Frigga Cloud Labs – Assignment

(Associate Software Engineer)

Position: Associate Software Engineer (ASE)

Location: Whitefield, Bangalore (On-Site)

Qualification: B.Tech in CS/IT Graduates Only (2024 or 2025 Pass outs Only) – this qualification is mandatory, else your application is auto-rejected.

Salary Structure (₹5.3L):

- Fixed Component: ₹3.24L
- Joining Bonus/Relocation Cost: ₹10K (paid after 3 months)
- Annual Bonus: ₹50K
- ESOPs: ₹1.5L

Assignment Details

Building Knowledge Base Platform (Confluence-Like)

We're not just evaluating your ability to implement features—we're seeking someone who can truly innovate and craft something so compelling it leaves a lasting impression. Think of this assignment as your stage to showcase your creativity, originality, and the power of your ideas. This is your chance to stand out and impress us with a vision that goes beyond expectations.

Introduction

In today's fast-paced digital environment, organizations require robust tools to capture, organize, and share knowledge. Platforms like Confluence have become essential for collaborative documentation and information management. This assignment challenges you, as a full-stack developer, to design and implement a scalable knowledge base platform that emulates core features of Confluence, with your own creative and technical contributions.

Assignment Overview

You will architect and develop a web-based Knowledge Base Platform. The platform is intended for teams to create, edit, organize, and search documents collaboratively. The solution should prioritize usability, performance, modularity, and security. Your implementation will be evaluated on functionality, code structure, UI/UX quality, scalability, and adherence to best practices.

AI Tool Usage

- You are encouraged to use AI tools (ChatGPT, Claude, Copilot, etc.) for assistance
- We want to see how effectively you can leverage AI to build quality software
- Focus on understanding and customizing AI-generated code rather than blind copy-paste

Evaluation Criteria

You will be evaluated on the following parameters. **Minimum passing score: 13/20**

Parameters	Weightage	Description
Code Structure & Organization	4/20	Clean folder structure, proper separation of modules.
Code Quality & Best Practices	4/20	Clean code principles, proper naming conventions, and code readability
Implementation Architecture	3/20	System design, API structure, component architecture.
Database Design	3/20	Schema design, relationships, indexing, data integrity
Feature Completeness	3/20	All required features are implemented and working
User Experience	2/20	Intuitive interface, responsive design, error handling
Documentation Quality	1/20	Clear setup instructions, code comments, API docs.

Required Features

User Authentication System

- User registration with email.
- Login with email/password
- Forgot password functionality with email reset
- JWT-based authentication for API security

Document Management

- Document Listing: Display all accessible documents with metadata (title, author, last modified, visibility status)
- Document Creation: Rich WYSIWYG editor for creating formatted documents
- Document Editing: In-place editing with auto-save functionality
- Search Functionality: Global search across document titles and content

User Collaboration

- User Mentions: @username functionality that triggers notifications
- Auto-sharing: When a user is mentioned, they automatically get read access to the document

Privacy Controls

- Public Documents: Accessible to anyone with the link (no login required)
- Private Documents: Only accessible to the author and explicitly shared users
- Sharing Management: Add/remove user access with different permission levels (view/edit)

Bonus Feature: Version Control & History

- Track all document changes with timestamps
- Display version history with ability to view previous versions
- Show who made changes and when
- Compare versions (basic diff view)

Submission Instructions

- Submit source code via a public Git repository (GitHub, GitLab, or Bitbucket) with clear commit history.
- Include a README with setup steps, architecture explanation, and credentials for demo accounts.
- Deploy a live demo video (optional but strongly encouraged) and share the URL.

Technical Requirements

- **Backend:** Any
- **Frontend:** Any (Preferred React/Next)
- **Database:** Any
- **Authentication:** JWT or session-based authentication
- **WYSIWYG Editor:** AnyAPI Design
- **Architecture:** Modular.

Tips for Success

- **Start with core features** before adding bonus functionality
- **Use AI effectively** - don't just copy-paste, understand and adapt
- **Test thoroughly** - ensure all features work as expected
- **Keep it simple** - focus on clean, working implementation over complex features
- **Document well** - clear instructions help us evaluate your work
- **Plan your time** - allocate time for testing, documentation, and deployment

Hint for the next round:

- Code Explanation
- Deploying it on a VM
- Building a small feature over a face-to-face session.

Send your assignment to shafan@frigga.cloud and cc: veer@frigga.cloud, rakshitha@frigga.cloud

All the Best!