**1. INTRODUCTION**

**1.1. Project Overview**

TransLingua is an AI-powered language translation web application designed to provide accurate and context-aware translations based on user input. The application allows users to enter text, select the source and target languages, and instantly receive translated output. By leveraging a pre-trained generative AI model, the system delivers natural, grammatically correct, and contextually meaningful translations within seconds.

The project aims to simplify multilingual communication for students, professionals, researchers, travelers, and content creators. Instead of relying on basic rule-based translators that often produce literal or inaccurate results, users can depend on TransLingua to generate high-quality translations quickly and efficiently. The application is built using Streamlit for the user interface and integrates the Google Gemini 2.5 Flash model (models/gemini-2.5-flash-latest) for fast and reliable AI-driven translation.

Overall, TransLingua demonstrates the practical implementation of generative AI in multilingual communication by providing a user-friendly, secure, and time-saving solution for real-time language translation.

**1.2. Objectives**

The main objectives of the TransLingua: AI-Powered Language Translator project are:

* To develop a web-based application that performs multilingual translation using artificial intelligence.
* To allow users to translate text between multiple languages by selecting source and target languages.
* To reduce the time and effort required for manual translation and cross-language communication.
* To integrate a pre-trained generative AI model (Google Gemini 2.5 Flash) for fast, accurate, and context-aware translation.
* To provide a simple, intuitive, and user-friendly interface using Streamlit.
* To enhance user experience by delivering natural, grammatically correct, and meaningful translations in real time.

**2. Ideation Phase**

**2.1. Problem Statement**

In today’s global digital ecosystem, individuals frequently need to communicate across language barriers, whether for education, business, research, or social interaction. However, many existing translation tools lack contextual understanding, often generating literal translations that may be inaccurate or awkward. Additionally, some platforms have complex interfaces that reduce usability and make the translation process less efficient. Therefore, users require a fast, accurate, and AI-powered translation solution that supports multiple global languages, understands contextual meaning rather than word-by-word translation, provides an intuitive and accessible user interface, and ensures secure handling of user input.



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Problem  Statement  (PS) | I am | I’m trying to | But | Because | Which makes me feel |
| PS-1 | A student who needs to translate academic content and research materials. | Translate text accurately and quickly between multiple languages for assignments and international collaboration. | Existing translation tools often provide literal translations and fail to capture contextual meaning. | Many platforms rely on basic rule-based or statistical methods instead of advanced AI models that understand context and tone. | Frustrated, uncertain about accuracy, and less confident while submitting academic work. |
| PS-2 | A working professional communicating with international clients. | Translate business emails and documents clearly and professionally. | Available online translators sometimes generate awkward or grammatically incorrect sentences. | They lack advanced AI-driven contextual understanding and domain awareness. | Unsure about communication quality and worried about professional credibility |

**2.2. Empathy Map Canvas**

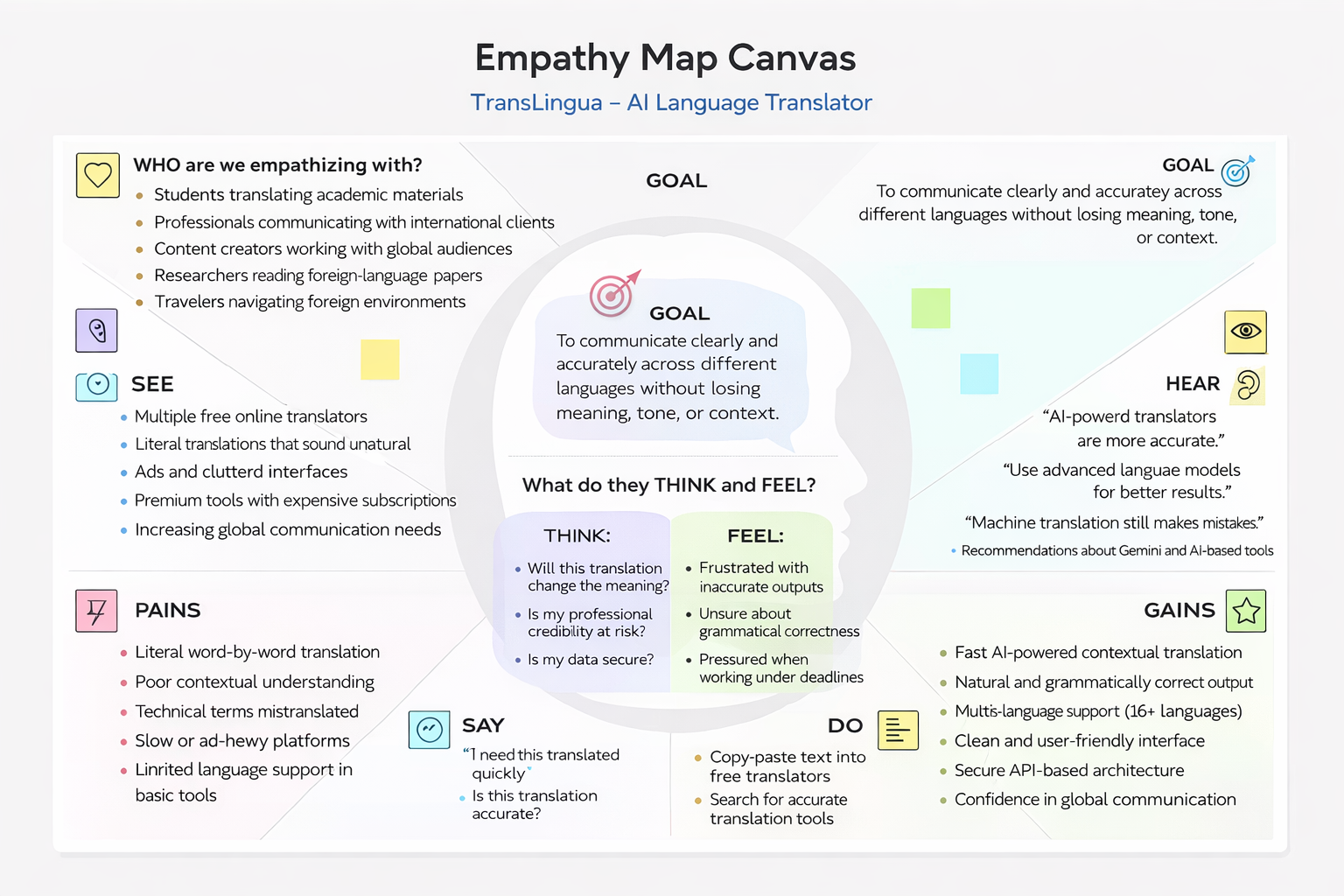
**Empathy Map Canvas:**

An empathy map is a simple, easy-to-digest visual that captures knowledge about a user’s behaviours and attitudes.

It is a useful tool to helps teams better understand their users.

Creating an effective solution requires understanding the true problem and the person who is experiencing it. The exercise of creating the map helps participants consider things from the user’s perspective along with his or her goals and challenges.

**Example:**



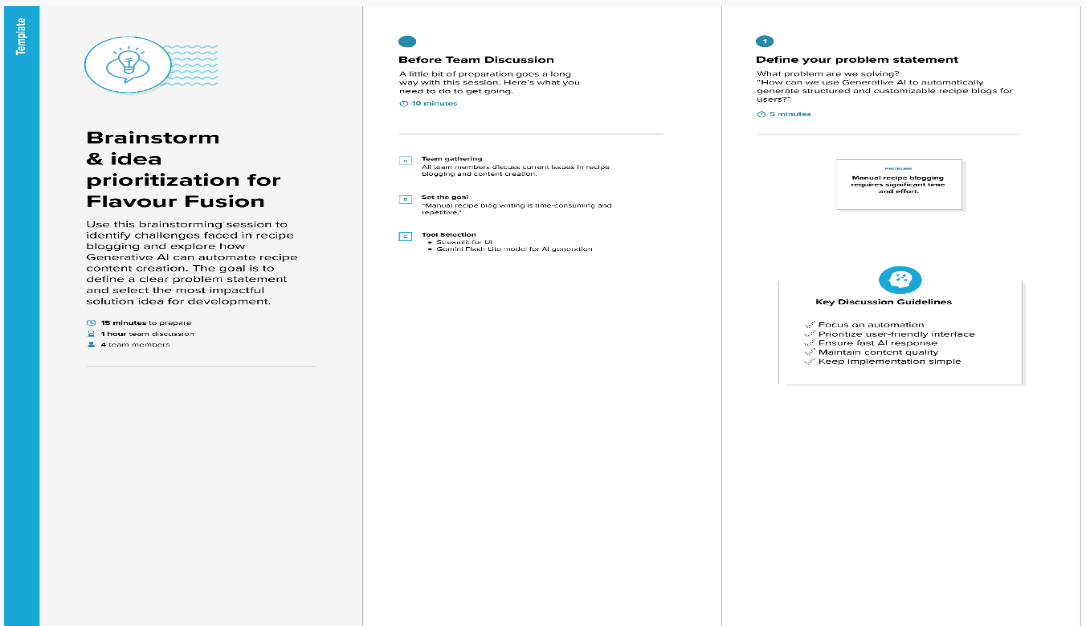
**2.3 Brainstorming**

**Brainstorm & Idea Prioritization:**

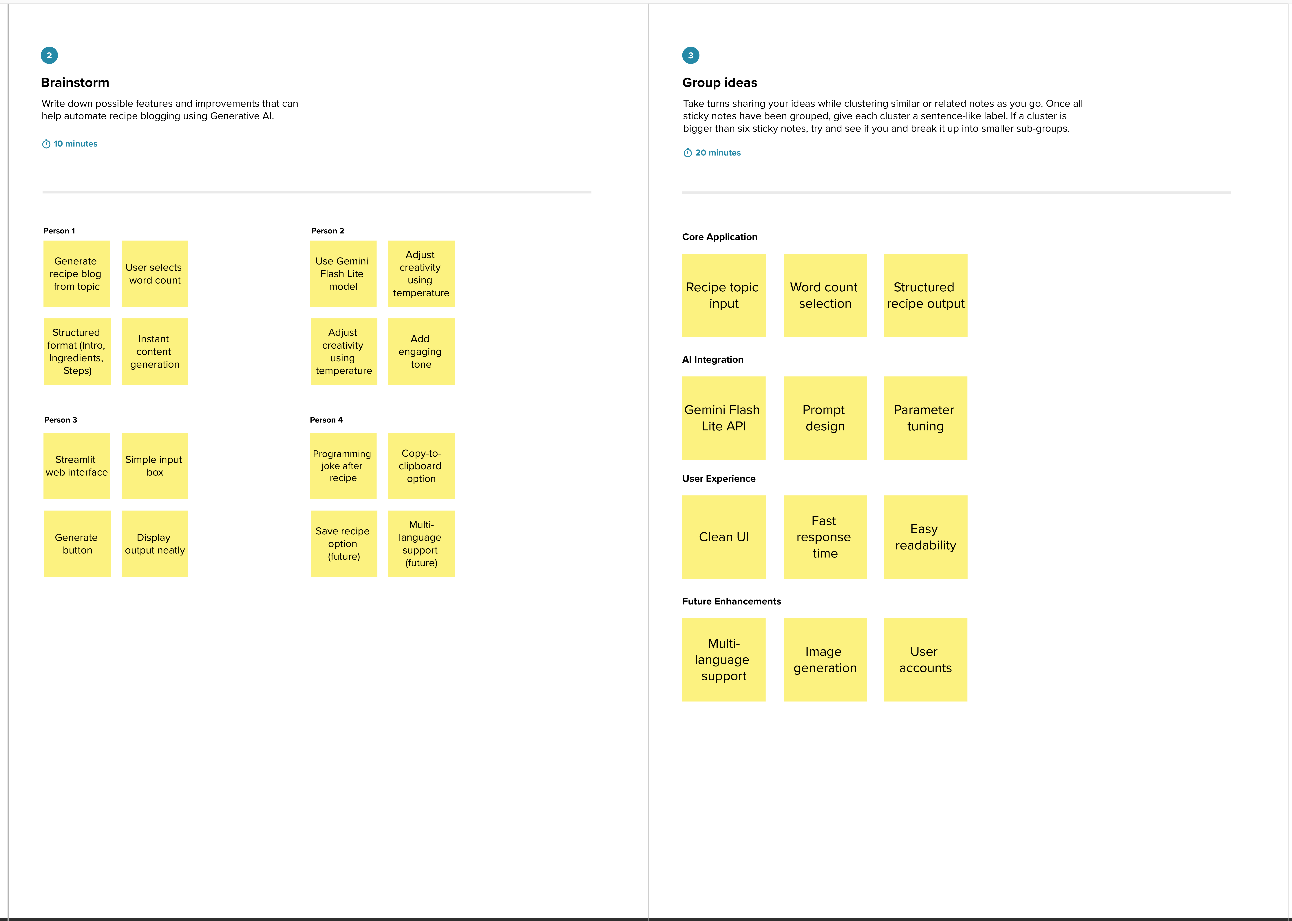
Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem solving. Prioritizing volume over value, out-of-the-box ideas are welcome and built upon, and all participants are encouraged to collaborate, helping each other develop a rich amount of creative solutions.

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

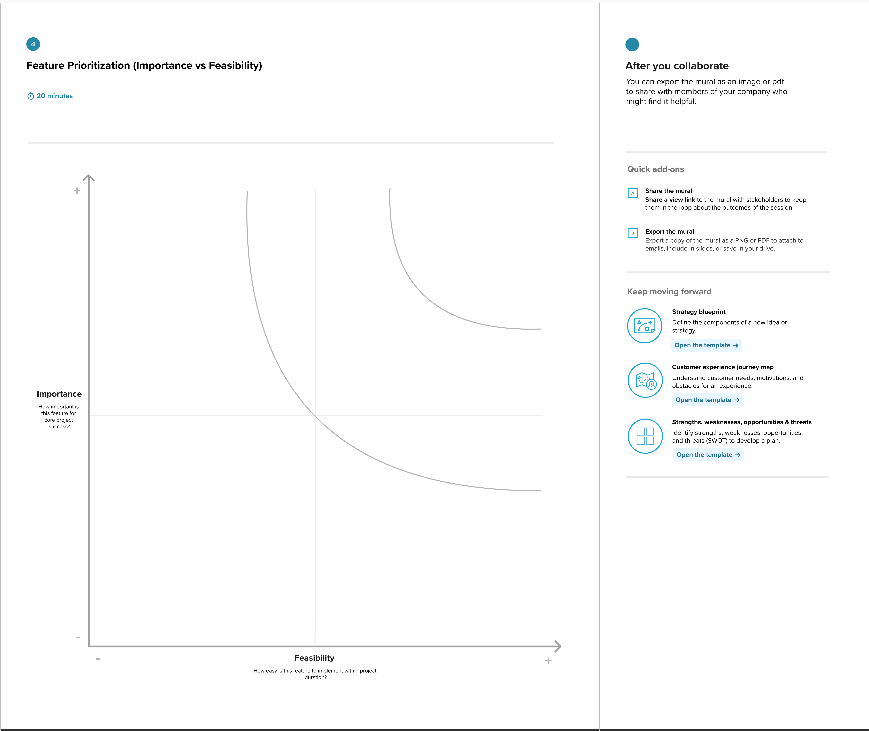
**Step-1: Team Gathering, Collaboration and Select the Problem Statement**



**Step-2: Brainstorm, Idea Listing and Grouping**



**Step-3: Idea Prioritization**

****

**3. Requirement Analysis**

**3.1. Solution Requirement**

**Functional Requirements:**

Following are the functional requirements of the proposed solution.

|  |  |  |
| --- | --- | --- |
| **FR No.** | **Functional Requirement (Epic)** | **Sub Requirement (Story / Sub-Task)** |
| FR-1 | |  | | --- | |  |  |  | | --- | | Text Input Module | | |  |  | | --- | --- | |  | Enter text to be translated |  |  |  |  | | --- | --- | --- | |  |  | Select source language |  |  |  |  | | --- | --- | --- | |  |  | Select target language | |
| FR-2 | Input Validation | |  | | --- | | Validate empty text input |  |  |  |  | | --- | --- | --- | |  |  | Ensure source and target languages are selected |  |  |  |  | | --- | --- | --- | |  |  | Display appropriate error messages | |
| FR-3 | AI Integration | |  | | --- | | Connect to Google Gemini API |  |  |  |  | | --- | --- | --- | |  |  | Send structured translation prompt |  |  |  |  | | --- | --- | --- | |  |  | Handle API response and fallback if needed | |
| FR-4 | Translation Processing | |  | | --- | | Generate context-aware translated text |  |  |  |  | | --- | --- | --- | |  |  | Preserve tone and meaning | |
| FR-5 | Output Display | |  | | --- | | Display translated text in UI |  |  |  |  | | --- | --- | --- | |  |  | Enable copy-to-clipboard feature | |
| FR-6 | |  | | --- | |  |  |  | | --- | | Deployment | | |  | | --- | | Deploy application on Streamlit Cloud |  |  |  |  | | --- | --- | --- | |  |  | Make application accessible via URL | |

**Non-functional Requirements:**

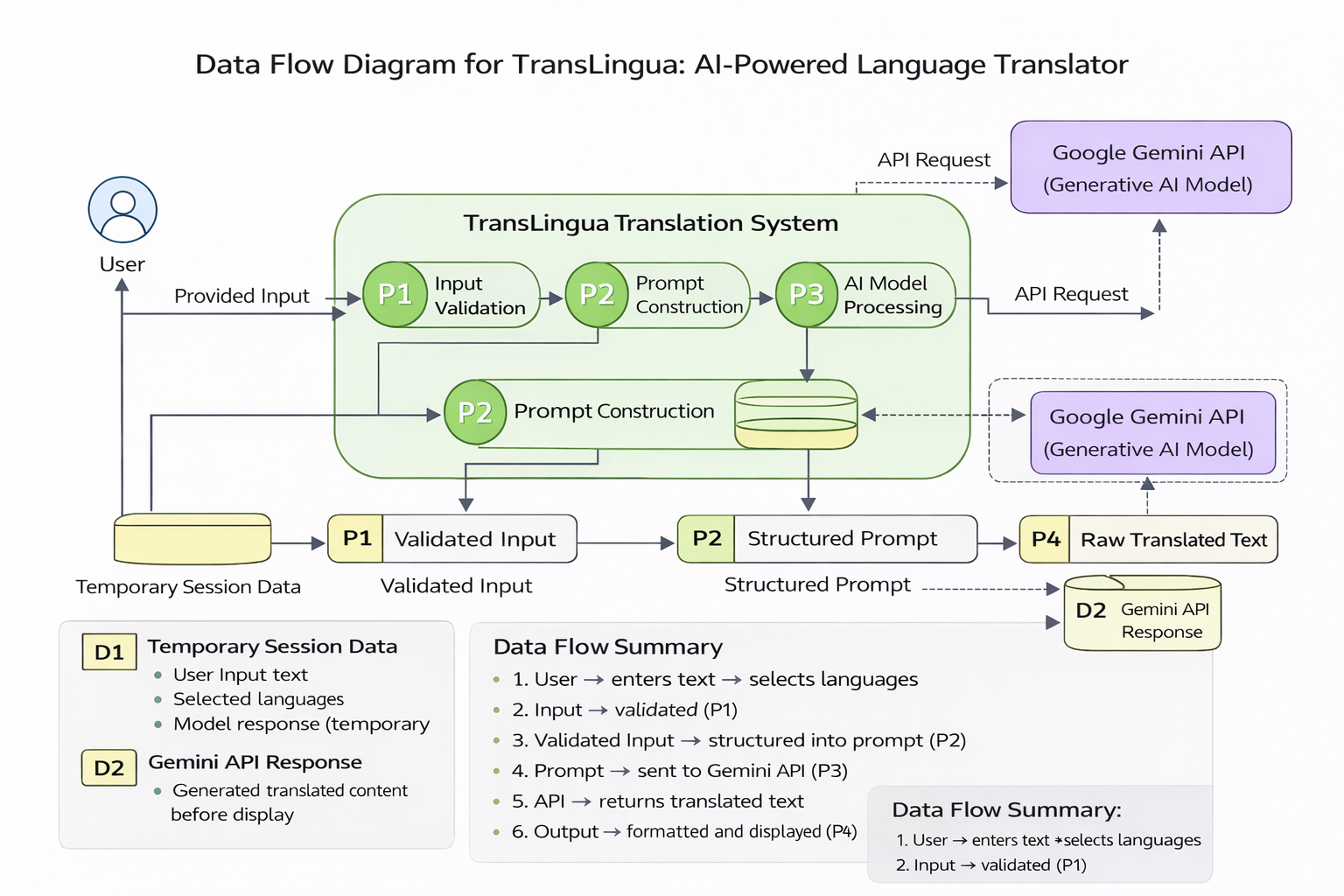
Following are the non-functional requirements of the proposed solution.

|  |  |  |
| --- | --- | --- |
| **FR No.** | **Non-Functional Requirement** | **Description** |
| NFR-1 | **Usability** | The application must have a simple and intuitive Streamlit interface that is easy to use. |
| NFR-2 | **Security** | API keys must be securely stored and not exposed in the frontend. |
| NFR-3 | **Reliability** | The system should generate consistent and structured outputs for valid inputs. |
| NFR-4 | **Performance** | Translation should be generated within a few seconds under normal network conditions. |
| NFR-5 | **Availability** | The application should be accessible online whenever deployed. |
| NFR-6 | **Scalability** | The system should handle multiple users without significant performance degradation. |

**3.2. Data Flow Diagram**

**Data Flow Diagrams:**

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.

****

**User Stories**

Use the below template to list all the user stories for the product.

| **User Type** | **Functional Requirement (Epic)** | **User Story Number** | **User Story / Task** | **Acceptance criteria** | **Priority** | **Release** |
| --- | --- | --- | --- | --- | --- | --- |
| Customer (Mobile user) | User Interface Setup | USN-1 | As a user, I can access a Streamlit-based interface to enter text and select source and target languages. | I can enter text and choose languages successfully. | High | Sprint-1 |
| Administrator | Input Validation | USN-2 | |  | | --- | |  |  |  | | --- | | As a user, I want the system to validate my input text and selected languages before translation. | | I receive validation messages for empty or incorrect inputs. | High | Sprint-1 |
| System | AI Model Integration | USN-3 | As a user, I want the system to translate text using the Google Gemini API. | The system successfully integrates with Gemini and returns translated text. | High | Sprint-2 |
| Administrator | |  | | --- | |  |  |  | | --- | | Prompt Engineering | | USN-4 | As a developer, I want to construct structured prompts to ensure contextual and accurate translation. | Prompts are correctly formatted and produce context-aware translations. | Medium | Sprint-2 |
| Customer (Mobile user) | Output Display | USN-5 | |  | | --- | |  |  |  | | --- | | As a user, I want to view the translated text clearly on the screen. | | Translated output is displayed in a readable format. | High | Sprint-3 |
| Customer Care Executive | Deployment | USN-6 | As a developer, I want to deploy the application on Streamlit Cloud so users can access it online. | successfully deployed and accessible via URL. | Medium | Sprint-3 |

**3.3. Technology Stack**

**Technical Architecture:**

The Deliverable shall include the architectural diagram as below and the information as per the table1 & table 2

**Table-1 : Components & Technologies:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Component** | **Description** | **Technology** |
|  | User Interface | Web-based interface where users enter text and select source and target languages | Streamlit (Python Web Framework) |
|  | Application Logic-1 | Input validation and prompt construction logic | Python |
|  | Application Logic-2 | AI request handling and response processing | Google Generative AI API |
|  | Application Logic-3 | Translation formatting and output structurin | Python |
|  | Temporary Data Handling | Stores user input and session data temporarily | Streamlit Session State |
|  | External API-1 | Generative AI service for language translation | Gemini 2.5 Flash (models/gemini-2.5-flash) |
|  | Machine Learning Model | Pre-trained generative AI model for contextual translation | Google Gemini Model |
|  | Infrastructure (Server / Cloud) | Deployment of application | Streamlit Cloud / Local Deployment |

**Table-2: Application Characteristics:**

| **S.No** | **Characteristics** | **Description** | **Technology** |
| --- | --- | --- | --- |
|  | Open-Source Frameworks | Web framework and development tools used | Streamlit, Python |
|  | Security Implementations | Secure storage of API keys and environment variables | Environment Variables (.env), Streamlit Secrets |
|  | Scalable Architecture | Web-based architecture supporting multiple users | Cloud-based deployment (Streamlit Cloud) |
|  | Availability | Application accessible online after deployment | Streamlit Cloud Hosting |
|  | Performance | Fast response generation using lightweight AI model | Gemini Flash Lite (optimized for low latency) |

**4. PROJECT DESIGN**

**4.1. Problem Solution Fit**

**Problem – Solution Fit Template:**

The Problem-Solution Fit simply means that you have found a problem with your customer and that the solution you have realized for it actually solves the customer’s problem. It helps entrepreneurs, marketers and corporate innovators identify behavioral patterns and recognize what would work and why

**Purpose:**

* Solve complex problems in a way that fits the state of your customers.
* Succeed faster and increase your solution adoption by tapping into existing mediums and channels of behavior.
* Sharpen your communication and marketing strategy with the right triggers and messaging.
* Increase touch-points with your company by finding the right problem-behavior fit and building trust by solving frequent annoyances, or urgent or costly problems.
* **Understand the existing situation in order to improve it for your target group.**

Calendar

Description automatically generated**Template:**

**4.2. Proposed Solution**

**Proposed Solution Template:**

Project team shall fill the following information in the proposed solution template.

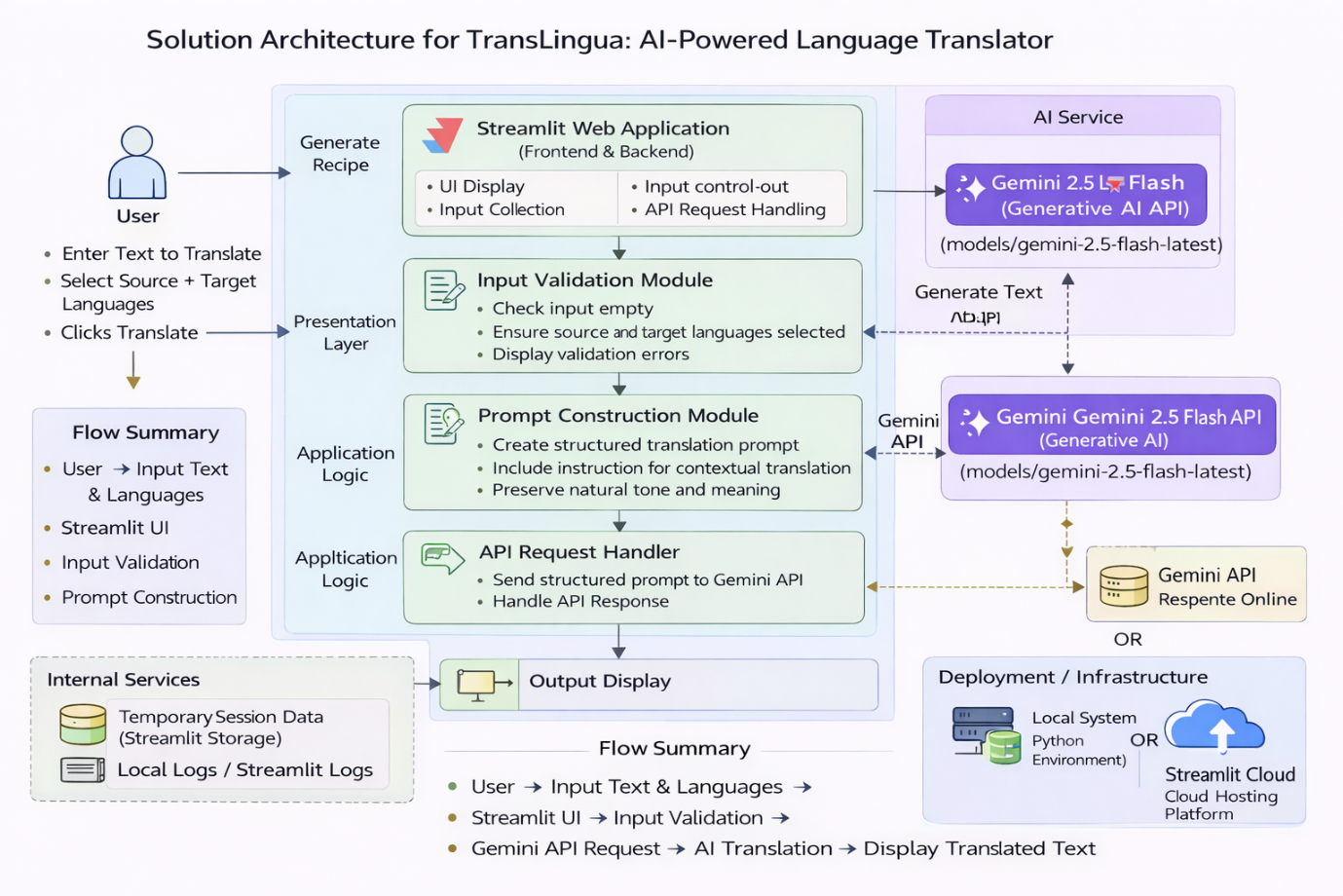
|  |  |  |
| --- | --- | --- |
| **S.No.** | **Parameter** | **Description** |
|  | Problem Statement (Problem to be solved) | Many students, professionals, and global users struggle to communicate effectively across different languages. Existing translation tools often provide literal translations, lack contextual understanding, and sometimes produce grammatically incorrect or unnatural outputs. There is a need for an AI-powered system that delivers accurate, context-aware, and fast translations through a simple and user-friendly interface.. |
|  | Idea / Solution description | TransLingua is an AI-powered web application that translates text between multiple languages using Google’s Gemini 2.5 Flash model. Users enter text, select source and target languages, and receive context-aware translations instantly. The system constructs structured prompts and processes them via the Gemini API to generate accurate and natural-sounding translations through a clean Streamlit-based interface. |
|  | Novelty / Uniqueness | Unlike traditional rule-based translators, TransLingua leverages advanced Generative AI to understand context, tone, and meaning rather than translating word-by-word. The system dynamically generates natural translations in real-time and provides a modern, responsive interface with secure API integration. |
|  | Social Impact / Customer Satisfaction | The solution enhances global communication by enabling students, researchers, professionals, and travelers to translate content accurately and efficiently. It improves productivity, reduces misunderstandings, and builds confidence in multilingual communication. The simple interface makes advanced AI translation accessible to non-technical users. |
|  | Business Model (Revenue Model) | The application can adopt a freemium model where basic translation is free, and advanced features (higher request limits, document translation, tone customization, enterprise API usage) are available through subscription plans. Revenue can also be generated through API licensing or enterprise integration. |
|  | Scalability of the Solution | The application is cloud-deployable and uses a scalable generative AI API. It can handle multiple users simultaneously when deployed on cloud infrastructure. |

**4.3 Solution Architecture**

**Solution Architecture:**

The solution architecture of TransLingua consists of a Streamlit-based web application that collects user input (text, source language, and target language) and validates it. The application constructs a structured translation prompt and sends it to the Google Gemini 2.5 Flash generative AI model through a secure API call.

The AI model processes the request and generates context-aware translated text. The application then formats the response and displays it clearly to the user. Temporary session data is managed using Streamlit Session State. The system can be deployed locally or on a cloud platform such as Streamlit Cloud

****

**5.PROJECT PLANNING & SCHEDULING**

**5.1. Project Planning**

**Product Backlog, Sprint Schedule, and Estimation (4 Marks)**

Use the below template to create product backlog and sprint schedule

| **Sprint** | **Functional Requirement (Epic)** | **User Story Number** | **User Story / Task** | **Story Points** | **Priority** | **Team Members** |
| --- | --- | --- | --- | --- | --- | --- |
| Sprint-1 | User Interface Setup | USN-1 | As a user, I can access a Streamlit-based interface to enter text and select source and target languages. | 2 | High | All Team Members |
| Sprint-1 | Input Validation | USN-2 | |  | | --- | |  |  |  | | --- | | As a user, I want the application to validate text input and language selection before translation. | | 1 | High | All Team Members |
| Sprint-2 | AI Model Integration | USN-3 | |  | | --- | |  |  |  | | --- | | As a user, I want the system to translate text using Google Gemini 2.5 Flash model. | | 3 | High | All Team Members |
| Sprint-2 | |  | | --- | |  |  |  | | --- | | Prompt Engineering | | USN-4 | |  | | --- | |  |  |  | | --- | | As a developer, I want to construct structured prompts for context-aware translation | | 1 | Medium | All Team Members |
| Sprint-3 | Output Display | USN-5 | |  | | --- | |  |  |  | | --- | | As a user, I want to view the translated text clearly on the screen. | | 2 | High | All Team Members |
| Sprint-3 | Deployment | USN-6 | As a user, I want the application to be deployed and accessible online. | 2 | Medium | All Team Members |

**Project Tracker, Velocity & Burndown Chart: (4 Marks)**

| **Sprint** | **Total Story Points** | **Duration** | **Sprint Start Date** | **Sprint End Date (Planned)** | **Story Points Completed (as on Planned End Date)** | **Sprint Release Date (Actual)** |
| --- | --- | --- | --- | --- | --- | --- |
| Sprint 1 | 20 | 4 Days | 28 January 2026 | 31 January 2026 | 20 | 31 January 2026 |
| Sprint 1 | 20 | 4 Days | 28 January 2026 | 31 January 2026 | 20 | 31 January 2026 |
| Sprint 2 | 20 | 8 Days | 02 February 2026 | 09 February 2026 | 20 | 09 February 2026 |
| Sprint 2 | 20 | 8 Days | 02 February 2026 | 09 February 2026 | 20 | 09 February 2026 |
| Sprint 3 | 20 | 7 Days | 12 February 2026 | 18 February 2026 | 20 | 18 February 2026 |
| Sprint 3 | 20 | 7 Days | 12 February 2026 | 18 February 2026 | 20 | 18 February 2026 |

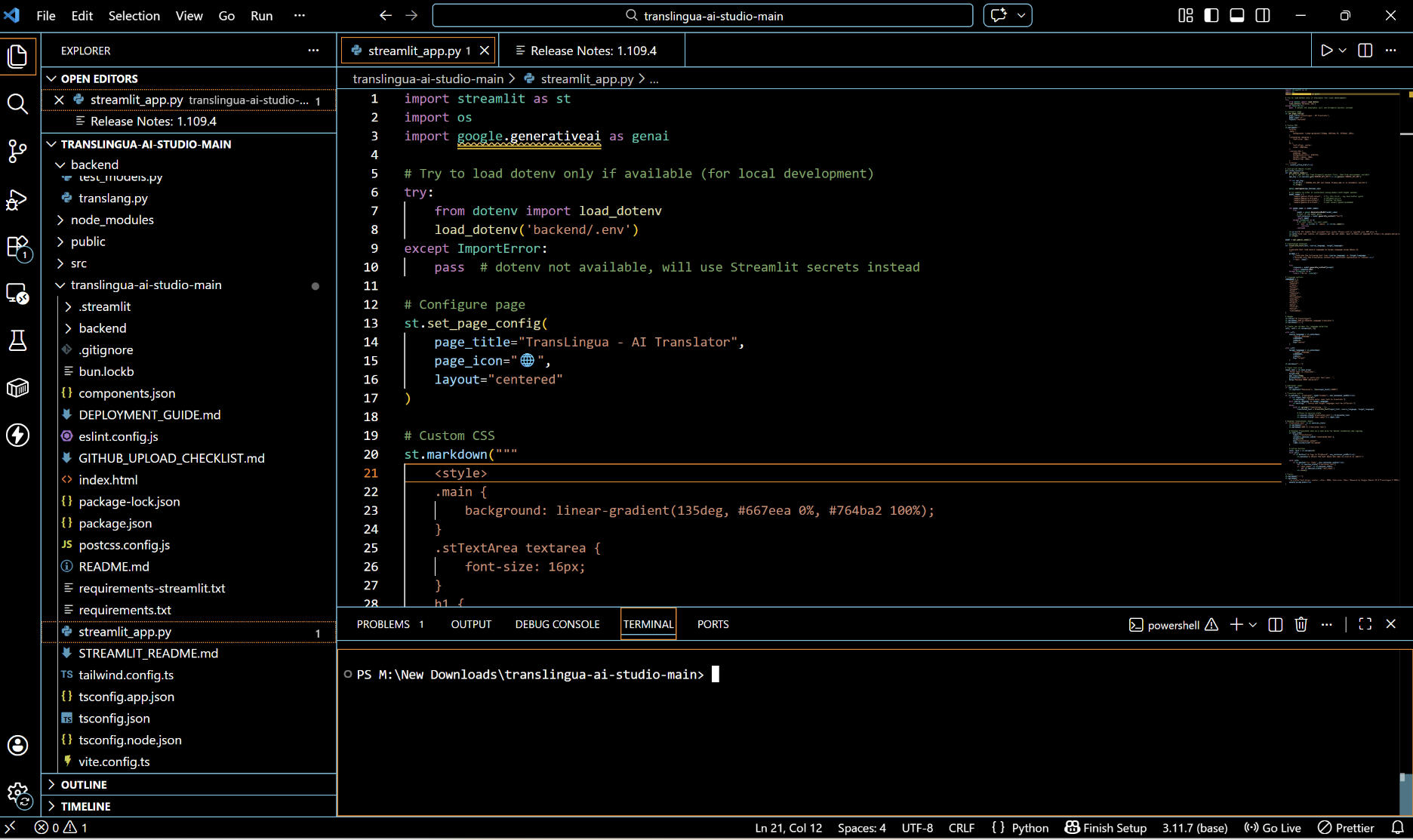
**6. FUNCTIONAL AND PERFORMANCE TESTING**

**6.1 Performance Testing**

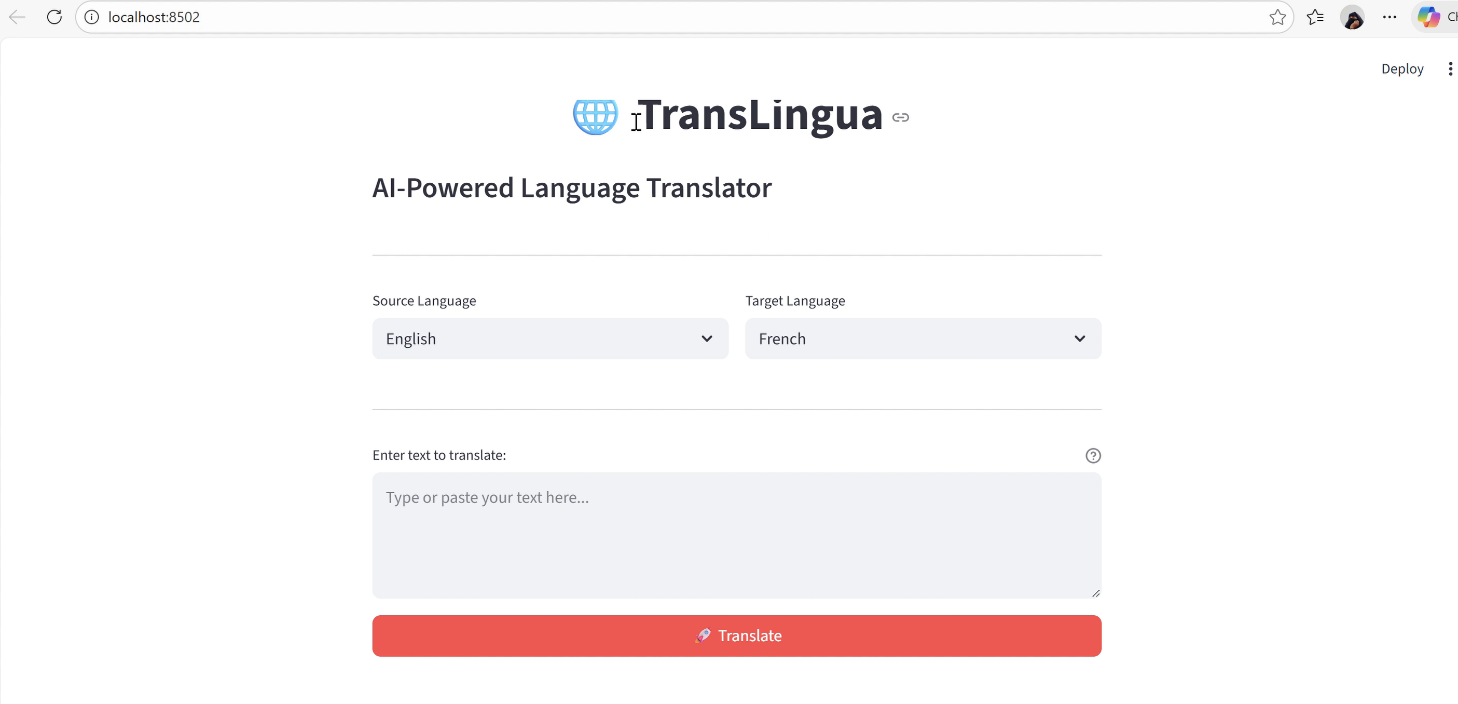
## **Test Scenarios & Results**

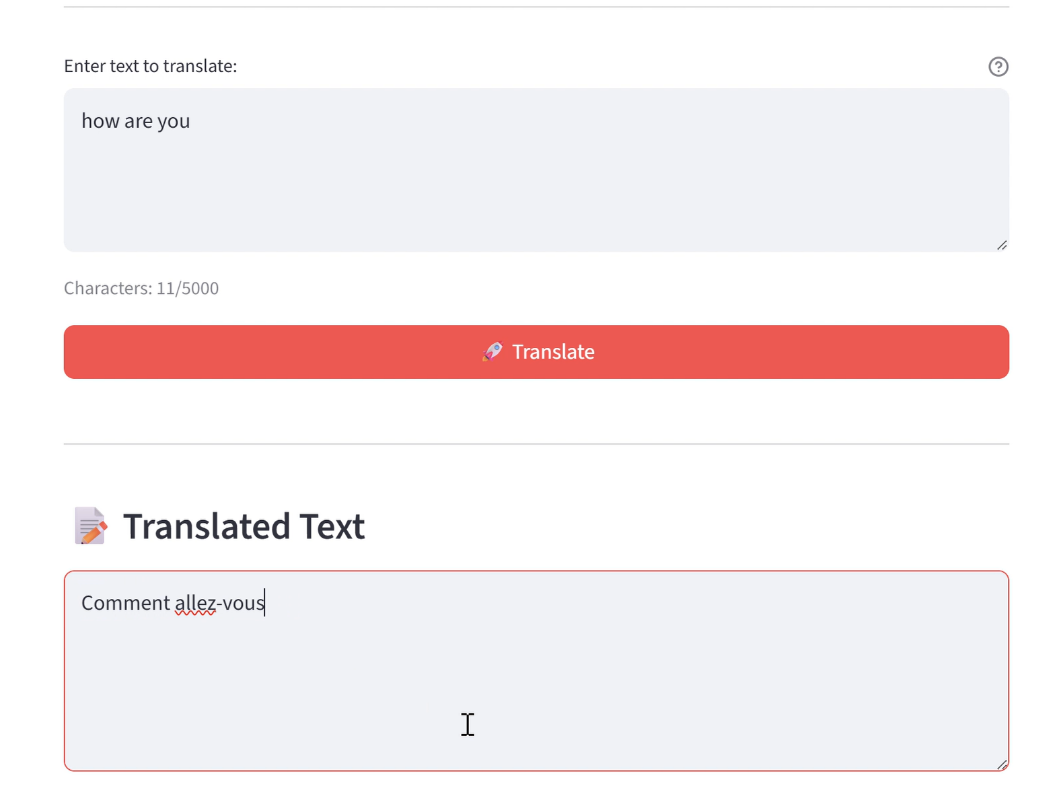
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Scenario (What to test)** | **Test Steps (How to test)** | **Expected Result** | **Actual Result** | **Pass/Fail** |
| **FT-01** | Text Input Validation | Enter valid text and try submitting with empty input | Valid input accepted, error shown for empty input | As Expected | Pass |
| **FT-02** | |  | | --- | |  |  |  | | --- | | Language  Selection  Validation | | Select source and target languages, try leaving one unselected | System shows validation warning if languages not selected | As Expected | Pass |
| **FT-03** | AI Translation Generation | Enter text and click “Translate”” | Context-aware translated text generated correctly | As Expected | Pass |
| **FT-04** | Gemini API Connection Check | Trigger translation with valid API key | API responds successfully and returns translation | As Expected | Pass |
| **FT-05** | |  | | --- | |  |  |  | | --- | | Copy-to-  Clipboard Feature | | Generate translation and click copy button | |  | | --- | | and click copy button |  |  | | --- | | Translated text copied successfully | | As Expected | Pass |
| **PT-01** | Response Time Test | Measure time after clicking generate | Translation generated within 3–5 seconds | Within Limit | Pass |
| **PT-02** | Multiple Request Handling | Perform multiple translations sequentially | Application handles requests without crash | Stable | Pass |
| **PT-03** | Deployment Test | Access deployed app via browser | Application loads and works correctly online | Working | Pass |

**7. RESULTS**

**7.1. Output Screenshots**

.





**8. ADVANTAGES AND DISADVANTAGES**

**Advantages**

* Saves time with instant AI-powered language translation
* Reduces manual effort in translating text across languages
* Provides context-aware and natural translations
* Supports multiple global languages
* User-friendly interface built with Streamlit
* Fast translation using pre-trained Gemini 2.5 Flash model
* No need for dataset collection or custom model training

**Disadvantages**

* Requires an active internet connection
* Depends on third-party AI APIs (Google Gemini)
* Limited to text-based translation (no voice/image translation in current version)
* Output quality may vary depending on clarity and complexity of input text
* API usage limits may apply under free-tier plans

**9. CONCLUSION**

The **TransLingua: AI-Powered Language Translator** project successfully demonstrates how generative AI can be utilized to enable accurate and context-aware multilingual communication. The application allows users to translate text between different languages by selecting the source and target languages, significantly reducing the time and effort required for manual translation. By integrating a pre-trained generative AI model (Google Gemini 2.5 Flash) with a simple and user-friendly Streamlit interface, the project delivers fast, reliable, and natural-sounding translations, making it a valuable tool for students, professionals, researchers, and global users.

**10. FUTURE SCOPE**

The **TransLingua** project can be further enhanced by incorporating additional language support to expand global accessibility and reach a wider audience. Future improvements may include adding **voice-to-text and speech-to-speech translation**, enabling users to translate spoken language in real time. Introducing **document translation (PDF, Word files)** would allow users to translate complete documents instead of only text input.

Adding **user authentication and accounts** would enable users to save translation history and access it later. The system could also provide **personalized translation suggestions**, tone adjustments (formal/informal), and domain-specific translations (technical, business, academic). Additionally, deploying the application as a **mobile app** or integrating it with messaging platforms would make the solution more convenient, accessible, and user-friendly across different devices and environment

**11. APPENDIX**

**11.1. Source Code**

The source code for the TransLingua: AI-Powered Language Translator project includes the implementation of the Streamlit-based user interface, integration of the Google Gemini 2.5 Flash model using the Google Generative AI API, translation prompt construction logic, and output formatting functionality. The application handles user input validation, API communication, and response processing to generate accurate and context-aware translations. The code is written in Python and follows a modular, structured, and readable approach to ensure maintainability and scalability.

**11.2. Github & Project Demo Link**

**Github Repository Link:** <https://github.com/ayeshasheik611/TransLingua-AI-Powered-Multi-Language-Translator-.git>

**Demo Link:** <https://drive.google.com/file/d/1xIqBx0-uOsE3Z3WVbOKXsGq7hEA9Lg4Z/view?usp=drive_link>