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

**Hackathon Project Phases Template**

**Project Title:**

**LogoCraft: Innovative Logo Generation with Diffusion Technology**

**Team Name: Squid Squad**

**Team Members:**

* Abhinaw Siddhrath Y
* Shivakaran N
* Avinash Reddy K
* Vasantha Kumar K
* Akash G

**Phase-1: Brainstorming & Ideation**

**Objective:**

LogoCraft aims to provide an AI-powered platform that uses diffusion technology to quickly generate high-quality, customized logos for businesses, startups, and individuals, making professional design accessible, affordable, and efficient

**Key Points:**

1. **Problem Statement:**

Designing a professional logo is often expensive, time-consuming, and requires design expertise, creating barriers for small businesses and individuals who need affordable and quick solutions.

1. **Proposed Solution:**

LogoCraft leverages AI and diffusion technology to generate unique logos based on user preferences. The platform allows easy customization, fast iterations, and high-quality outputs, making logo creation quick, affordable, and accessible to all.

1. **Target Users:** Small businesses and startups

 Freelancers and solopreneurs

 Marketing agencies

 Non-designers

1. **Expected Outcome:**

 Empower users to create professional logos without needing design skills or large budgets.

 Speed up the design process and reduce costs.

 Help users build strong, recognizable brand identities with high-quality logos..

**Phase-2: Requirement Analysis**

**Objective:**  
Define the technical and functional requirements for the LogoCraft App, a platform that utilizes diffusion technology to generate innovative, unique, and customizable logos efficiently and effectively.

### ****Key Points:****

#### ****1. Technical Requirements:****

* **Programming Language:** Python
  + Python will be used for backend development, especially for integration with AI-based models, including diffusion technology, and handling user inputs for logo customization.
* **Backend:** Diffusion Technology Model (AI-powered)
  + The backend will leverage diffusion technology models to generate logos by refining random noise into creative outputs based on user preferences. The AI will be powered using cloud-based processing to ensure scalability.
* **Frontend:** Streamlit Web Framework
  + Streamlit will be used to build the user interface, making it interactive and easy to use for both technical and non-technical users. It will allow users to input their preferences, customize logos, and see real-time results.
* **Database:**
  + Initially, a database may not be required, as the platform will use the API to generate logos. However, user preferences and generated logos can be stored temporarily for quick access and iteration during the design process.

#### ****2. Functional Requirements:****

* **Fetch Logo Details Using AI and Diffusion Technology:**
  + The app must generate logos using AI models, allowing users to input preferences such as color schemes, icon styles, and typography to generate a variety of logos automatically.
* **Logo Customization Features:**
  + Users should be able to tweak logos with specific adjustments, such as modifying colors, fonts, icon positions, and shapes, providing a truly customizable experience.

#### ****3. Constraints & Challenges:****

* **Real-Time Updates from Diffusion Models:**
  + Ensuring that the AI models generate logos quickly and can process user input in real time without significant delays. This requires efficient model optimization and cloud processing.
* **Handling AI Model Processing Limitations:**
  + Diffusion models can be computationally intensive. Managing the time it takes to generate logos while keeping the platform responsive and ensuring smooth user experience is a key challenge.
* **Optimizing User Interface for Smooth Experience:**
  + Streamlit’s simplicity needs to be leveraged to create an intuitive UI that handles multiple logo generations, customizations, and iterations in a seamless manner without performance lags.
* **Balancing Creativity with Brand Identity:**
  + The challenge of ensuring that the generated logos align with the user’s brand identity, values, and goals while maintaining creative uniqueness and appeal.

**Phase-3: Project Design**

**Objective:** Develop the architecture and user flow of LogoCraft, an application utilizing diffusion technology for innovative logo generation.

**Key Points:**

**System Architecture:**

* **User Input via UI:**
  + Users enter design prompts, style preferences, color palettes, and any relevant keywords through a user-friendly interface.
  + Options for uploading existing images or sketches as inspiration.
* **Prompt Processing & Diffusion API Integration:**
  + The user's input is processed and transformed into a structured prompt suitable for the diffusion model.
  + Integration with a diffusion model API (e.g., Stable Diffusion, DALL-E 2, or a custom-trained model) to generate logo variations.
* **AI Model Processing & Generation:**
  + The diffusion model generates multiple logo variations based on the processed prompt.
  + Post processing of the image, to clean, and vectorise the logo.
* **Frontend Display & Refinement:**
  + The generated logo variations are displayed in a gallery format on the frontend.
  + Users can select, refine, and download their chosen logo.
  + Options to regenerate variations based on a selected logo.

**User Flow:**

* **Step 1: User Input & Prompt Creation:**
  + User enters a detailed prompt describing their desired logo (e.g., "Minimalist logo for a tech startup, using blue and white, with a stylized circuit board").
  + Users select desired style, and color pallete.
* **Step 2: Diffusion Model API Call & Logo Generation:**
  + The backend sends the processed prompt to the diffusion model API.
  + The API returns multiple generated logo images.
* **Step 3: Display, Selection, & Refinement:**
  + The app displays the generated logos in a user-friendly gallery.
  + Users can browse, select, and refine their chosen logo.
  + User can download the logo in various formats.

**UI/UX Considerations:**

* **Intuitive Prompt Interface:**
  + A clear and easy-to-use interface for entering detailed prompts.
  + Suggestions and examples to guide users.
* **Visual Style Selection:**
  + Predefined style options (e.g., minimalist, abstract, geometric) for quick selection.
  + Color palette selectors and the ability to upload custom palettes.
* **Interactive Refinement Tools:**
  + Tools for fine-tuning generated logos (e.g., color adjustments, minor shape modifications).
  + Ability to regenerate variations of a selected logo.
* **Minimalist & User-Friendly Design:**
  + Clean and uncluttered interface for a seamless user experience.
  + Dark and light mode options for user preference.
* **Download & Export Options:**
  + Ability to download logos in various formats (SVG, PNG, JPG).
  + Vectorization options for scalable logos.

**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 | Vasanth | Google API Key, Python, Streamlit setup | API connection established & working |
| Sprint 1 | Frontend UI Development | 🟡 Medium | 2 hours (Day 1) | End of Day 1 | Akash | API response format finalized | Basic UI with input fields |
| Sprint 2 | Error Handling & Debugging | 🔴 High | 3 hours (Day 2) | Mid-Day 2 | Abhinaw Siddharth | API logs, UI inputs | Improved API stability |
| Sprint 3 | Testing & UI Enhancements | 🟡 Medium | 1.5 hours (Day 2) | Mid-Day 2 | Shiva karan and Avinash | 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 LogoCraft App.

**Key Points:**

1. **Technology Stack Used:**
   * **Frontend:** Streamlit
   * **Backend:** Google Gemini Flash API
   * **Programming Language:** Python
2. **Development Process:**
   * Implement **API key authentication** and **Gemini API integration**.
   * Develop **vehicle comparison and maintenance tips logic**.
   * Optimize **search queries for performance and relevance**.
3. **Challenges & Fixes:**
   * **Challenge:** Delayed API response times.  
      **Fix:** Implement **caching** to store frequently queried results.
   * **Challenge:** Limited API calls per minute.  
      **Fix:** Optimize queries to fetch **only necessary data**.

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

**Objective:**

Ensure that the LogoCraft App works as expected.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Category** | **Test Scenario** | **Expected Outcome** | **Status** | **Tester** |
| 1 | Functional Testing | Prompt: futuristic rocket in sky | Logo containing rocket in sky |  |  |
| 2 | Performance Testing | Image generation in 30 minutes | Should return exact logo |  |  |
| 3 | Bug fixes and improvements | Correcting weak prompts | Image related to given prompt |  |  |
| 4 | Final validation | Ensure UI is responsive | UI will work on cpu |  |  |
| 5 | Deployment testing | Host the app using streamlit | Should be accessible online |  |  |

**Final Submission**

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