North East University Bangladesh

Project Documentation

Course Code: CSE-334

Course Title: Software Engineering

Submitted To

Sabuj Chandra Paul

Guest Lecturer at NEUB

Dept. of CSE

Submitted By

Name: Lubna Yesmin

Reg:0562220005101006

Name:Fabiha Bushra Chowdhary

Reg:0562220005101016

Name:Farhana Akter Dina

Reg:0562220005101031

Semester:6th

Blog Website Project Documentation

---

1. Introduction

Overview

The Blog Website is a full-stack web application designed to allow users to write, edit, and read blog posts. The platform provides a space for users to express thoughts, share information, and engage with content through likes and user profiles. Admin users have additional privileges to manage site content and users.

---

2. Background of the Project

Problem Statement

In the age of digital media, many users seek a simple, reliable platform to publish and manage personal blogs. Existing platforms are often overloaded with features, difficult to navigate, or lack customization options for admin-level control.

Motivation

Our motivation was to build a minimal, scalable blog website using modern technologies where both users and admins have defined, efficient workflows.

---

3. Objectives

- Enable users to register, login, and manage their blog posts.
- Provide an editor to create and publish content.
- Implement functionality to like posts.
- Enable admin-level content and user management.
- Build a clean and responsive UI using modern frontend technologies.

---

4. Scope

In Scope:

- User registration and login.
- Writing, editing, and deleting blog posts.
- User profile management.
- Admin dashboard for user and post control.
- MongoDB-based backend with REST API.

Out of Scope:

- Commenting system.
- Blog post search.
- Third-party social login.
- Push notifications.

5. Literature Review / Related Work

Existing Systems

Platforms like WordPress, Medium, and Blogger allow content creation and sharing. However, they are either too complex or closed-source.

References

- Medium UI structure.
- MongoDB and REST API-based blog architecture from educational resources.
- Mongoose ODM models for structured data handling.

---

6. Methodology

Technologies Used:

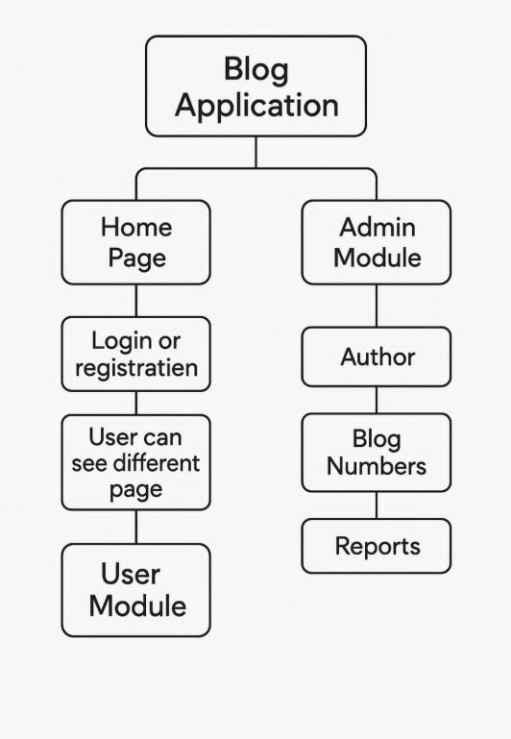
- **Frontend**: React.js, Tailwind CSS / Bootstrap
- **Backend**: Node.js, Express.js
- **Database**: MongoDB (NoSQL)
- **Other Tools**: Postman, VS Code, GitHub

Workflow:

- Requirements gathering

- UI/UX design
- API development
- Frontend integration
- Admin module development
- Testing & debugging
- Deployment

Diagrams:



---

7. Implementation / Development

Features Developed:

- User login & registration with JWT.
- Blog post creation with title, content, tags.
- Like functionality per user-post.
- Editable user profiles.
- Admin panel to manage authors, posts, categories, and reports.

Folder Structure:

```
bash
CopyEdit
/client
  /src
    /pages
    /components
    /admin
  App.js

/server
  /routes
  /models
  /controllers
  index.js
```

Code Sample:

```
js
CopyEdit
// Node.js Route - Creating a Post
router.post("/create", authMiddleware, async (req, res) => {
  const post = new Post(req.body);
  await post.save();
  res.status(201).send(post);
});
```

Database Schema:

Collections:

- Users
- Posts
- Categories
- Likes

---

8. Results / Analysis

The system meets the minimum viable requirements:

- User can create, edit, and delete posts.
- Admin can moderate the entire platform.
- Posts are stored and retrieved efficiently from the database.
- Frontend and backend are fully integrated.

Screenshots of the UI and backend logs should be included here.

---

9. Challenges We Faced

As students building a full-stack blog website, we faced several challenges during development:

- **Handling User Login Expiry:**
  At first, we didn't know how to keep users logged in properly. Later, we learned about token-based authentication and added a way to refresh tokens so users don't get logged out quickly.
- **Working with Database Schema:**
  Designing how data should be stored in MongoDB was confusing at first. We used Mongoose to help us define proper data structures and relationships between users and posts.
- **Securing the Backend:**
  Making sure only logged-in users could access certain parts of the website was tricky. We solved this by learning how to use middleware to protect routes.
- **Making the Design Consistent:**
  Our pages didn't look the same at the beginning. To fix this, we used Tailwind CSS and followed the same layout style across all pages to make the UI look clean and uniform.

---

10. Conclusion

The blog website project successfully delivers a functioning content management system with full CRUD operations, authentication, and admin control. The modular architecture ensures easy maintenance and future scaling.

---

11. Future Scope (Optional)

- Add comment and reply system.
- Implement search and filter features.
- Enable social sharing of blogs.
- Use cloud storage for media upload.
- Convert to PWA for offline reading.

-