conclusion

The Live Language Translator webapp is a user-friendly chat platform that helps people from different countries talk to each other, even if they don't speak the same language. It automatically translates messages in real time, so conversations feel smooth and natural, just like chatting on WhatsApp or Messenger. Built with Python and smart language technology (NLP), it makes global communication easy and instant — perfect for travel, teamwork, learning new languages, or making new friends around the world.

3 Functional Requirements

3.1 Software Requirement Specifications

- Google Gemini AI API for translation services
- Clerk authentication service
- Next.js framework for web application
- WebSocket protocol for real-time messaging
- RESTful API endpoints for data operations

3.2 Hardware Requirements Specifications

- Processor: Intel i5 or above
- **Memory**: 4GB RAM minimum
- **Storage**: 128GB HDD/SSD
- **Network**: Stable internet connection

3.3 UML Diagrams

3.3.1 Usecase Diagram:

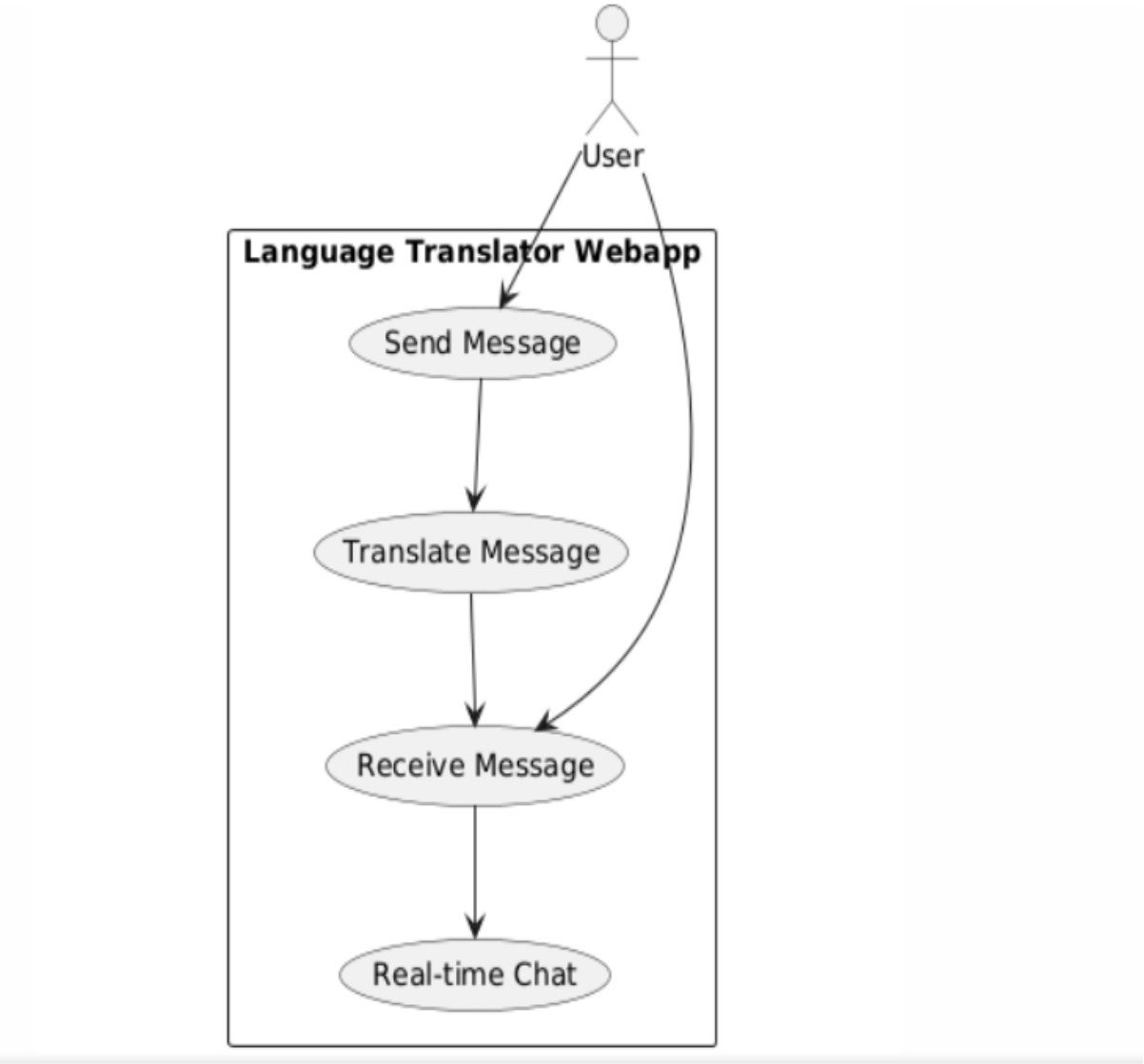

fig1: Use-case Diagram

3.3.2 Activity Diagram:

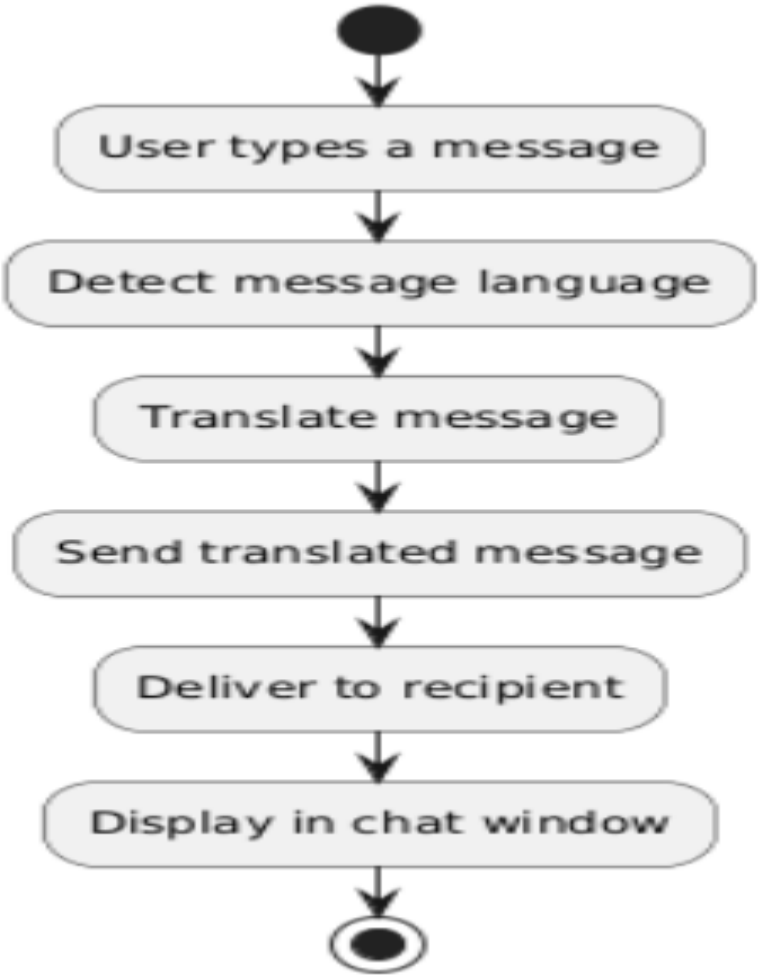

fig 2: Activity Diagram

3.3.3 Class Diagram:

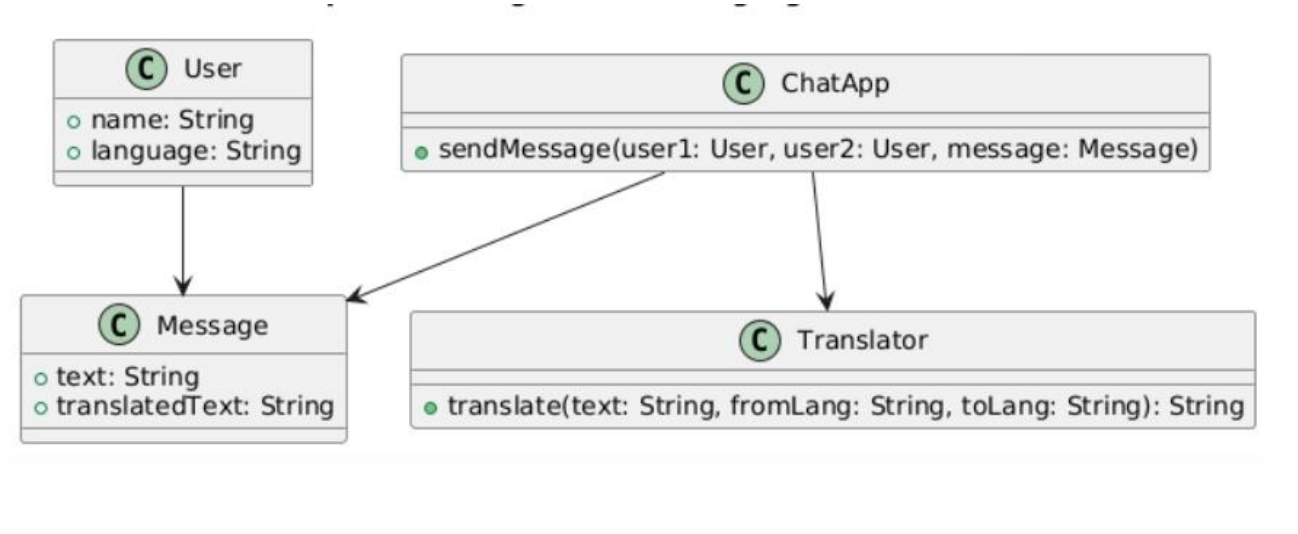

fig 3: Class Diagram

3.3.4 Sequence Diagram:

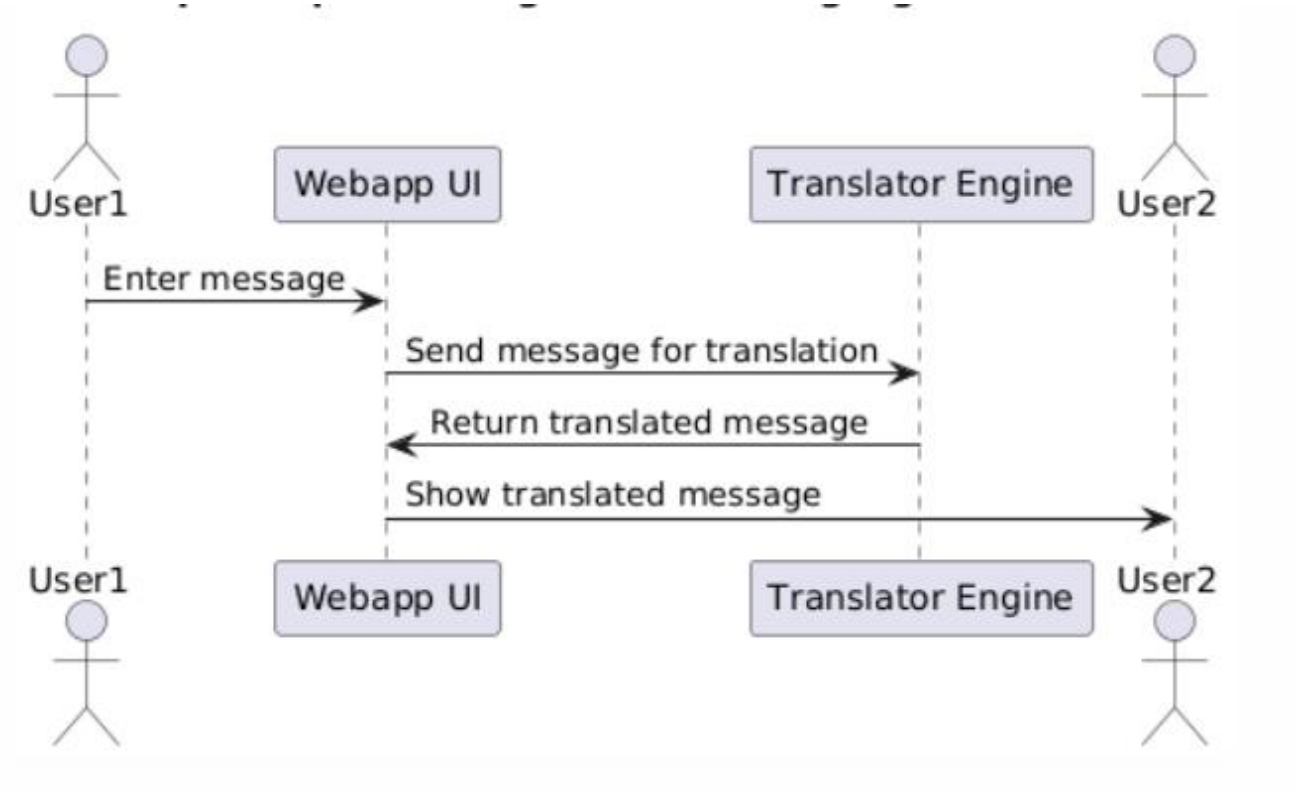

fig 4: Sequence Diagram

4 Other Non-functional Requirements

4.1 Performance Requirements

- Real-time message delivery (<1 second latency).
- Translation response time: under 2 seconds.

4.2 Safety and Security Requirements

- User authentication with Clerk.
- **Secure communication** (HTTPS, encryption).
- **Data privacy** – no storage of sensitive messages.

4.3 Software Quality Attributes

The following attributes are critical for ensuring the platform's usability, maintainability, and scalability.

- **Reliability**: 24/7 availability.
- **Usability**: Simple, user-friendly UI.
- **Maintainability**: Modular codebase.
- **Scalability**: Can handle multiple concurrent users.

5 Other Requirements

5.1 Database Requirements:

- Store user profiles, chat history, language preferences.
- Database: SQL/NoSQL (e.g., Firebase or MongoDB).

5.2 Internationalization Requirements:

- Support multiple global languages.
- Dynamic language detection.

5.3 Legal Requirements:

Legal considerations for the project include:

- Must comply with data protection laws (GDPR, IT Act).
- Respect copyright/licensing of translation API.

5.4 Reuse Objectives:

- Reusable API modules for chat and translation.
- Modular architecture for **future expansion**

5.5 Development Environment Requirements:

- OS: Windows/Linux/Mac
- **IDE**: VS Code, PyCharm
- **Tools**: GitHub, Docker (optional for deployment)

5.6 Documentation Requirements:

- Developer documentation for APIs.
- User manual for application usage.
- UML diagrams for future maintenance.

6 References

- Google Gemini AI API Documentation
- Next.js Official Documentation
- Clerk Authentication Service Docs
- WebSocket Protocol Standards
- Research papers on NLP-based Translation Systems

SRS DOCUMENT REVIEW

CERTIFICATION

This Software Requirement Specification (SRS) Document is reviewed and certified to proceed for the project development by the Departmental Review Committee (DRC).

Date of SRS Submitted:	
Date of Review:	
Supervisor Comments:	
Supervisor Sign. & Date.	
Coordinator Sign. & Date	
HOD Sign. & Date	
Dept. Stamp	