Project Design Phase-II
Technology Stack (Architecture & Stack)

Date	31 January 3035
Team ID	PNT2025TMID01602
Project Name	SmartResume Generator
Maximum Marks	4 Marks

Technical Architecture:

The Deliverable shall include the architectural diagram as below and the information as per the table1 & table 2

Smart Resume Generator:

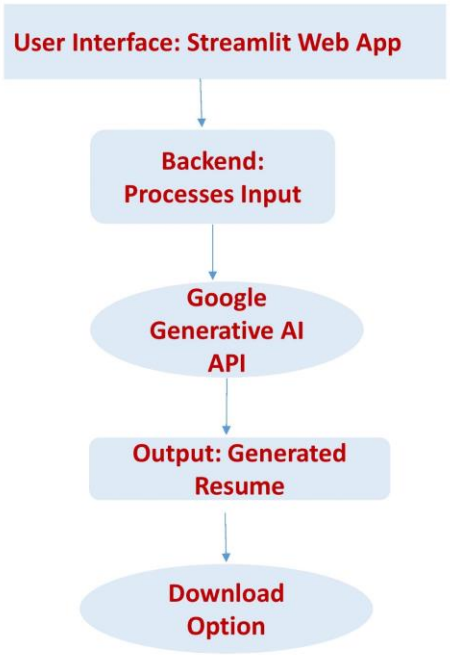

Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	The user interacts with the application via a web interface.	Streamlit (Python)
2.	Application Logic-1	Handles user input and processes it for resume generation.	Python
3.	Application Logic-2	Interacts with the Google Generative AI API to generate resume content.	Google Generative AI API
4.	Application Logic-3	If you have additional logic, such as cleaning the resume text.	Python
5.	Database	If you are storing user data or resumes.	MySQL, NoSQL, etc.
6.	Cloud Database	Database Service on Cloud	Firebase / AWS DynamoDB (if used)
7.	File Storage	If you are storing generated resumes as files.	Local Filesystem / AWS S3 (if used)
8.	External API-1	Google Generative AI API for generating resume content.	Google Generative AI API
9.	External API-2	If you are using any other external APIs.	N/A
10.	Machine Learning Model	Google Generative AI model for generating resume content.	Gemini 1.5 Flash / Gemini 1.0 Pro
11.	Infrastructure (Server / Cloud)	Application deployment on a local system or cloud.	Local / Streamlit Sharing / Heroku

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	List the open-source frameworks used.	Streamlit, Python, Google AI
2.	Security Implementations	Security measures implemented (e.g., API key management).	Environment Variables (.env)
3.	Scalable Architecture	Justify the scalability of architecture (3 – tier, Micro-services)	Microservices (if applicable)
4.	Availability	Justify the availability of application (e.g. use of load balancers, distributed servers etc.)	Streamlit Sharing / Heroku
5.	Performance	Design consideration for the performance of the application (number of requests per sec, use of Cache, use of CDN's) etc.	Google Generative AI API

References:

https://c4model.com/

https://developer.ibm.com/patterns/online-order-processing-system-during-pandemic/

https://www.ibm.com/cloud/architecture

https://aws.amazon.com/architecture

https://medium.com/the-internal-startup/how-to-draw-useful-technical-architecture-diagrams-2d20c9fda90d