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

# **Hackathon Project Phases Template**

## **Project Title:**

**Audio2Art: Transforming Voice Prompts into Visual Creations Using Transformers**

## **Team Name:**

Code Blooded

## **Team Members:**

* Sumana Sri Aashily
* Ceelasagaram Srivani
* Abboju Haarika
* Arelly Tejaswi

## **Phase-1: Brainstorming & Ideation**

### **Objective:**

Develop an AI-powered system that transforms voice prompts into unique visual creations using GenAI.

### **Key Points:**

1. **Problem Statement:**
   * Artists and designers often struggle with generating visual content from abstract ideas.
   * Traditional text-to-image AI models require manual text input, limiting creative spontaneity.
   * There is a need for an intuitive way to convert spoken ideas into visual art effortlessly.
2. **Proposed Solution:**
   * Audio2Art will leverage GenAI to process voice inputs and generate visually appealing artwork.
   * The model will interpret speech, extract keywords, and create images using a transformer-based deep learning model.
   * Users can fine-tune the output with additional voice commands for modifications.
3. **Target Users:**
   * Digital artists and designers seeking AI-assisted creativity.
   * Content creators who want instant visuals based on spoken ideas
   * Anyone interested in AI-generated art from natural voice input.
4. **Expected Outcome:**
   * A functional GenAI-powered application that transforms spoken words into visual art, providing a seamless, voice-driven creative process.

## **Phase-2: Requirement Analysis**

### **Objective:**

Define the technical and functional requirements for Audio2Art.

### **Key Points:**

1. **Technical Requirements:**
   * Programming Language: **Python**
   * Backend: **GenAI API (TRansformer-based model)**
   * Frontend: **Streamlit Web Framework**
   * Database: **Not required initially (API-based queries)**
2. **Functional Requirements:**
   * Convert speech-to-text and extract relevant keywords.
   * Generate images based on voice input using GenAI.
   * Provide real-time modifications via additional voice commands.
   * Display and allow users to download generated artwork.
3. **Constraints & Challenges:**
   * Ensuring accurate voice-to-text conversion.
   * Optimizing GenAI image generation for high-quality results.
   * Managing API rate limits and response times.

## **Phase-3: Project Design**

### **Objective:**

Develop the architecture and user flow for Audio2Art.



### **Key Points:**

1. **System Architecture:**
   * 1. User speaks a creative prompt (e.g., "A futuristic city at sunset").
   * 2. Speech is converted into text using an ASR (Automatic Speech Recognition) model.
   * 3. Text is processed to extract relevant keywords.
   * 4. GenAI model generates an image based on the processed text.
   * 5. Image is displayed in the frontend, allowing user refinements.
2. **User Flow:**
   * 1. User speaks a prompt into the app.
   * 2. The backend converts voice to text and processes keywords.
   * 3. The GenAI model generates an image from the extracted text.
   * 4. The user can request modifications through additional voice prompts.
   * 5. The final image is displayed and available for download.
3. **UI/UX Considerations:**
   * Simple and intuitive interface for easy interaction.
   * Dark & light mode for a better visual experience.
   * Voice-controlled interactions for seamless workflow.

## 

## **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 | Member 1 | Google API Key, Python, Streamlit setup | API connection established & working |
| Sprint 1 | Frontend UI Development | 🟡 Medium | 2 hours (Day 1) | End of Day 1 | Member 2 | API response format finalized | Basic UI with input fields |
| Sprint 2 | Vehicle Search & Comparison | 🔴 High | 3 hours (Day 2) | Mid-Day 2 | Member 3 | API response, UI elements ready | Search functionality with filters |
| Sprint 2 | Error Handling & Debugging | 🔴 High | 1.5 hours (Day 2) | Mid-Day 2 | Member 1&4 | API logs, UI inputs | Improved API stability |
| Sprint 3 | Testing & UI Enhancements | 🟡 Medium | 1.5 hours (Day 2) | Mid-Day 2 | Member 3 | 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 **GenAI API** and **Speech-to-Text API.**  
 **(🟡 Medium Priority)** DEvelop a **basic UI with voice input functionality**.

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

**(🔴 High Priority)** Implement **voice-to-image generation using GenAI**.  
 **(🔴 High Priority)** Debug API responses and optimize latency.

### **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 Audio2Art.

### **Key Points:**

1. **Technology Stack Used:**
   * **Frontend:** Streamlit
   * **Backend:** GenAI APIfor image generation
   * **Programming Language:** Python
2. **Development Process:**
   * Implement voice-to-text conversion.
   * Develop an image-generation pipeline using GenAI.
   * Optimize the system for smooth user interaction.
3. **Challenges & Fixes:**
   * **Challenge:** Inaccurate keyword extraction from speech.
   * **Fix:** Use NLP techniques to improve keyword processing.
   * **Challenge:** API latency for image generation.
   * **Fix:** Implement caching to store frequently used prompts.

## **Phase-6: Functional & Performance Testing**

### **Objective:**

Ensure that the AutoSage App works as expected.

| **Test Case ID** | **Category** | **Test Scenario** | **Expected Outcome** | **Status** | **Tester** |
| --- | --- | --- | --- | --- | --- |
| TC-001 | Functional Testing | Speak a prompt: “A dragon in a mystical forest” | A detailed image of a dragon in a mystical forest. | ✅ Passed | Tester 1 |
| TC-002 | Functional Testing | Request modification: “Make it night-time” | Image updates with a night-time setting. | ✅ Passed | Tester 2 |
| TC-003 | Performance Testing | API response time under 500ms | Image should be generated quickly. | ⚠ Needs Optimization | Tester 3 |
| TC-004 | Bug Fixes & Improvements | Fixed incorrect speech-to-text conversion. | Better accuracy in processing spoken prompts. | ✅ Fixed | Developer |
| TC-005 | UI | Ensure UI is responsive across devices. | UI should work on mobile & desktop. | ❌ Failed - UI broken on mobile | Tester 4 |
| TC-006 | Deployment Testing | Host the app using Streamlit Sharing | 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**