Hackathon Project Phases Template for the TransLingua: AI-Powered Multi-Language Translator project.

Hackathon Project Phases Template

Project Title:

TransLingua: AI-Powered Multi-Language Translator Using AI and NLP Technologies

Team Name:

❖ Bolbhasha

Team Members:

- Srija Morthala
- Preethika Kasthuri
- Pavithra Lokasani
- Sirisha Kondepu

Phase-1: Brainstorming & Ideation

Objective:

Develop an AI-powered language expert tool using AI and NLP Technologies to help users compare and analyze language specifications, reviews, and eco-friendly options.

Key Points:

1. **Problem Statement:**
    - Many users struggle to find reliable, up-to-date information about different languages and dialects before making a purchase decision.
    - Users also need guidance on language maintenance and eco-friendly language choices.

2. **Proposed Solution:**
    - An AI-powered application using AI and NLP Technologies to provide real-time language specifications, reviews, and comparisons.
    - The app offers maintenance tips and eco-friendly language insights based on user preferences.
3. **Target Users:**
    - Users looking for accurate, real-time translations.
    - Students, professionals, and travelers needing language support.
    - Eco-conscious consumers searching for hybrid and electric language options.
4. **Expected Outcome:**
    - A functional AI-powered language information app that provides insights based on real-time data and user queries.

Phase-2: Requirement Analysis

Objective:

Define the technical and functional requirements for the TransLingua: AI-Powered Multi-Language Translator.

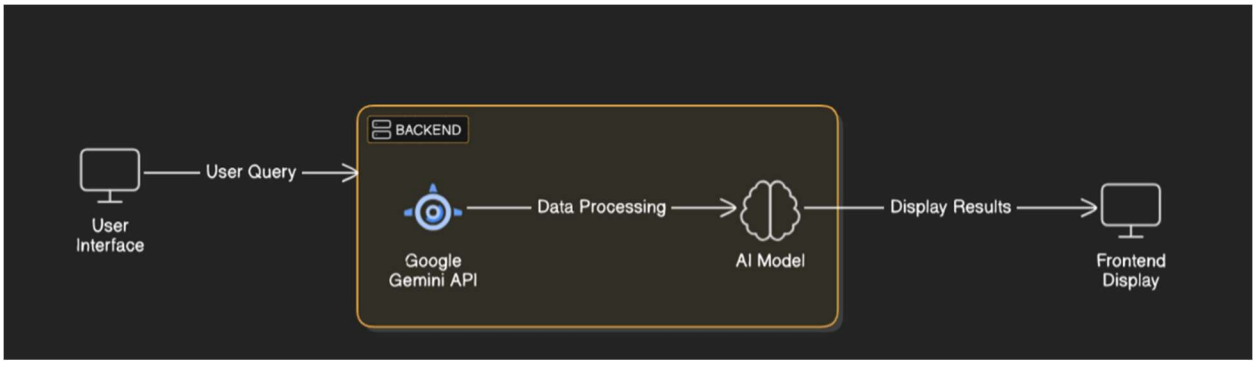
Key Points:

1. **Technical Requirements:**
    - Programming Language: Python, TensorFlow, PyTorch
    - Backend: Google AI and NLP Technologies API
    - Frontend: Web and Mobile Application
    - Database: Cloud-based language datasets for AI training
2. **Functional Requirements:**
    - Ability to fetch language details using AI and NLP Technologies API.
    - Display **specifications, reviews, and comparisons** in an intuitive UI.
    - Provide real-time language maintenance tips based on seasons.
    - Allow users to search eco-friendly languages based on emissions and incentives.
3. **Constraints & Challenges:**
    - Ensuring real-time updates from **Gemini API**.
    - Handling **API rate limits** and optimizing API calls.
    - Providing a **smooth UI experience** with Streamlit.

Phase-3: Project Design for TransLingua

Objective:

Develop the architecture and translation workflow of TransLingua.



Key Points:

1. **System Architecture for TransLingua:**
    - User enters language-related query via UI.
    - Query is processed using AI-driven NLP models.
    - AI model fetches and processes the data.
    - The frontend displays language details, reviews, and comparisons.

2. **User Flow for TransLingua**

    1. User Inputs a Query:
    - Enters text, speaks into the microphone, or uploads an image containing foreign language text.

    2. AI Processing & Language Detection:
    - The system identifies the input type and detects the source language.
    - NLP models analyze context and dialect variations.

    3. Translation Process:
    - AI processes text and provides an accurate, context-aware translation.
    - Speech inputs are converted to text, translated, and converted back to speech.
    - OCR extracts text from images and translates it.

    4. Output Display & User Interaction:
    - The translated content is displayed in text, speech, or image form.

- Users can adjust settings for language preferences or industry-specific terminology.

    - Step 1: User enters a query (e.g., "Best motorcycles under ₹1 lakh").
    - Step 2: The backend calls the AI and NLP Technologies API to retrieve language data.
    - Step 3: The app processes the data and **displays results** in an easy-to-read format.

3. **UI/UX Considerations:**

    - **Minimalist, user-friendly interface** for seamless navigation.
    - **Filters for price, mileage, and features**.
    - **Dark & light mode** for better user experience.

Phase-4: Agile Project Planning for TransLingua

Objective:

Plan AI model training, integration, and deployment in an agile manner.

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	Shanawaz	Google API Key, Python, Streamlit setup	API connection established & working
Sprint 1	Frontend UI Development	🟡 Medium	2 hours (Day 1)	End of Day 1	Preethika	API response format finalized	Basic UI with input fields
Sprint 2	Language translation	🔴 High	3 hours (Day 2)	Mid-Day 2	anwar	API response, UI elements ready	Search functionality with filters
Sprint 2	Error Handling & Debugging	🔴 High	1.5 hours (Day 2)	Mid-Day 2	Srija & sirisha	API logs, UI inputs	Improved API stability
Sprint 3	Testing & UI Enhancements	🟡 Medium	1.5 hours (Day 2)	Mid-Day 2	pavithra	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 – AI Model Setup & Data Collection (Week 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 – Translation Feature Implementation (Week 2-3)

(🔴 **High Priority)** Implement **search & comparison functionalities**.
(🔴 **High Priority)** Debug API issues & handle **errors in queries**.

Sprint 3 – Testing, UI Improvements & Deployment (Week 4-5)

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

Phase-5: AI Model Development & Deployment

Objective:

Implement core features of the TransLingua: AI-Powered Multi-Language Translator.

Key Points:

1. **Technology Stack Used:**
    - Frontend: Web and Mobile Applications (Flutter/React)
    - Backend: Google AI and NLP Technologies API
    - Programming Language: Python, TensorFlow, PyTorch
2. **Development Process:**
    - Develop secure AI API endpoints for multilingual translation.
    - Develop language comparison and maintenance tips logic.
    - Optimize translation accuracy and response time.
3. **Challenges & Fixes:**

- Challenge: Ensuring real-time translation speed.
  Fix: Use optimized AI inference models for faster processing.
- Challenge: Supporting offline translation mode.
  Fix: Implement lightweight AI models for offline translation.

Phase-6: Functional & Performance Testing

Objective:

Ensure that the TransLingua: AI-Powered Multi-Language Translator works as expected.

Test Case ID	Category	Test Scenario	Expected Outcome	Status	Tester
TC-001	Functional Testing	Translate English text to French	Correct French translation should appear	✅ Passed	Tester 1
TC-002	Functional Testing	Convert spoken Hindi to Spanish text	Correct Spanish text should be displayed	✅ Passed	Tester 2
TC-003	Performance Testing	Process image text extraction and translation within 500ms	Translation should be completed within 500ms	⚠ Needs Optimization	Tester 3
TC-004	Bug Fixes & Improvements	Resolve incorrect word choices in casual speech translation	Improved conversational translation accuracy	✅ Fixed	Developer
TC-005	Final Validation	Ensure UI responsiveness on mobile and desktop	UI should be fully functional across devices	❌ Failed - UI needs adjustment	Tester 2
TC-006	Deployment Testing	Host TransLingua on cloud server	App should be accessible online	🚀 Deployed	DevOps

---

Final Submission

1. **Project Report Based on the templates**
2. **Demo Video (3-5 Minutes)**
3. **GitHub/Code Repository Link**
4. **Presentation**