**Hackathon Project Phases Template** for the **Couture AI App** project.

**Hackathon Project Phases Template**

**Project Title:**

**CoutureAI: AI-Driven 3D Fashion Design & Augmented Reality Try-On**

**Team Name:**

AI Stitch

**Team Members:**

* Kondaparthi Anika
* Srilekha Jain
* Himaja Devarkonda
* Neha Sarvan

**Phase-1: Brainstorming & Ideation**

**Objective:**

To create CoutureAI, a user-friendly tool that utilizes AI image generation to revolutionize the online fashion industry by enabling customers to visualize personalized clothing designs before purchase, thereby increasing customer satisfaction, reducing return rates, and fostering a more engaging shopping experience.

**Key Points:**

* **Problem Statement:**
* Traditional online fashion platforms struggle to provide personalized visualizations of custom clothing designs, hindering user satisfaction and limiting the shopping experience.
* This lack of visualization creates a gap between a consumer's imagined design and the reality of available options, making it difficult to find or create truly unique and personalized clothing.
* **Proposed Solution:**
* Current online fashion platforms lack robust visualization tools for custom designs, limiting user satisfaction and hindering personalized shopping experiences**.**
* CoutureAI addresses this by integrating Stable Diffusion with a Streamlit interface, enabling users to generate realistic images of custom clothing based on their descriptions, bridging the gap between imagination and reality in fashion design.
* **Target Users:**
* **Fashion Enthusiasts/Consumers.**
* **Fashion Designers**
* **Online Retailers/Boutiques**
* **Expected Outcome:**
* Realistic, high-quality images of user-defined clothing designs generated from text descriptions and/or image inputs, displayed within a user-friendly Streamlit interface.

**Phase-2: Requirement Analysis**

**Objective:**

To elicit and document the user needs for personalized clothing visualization and translate them into functional and non-functional requirements for the CoutureAI platform.

**Key Points:**

* **Technical Requirements:**
* Programming Language: **Python**
* Backend: **Google Collab(API-Key)**
* Frontend: **Streamlit Web Framework**
* Database: **Not required initially (API-based queries)**
* **Functional Requirements:**
* Ability to **generate clothing images** using Stable Diffusion API.
* Display clothing designs with **customizable attributes**.
* Provide **style recommendations** based on user preferences and current trends.
* Allow users to **search for clothing designs** based on style, fabric, color, and other attributes.
* **Constraints & Challenges:**
* Ensuring high-quality and consistent image generation from **Stable Diffusion**.
* Managing Stable Diffusion's **computational demands** and optimizing **inference speed**.
* Providing a **smooth and responsive UI experience** with Streamlit.

**Phase-3: Project Design**

**Objective:**

To design and develop CoutureAI, a platform using Stable Diffusion and Streamlit to enable users to visualize and customize clothing designs.



**Key Points:**

* **System Architecture:**
* User enters clothing **design description** via UI.
* Design description is processed and sent to the **Stable Diffusion API**.
* Stable Diffusion generates the clothing image based on the input.
* The frontend displays the generated **clothing design with customization options**.
* **User Flow:**
* Step 1: User enters a clothing design description (e.g., "A flowing red dress with a floral pattern") or uploads a reference image.
* Step 2: The backend processes the **input and sends it to the Stable Diffusion API** for image generation.
* Step 3: CoutureAI displays the generated clothing **design on the Streamlit interface**, allowing the user to view and further customize it.
* **UI/UX Considerations:**
* **Intuitive and easy-to-use** interface for describing designs and viewing results.
* Options for **filtering and sorting generated designs** by style, color, fabric, etc.
* Visually appealing presentation of generated images**,with zoom and detail views**.

**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 | anika | Google API Key, Python, Streamlit setup | API connection established & working |
| Sprint 1 | Frontend UI Development | 🟡 Medium | 2 hours (Day 1) | End of Day 1 |  | API response format finalized | Basic UI with input fields |
| Sprint 2 | Data Processing & Comparison Logic | 🔴 High | 3 hours (Day 2) | Mid-Day 2 | anwar | 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 | 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:**

To build and deploy a functional CoutureAI platform with seamless Stable Diffusion integration for generating personalized clothing visualizations.

**Key Points:**

* **Technology Stack Used:**
* **Frontend:** Streamlit
* **Backend:** Google Gemini Flash API
* **Programming Language:** Python
* **Development Process:**
* Implement **Stable Diffusion integration** and **optimize image generation** pipeline.
* Develop **UI components** for design customization and image display.
* Implement **design filtering and search functionality** based on clothing attributes.
* **Challenges & Fixes:**
* **Challenge:** Slow image generation.

**Fix:** Optimize Stable Diffusion and use asynchronous tasks.

* **Challenge:** Inconsistent image quality.  
   **Fix:** Refine prompts and parameters, potentially fine-tune the model.

**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 | Query "Generate a red floral dress design"" | A realistic red floral dress image should be displayed. | ✅ Passed | Anika |
| TC-002 | Functional Testing | Query "Casual summer outfit with pastel colors"" | A visually appealing summer outfit should be generated. | ✅ Passed | Srilekha |
| TC-003 | Performance Testing | Model generates an image within 5 seconds | Image should be generated quickly | ⚠ Needs Optimization | Himaja |
| TC-004 | Bug Fixes & Improvements | Fixed incorrect clothing style outputs. | More accurate style representations. | ✅ Fixed | Neha |
| TC-005 | Final Validation | Ensure UI is responsive in Streamlit | UI should work properly across devices. | ❌ Failed - UI broken on mobile | Srilekha |
| TC-006 | Deployment Testing | Deploy the app using Streamlit Sharing | App should be accessible online. | 🚀 Deployed | DevOps |

**Final Submission**

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