JobSwift

Project Title:

JobSwift : Streamline your job search with AI-powered application.

Team Name:

ZeroBug Samurai

Team Members:

- Gajanan Gourishettishwar
- Sayyad Qamar
- Kharishma Shaik
- Vishal Buyyarapu
- Thilak Murga

Phase-1: Brainstorming & Ideation

Objective:

To simplify and accelerate the job application process by using AI to find suitable job opportunities, automatically customize resumes to match job titles and descriptions, and autofill application details seamlessly through a browser extension.

Key Points:

1. **Problem Statement:**

    - **Time-Consuming Job Search and Application Process:** Job seekers often spend significant time searching for relevant job openings, tailoring resumes for each position, and manually filling out application forms, which can be tedious and inefficient.
    - **Lack of Personalization and Automation in Job Applications:** Many job applications require repetitive data entry and a generic resume, leaving

candidates with little room to stand out or efficiently apply to multiple opportunities without reinventing the wheel each time.

2. **Proposed Solution:**

    - **AI-Driven Job Matching:** The app uses AI to intelligently match users with suitable job opportunities based on their skills, experience, and preferences, streamlining the job search process.
    - **Customized Resume and Coverletter Enhancement:** The app automatically customizes resumes for each job, tailoring content to match job titles and descriptions, improving the chances of getting noticed by employers.
    - **Seamless Autofill with Browser Extension:** A Chrome extension simplifies the application process by autofilling personal details and job-specific information, saving users time and reducing the chances of errors in the application process.
    - **Interactive Chatbot Assistance:** The integrated chatbot guides users through the job application process, offering personalized advice, answering queries, and ensuring that all application steps are completed smoothly.

3. **Expected Outcome:**

    - A functional AI-powered job application app that streamlines the job search process, enhances resumes based on job descriptions, automates application form filling through a browser extension, and provides personalized guidance via a chatbot, ultimately increasing job seekers' chances of securing interviews and job offers with greater efficiency and ease.

Phase-2: Requirement Analysis

Objective:

Define the technical and functional requirements for the JobSwift App.

Key Points:

Technical Requirements

1. Application

- **Programming Language:**
  - **Dart** (for Flutter frontend development)
  - **React** (for web-based chrome extension)
- **Frontend Framework:**
  - **Flutter** (for cross-platform mobile app development, targeting iOS and Android)
- **Backend:**
  - **Node.js** (JavaScript runtime environment)
  - **Express.js** (web application framework for Node.js)
- **API Integration:**
  - **Google PaLM's text-bison-001(Pathways Language Model)** for AI-driven features like job matching, resume enhancement, and chatbot responses
- **State Management:**
  - **Riverpod** (for efficient state management in Flutter)

2. Chatbot

- **Programming Language:**
  - **Dart**
- **API Integration:**
  - **Google PaLM's text-bison-001** for generating intelligent responses based on user queries.
- **Integration with App:**
  - Use **Flutter WebView** or **custom UI components** to embed the chatbot directly within the mobile app.

3. Chrome Extension

- **Programming Language:**

    - **JavaScript** for extension development
- **Extension Framework:**
    - **Chrome Extensions API** for building and managing browser extensions using manifest.json
- **Backend API Integration:**
    - Communicate with the backend (Node.js + ExpressJS + Multer) to store and retrieve application data and gemini api for formatting and reteriving the exact information in json.
- **Functionality:**
    - Autofill personal details and job-specific information from a user's profile onto job application forms using **DOM manipulation** in the extension.

4. LaTeX Code for Resume

- **Programming Language:**
    - **LaTeX** (for generating and formatting resumes)
- **Resume Template:**
    - Pre-designed templates that can be dynamically populated using user data.
- **Backend:**
    - **Node.js** to generate LaTeX code server-side or client-side using a LaTeX rendering engine.
- **API for LaTeX Rendering:**
    - Integration with a LaTeX engine (such as **Overleaf** or **TeX Live**) for rendering the final PDF resume.
- **Custom Fields and Dynamic Content:**
    - Automatically generate sections like **Skills**, **Experience**, **Education**, etc., based on the user's data.

Functional Requirements

1. Application

- **Job Matching:**

    - Automatically suggest job listings based on the user's profile, experience, and preferences. Use AI to match the best-fit jobs.
- **Resume Customization:**

    - Automatically update resumes to match job descriptions and titles using AI-powered algorithms.
- **Job Application Autofill:**

    - Seamlessly autofill application forms with personal details, job preferences, and previous experience using the Chrome extension.
- **User Authentication:**
    - Allow users to register, log in, and manage their profiles via secure login (email, Google, or other authentication methods).

2. Chatbot

- **Personalized Job Search Assistance:**
    - Provide users with job search guidance, recommendations, and job search tips based on queries.
- **Interactive Help:**
    - Answer user questions regarding the application process, job details, or the resume-building process.
- **Real-Time Suggestions:**
    - Offer real-time feedback on resume content, application progress, and any missing fields.

3. Chrome Extension

- **Autofill Job Application Details:**
    - Automatically fill out job application forms on websites using personal and job-specific data retrieved from the app.
- **Profile Syncing:**
    - Sync the user's app profile with the Chrome extension for real-time updates and autofill accuracy.
- **Form Detection:**
    - Detect and autofill compatible application forms across job portals and websites.
- **Data Security:**
    - Ensure secure handling of user data, especially sensitive details like personal info and work history.

4. LaTeX Code for Resume

- **Dynamic Resume Generation:**
    - Automatically generate a professional resume in LaTeX format based on the user's profile data and customized job application needs.

- **Resume Customization:**
    - Allow users to modify the layout, font, and section ordering of their resume templates before generating the final version.
- **PDF Export:**
    - Provide users with the option to download the generated resume as a PDF or share it directly with job portals.

Constraints & Challenges:

1. API Credits

Managing usage limits and costs for third-party APIs (e.g., Google PaLM) for job matching and resume customization.

2. Integration of Chrome Extension with Flutter

Integrating a browser-based Chrome extension with a mobile Flutter app.

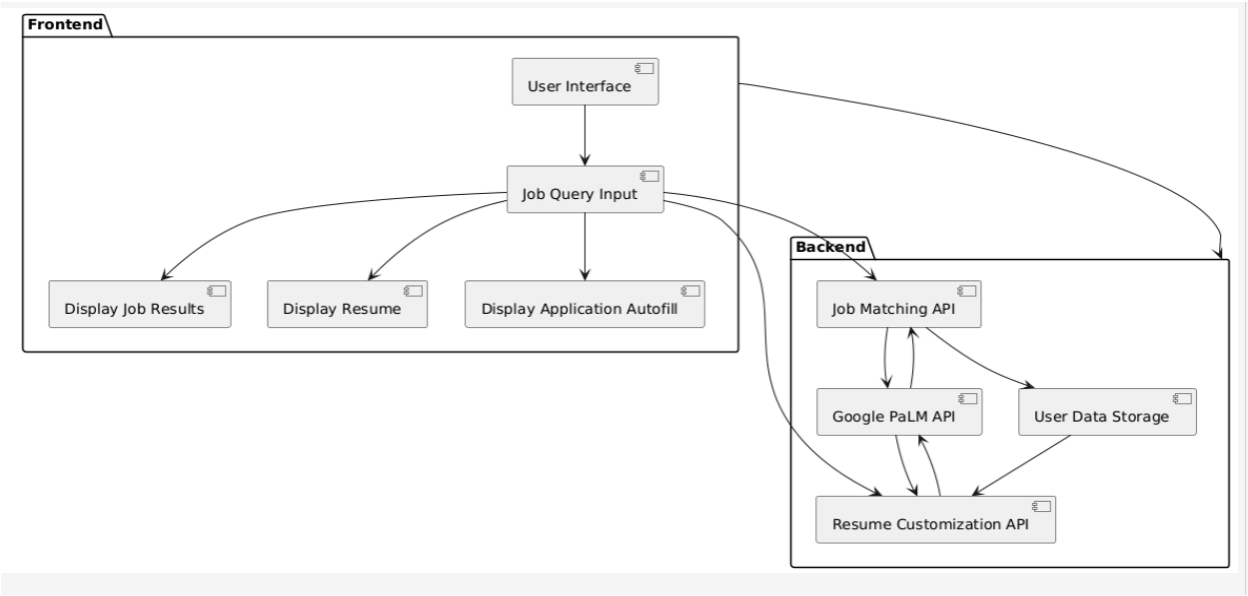
3. LaTeX Code Generation & Overleaf Integration

Dynamically generating LaTeX resumes and integrating with Overleaf for rendering.

Phase-3: Project Design

Objective:

Develop the architecture and user flow of the application.



Key Points:

System Architecture:

- User enters job-related query or profile details via UI.
- Query is processed using **Google PaLM API** for job matching and resume customization.
- AI model fetches and processes relevant job data, enhances resumes, and suggests suitable positions.
- The frontend displays job details, personalized resume suggestions, and autofill options for application forms.

User Flow:

- **Step 1:** User enters a job-related query or uploads a resume (e.g., “Software Developer jobs in New York”).
- **Step 2:** The backend calls the **Google PaLM API** to retrieve relevant job listings and enhance the resume according to job descriptions.
- **Step 3:** The app processes the data, customizes the resume, and displays job listings with autofill options for application forms.

UI/UX Considerations:

- Clean and intuitive interface for smooth navigation through job searches and application processes.
- Filters for job title, location, salary range, and experience level.
- Customizable dark and light mode for a comfortable user experience across different environments.Simple, streamlined resume editor and autofill form for job applications.

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	Vishal	Google API Key, Node.js setup	API connection established & working
Sprint 1	UI/UX Design for Job Search & Resume Customization	🟡 Medium	2 hours (Day 1)	End of Day 1	Gajanan	API response structure ready	Basic UI with job search and resume customization input
Sprint 2	Job Search & AI Integration (Google PaLM)	🔴 High	3 hours (Day 2)	Mid-Day 2	Qamar	API integration, UI structure	Functional job search based on user input using Google PaLM API
Sprint 2	Resume Customization & LaTeX Code Generation	🔴 High	3 hours (Day 2)	Mid-Day 2	Vishal	Resume template, job descriptions	Resume auto-customized for job description, LaTeX code generation
Sprint 3	Chatbot Implementation for Suggestions	🟡 Medium	2 hours (Day 2)	Mid-Day 2	Thilak	Job search data, UI input fields	Chatbot providing real-time job recommendations and resume tips
Sprint 3	Error Handling & Debugging	🔴 High	1 hour (Day 2)	Mid-Day 2	Entire Team	Functional app, API calls	Bug-free app with error-free interactions across features
Sprint 3	Final Testing, Deployment & Demo Preparation	🟢 Low	1 hour (Day 2)	End of Day 2	Entire Team	All features complete	Demo-ready, fully functional app ready for presentation

Sprint Planning with Priorities

Sprint 1 – Setup & Integration (Day 1)

- **(🔴 High Priority)** Set up the development environment & install dependencies (Node.js, Flutter, Google PaLM API).
- **(🔴 High Priority)** Integrate Google PaLM API for job matching and resume customization.
- **(🟡 Medium Priority)** Build a basic UI with input fields for job search and resume upload.
- **(🟡 Medium Priority)** Set up Chrome Extension environment for autofill functionality and data transfer between the extension and the mobile app.

Sprint 2 – Core Features & Debugging (Day 2)

- **(🔴 High Priority)** Implement job search functionality using AI (Google PaLM) and display job recommendations.
- **(🔴 High Priority)** Implement resume customization feature, auto-generate a customized resume using job descriptions, and integrate LaTeX code generation.
- **(🔴 High Priority)** Implement the Chrome Extension for autofill feature on job application forms, syncing with the mobile app.
- **(🔴 High Priority)** Integrate chatbot for real-time job suggestions, resume tips, and job-related queries.
- **(🔴 High Priority)** Debug API-related issues and handle errors in job queries, resume data, autofill feature, and chatbot interactions.

Sprint 3 – Testing, Enhancements & Submission (Day 2)

- **(🟡 Medium Priority)** Test API responses, refine UI design, and fix UI-related bugs for an improved user experience.
- **(🟡 Medium Priority)** Test Chrome Extension integration and ensure autofill works seamlessly on job application forms.
- **(🟡 Medium Priority)** Test chatbot interactions for smooth communication and job suggestions.
- **(🟢 Low Priority)** Final demo preparation, deployment, and making the app ready for presentation.

Phase-5: Project Development

Objective:

Implement the core features of the **JobSwift App** (Job Application Filling App), focusing on job search, resume customization, chatbot, and Chrome Extension integration.

Key Points:

Technology Stack Used:

- **Frontend**: Flutter (for mobile app interface)
- **Backend**: Google PaLM API (AI-powered job search and resume customization)
- **Programming Language**: Dart (for Flutter) & Node.js (for backend)
- **Chrome Extension**: JavaScript (for autofill functionality)

Development Process:

1. **API Key Authentication & Gemini API Integration:**
    - Integrate Google PaLM API for job search and resume customization.
    - Ensure proper API key authentication to securely fetch job listings and enhance resumes based on job descriptions.
2. **Job Search Functionality:**
    - Implement job search logic using AI (Google PaLM) to fetch suitable job recommendations based on user input (job title, location, experience level, etc.).
    - Integrate relevant filters such as salary range, job type, and experience.
3. **Resume Customization:**
    - Implement the resume customization feature that auto-generates and tailors resumes to match the job description.
    - Use **LaTeX code generation** to format and structure resumes for easy download and submission.
4. **Chatbot for Job Suggestions and Resume Tips:**
    - Integrate a chatbot that provides real-time job suggestions, resume tips, and answers to user queries.
    - Train the chatbot to understand common job-related questions and provide personalized recommendations.

5. **Chrome Extension for Autofill:**

    - Implement a Chrome Extension that integrates with the mobile app to autofill job application forms on external websites (like LinkedIn, Indeed, etc.).
    - Sync data from the mobile app to the extension to allow seamless autofill based on user preferences.

Challenges & Fixes:

- **Challenge 1**: **Delayed API Response Times**
    - **Fix**: Implement **caching** on frequently queried data (e.g., common job searches and job listings) to improve response times and user experience.
- **Challenge 2**: **Limited API Calls Per Minute**
    - **Fix**: Optimize queries to the **Google PaLM API** by fetching only essential data (e.g., job title, location, salary) and reducing unnecessary calls. Implement **batch processing** to group multiple queries into a single API call where possible.
- **Challenge 3**: **Chrome Extension Integration with Flutter**
    - **Fix**: Use a **JavaScript-based backend** to handle Chrome Extension communication and ensure data is transferred securely between the extension and the mobile app using shared APIs.
- **Challenge 4**: **Handling Complex Resume Customization**
    - **Fix**: Use AI-based logic to identify key skills and experiences from both the job description and user-uploaded resume. Tailor the resume dynamically, ensuring accuracy and relevancy based on the job listing.

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	Query "Best jobs for software engineers"	Relevant job listings for software engineers should be displayed	✅ Passed	Qamar
TC-002	Functional Testing	Query "Resume tips for marketing managers"	Resume should be auto-customized with relevant tips for marketing	✅ Passed	Vishal
TC-003	Functional Testing	Enter job description and customize resume	Resume should be auto-generated based on the job description	✅ Passed	Kharishma
TC-004	Performance Testing	API response time under 500ms	API should return results within 500ms for job searches and resume updates	⚠ Needs Optimization	Gajanan
TC-005	Bug Fixes & Improvements	Fixed issues with Chrome Extension autofill	Chrome Extension should autofill job application forms correctly	✅ Fixed	Gajanan
TC-006	Bug Fixes & Improvements	Fixed chatbot responses for job-related queries	Chatbot should provide accurate job recommendations and resume tips	✅ Fixed	Qamar
TC-006	Final Validation	Ensure UI is responsive across devices (mobile & desktop)	UI should work seamlessly across mobile and desktop devices.	✅ Passed	Thilak
TC-007	Deployment Testing	Complete app using Flutter	App should be accessible online.	⚠ Needs Optimization	

Final Submission

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