**Hackathon Project Phases Template** for the **Smart Resume Generator** project.

Hackathon Project Phases Template

# Project Title:

**Smart Resume Generator**

# Team Name:

Tech Titans

# Team Members:

* S.Anitha
* K.Bhuvaneshwari
* K.Harshini
* A.Jyotsna Sravani

# Phase-1: Brainstorming & Ideation

## Objective:  To build a generative model that can craft tailored resumes based on user inputs such as personal information, job experience, and career goals.

## Key Points:

1. **Problem Statement:**
   * Many job seekers struggle with creating a well-structured and professional resume due to a lack of design skills, writing expertise, or knowledge of industry-specific resume standards.
   * A resume generator aims to simplify and streamline the resume creation process by providing users with pre-designed templates, guided content suggestions, and ATS-friendly formatting.
2. **Proposed Solution:**
   * A user-friendly tool that provides guided input fields and AI-powered content suggestions to help users quickly generate professional, ATS-friendly resumes.
   * Allows users to personalize resumes with different layouts, fonts, and colors while offering multiple download formats (PDF, DOCX) for easy job applications.
3. **Target Users:**
   * Job Seekers & Fresh Graduates – Individuals looking for their first job who need guidance in creating a professional resume.
   * Working Professionals & Career Switchers – Employees seeking new opportunities or transitioning to a different industry who need updated, well-structured resumes.
   * Freelancers & Remote Workers – Independent professionals who require tailored resumes to showcase their skills and experience for project-based or remote job applications.
4. **Expected Outcome:**
   * A streamlined resume creation process that enables users to quickly generate professional, ATS-friendly resumes with minimal effort. The tool increases job application success rates by providing well-structured, customizable, and industry-optimized resume templates.

# Phase-2: Requirement Analysis

## Objective:

Define the technical and functional requirements for the Smart Resume Generator.

## Key Points:

1. **Technical Requirements:**
   * Programming Language: **Python**
   * Backend: **Google Gemini Flash API**
   * Frontend: **Streamlit Web Framework**
2. **Functional Requirements:**
   * Create a simple and user-friendly registration and login interface using Streamlit.
   * Handle user authentication and account management using Google Gemini Flash API.
   * Store and update user profile data using Google Gemini Flash API.
   * Use Google Gemini Flash API to fetch and update resume data.
   * Develop an admin interface using Streamlit.
3. **Constraints & Challenges:**
   * Seamlessly integrating with external APIs like Google Gemini Flash.
   * Handling **API rate limits** and optimizing API calls.
   * Providing a **smooth UI experience** with Streamlit.

# Phase-3: Project Design

## Objective:

## A Smart Resume Generator employs a three-tiered architecture: a user-friendly frontend for data input and a robust backend for data processing, AI integration, and document generation. Users follow a streamlined flow, starting with profile creation or import, progressing through structured data entry with real-time previews and AI-powered suggestions, and culminating in resume generation and download.

## Key Points:

1. **System Architecture:**
   * User enters the information required for a resume via UI.
   * The details are processed using **Google Gemini API**.
   * AI model fetches and processes the data.
   * The frontend displays **modified resume, reviews, and suggestions**.
2. **User Flow:**
   * Step 1: User enters a detail (e.g., "Name: Anitha").
   * Step 2: The backend **calls the Gemini Flash API** to retrieve data.
   * Step 3: The app processes the data and **displays results** in a desired resume template selected by the user.
3. **UI/UX Considerations:**
   * **Minimalist, user-friendly interface** for seamless navigation.
   * **Filters for price, mileage, and features**.
   * **Dark & light mode** for better user experience.

# 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 | Resume parsing & data extraction | 🔴 High | 3 hours  (Day 2) | Mid-Day 2 | Member 1& 2 | 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 2 | Skill Experience &  Matching  Algorithm | 🔴 High | 3 hours  (Day 2) | Mid-Day 2 | Member 1&3 | Parsed resume  data, ML  model setup | Match skills  with job  descriptions |
| Sprint 3 | Resume  Formatting & PDF  generation | 🟡  Medium | 2 hours  (Day 2) | Mid-Day 2 | Member 2&3 | Parsed  data, template  designs | Well-  Structured  resume output |
| 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 development environment and install dependencies (e.g., frameworks, libraries).
* **(🟡 Medium Priority)** Set up a basic UI to allow user input for personal information (name, contact details, etc.).

**Sprint 2 – Core Features & Functionality (Day 2)**

* **(🔴 High Priority)** Implement user input forms for work experience, skills, education, and certifications.
* **(🔴 High Priority)** Implement the logic for generating dynamic bullet points for work experience using AI or template-based suggestions.
* **(🟡 Medium Priority)** Develop a preview feature where users can see how their resume looks in real-time.

**Sprint 3 – Testing, Enhancements & Finalization (Day 2)**

* **(🟡 Medium Priority)** Conduct testing on various input types and edge cases (e.g., long job titles, empty fields).
* **(🟡 Medium Priority)** Refine UI design for better user experience.
* **(🟢 Low Priority)** Add ATS (Applicant Tracking System) optimization features (keyword matching for job descriptions).
* **(🟢 Low Priority)** Final demo preparation, polish user interface.

# Phase-5: Project Development

## Objective:

A Smart Resume Generator's core functionality centers on simplifying and enhancing the resume creation process through automation and intelligence.

## 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 (for secure and functional AI capabilities).
   * Develop resume content generation and optimization logic (including skill extraction, keyword optimization, and tailored content suggestions).
   * Optimize prompts and data parsing for performance and accuracy (ensuring efficient and relevant resume generation).
3. **Challenges & Fixes:**
   * **Challenge:** Inconsistent or slow Gemini API response times when generating complex resume sections.

**Fix:** Implement caching to store frequently generated resume sections and optimize prompt structure for faster responses.

* + **Challenge:** Reaching API usage limits during peak user activity.

**Fix:** Implement efficient data handling to minimize API calls, and optimize prompts to consolidate requests and fetch only the required data.

# Phase-6: Functional & Performance Testing

## Objective:

Ensure that the project works as expected.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Category** | **Test Scenario** | **Expected Outcome** | **Status** | **Tester** |
| TC-001 | Functional Testing | Query "Summery should be generated using AI" | Resume with given details should be displayed. | ✅ Passed | Tester 1 |
| TC-002 | Functional Testing | Query "Work experience section with 0" | Work experience  should be skipped (not  be displayed). | ✅ Passed | Tester 2 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 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 using Streamlit Sharing | Website should be accessible online. | 🚀 Deployed | DevOps |

# Final Submission

1. **Project Report Based on the templates**
2. **GitHub/Code Repository Link**
3. **Presentation**