**Hackathon Project Phases Template** for the **SMART RESUME GENERATOR** project.

**Hackathon Project Phases Template**

**Project Title:**

**Smart Resume Generator Customised Resumes for Every Opportunity**

**Team Name:**

**SPARTANS**

**Team Members:**

* L. Himavarsha
* M. Sai Vaishnavi
* K. Jagruthi
* K. Ushasri

**Phase-1: Brainstorming & Ideation**

**Objective:**

"To design and develop an intelligent resume generator that utilizes Al-powered algorithms to create tailored, high-quality resumes for job seekers, increasing their chances of landing their desired role."

**Key Points:**

1. **Problem Statement:**
   * “Despite the abundance of resume-building resources, job seekers continue to struggle with creating effective, tailored resumes that showcase their skills and experiences, resulting in low response rates and missed opportunities.
   * The existing resume-building solutions often require extensive manual customization, lack personalization, and fail to optimize resumes for applicant tracking systems (ATS), leading to a significant gap between job seekers' expectations and the reality of the job market."
2. **Proposed Solution:**
   * 1. AI-powered Resume Generation
   * 2. Automated Keyword Optimization
   * 3. Real-time Job Matching
   * 4. User-Friendly Interface
   * 5. Cloud-Based Resume Storage
   * 6. Collaborative Review & Feedback
3. **Target Users:**
   * 1. Job Seekers: Fresh graduates, entry-level professionals, and experienced individuals looking for new job opportunities.
   * 2. Career Changers: Professionals transitioning to new industries or roles.
   * 3. Freelancers: Independent contractors and freelancers seeking to showcase their skills and services.
   * 4. Students: University students and recent graduates seeking internships or entry-level positions.
   * 5. Career Counselors: Professionals assisting clients with resume building and job search strategies.
   * 6. Recruitment Agencies: Agencies seeking to streamline their resume screening and candidate matching processes.
4. **Expected Outcome:**
   * 1. Increased Efficiency: Automate resume building, saving users time and effort.
   * 2. Improved Resume Quality: Generate high-quality, tailored resumes that showcase users' skills and experiences.
   * 3. Enhanced Job Matching: Provide users with relevant job suggestions, increasing their chances of landing their desired role.
   * 4. Better Career Outcomes: Empower users to achieve their career goals through effective resume building and job matching.
   * 5. Increased User Engagement: Offer a user-friendly interface, encouraging users to actively manage their resumes and job searches.
   * 6. Reduced Unemployment: Contribute to reducing unemployment rates by connecting job seekers with relevant job opportunities.

**Phase-2: Requirement Analysis**

**Objective:**

Defining the Functional and Non-functional requirements for the Smart Resume Generator.

**Key Points:**

**Functional Requirements:**

1. User Registration and Login

2. Resume Building with AI-powered suggestions

3. Customization options for resume templates, fonts, and colors

4. Integration with job databases for real-time job matching

5. Automated keyword optimization for ATS compatibility

6. Cloud-based resume storage and management

7. Collaborative review and feedback features

**Non-Functional Requirements:**

1. User-friendly interface with intuitive navigation

2. Fast and accurate AI-powered resume generation

3. High-quality, visually appealing resume output

4. Secure and reliable cloud-based storage

5. Scalability to accommodate a large user base

6. Compatibility with various devices and browsers

7. Regular updates and maintenance for optimal performance

**Performance Requirements:**

1. Response time: < 3 seconds for resume generation

2. Accuracy: > 90% for AI-powered suggestions

3. Uptime: > 99.9% for cloud-based storage

4. Data security: encryption and secure authentication protocols

**Usability Requirements:**

1. Easy registration and login process

2. Intuitive resume building and customization options

3. Clear and concise job matching results

4. Simple and secure collaborative review and feedback features

**Security Requirements:**

1. Data encryption for user information and resumes

2. Secure authentication and authorization protocols

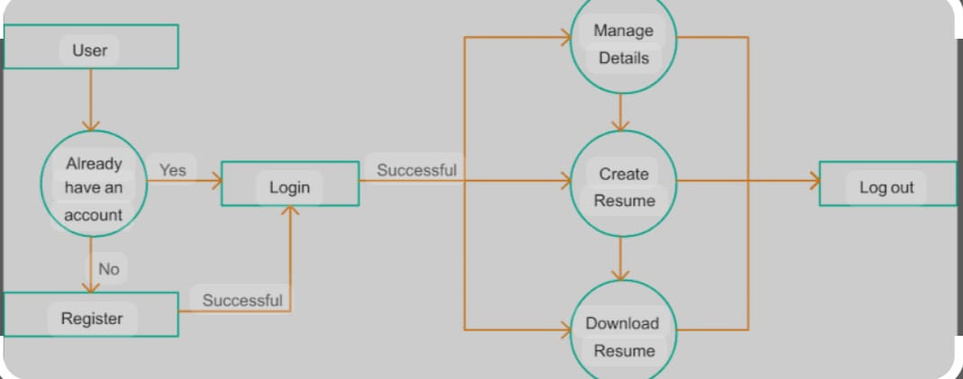
3. Regular security updates and patches

4. Compliance with relevant data protection regulations

**Phase-3: Project Design**

**Objective:**

Develop the architecture and user flow of the application.



**Key Points:**

**Project Architecture**

1. Frontend: User Interface built using HTML, CSS, JavaScript, and a framework like React or Angular.

2. Backend: Server-side logic built using a language like Python, Java, or Node.js, and a framework like Django, Spring, or Express.

3. Database: Cloud-based database management system like AWS Aurora, Google Cloud SQL, or Microsoft Azure SQL Database.

4. AI/ML Engine: Integration with AI/ML libraries like TensorFlow, PyTorch, or scikit-learn for resume analysis and generation.

**System Components**

1. User Module: Handles user registration, login, and profile management.

2. Resume Builder Module: Allows users to input their information and generates a customized resume.

3. AI-powered Resume Analysis Module: Analyzes user input and generates suggestions for improvement.

4. Job Matching Module: Integrates with job databases and suggests relevant job openings to users.

5. Collaborative Review Module: Enables users to share their resumes with others for review and feedback.

**Data Flow**

1. User Input: Users enter their information and resume details.

2. Resume Generation: The system generates a customized resume based on user input.

3. AI-powered Analysis: The AI/ML engine analyzes the resume and provides suggestions for improvement.

4. Job Matching: The system suggests relevant job openings to the user.

5. Collaborative Review: Users share their resumes with others for review and feedback**.**

**User Interface**

1. Responsive Design: A mobile-friendly and responsive design to ensure accessibility across devices.

2. Intuitive Navigation: Clear and concise navigation to guide users through the resume building and job matching process.

3. Real-time Feedback: Instant feedback and suggestions for improvement during the resume building process.

**Technical Requirements**

1. Cloud Hosting: Host the application on a cloud platform like AWS, Google Cloud, or Microsoft Azure.

2. Scalability: Design the system to scale horizontally to handle increased traffic and user growth.

3. Security: Implement robust security measures to protect user data and ensure compliance with relevant regulations.

**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 | Member 1 | Python, Streamlit setup | API connection established & working |
| Sprint 1 | Frontend UI Development | 🟡 Medium | 2 hours (Day 1) | End of Day 1 | Member 2&4 | API response format finalized | Basic UI with input fields |
| Sprint 2 | Error Handling & Debugging | 🔴 High | 3 hours (Day 2) | Mid-Day 2 | Member 3 | API logs, UI inputs | Improved API stability |
| Sprint 3 | Final Presentation & Deployment | 🟢 Low | 1 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 SMART RESUME GENERATOR.

**Key Points:**

1. **Technology Stack Used:**
   * **Frontend:** Streamlit
   * **Backend:** Python
   * **Programming Language:** Python
2. **Development Process:**
   * Implement Ul components and routing
     + Implement state management and API integration
   * Set up backend framework and implement API endpoints
   * Implement authentication, authorization, and error handling
   * Unit testing, integration testing, and user acceptance testing
3. **Challenges & Fixes:**
   * **Challenges:** Users may resist adopting the new resume generator tool.
     + **Fix:** Provide clear user documentation, training, and support to ensure a smooth transition.
   * **Challenges:** User data and resumes may be vulnerable to security breaches.
     + **Fix**: Implement robust security measures, including encryption, access controls, and regular security audits.

**Phase-6: Functional & Performance Testing**

**Objective:**

Ensure that the SMART RESUME GENERATOR works as expected.

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID** | **Category** | **Status** | **Tester** |
| TC-001 | Functional Testing | ✅ Passed | Tester 1 |
| TC-002 | Functional Testing | ✅ Passed | Tester 2 |
| TC-003 | Performance Testing | ⚠ Needs Optimization | Tester 3 |
| TC-004 | Bug Fixes & Improvements | ✅ Fixed | Developer |
| TC-005 | Final Validation | ❌ Failed - UI broken on mobile | Tester 2 |
| TC-006 | Deployment Testing | 🚀 Deployed | DevOps |

**Final Submission**

1. **Project Report Based on the templates**
2. **GitHub/Code Repository Link**
3. **Presentation**