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

# Hackathon Project Phases Template

**Project Title:**

**SMART RESUME GENERATOR**

**Team Name:**

**Team-risers**

**Team Members:**

* CH.VINAY KUMAR
* CH.SANJAY
* CH.BHANU MAHENDER
* B.SUNIL
* B.SATHVIK

## Phase-1: Brainstorming & Ideation

**Objective:**

Develop an AI-powered resume-building tool that generates professional, ATS-friendly resumes tailored to industry standards

**Key Points:**

1. **Problem Statement:**

 Many job seekers struggle to create well-structured, visually appealing resumes.

 Recruiters use ATS (Applicant Tracking Systems), which often reject resumes that don't follow specific formats.

 Users need a smart tool to automatically format resumes based on job roles and industrial trends.

.**Proposed Solution:**

** An AI-powered resume generator that creates optimized resumes in multiple formats.**

** Uses machine learning to suggest skills, achievements, and summaries tailored to specific job roles.**

** Provides ATS-friendly templates and real-time feedback on resume quality.**

.

1. **Target Users:**

** Job seekers (students, freshers, and professionals).**

** Career changers looking for optimized resume formats.**

** HR professionals who need quick resume formatting tools.**

.

1. **Expected Outcome:**

* 1. A fully functional Smart Resume Generator that provides users with optimized resumes, improving their job application success rate.

## Phase-2: Requirement Analysis

**Objective:** Define the technical and functional requirements for the Smart Resume Generator.

**Key Points:**

**. Technical Requirements:**

* **Programming Language:** Python
* **Backend:** OpenAI/Gemini API for AI-powered suggestions
* **Frontend:** Streamlit Web Framework
* **Database:** Firebase/SQL for storing user-generated resumes

**2. Functional Requirements:**

* Ability to generate and customize resumes based on job roles.
* Real-time grammar and keyword optimization suggestions.
* Export resumes in PDF and DOCX formats.
* ATS compatibility checker to ensure resume success.

**3. Constraints & Challenges:**

* Ensuring resumes are ATS-friendly and industry-compliant.
* Handling multiple formats and layouts dynamically.
* Optimizing AI-generated suggestions to be concise and relevant.

## Phase-3: Project Design

**Objective:** Develop the architecture and user flow of the application

Develop the architecture and user flow of the application.

**Key Points:**

**1. System Architecture:**

* User inputs job role and experience details via UI.
* AI model processes the input and suggests resume content.
* The frontend dynamically generates a formatted resume.
* Users can edit, download, or share their resume.

**2. User Flow:**

1. Step 1: User selects a job role and fills in basic details.
2. Step 2: AI suggests optimized content and formatting.
3. Step 3: Users can edit sections, choose a design, and preview the resume.
4. Step 4: Downloadable resume in multiple formats (PDF, DOCX).

**3. UI/UX Considerations:**

* Simple and intuitive resume-building process.
* Multiple templates with preview options.
* Mobile-friendly UI for on-the-go resume editing.

## 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 | Ch.vinay kumar | Google API Key,  Python, Streamlit setup | API connection established & working |
| Sprint 1 | Frontend UI Development | 🟡  Medium | 2 hours  (Day 1) | End of Day  1 | Ch.sanjay | API response format finalized | Basic UI with input fields |
| Sprint 2 | Resume generation&formating | 🔴 High | 3 hours  (Day 2) | Mid-Day 2 | Ch.vinay kumar & Ch.sanjay | API response, UI elements ready | Search functionality with filters |
| Sprint 2 | Error Handling &  Debugging | 🔴 High | 1.5 hours  (Day 2) | Mid-Day 2 | Ch.bhanu mahender & B.sunil | API logs, UI inputs | Improved API stability |
| Sprint 3 | Testing & UI  Enhancements | 🟡  Medium | 1.5 hours  (Day 2) | Mid-Day 2 | B.sunil & B.sathvik | 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 **SMART RESUME GENERATOR**

**(**🟡 **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 smart resume generator

**Key Points:**

1. **Technology Stack Used:**

* 1. **Frontend:** Html & css

○ **Backend:** java script & python

○ **Programming Language:** Python & java script

1. **Development Process:**

 Implement AI-driven resume suggestions and formatting logic.

 Optimize keyword detection for ATS compatibility.

 Implement resume template selection and editing

1. **Challenges & Fixes:**

** Challenge: Formatting issues across different templates.**

* **Fix: Standardize layouts with HTML-to-PDF libraries.**

** Challenge: AI-generated text sometimes too generic.**

* **Fix: Fine-tune AI prompts and offer manual edits.**

## 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 "Best budget cars under ₹10 lakh" | Relevant budget cars should be displayed. | ✅ Passed | shanwa z |
| TC-002 | Functional  Testing | Query "Motorcycle maintenance tips for  winter" | Seasonal tips should be provided. | ✅ Passed | anwar |
| TC-003 | Performance  Testing | API response time under  500ms | API should return results quickly. | ⚠ Needs Optimization | Tester 3 |
| TC-004 | Bug Fixes & Improvements | Fixed incorrect API responses. | Data accuracy should be improved. | ✅ Fixed | Develop er |
| TC-005 | Final Validation | Ensure UI is responsive across devices. | UI should work on mobile & desktop. | ❌ Failed - UI broken on mobile | Tester 2 |
| 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**