**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:

**“Byte Bots”**

## Team Members:

- Rishitha Patel Puppala
- Bala Supraja Pasumarthy
- Nathi Roshini
- Siri Chandana Palai

---

## Phase-1: Brainstorming & Ideation

### Objective:

Develop an expert tool that leverages voice prompts to generate visual creations, combining speech recognition with transformer models to produce images that align with spoken descriptions.

### Key Points:

- **Problem Statement:**

○ Audio2Art addresses the gap in translating voice prompts into personalized visuals by combining speech recognition with transformer models.

○ The challenge lies in accurately interpreting speech tones, emotions, and context to generate meaningful, emotion-driven art.

- Existing technologies typically rely on text inputs, limiting creative expression

- **Proposed Solution:**
  - Audio prompts are transcribed into text using speech recognition models, which are then processed by NLP models to understand context and style.
  - The interpreted text is fed into a text-to-image model like DALL·E or Stable Diffusion to generate visuals.
  - Fine-tuning on paired datasets ensures accurate and meaningful image creation based on voice input.

- **Target Users:**
  - **Artists and Designers** who want to quickly generate visual concepts or inspiration from voice descriptions.
  - **Content Creators** in industries like marketing, advertising, or social media, who need unique visual assets based on verbal ideas.
  - **Educators and Students** in creative fields who are exploring new ways to translate verbal prompts into visual art for learning or projects.

- **Expected Outcome:**
  - The expected outcome is to seamlessly transform voice prompts into visually compelling artworks, enabling users to create unique images effortlessly from verbal descriptions.

---

# Phase-2: Requirement Analysis

## Objective:

Define the technical and functional requirements for the Audio2Art website.

## Key Points:

1. **Technical Requirements:**
   - Programming Language: **Python**
   - Frontend: **Python web frameworks**
   - Database: **Not required initially (API-based queries)**
2. **Functional Requirements:**

- **Voice Input & Processing**: The system should capture and convert voice prompts into text through accurate speech-to-text conversion.
- **Text-to-Image Generation**: Based on the transcribed text, the system must generatecorresponding high-quality images using a Transformer-based model.
- **UserInteraction**: The system should provide real-time feedback, display generated images, and store user data while ensuring data privacy and security.

3. **Constraints & Challenges:**

   ○ **Speech-to-text accuracy** can be affected by noise and unclear speech.

   ○ **High computational costs** for real-time image generation.

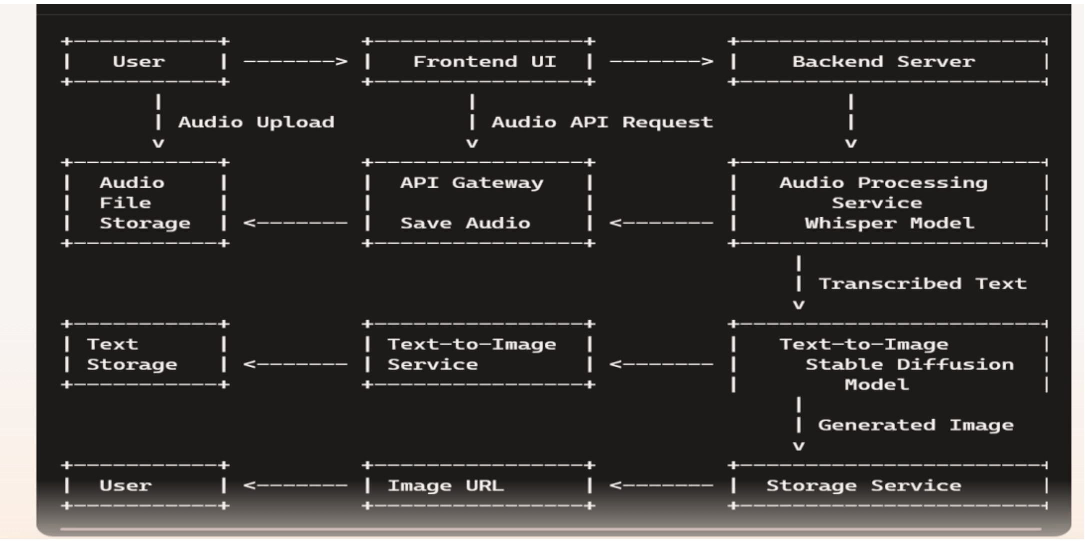
   ○ **Latency** in processing voice prompts and rendering images.

---

# Phase-3: Project Design

## Objective:

Develop the architecture and user flow of the application.



```
+-----------+          +-----------------+          +-----------------------+
|   User    | --------> |   Frontend UI   | --------> |    Backend Server     |
+-----------+          +-----------------+          +-----------------------+
       |                        |                             |
       | Audio Upload           | Audio API Request           |
       v                        v                             v
+-----------+          +-----------------+          +-----------------------+
|  Audio    |          |  API Gateway    |          |   Audio Processing    |
|  File     |          |                 |          |       Service         |
|  Storage  | <-------- |  Save Audio     | <-------- |     Whisper Model     |
+-----------+          +-----------------+          +-----------------------+
                                                         |
                                                         | Transcribed Text
                                                         v
+-----------+          +-----------------+          +-----------------------+
| Text      |          | Text-to-Image   |          |   Text-to-Image       |
| Storage   | <-------- | Service         | <-------- |     Stable Diffusion  |
+-----------+          +-----------------+          |        Model          |
                                                         |
                                                         | Generated Image
                                                         v
+-----------+          +-----------------+          +-----------------------+
|  User     | <-------- | Image URL       | <-------- |  Storage Service      |
+-----------+          +-----------------+          +-----------------------+
```

## Key Points:

1. **System Architecture:**
    - User enters audio query via UI.
    - AI model fetches and processes the data.
    - The frontend displays **audio-text and text-image.**
2. **User Flow:**
    - Step 1: User enters a query (e.g., "audio like A girl sitting on the bench").
    - Step 2: The backend **calls the python** to retrieve audio data.
    - Step 3: The app processes the data and **displays results** in an easy-to-read format.
3. **UI/UX Considerations:**
    - Microphone button for voice input with visual feedback.
    - Show generated images in a gallery with zoom and download options.
    - Ensure an accessible design with easy navigation and user guidance.

---

# 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 | Audio processing & Text transcription | 🔴 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 2& 3 | API response, UI layout completed | Responsive UI, better user experience |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Sprint 3 | Image Generation from Text | 🔴 High | 2 hours (Day 2) | Mid-Day 2 | Member 4 | Diffusers,Torch | Text → Image |
| Sprint 3 | Final Presentation & Deployment | 🟡 Medium | 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.
(🟡 **Medium Priority)** Build a **Frontend UI development.**

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

(🔴 **High Priority)** Implement **Audio processing & Text transcription**.
(🔴 **High Priority)** Debug API issues & handle **errors in queries**.

### Sprint 3 – Testing, Enhancements & Submission (Day 2)

(🟡 **Medium Priority)** Test API responses **Testing & UI Enhancements.**

(🔴 **High Priority) Image Generation from Text.**
(🟡 **Medium Priority)** Final **demo preparation & deployment**.

---

# Phase-5: Project Development

### Objective:

Implement core features of the Audio2Art.

### Key Points:

1. **Technology Stack Used:**
    - **Frontend:** Streamlit

   - **Backend:** Audio Processing
   - **Programming Language:** Python
2. **Development Process:**

   - Implement **Speech-to-Text Conversion**.
   - Develop **image from audio.**
   - Optimize **generates high-quality images based on transcribed text using Stable Diffusion.**
3. **Challenges & Fixes:**

   - **Challenge:** Audio Format Inconsistencies
   - **Fix: Implementing an audio conversion step using ffmpeg ensures that the audio is standardized to a consistent format (16kHz mono WAV), improving compatibility for transcription.**

# Phase-6: Functional & Performance Testing

## Objective:

Ensure that the Audio2Art works as expected.

| Test Case ID | Category | Test Scenario | Expected Outcome | Status | Tester |
|---|---|---|---|---|---|
| TC-001 | Functional Testing | Provide an audio file with clear speech (e.g., "Describe a sunny day") | Text should be accurately transcribed from audio.. | ✅ Passed | Tester 1 |
| TC-002 | Functional Testing | Provide an audio file with background noise (e.g., "Describe a forest") | Text should still be transcribed with minimal errors. | ✅ Passed | Tester 2 |
| TC-003 | Performance Testing | API response time under 500ms | API should process the audio and transcribe text within 10 seconds. | ⚠ Needs Optimization | Tester 3 |
| TC-004 | Bug Fixes & Improvements | API should process the audio and transcribe text within 10 seconds. | Image generation time should be under 15 seconds. | ✅ Fixed | Developer |
| TC-005 | Final Validation | Ensure system is accessible via Streamlit | System should work seamlessly both on desktop and mobile. | ❌ Failed - UI broken on mobile | Tester 2 |
| TC-006 | Deployment Testing | Host the  using Streamlit Sharing | should be accessible online without issues.. | 🚀 Deployed | DevOps |

# Final Submission

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