# Hackathon Project Phases Template

**Project Title:**

**TransLingua: AI-Powered Multi-Language Translator**

**Team Name:**

**Lingua Franca**

**Team Members:**

* Vyshnavi
* Guru Meghana
* Vrinda kishoree
* Ramadevi

## Phase-1: Brainstorming & Ideation

**Objective:**

To create a simple and efficient tool for translating text between multiple languages using AI. It helps users communicate easily by providing fast and accurate translations. The project aims to be user-friendly, scalable, and useful for businesses, students.

**Key Points:**

1. **Problem Statement:**

A company planning to expand its market reach to non-English speaking regions needs to translate business documents, marketing materials, and customer communications into multiple languages. TransLingua enables the company to quickly translate promotional content and technical documents, ensuring consistency and accuracy across different languages. This helps the company effectively communicate with a broader audience and tailor its messaging to local markets.

**Proposed Solution:**

* 1. AI-powered web app using **Google’s Generative AI** for accurate translations.

○ Simple **Streamlit-based** interface for easy text input and language selection.,gTTs, speech recognition,transformers

1. **Target Users:**

* **Businesses** for global communication.

○ **Students/Educators** for language learning

○  **Travelers** for real-time translations.

1. **Expected Outcome:**

* 1. A user-friendly and scalable AI translation tool that enables seamless multilingual communication for diverse users.

## Phase-2: Requirement Analysis

**Objective:**

Defining the technical and functional requirements for the **TransLingua** app

**Key Points:**

1. **Technical Requirements:**

* 1. Programming Language: **Python**

○ Backend**:** **Google Gemini AI API for translation**

○ Frontend: **Streamlit Web Framework**

○ Database: **Not required initially (API-based queries)**

1. **Functional Requirements:**

* 1. Ability to **translate text** between multiple languages using **Google’s Generative AI**.

○ Provide **context-aware, high-quality translations** with user-friendly UI.

○ Support **multiple language selections** for both source and target languages.

○ Ensure **fast response time** for real-time translations.

1. **Constraints & Challenges:**

* 1. Ensuring **accurate** and **context-aware** translations for different languages.

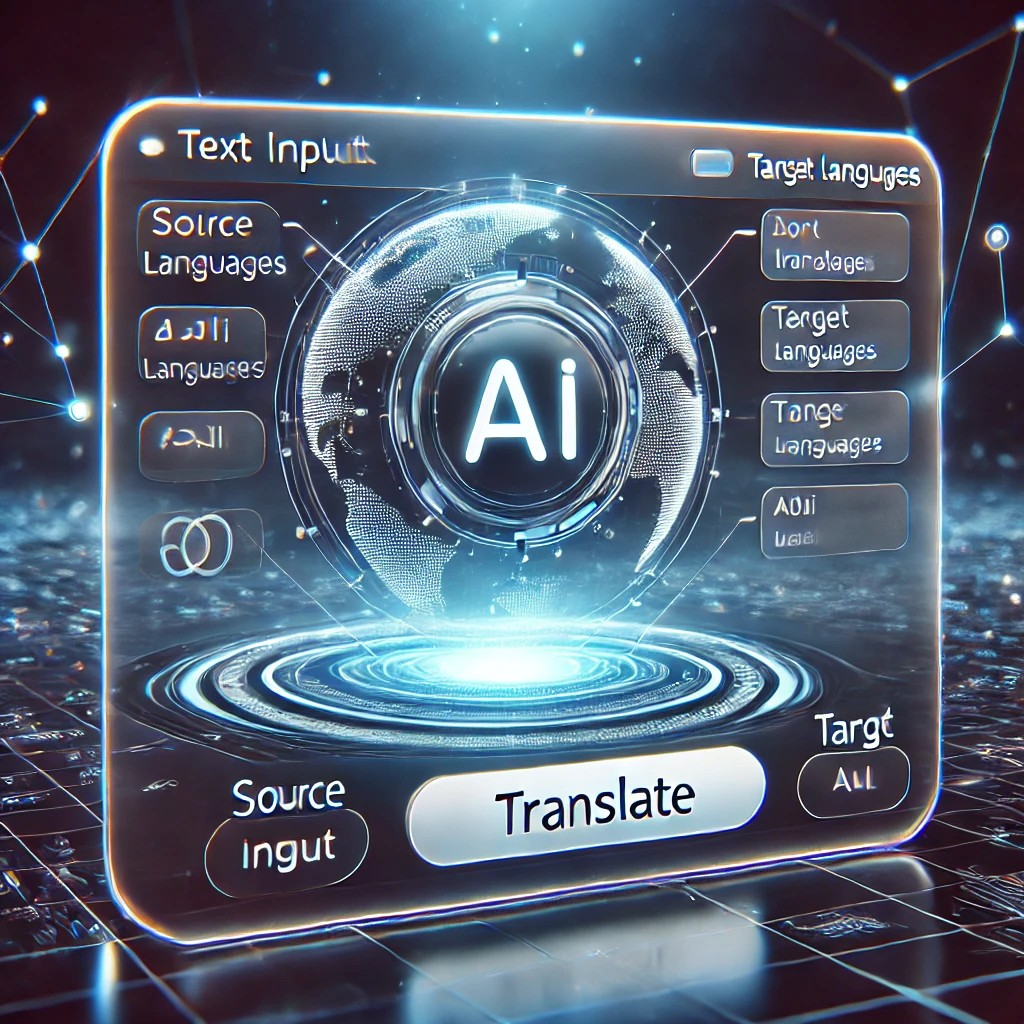
○ Handling **API rate limits** and optimizing API calls for performance.

○ Providing a **smooth and interactive UI** with Streamlit for an intuitive user experience.

## Phase-3: Project Design

**Objective:**

Develop the architecture and user flow of the **TransLingua** application.



**Key Points:**

1. **System Architecture:**

* 1. **User enters text** and selects source & target languages via the UI.

○ **Query is processed** using **Google Gemini AI API**.

○ **AI model translates the text** and returns the result.

○ **Frontend displays** the translated text in an easy-to-read format.

**User Flow:**

* Step 1: User enters text and selects source & target languages.

○ Step 2: The backend sends the query to **Google Gemini AI** for translation.

○ Step 3: The app processes the AI response and **displays the translated text** instantly.

1. **UI/UX Considerations:**

* 1. **Minimalist, user-friendly interface** for easy navigation.

○ **Dropdown menus for language selection** to enhance usability

○ **Dark & light mode support** for better readability and accessibility.

## Phase-4: Project Planning (Agile Methodologies)

**Objective:**

Break down development tasks for efficient completion.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Sprint** | **Task** | **Priority** | **Duration** | **Deadline** | **Assigned To** | **Dependencies** | **Expected**  **Outcome** |
| Sprint 1 | Environment Setup & API Integration | 🔴 High | 6 hours  (Day 1) | End of Day  1 | Vrinda | Google API Key, Python, Streamlit setup | API connection established & working |
| Sprint 1 | Frontend UI Development | 🟡  Medium | 2 hours  (Day 1) | End of Day  1 | Meghana | API response format finalized | Basic UI with input fields |
| Sprint 2 | Translation Function Implementation | 🔴 High | 3 hours  (Day 2) | Mid-Day 2 | Vyshnavi | API response, UI elements ready | Text translation feature working |
| Sprint 2 | Error Handling & Debugging | 🔴 High | 1.5 hours  (Day 2) | Mid-Day 2 | Entire Team | API logs, UI inputs | Improved API stability |
| Sprint 3 | Testing & UI Enhancements | 🟡  Medium | 1.5 hours  (Day 2) | Mid-Day 2 | Rama | API response, UI layout completed | Responsive UI, better user experience |
| Sprint 3 | Final Presentation & Deployment | 🟢 Low | 1 hour  (Day 2) | End of Day  2 | Entire Team | Working prototype | Demo-ready project |

**Sprint Planning with Priorities**

**Sprint 1 – Setup & Integration (Day 1)**

**(**🔴 **High Priority)** Set up the **environment** & install dependencies.

**(**🔴 **High Priority)** Integrate **Google Gemini API**.

**(**🟡 **Medium Priority)** Build a **basic UI with input fields**.

**Sprint 2 – Core Features & Debugging (Day 2)**

**(**🔴 **High Priority)** Implement **search & comparison functionalities**. **(**🔴 **High Priority)** Debug API issues & handle **errors in queries**. **Sprint 3 – Testing, Enhancements & Submission (Day 2)**

**(**🟡 **Medium Priority)** Test API responses, refine UI, & fix UI bugs. **(**🟢 **Low Priority)** Final **demo preparation & deployment**.

## Phase-5: Project Development

**Objective:**

Implement core features of the **TransLingua** application.

**Key Points:**

1. **Technology Stack Used:**

* 1. **Frontend:** Streamlit

○ **Backend:** Google Gemini Flash API

○ **Programming Language:** Python

1. **Development Process:**

* 1. Implement **API key authentication** and integrate **Google Gemini AI API** for translations.

○ Develop **language selection logic** for source and target languages

○ Optimize **translation queries** for speed and accuracy.

1. **Challenges & Fixes:**

* 1. **Challenge:** Delayed API response times.

**Fix:** Implement caching to store frequently translated phrases

○ **Challenge:** Limited API calls per minute.

**Fix:** Optimize queries to send only essential data and minimize redundant API calls.

## Phase-6: Functional & Performance Testing

**Objective:**

Ensure that the **TransLingua** application works as expected.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Category** | **Test Scenario** | **Expected Outcome** | **Status** | **Tester** |
| TC-001 | Functional  Testing | Translate "Hello, how are you?" from English to Spanish | Correct Spanish translation should be displayed | ✅ Passed | Vrinda |
| TC-002 | Functional  Testing | Translate "Good morning" from French to Japanese | Correct Japanese translation should be displayed | ✅ Passed | Vyshnavi |
| TC-003 | Performance  Testing | API response time under **500ms** | API should return translations quickly | ⚠ Needs Optimization | Rama |
| TC-004 | Bug Fixes & Improvements | Fixed incorrect translations for complex sentences | Translation accuracy should be improved | ✅ Fixed | Develop er |
| TC-005 | Final Validation | Ensure UI is responsive across devices | UI should work on mobile & desktop | ❌ Failed - UI broken on mobile | Meghana |
| TC-006 | Deployment  Testing | Host the app using **Streamlit Cloud** | App should be accessible online | 🚀 Deployed | DevOps |