GENAI HACKATHON

Project Title:

Job Swift - AI-Powered Career Accelerator

Team Members:

Dileep Kumar P
Karthik G
Daivik Y
Sai Rathan G

Phase-1: Brainstorming & Ideation

Objective:

To build an AI-powered career acceleration platform that leverages Google's Palm text-bison-001 model to help job seekers with resume enhancement, job matching, and AI-driven interview preparation.

Key Points:

1. Problem Statement:

Job seekers often struggle with:

- Unoptimized resumes that don’t pass Applicant Tracking Systems (ATS).
- Difficulty in crafting personalized cover letters for different job applications.
- Lack of targeted job recommendations based on their skills.
- Inadequate interview preparation due to a lack of structured feedback.
- Limited awareness of career growth opportunities and required skill sets.

Traditional job search methods are time-consuming and inefficient, often leading to missed opportunities and delayed career progression.

2. Proposed Solution:

Job Swift is an AI-powered career acceleration platform that leverages Google's Palm text-bison-001 to enhance job-seeking processes through:

Key Features:

1. AI Resume Enhancer – Automatically analyzes and optimizes resumes for ATS compatibility and industry standards.
2. Personalized Job Matching – Uses GenAI-powered algorithms to recommend jobs based on user skills and experience.
3. AI-Powered Cover Letter Generator – Dynamically generates tailored cover letters for job applications.
4. Career Insights & Skill Enhancement – Provides career growth suggestions, including recommended courses and skills to improve employability.

3. Target Users:

- **Job Seekers**: Fresh graduates, professionals switching careers, and experienced individuals looking for career growth.
- **Recruiters & HR Professionals**: To identify well-matched candidates efficiently.
- **Universities & Career Counselors**: To help students prepare for the job market.
- **Freelancers & Gig Workers:** To enhance their portfolios and find better work opportunities.
    - .

4. Expected Outcome:

- Faster Job Placements: Reduces job search time by providing optimized job matches.
- Higher Resume Visibility: Ensures ATS-friendly resumes for better recruiter engagement.
- Improved Interview Readiness: AI-driven feedback enhances candidate confidence and response quality.
- Enhanced Career Growth: Offers personalized learning paths to upskill job seekers.
- Streamlined Hiring Process: Recruiters can find better-qualified candidates more efficiently.

Phase-2: Requirement Analysis

Objective:

Define the technical and functional requirements for **JobSwift**.

Key Points:

1. Technical Requirements:

- **Programming Language:** Python
- **Backend:** Google's **PaLM text-bison-001** API(GOOGLE GEMINI API)
- **Frontend:** Streamlit Web Framework
- **Database:** Not required initially (API-based queries)

2. Functional Requirements:

- Ability to **generate optimized resumes** based on job roles and ATS requirements.
- AI-powered **cover letter generation** tailored to job applications.
- **Job recommendations** based on user skills, experience, and preferences.
- **Mock interview feature** with real-time AI feedback.
- Career growth suggestions and **skill enhancement recommendations**.

3. Constraints & Challenges:

- Ensuring **real-time processing** of AI-generated resumes and cover letters.
- Handling API **rate limits** and optimizing queries for efficiency.

- Providing a **seamless and interactive UI** with Streamlit.

Phase-3: Project Design

Objective:

Develop the **architecture** and **user flow** of the application.

Key Points:

1. System Architecture:

- **User enters a job-related query** or uploads a resume.
- Query is processed using **Google's PaLM API**.
- AI **analyzes, optimizes, and generates** resume/cover letter/job recommendations.
- The frontend **displays results** in an interactive and easy-to-read format.

2. User Flow:

- **Step 1:** User enters a query (e.g., *"Optimize my resume for Data Analyst role"*).
- **Step 2:** Backend calls **Google PaLM API** to process the resume.
- **Step 3:** AI generates **optimized content** and displays it in the UI.
- **Step 4:** User **downloads the resume/cover letter** or views job recommendations.

3. UI/UX Considerations:

- **Minimalist, user-friendly** interface for seamless navigation.
- **Filters for job roles, industries, and experience levels**.
- **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	G. Karthik	streamlit, google-generativeai, requests	API connection established & working
Sprint 1	Frontend UI Development	☐ Medium	2 hours (Day 1)	End of Day 1	Sai Rathan	streamlit, Pillow, datetime	Basic UI with input fields
Sprint 2	Resume & Cover Letter Generation	☐ High	3 hours (Day 2)	Mid-Day 2	Y.Daivik	fpdf, pdfplumber, pytesseract, re	Resume & Cover Letter generator working
Sprint 2	Error Handling & Debugging	☐ High	1.5 hours (Day 2)	Mid-Day 2	P.Dileep Kumar	loguru, requests	Improved API stability

Sprint 3	Testing & UI Enhancements	□ Medium	1.5 hours (Day 2)	Mid-Day 2	Y.Daivik	streamlit, Pillow, datetime	Responsive UI, better user experience
Sprint 3	Final Presentation & Deployment	□ Low	1 hour (Day 2)	End of Day 2	P.Dileep Kumar	streamlit, fpdf, requests	Demo-ready project

Sprint Planning with Priorities:

- **Sprint 1 – Setup & Integration (Day 1)**
  - High: Set up the **development environment** & install dependencies.
  - High: Integrate **Google PaLM API**.
  - Medium: Build a **basic UI** with input fields.
- **Sprint 2 – Core Features & Debugging (Day 2)**
  - High: Implement **resume & cover letter generation**.
  - High: Debug API issues & handle **error management**.
- **Sprint 3 – Testing, Enhancements & Submission (Day 2)**
  - Medium: Test **API responses**, refine UI & fix UI bugs.
  - Low: Final demo preparation & deployment.

Phase-5: Project Development

Objective:

Implement core features of **Job Swift**.

Key Points:

1. Technology Stack Used:

- **Frontend:** Stream lit
- **Backend:** Google **PALM text-bison-001 API(GOOGLE's GEMINI API)**
- **Programming Language:** Python

2. Development Process:

- Implement **API key authentication** and **Google PALM API integration**.
- Develop **resume optimization, cover letter generation, and job recommendation logic**.
- Optimize queries for performance and relevance.

Phase-6: Functional & Performance Testing

Objective:

Ensure that **JobSwift** works as expected.

Test Case ID	Category	Test Scenario	Expected Outcome	Status	Tester
TC-001	Functional Testing	Test Resume Optimization feature with a valid job description	Optimized resume should be generated successfully	Provided the optimized resume in ATS friendly format	Daivik
TC-002	Functional Testing	Test Cover Letter Generation with valid user input	AI should generate a well-structured cover letter	Generated a structured download-able cover letter	Sai Rathan
TC-003	Performance Testing	Evaluate API response time under normal load	API should respond within 2 seconds	API took around 4 seconds to respond	Karthik
TC-005	Error Handling	Enter invalid inputs (e.g., empty job description)	System should display an appropriate error message	Displayed should fill all required fields	Dileep
TC-006	UI/UX	Check responsiveness on different screen sizes	UI should adapt correctly on mobile and desktop	It adapted in all different devices	Dileep

---

Final Submission

Github Repolink: https://github.com/dileepkumar-09/Job--Swift_SIGMA

Deployed applink: https://jobswift-sigma.streamlit.app/