Business Requirements Document (BRD)

Project Title: Next.js Blog Post Application **Prepared For**: Stakeholders, Product Owners **Prepared By**: Development Team **Date**: May 2025

---

1. Executive Summary

The goal of this project is to develop a **feature-rich blog post application** using **Next.js 15**, showcasing key features of the modern web stack while serving as both a **demonstration project** and a **functional blogging platform**. It will include secure user authentication, content creation tools, commenting features, and an administrative dashboard—all built with performance, SEO, and scalability in mind.

---

2. Business Objectives

- Deliver a fully functional blog application that **demonstrates capabilities of Next.js 15**.
- Provide a **modern, user-friendly platform** for writing, reading, and managing blog posts.
- Allow **authenticated users** to interact with content (likes, comments).
- Allow **administrators** to manage posts, categories, tags, and comments securely.
- Ensure the platform is **scalable, secure, SEO-optimized, and mobile-friendly**.
- Serve as a **reference project** for showcasing technical skills and frontend/backend integration.

---

3. Key Stakeholders

Stakeholder	Role	Responsibilities
Product Owner	Oversees vision and priorities	Defines feature requirements and goals
Development Team	Builds the application	Implements the frontend, backend, and DevOps flows
Content Creators	End users	Writes and manages blog posts
Admin Users	System administrators	Manages users, posts, comments, and metadata
Readers / Visitors	Public users	Browse, read, and interact with content

---

4. Scope of Work

In Scope:

- Blog post creation, editing, deletion (admin-only)
- User authentication via Google/GitHub (OAuth)
- Commenting system with optional threading
- Admin dashboard for managing posts/tags/categories
- Search and filtering capabilities
- Mobile-friendly, responsive UI

- SEO and performance optimization
- Deployment to Vercel or container-based platforms

Out of Scope (initial version):

- Paid subscriptions or monetization
- Multi-language support
- Complex analytics dashboards
- CMS integration (e.g., WordPress headless)

---

5. Business Requirements

ID	Requirement
BR1	The system must allow visitors to browse and read blog posts freely.
BR2	Authenticated users must be able to comment on blog posts.
BR3	Admins must have a secure dashboard to create, edit, or delete content.
BR4	Blog content must support rich text formatting via a Markdown/WYSIWYG editor.
BR5	The platform must support category and tag-based filtering.
BR6	Users must be able to sign in using GitHub or Google accounts.
BR7	The application must be optimized for mobile and SEO performance.
BR8	All user-generated content must be sanitized and validated.
BR9	System should support both static and server-side rendered pages.
BR10	The application must maintain consistent layout and branding.

---

6. Success Criteria

- Admins can securely manage blog content.
- Authenticated users can comment and like posts.
- Blog post pages load quickly with SEO metadata.
- Users can filter and search blog content effectively.
- Deployed system runs with zero critical bugs.
- Responsive design verified on mobile and desktop.
- Application deployed successfully to Vercel or Docker platform.
- Basic test coverage in place for frontend and backend logic.

---

7. Assumptions

- Admin and content creators are familiar with Markdown or rich text formatting.
- All user interactions will be session-based and require login for protected actions.
- SEO and image optimization are considered valuable for blog visibility.

---

8. Constraints

- Must use **Next.js 15 App Router**.
- Must use **TypeScript** and follow modern coding standards.
- Deployment should be possible via **Vercel** or **Docker-compatible platforms**.
- Must be completed using open-source or free-tier tools for demo purposes.

---

9. Risk Factors

Risk	Mitigation Strategy
OAuth service downtime	Provide fallback or error UI
Incomplete authentication flow	Use official `next-auth` integration
Security vulnerabilities	Input sanitization and strict access control
SEO misconfiguration	Leverage `generateMetadata()` and structured data
Deployment platform limits (Vercel)	Support Docker for flexible alternatives (e.g. Railway)

---

10. Future Considerations (Phase 2)

- Integration with external CMS (e.g., Sanity, Contentful)
- Analytics and performance tracking dashboards
- Multi-author roles and content approval flows
- Localization and language support
- Email notifications and subscriptions
