Generating resumes based on user input, ensuring customized and tailored resumes for every opportunity.

SMART-RESUME GENERATOR

Project Title:

Smart-Resume Generator: Customized Resumes for Every Opportunity

Team Name:

Next-Gen Coders

Team Members:

- Sk Rehana Azmee.
- Bhukya Navya.
- Ambati Indhu.
- Malligari Poojitha.

Phase-1: Brainstorming & Ideation

Objective:

To generate and refine innovative ideas and approaches for the Smart Resume Generator, ensuring a clear, creative, and feasible plan for development.

Key Points:

1. **Problem Statement:**
    - Job seekers face difficulties creating tailored, high-quality resumes that meet job requirements and showcase their strengths effectively.
    - The manual process of building resumes is time-consuming and prone to errors, leading to missed opportunities for job applicants.

2. **Proposed Solution:**
    - Develop an AI-powered tool to create **personalized, professional resumes** quickly and accurately.
    - Implement an intuitive **user interface for easy customization and real-time feedback,** ensuring tailored resumes that match job requirements.
3. **Target Users:**
    - **Job Seekers**: Create high-quality, tailored resumes quickly.
    - **Career Changers:** Highlight relevant skills for new fields**.**
    - **Students and Graduates:** Gain a competitive edge in the job market**.**
4. **Expected Outcome:**
    - The Smart Resume Generator will produce personalized, professional resumes quickly, **enhancing job seekers' chances of success** by effectively highlighting their strengths.

Phase-2: Requirement Analysis

Objective:

Identify and document technical and functional requirements for the Smart Resume Generator.

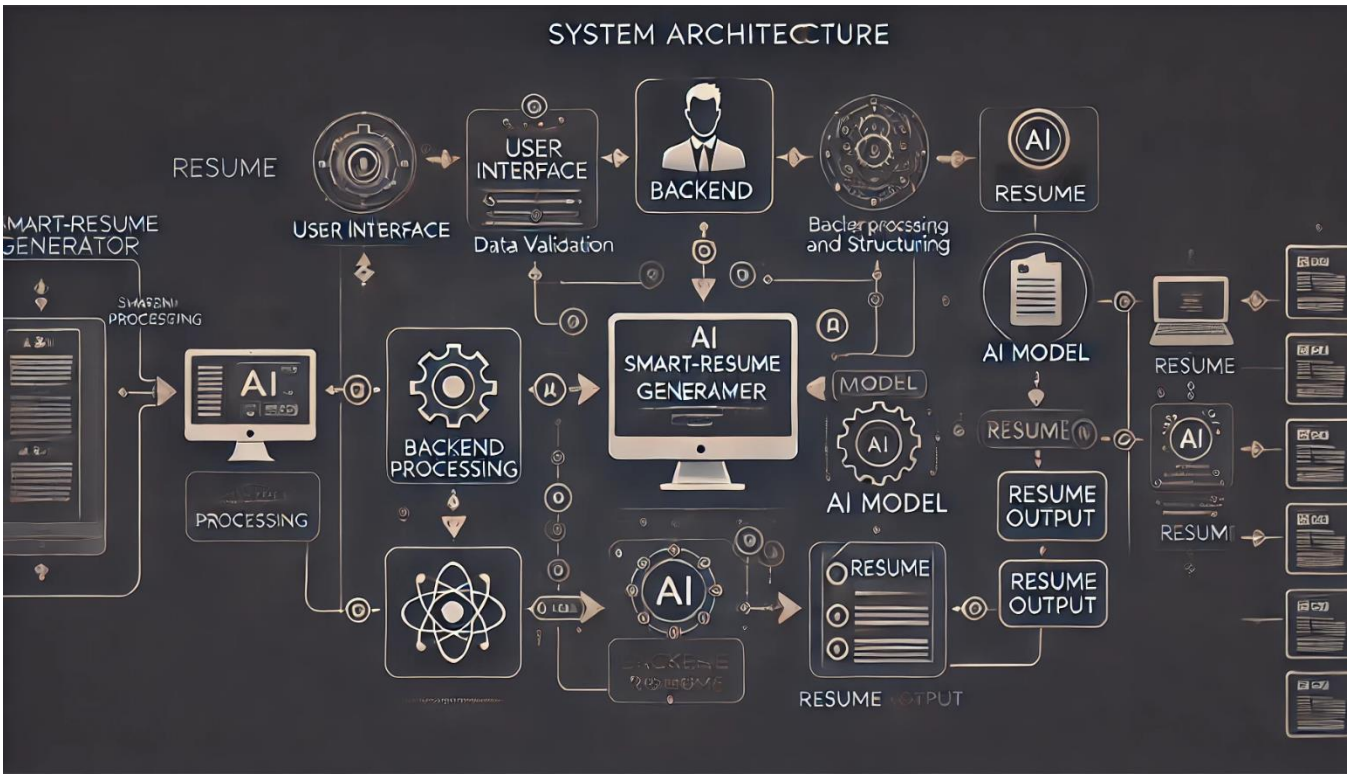
Key Points:

1. **Technical Requirements:**
    - Programming Language: **Python**
    - AI Models: **Natural Language Processing (NLP)** and **Machine Learning (ML)** algorithms.
    - Database: **Azure SQL Database**
    - Framework: **Microsoft Bot Framework**
2. **Functional Requirements:**
    - Enable users to **create** and **manage profiles**.
    - Allow users to **input data** and **generate customized resumes**.
    - Provide various **resume templates** for users to choose from.
    - Offer **suggestions** and **corrections** to improve resume content and formatting.
3. **Constraints & Challenges:**
    - Ensuring timely data retrieval despite **API rate limits**.
    - **Safeguarding** user data and **complying** with regulations.
    - Creating a seamless**,** intuitive interface with **real-time feedback.**

Phase-3: Project Design

Objective:

Develop a clear architecture and user flow for the Smart Resume Generator.



Key Points:

1. **System Architecture:**
   - **Captures user data** and job preferences through an intuitive interface.
   - Utilizes **AI** and **NLP** models to analyze and organize input data.
   - Provides various **resume templates** for customization and formatting.
   - Offers **real-time suggestions** and **improvements** to ensure high-quality resumes.
2. **User Flow:**
   - Step 1: User enters **personal details**, **job preferences**, and selects a r**esume template**.
   - Step 2: **AI processes** the input data, generates a personalized resume, and provides **real - time feedback**
   - Step 3: User **reviews** the resume, makes any necessary edits, and downloads the final version.

3.UI/UX Considerations:

- **Intuitive Design**: User-friendly and easy navigation.
- **Customization:** Flexible template and content customization**.**
- **Responsive Design**: Consistent experience across devices**.**

Phase-4: Project Planning (Agile Methodologies)

Objective:

Develop an AI-powered Smart-Resume Generator with Agile for iterative, user-centric, and scalable deployment.

Sprint	Task	Priority	Duration	Deadline	Assigned To	Dependencies	Expected Outcome
Sprint 1	Environment Setup & API Integration	⍰ High	3 hours	Day 1 (mid)	Member 4	Python, Flask, Open AI, API keys	API connected and Functional
Sprint 1	Frontend UI Development	⍰ Medium	3 hours (Day 1)	End of Day 1	Member 2	API response format finalized	Basic UI with input fields
Sprint 2	Resume Data processing & Structuring	⍰ High	4 hours (Day 1)	Mid-Day 2	Member 3& 1	User input fields, API response	Well – Structured resume sections
Sprint 2	AI model for Resume Generation	⍰ High	4 hours	Day 2 (End)	Member 3	Data Structuring Completed.	AI – Generated resume content.
Sprint 3	Error Handling & Debugging	⍰ Medium	3 hours	Mid-Day 2	Member 1 & 4	API logs, UI inputs	Improved stability & bug fixes.
Sprint 3	UI Enhancements & Formatting	◻ Medium	2 hour	Mid of Day 2	Member 2 & 3	Finalized resume sections, AI output	User – friendly, well-formatted UI
Sprint 4	Export & Download Feature	High	3 hours	Mid of day 2	Member 1	Resume content finalized	Resume export in PDF/Word format
Sprint 4	Final presentation & Deployment	◻ Low	1 hour	End of Day 2	Entire Team	Fully functional System	Demo-ready project

Sprint Planning with Priorities

Sprint 1 – Setup & Integration (Day 1)

(⍰ **High Priority)** Set up the **environment** & install dependencies.
(⍰ **High Priority**) Integrate **OpenAI** for AI-based resume generation.
(⍰ **Medium Priority)** Build a **basic UI with input fields**.

Sprint 2 – Core Features & Debugging (Day 2)

(⚠ High Priority) Process and **structure** user-inputted resume data**. (⚠ High Priority)** Implement AI-based resume generation using **OpenAI.**

(● High Priority) Debug API issues and **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:

Build a functional and user-friendly Smart Resume Generator.

Key Points:

1. **Technology Stack Used:**
    - **Frontend:** html, CSS
    - **Backend:** Flask frameworks with python.
    - **Programming Language:** Python
2. **Development Process:**
    - Integrate **AI models** and **backend functionality** to generate resumes.
    - Build an **intuitive** and **responsive user interface** for input and customization.
    - Conduct thorough **testing** to ensure performance, accuracy, and user satisfaction.
3. **Challenges & Fixes:**
    - **Challenge:** Delayed API Response Times
      Fix: Implement caching**.**
    - **Challenge:** Limited API Calls per Minute**:**
      Fix: Optimize queries for necessary data**.**

Phase-6: Functional & Performance Testing

Objective:

Ensure correct functionality and optimal performance.

Test Case ID	Category	Test Scenario	Expected Outcome	Status	Tester
TC-001	Functional Testing	User inputs resume details & submits form	Resume data should be processed correctly	✅ Passed	Tester 1
TC-002	Functional Testing	AI generates resume based on input data	AI should generate structured resume content	✅ 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 resume formatting	Resume layout should be accurate	✅ Fixed	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	DevOps

---

Final Submission

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