CleverDocs PRD

CleverDocs is an AI-powered onboarding and knowledge-sharing platform that accelerates new-hire ramp-up and empowers engineers to formalize and distribute institutional knowledge as polished, multi-level tech blogs and custom guides.

Problem Statement:

- Onboarding friction: New engineers struggle to find consistent, up-to-date documentation and lose days context-switching among outdated wikis, internal threads, and scattered notes.
- Knowledge hoarding: Senior contributors accumulate years of quick-jot notes but lack an efficient way to structure and publish their insights.
- Tutorial overload: Learners bounce among official docs, community tutorials, and repos without a clear path or level of depth.

Objectives:

- Rapid blog generation: Transform raw notes into well-structured, accurate, and typo-free tech blogs in under five minutes.
- Dynamic expertise slider: Offer on-demand Beginner, Intermediate, and Expert variants of each blog.
- Custom guide builder: Assemble topic-specific blogs into streamlined, use-case–tailored playbooks.
- Community & curation: Support Official (admin-curated) and Community sections with ratings, comments, and leaderboards.

Key Features:

1. Note → Blog Conversion

Allow users to turn years of cluttered, quickly jotted notes into well-documented, in-depth tech blogs with examples and share them with the community.

- Input: File upload, text paste, or URL link
- Pipeline:
- Chunk and embed notes via AWS Bedrock or SageMaker
- Summarize into blog structure: Introduction, Steps, Examples, Conclusion
- Secrets detection and redaction via Amazon Macie or custom regex

- QA:
- Automated grammar and spell checks using Amazon Comprehend and custom dictionaries
- Link validity scanner for outdated URLs
- Human-in-the-loop review queue for Official content publications
- Custom Content for Each Reader’s Level of Expertise

Every tech blog’s content can be dynamically adapted for different levels of expertise of its reader.

- Levels:
- Beginner: High-level explanations, analogies, inline definitions, minimal prerequisites
- Intermediate: Clear code examples, configuration details, trade-off discussions
- Expert: Deep dives on optimizations, edge cases, and advanced patterns
- Implementation: On-the-fly prompts to the Bedrock LLM with the original blog context and level parameter
- Custom Guide Builder

The platform uses AI to combine multiple blogs into one easy-to-follow, step-by-step custom guide tailored to individual use cases.

- Tagging and Indexing: Amazon OpenSearch Serverless for vector and full-text search of blogs
- Source Ranking: Metadata-driven (Official vs. Community rating, recency)
- Interfaces:
- Wizard UI: Dropdowns and checkboxes for tech-stack selections (e.g., React, Kubernetes)
- Chatbot: Conversational refinement of scope and content ordering
- Community and Official Sections
- Community: Open uploads, upvotes, downvotes, comments, version history
- Official: Admin-only publishing, audit logs, gated review, and comment moderation
- Gamified Leaderboards

- Points for blog contributions, ratings received, and comments posted
- Leaderboards by team or organization, with badges awarded for milestones

System Architecture

- Frontend: React, Vite, TypeScript, Chakra UI
- Backend: Python FastAPI, AWS Step Functions for AI pipelines
- Storage: S3 for raw uploads and generated content; DynamoDB for metadata and ratings; OpenSearch Serverless for search and retrieval
- AI Services: AWS Bedrock (LLM + RAG) with OpenSearch knowledge base; Amazon Comprehend for entity recognition; Macie for secrets detection
- Auth & Security: AWS Cognito, IAM roles, VPC endpoints
- CI/CD: GitHub Actions, Terraform