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

# **Hackathon Project Phases Template**

## **Project Title:**

**CoutureAI App**

## **Team Name:**

TechGen

## **Team Members:**

* Gurijala Radha Sivani
* Melupaka Manozgna
* Tippanwar Bhavani

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

#### **Objective:**

* Enable users to generate personalized clothing images based on textual descriptions.
* Enhance the online shopping experience by offering a highly visual and interactive process.
* Provide users with the ability to express their unique style, leading to better satisfaction in clothing purchases.

### **Key Points:**

1. **Problem Statement:  
   Current Shopping Limitations:**

* Online shopping platforms don’t provide adequate visualization of customized clothing.
* Consumers often have difficulty finding clothing that matches their specific descriptions or designs.
* Traditional fashion e-commerce platforms focus on catalog-based browsing without dynamic customization options.

1. **Proposed Solution:**

The proposed solution for **CoutureAI** using **Google Colab** involves generating realistic clothing images from text descriptions with **Stable Diffusion**. Users can customize fabrics, styles, and features, with **real-time previews** of their designs. The platform will use **Streamlit** or **Gradio** for a user-friendly interface. Users can view their designs from multiple angles, **save** or **share** them, and connect with **tailors** for custom creations.

1. **Target Users:**

**Primary Audience:**

* **Fashion Enthusiasts & Designers:** People who are deeply interested in fashion and appreciate the power of personalized design.
* **Tailors & Custom Clothing Services:** Those who want to provide customers with designs based on their ideas or clothing descriptions.
* **Consumers Seeking Bespoke Clothing:** Customers who are looking for custom-made or personalized outfits but don’t have the skills to design them themselves.

**Secondary Audience:**

* **Clothing Brands & Retailers:** Those who might integrate this tool into their websites for a more interactive and personalized shopping experience.
* **Technology Enthusiasts:** Users who are curious about AI-driven creative tools and would enjoy experimenting with the fashion design generation feature.

1. **Expected Outcome:**

The expected outcome of **CoutureAI** is to enable users to generate realistic clothing designs from text descriptions, with real-time previews and customization options. Users can refine, save, and share their designs, and connect with tailors for custom creations, offering a personalized and interactive fashion experience.

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

### **Objective:**

* Identify user needs and key functionalities.
* Define system, technical, and performance requirements.
* Address constraints like data quality and computational resources.
* Assess feasibility and plan development steps.

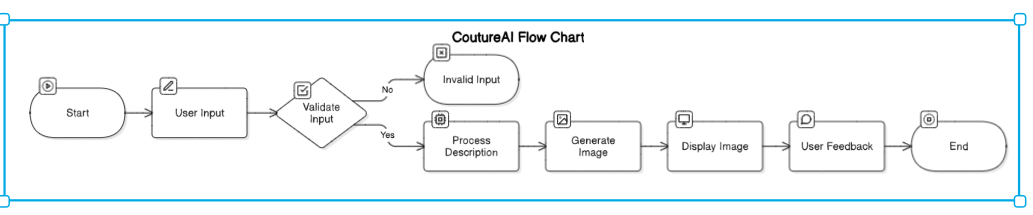
### **Key Points:**

1. **Technical Requirements:**
   * **AI Model: Use Stable Diffusion for text-to-image generation, fine-tuned on fashion data.**
   * **Platform**: Deploy on **Google Colab** with **Streamlit** or **Gradio** for the interface.
   * **Programming**: **Python** for backend, **HTML/CSS/JavaScript** for frontend (if needed).
   * **Cloud Services**: Store images and data on **Google Cloud** or **AWS**.
   * **Database**: Use **SQLite** or **Firebase** for storing user data and designs.
   * **Security**: Implement **SSL encryption** and **OAuth** for user authentication.
   * **Computational Resources**: Leverage **GPU/TPU** in Google Colab for fast image generation.
   * **Real-Time Updates**: Ensure immediate design updates as users modify descriptions.
2. **Functional Requirements:**
   * **Text-to-Image Generation**: Use **Stable Diffusion** for realistic clothing images from text descriptions.
   * **Customization Options**: Allow users to choose fabric, style, and design features.
   * **Real-Time Previews**: Provide instant visual updates as users modify designs.
   * **User Interface**: Simple, interactive UI built with **Streamlit** or **Gradio**.
   * **Multiple Viewpoints**: Display designs from different angles.
   * **Save & Share**: Enable users to save and share designs.
   * **Tailor Integration**: Allow users to connect with tailors for custom creations.
3. **Constraints & Challenges:**
   * **Data Quality**: Limited or biased datasets can affect design accuracy.
   * **Computational Resources**: High costs for real-time image generation.
   * **Model Limitations**: Struggles with complex or non-standard designs.
   * **Performance**: Ensuring smooth real-time updates without lag.

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

### **Objective:**

the architecture and user flow of the application.



### **Key Points:**

1. **System Architecture:**
   * Define the overall structure of the platform, including backend, frontend, and AI model integration.
   * Choose cloud infrastructure (e.g., **Google Cloud**, **AWS**) for deployment and scalability.
2. **User Flow:**
   * **Input Description**: User enters a detailed clothing description.
   * **Customization**: Select fabric, color, and style options.
   * **Generate Design**: AI generates the clothing design.
   * **Real-Time Preview**: User sees and adjusts the design.
   * **Save/Share**: Save or share the generated design.
   * **Connect with Tailor**: Option to send the design to a tailor.
3. **UI/UX Considerations:**
   * **Simplicity**: Clean, intuitive interface.
   * **Customization**: Easy clothing customization with real-time feedback.
   * **Responsive Design**: Mobile-friendly.
   * **Visual Appeal**: High-quality images and modern design.
   * **Error Handling**: Clear error messages.
   * **Save/Share Options**: Easy saving and sharing of designs.

## **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 | Bhavani | Google API Key, Python, Streamlit setup | API connection established & working |
| Sprint 1 | Frontend UI Development | 🟡 Medium | 2 hours (Day 1) | End of Day 1 | Manozgna | API response format finalized | Basic UI with input fields |
| Sprint 2 | Clothing Description Input | 🔴 High | 3 hours (Day 2) | Mid-Day 2 | Bhavani | API response, UI elements ready | Search functionality with filters |
| Sprint 2 | AI Model Integration | 🔴 High | 1.5 hours (Day 2) | Mid-Day 2 | Entire Team | API logs, UI inputs | Improved API stability |
| Sprint 3 | Real-Time Preview & Customization | 🟡 Medium | 1.5 hours (Day 2) | Mid-Day 3 | Manozgna | API response, UI layout completed | Responsive UI, better user experience |
| Sprint 3 | UI Enhancements & Debugging | 🟢 Low | 1 hour (Day 2) | End of Day 3 | Entire Team | Working prototype | Demo-ready project |
| Sprint 4 | Testing & User Feedback | 🔴 High | 2 hours (Day 4) | Mid-Day 4 | Radha Sivani | All previous sprints completed | Functional platform with user feedback |
| Sprint 4 | Final Presentation & Deployment | 🟢 Low | 1 hour (Day 4) | End of Day 4 | Radha Sivani | Working prototype | Demo-ready project with deployment |

### 

### **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 CoutureAI

### **Key Points:**

1. **Technology Stack Used:**
   * **Technology Stack Used**:
   * **Frontend**: Streamlit
   * **Backend**: Google Gemini Flash API
   * **Programming Language**: Python
   * **AI Model**: Stable Diffusion
2. **Development Process:**
   * **API Integration**: Set up API key authentication and Gemini API for image generation.
   * **UI Development**: Build user-friendly interface for clothing design input.
   * **Model Integration**: Connect Stable Diffusion for personalized clothing generation.
3. **Challenges & Fixes:**
   * **Delayed Response**: Implemented caching to improve speed.
   * **Limited API Calls**: Optimized queries for efficient data fetching.
   * **UI Responsiveness**: Improved mobile UI optimization.

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

### **Objective:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Category** | **Test Scenario** | **Expected Outcome** | **Status** | **Tester** |
| TC-001 | Functional Testing | Query "Red satin dress with lace sleeves" | A red satin dress with lace sleeves should be displayed. | ✅ Passed | Bhavani |
| TC-002 | Functional Testing | Query "Evening gown with sweetheart neckline" | An evening gown with sweetheart neckline should appear. | ✅ Passed | Manozgna |
| TC-003 | Performance Testing | API response time under 500ms | API should return results quickly (under 500ms) | ⚠ Needs Optimization | Bhavani |
| TC-004 | Bug Fixes & Improvements | Fixed incorrect API responses. | Data accuracy should be improved. | ✅ Fixed | Radha sivani |
| TC-005 | Final Validation | Fix incorrect API responses | UI should work on mobile & desktop. | ❌ Failed - UI broken on mobile | Manozgna |
| TC-006 | Deployment Testing | Ensure UI is responsive across | App should be accessible online. | 🚀 Deployed | Radha sivani |