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

Hackathon Project Phases Template

Project Title:

Smart Resume Generator – Customized Resumes for Every Opportunity

Team Name:

Byte Strom

Team Members:

- Pranavi Neela
- Sathwika Nizampet
- Nalla Akshitha
- P.Himavarshini

Phase-1: Brainstorming & Ideation

Objective:

To develop an AI-driven tool that automates resume creation, generating personalized and well-structured resumes based on user inputs. This tool streamlines the process, helping job seekers effectively showcase their skills and improve their employment prospects.

Key Points:

1. **Problem Statement:**

- **Time-Consuming & Complex** – Manual resume creation is tedious, with formatting and personalization challenges.
- **Need for Efficiency** – A quick, AI-driven solution can streamline the process and improve job prospects.

2. **Proposed Solution:**

- **AI-Powered Automation** – Develop a tool that generates well-structured, personalized resumes based on user inputs.
- **Efficiency & Optimization** – Streamline the resume creation process, ensuring professional quality and improved job prospects.

3. **Target Users:**

- **Job Seekers** – Individuals looking for employment, from fresh graduates to experienced professionals.

- **Career Changers** – Professionals transitioning to new industries or roles who need tailored resumes.
- **Recruitment Agencies** – Organizations assisting clients in creating polished, job-ready resumes.

4. **Expected Outcome:**

- AI-driven, personalized resume generation that streamlines the process, ensuring well-structured, professional resumes to enhance job prospects.

Phase-2: Requirement Analysis

Objective:

Define the technical and functional, system, and user requirements to create an efficient, secure, and user-friendly AI-driven Resume Generator.

.

Key Points:

1. **Technical Requirements:**

- **AI Model** – Python-based NLP for resume generation.
- **Web Development** – Front-end (HTML, CSS, JavaScript) and back-end (Django/Flask).
- **Database** – SQL/No SQL for storing user data and resumes.
- **Export & Storage** – Multi-format export and secure cloud storage.

2. **Functional Requirements:**

- **User Input** – Collect personal, career, and skills data.
- **Resume Generation** – AI creates tailored, professional resumes.
- **Customization** – Options to modify templates and content.
- **Export** – Ability to download resumes in various formats..

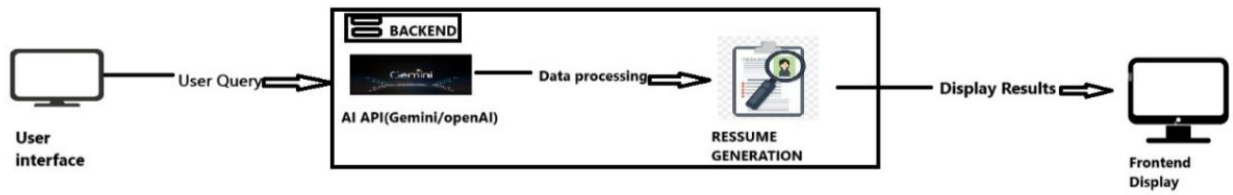
3. **Constraints & Challenges:**

- **Data Accuracy** – Ensuring AI-generated resumes are error-free and relevant.
- **User Customization** – Balancing automation with flexibility for edits.
- **Privacy & Security** – Protecting user data from breaches and unauthorized access.

Phase-3: Project Design

Objective:

To develop the architecture and user flow of the Resume Generator application, ensuring a seamless interaction between the user interface, backend processing, AI model, and resume output display for an efficient and user-friendly experience.



Key Points:

1. **System Architecture:**
    - User enters vehicle-related query via UI.
    - Query is processed using **Google Gemini API**.
    - AI model fetches and processes the data.
    - The frontend displays **vehicle details, reviews, and comparisons**.
2. **User Flow:**
    - Step 1: User enters job details and selects preferences.
    - Step 2: Backend processes input using AI.
    - Step 3: The app generates and displays a customized resume.
    - Step 4: User downloads or edits the resume before finalizing.
3. **UI/UX Considerations:**
    - Minimalist, user-friendly interface for easy navigation.
    - Template selection for different job types.
    - Dark & light mode for better accessibility.
    - Interactive resume preview for real-time edits..

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	Python, Flask, Open AI API	API setup & working
Sprint 1	Frontend UI Development	Medium	2 hours (Day 1)	End of Day 1	Member 2	Basic HTML, CSS, Boots trap	Basic UI with input fields
Sprint 2	Resume Data Processing	High	3 hours (Day 2)	Mid-Day 2	Member 3	Jinja2 template ,Data Parsing	Resume customization working
Sprint 2	Error Handling & Debugging	High	1.5 hours (Day 2)	Mid-Day 2	Member 1&4	API logs, input validation	Improved stability
Sprint 3	Testing & UI Enhancements	Medium	1.5 hours (Day 2)	Mid-Day 2	Member 2& 3	API response,I layout updates	Better UI 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 **Open AI API for** resume content suggestions.
(Medium Priority) Build a **basic UI with input fields**.

Sprint 2 – Core Features & Debugging (Day 2)

(High Priority) Implement **resume customization and formatting .**
(High Priority) Debug API issues & handle **errors in data processing**.

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 App.

Key Points:

1. **Technology Stack Used:**
    - **Frontend:** HTML, CSS, Bootstrap
    - **Backend:** Flask Open AI API
    - **Programming Language:** Python
2. **Development Process:**
    - Implement API key authentication and OpenAI API integration for resume content generation.
    - Develop resume formatting and customization logic for different job roles.
    - Optimize job description parsing to match resume content accurately.

3. **Challenges & Fixes:**
    1. Challenge: Inaccurate keyword matching between job descriptions and resumes.

       Fix: Use NLP-based keyword extraction and similarity matching to improve accuracy.

    - 2. Challenge: Slow response time during resume generation.

       Fix: Optimize API calls, implement caching, and process requests asynchronously.

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	Generate resume for “Software Engineer with 5 years of experience”	Resume should be tailored with relevant experience and skills..	✅Passed	Tester 1
TC-002	Functional Testing	Parse job description for “Data Scientist “ and auto-generate resume	Resume should highlight relevant skills like Python, ML, and Data Analysis.	✅Passed	Tester 2
TC-003	Performance Testing	Generate 5 resumes in 1 minute	System should handle bulk requests efficiently	⚠In Progress	Tester 3
TC-004	UI Testing	Check responsive design mobile & desktop	UI should adjust properly to different screen sizes	✅Passed	Developer
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	Dev Ops

---

Final Submission

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