Hackathon Project Phases Template for the **Audio2Art App** project.

Hackathon Project Phases Template

Project Title:

Audio2Art: Transforming Voice Prompts into Visual Creations using Transformers

Team Name:

The Hacky Fab Four

Team Members:

- Akella Sai Gayathri Keerthana
- Prerana Joshi
- K. Rithika
- G. Swathi

Phase-1: Brainstorming & Ideation

Objective:

To develop an AI-powered system that converts audio prompts into visual representations using transformer models, enhancing the creative workflow for artists and designers.

Key Points:

1.Problem Statement:

- Artists and designers often spend significant time creating initial sketches based on ideas. Traditional methods require manual effort, limiting productivity and creativity.

- There is a need for an AI-driven solution that translates spoken descriptions into visual concepts efficiently.

2. Proposed Solution:

- Utilizing advanced transformer models for **natural language processing** and **image generation**, Audio2Art will interpret audio descriptions and generate high-quality visuals.
- The system will incorporate **speech-to-text conversion, contextual analysis**, and **AI-powered image synthesis.**

3. Target Users:

- **Digital artists** and **designers.**
- **Storytellers** and **content creators.**
- **Game developers** and **concept artists.**
- **Educators** and **researchers** in visual learning.
- **Marketing** and **advertising professionals.**

4. Expected Outcome:

- Faster and more efficient **visual ideation** process.
- Enhanced creativity with **AI-assisted concept generation.**
- Seamless integration of **auditory and visual creativity.**
- Improved accessibility for users with visual impairments through **AI-generated visual aids.**

Phase-2: Requirement Analysis

Objective:

Define the technical and functional requirements for the Audio2Art App.

Key Points:

1. Technical Requirements:

- Programming Language: **Python.**
- Backend:
  - **Audio Processing:** py dub, speech recognition.
  - **Language Processing**: google trans.
  - **AI Model Inference:** diffusers, transformers, torch, accelerate.

- **Image Processing:** Pillow.

- Frontend: **JavaScript (Media Recorder API)**
- Database**:** There is no database in the current setup.
- Speech Processing**: Speech Recognition Library** (Python) – Uses **Google Web Speech API** to transcribe audio. **pydub -** Converts audio format (Web M → WAV).
- **Torch & CUDA Support**: Enables GPU acceleration if available for Stable Diffusion, improving processing speed.
- **Stable Diffusion Model (run way ml /stable-diffusion-v1-5)**: A pre-trained model for text-to-image generation.

2. **Functional Requirements**:

- The system should support **multiple languages** for audio input.
- The system must **transcribe spoken words into text** accurately.
- The system should process the transcribed text to **generate an image**.
- The system should provide a **dropdown menu for language selection**.
- The system should **process audio and generate images in real time.**

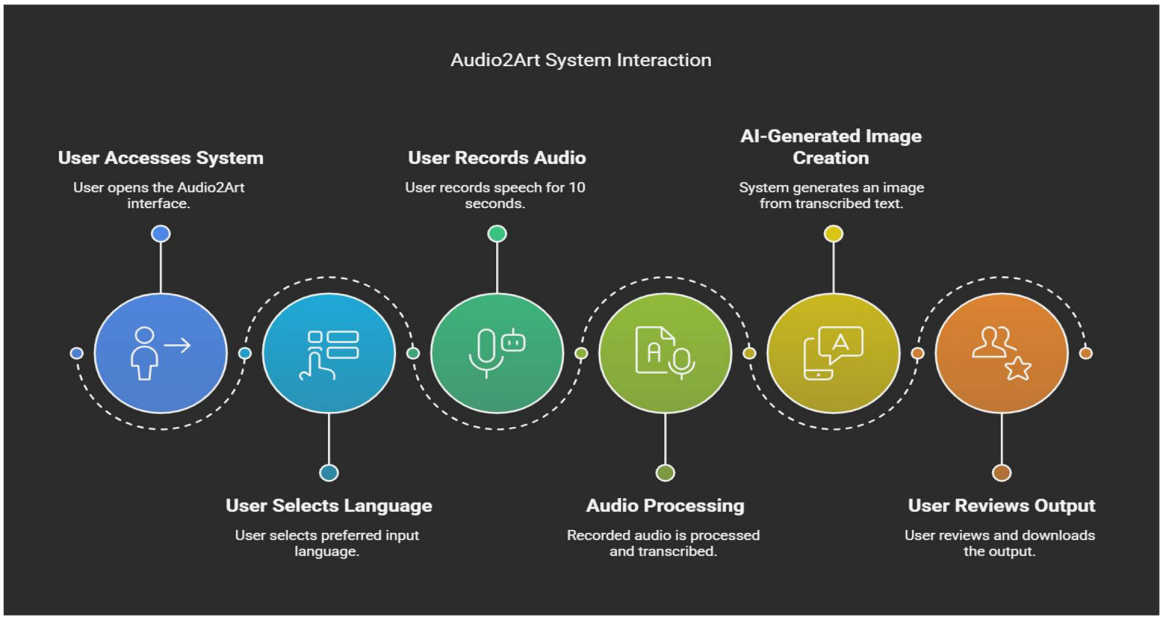
3. Constraints & Challenges:

- The accuracy of Google Web Speech API (or other ASR models) can be affected by**: Background noise**, **accents and pronunciation variations**, **ambiguous or complex words, multiple speakers**.
- Some prompts might **not be understood correctly by the model**, leading to unpredictable results.
- High server load may **slow down image generation**.

Phase-3: Project Design

Objective:

Develop the architecture and user flow of the application.



Key Points:

1. **System Architecture:**

   User Interface (Google colab / Web App)

   - UI Components (Buttons, Dropdowns)
   - JavaScript (Audio Recording)
   - Display Output (Text & Images)

   Backend Processing (Python)

   - Audio Recording (Media Recorder API)
   - Convert Web M → WAV (Py dub / FF m peg)
   - Speech Recognition (Google Web Speech API)
   - AI Image Generation (Stable Diffusion via Replicate API)

   APIs & External Services

   - Google Web Speech API (for transcription)

   AI Processing Layer (Stable Diffusion)

   - diffusers (Stable Diffusion Model – run way ml/stable-diffusion-v1-5)
   - transformers, accelerate, and torch for deep learning acceleration.

2. **User Flow:**
   - User Records Audio
   - System Transcribes & Generates Art
   - User Receives Output
3. **UI/UX Considerations:**
   - Minimalist, user-friendly interface

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	**Keerthana**	Python, Replicate API, Google API, FFmpeg	API connection established & working
Sprint 1	Frontend UI Development	🟡 Medium	2 hours (Day 1)	End of Day 1	**Rithika**	API response format finalized	Basic UI with input fields
Sprint 2	Audio Recording & Processing	🔴 High	3 hours (Day 2)	Mid-Day 2	**Keerthana & Prerana**	WebM to WAV conversion, Pydub, FFmpeg	Properly processed & transcribed audio
Sprint 2	Error Handling & Debugging	🔴 High	1.5 hours (Day 2)	Mid-Day 2	**Keerthana & Swathi**	API logs, UI inputs	Improved API stability
Sprint 2	UI Improvements & Styling	🟡 Medium	2 hours (Day 2)	Mid-Day 2	**Swathi**	Frontend UI, CSS, JavaScript	Better visual design and user experience
Sprint 3	AI Image Generation Integration	🟡 Medium	3 hours (Day 2)	Mid-Day 2	**Prerana & Rithika**	Stable Diffusion (Replicate API)	AI-generated images based on text
Sprint 3	Testing & UI Enhancements	🟡 Medium	1.5 hours (Day 2)	Mid-Day 2	**Prerana & Rithika**	API response, UI layout completed	Responsive UI, better user experience
Sprint 3	User Guide Documentation	🟡 Medium	2 hours (Day 2)	Mid-Day 2	**Swathi**	API features, frontend workflow	Clear user instructions & project summary
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 environment, install dependencies, and integrate Replicate API.
- 🟡 **Medium Priority:** Develop a basic UI with input fields.

✅ **Sprint 2 - Core Features & Debugging (Day 2)**

- 🔴 **High Priority:** Implement **audio recording, processing, and debugging**.

✅ **Sprint 3 - AI Image Generation & Final Enhancements (Day 2)**

- 🟡 **Medium Priority:** Integrate AI-generated image display.
- 🟢 **Low Priority:** Final testing & deployment.

Phase-5: Project Development

Objective:

Implement core features of the Audio2Art App.

Key Points:

1. **Technology Stack Used:**
    - **Frontend:** JavaScript (Media Recorder API)
    - **Backend:**
        - **Audio Processing:** py dub, speech recognition.
        - **Language Processing**: google trans.
        - **AI Model Inference:** diffusers, transformers, torch, accelerate.
        - **Image Processing:** Pillow.
    - **Programming Language**: Python
2. **Development Process:**
    - Audio Recording & Transcription
    - Google trans (Google Translate API for text translation)
    - Translates non-English text into **English** (Stable Diffusion requires English prompts).
    - Text-to-Image Generation
    - Frontend & User Interaction
3. **Challenges & Fixes:**
    - **Challenge:** High latency in audio transcription.
      Fix: Optimized API calls and used WAV format for better accuracy.
    - **Challenge:** Limited API calls due to free-tier constraints.
      Fix: Implemented request throttling and optimized API usage.

Phase-6: Functional & Performance Testing

Objective:

Ensure that the Audio2Art App works as expected.

Test Case ID	Category	Test Scenario	Expected Outcome	Status	Tester
TC-001	Functional Testing	Generate a personalized email	Properly structured email	✅ Passed	Keerthana
TC-002	Functional Testing	Include special instructions	Instructions followed	✅ Passed	Prerana
TC-003	Performance Testing	Response time under 500ms	Fast API response	⚠️ Needs Optimization	Rithika
TC-004	Bug Fixes & Improvements	Handle incorrect API inputs	Proper error message	✅ Fixed	Developer
TC-005	Final Validation	Ensure UI responsiveness	Works on all devices	❌ Failed - Mobile UI issue	Swathi
TC-006	Deployment Testing	Host the app online	Accessible app	🚀 Deployed	DevOps

---

Final Submission

1. Project Report Based on the templates

2. Demo Video (3-5 Minutes)

3. GitHub/Code Repository Link

4. Presentation