

Hackathon Project Phases

Project Title:

SmartResume Generator: Customized Resumes For
Every Opportunity

Team Name:

Smart hackers

Team Members:

- Malve Chaitra
 - Sanadi Sadhvika
 - Areej Akram
 - Devari Lahari
-

Phase-1: Brainstorming & Ideation

Objective:

To assist students in creating industry-specific, polished resumes that highlight relevant skills and experiences, enhancing their competitiveness in the job market.

Key Points:

1.Problem Statement:

- Content & Clarity Issues – Lack of experience, unclear goals, weak skill highlighting, and outdated or cluttered content.
- Formatting & Optimization – Poor structure, missing keywords (ATS), grammar errors, and customization difficulties.
- Confidence & Presentation – Struggling to showcase strengths effectively.

2.Proposed Solution:

- Develop an AI-driven tool for automated resume generation.
- Build a generative model that tailors resumes based on user inputs
- Ensure resumes are well-structured, highlighting key skills, achievements, and qualifications.
- Streamline the resume creation process for quick and polished outputs.
- Enhance users' ability to present themselves effectively to employers, improving job application success.

2.Target Users:

- Job Seekers: New graduates or part timers,Career switchers
- Freelancers
- College Students
- Internship Seekers

3.Expected Outcome:

- Tailored Resumes Based on User Input
- Enhanced Efficiency in Resume Creation
- User-Friendly Experience: Step by Step Process

Phase-2: Requirement Analysis

Objective:

Define the technical and functional requirements for the SmartResume Generator

Key Points:

1. Technical Requirements:

- Programming Language: **Python**
- Backend: **Google and HuggingFace API keys**
- Frontend: **Streamlit Configuration**
- Database: **API-based queries**

2. Functional Requirements:

- Collects user's personal details,educational information,work experience,achievements and job description.

- Allow the user to choose between two AI models (Hugging Face and Google Gemini) for generating the resume content.
- Generate the resume text based on the user's inputs by sending a prompt to the chosen AI model (either Hugging Face or Google Gemini).

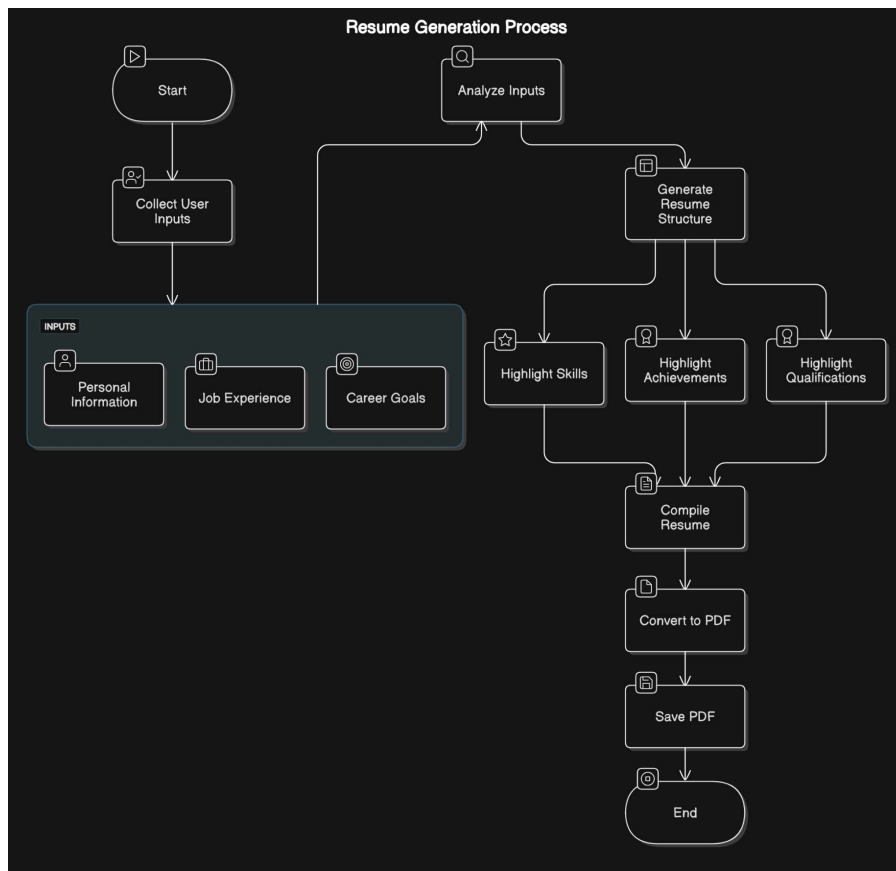
3. Constraints & Challenges:

- **Input Validation**
- Handling **API rate limits** .
- Displaying a progress message while the resume is being generated and allowing easy download of the PDF file improves usability.

Phase-3: Project Design

Objective:

Develop the architecture and user flow of the application.



Key Points:

1. **System Architecture:**
 - Front-End (UI/UX Layer) - Streamlit Application
 - Application Logic Layer: Streamlit Handlers, Test cleaning logic, PDF generation
 - AI model fetches and processes the data.
2. **User Flow:**
 - Step 1: User gets directed to the tool. User enters data asked.
 - Step 2: **Google Generative AI & Hugging Face API** process the user's input through pre-trained models to generate the resume content.
 - Step 3: The app processes the data and **displays results** in an easy-to-read format in PDF.
3. **UI/UX Considerations:**
 - **Minimalist, user-friendly interface** for seamless navigation.
 - **Responsive Layout**
 - The **Download Resume button** should be **prominently placed** and easy to find once the resume is generated.
-

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	Areej,Lahar,C haitra	Google API Key, Python, Streamlit setup	API connection established & working
Sprint 1	Frontend UI Development	● Medium	2 hours (Day 1)	End of Day 1	Sadvika	API response format finalized	Basic UI with input fields
Sprint 2	Developing user interface	● High	3 hours (Day 2)	Mid-Day 2	Sadhvika,Cha itra	API response, UI elements ready	Search functionality with filters

Sprint 2	Error Handling & Debugging	● High	1.5 hours (Day 2)	Mid-Day 2	Chaitra,Areej	API logs, UI inputs	Improved API stability
Sprint 3	Testing & UI Enhancements	● Medium	1.5 hours (Day 2)	Mid-Day 2	Sadhvika	API response, UI layout completed	Responsive UI, better user experience
Sprint 3	Final Presentation & Deployment	● Low	0.5 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:

Implement core features of the [SmartResume Generator](#)

Key Points:

1. Technology Stack Used:

- **Frontend:** Streamlit
- **Backend:** Google Gemini API and HuggingFace API Keys
- **Programming Language:** Python

2. Development Process:

- Implement **API key authentication** and **Gemini and HuggingFace API integration**.
- Develop Resume Layout for user input as well as output
- Optimize the app should have an **intuitive flow** that doesn't overwhelm users with complex options or steps. Users should know exactly what to do at each stage.

3. Challenges & Fixes:


- **Challenge:** Delayed API response times.
Fix: Implement **caching** to store frequently queried results.
- **Challenge:** Limited API calls per minute.
Fix: Optimize queries to fetch **only necessary data**.

Phase-6: Functional & Performance Testing

Objective:

Ensure that the tool works as expected.

Test Case ID	Category	Test Scenario	Expected Outcome	Status	Tester
TC-001	Functional Testing	Ensure all input fields (Name, Age, Email, Phone, LinkedIn, GitHub, Graduation, etc.) accept valid inputs.	User should be able to enter the details .	✅ Passed	Lahari,Areej
TC-002	Functional Testing	Ensure that the required fields (e.g., Name, Email, Phone, Graduation, Skills, Job Description) must be filled out before proceeding to generate the resume	The tool should accept the details correctly and user should be able to enter them with ease.	✅ Passed	Sadhvika,Chaitra
TC-003	Performance Testing	Ensure that the AI model (Hugging Face or Google Gemini) works properly. The user should be able to select only one model at a time.	API should return results quickly.	passed	Sadhvika,Areej, Chaitra
TC-004	Bug Fixes & Improvements	The resume is generated correctly based on the AI model's output. Verify that the PDF generation works correctly. Ensure the content of the generated resume is	The resume should be generated into a pdf format with proper layout.	✅ Fixed	Entire team

		displayed in the PDF output accurately.			
TC-005	Final Validation	Ensure UI is responsive each time of use.	UI should work on mobile & desktop.	passed	Sadhvik a
TC-006	Deployment Testing	Host the tool	The tool,I should be accessible to everyone.	 Deployed	Entire Team

Final Submission

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