MERN Stack Blogging Platform Documentation

Author: [Your Name]

Date: [Submission Date]

# 1. Introduction

## 1.1 Project Overview

A full-stack blogging platform built using:  
- MongoDB (Database)  
- Express.js (Backend API)  
- React.js (Frontend)  
- Node.js (Server)

## 1.2 Features

- User authentication (Signup/Login with JWT)  
- Role-based access (Admin, Author, Reader)  
- Create, read, update, and delete (CRUD) blog posts  
- Commenting system

# 2. Setup Instructions

## 2.1 Backend Setup

1. Install dependencies:  
```  
cd backend  
npm install  
```  
  
2. Set up environment variables (.env):  
```  
MONGO\_URI=your\_mongodb\_connection\_string  
JWT\_SECRET=your\_jwt\_secret\_key  
PORT=5000  
```  
  
3. Run the server:  
```  
npm start

## 2.2 Frontend Setup

1. Install dependencies:  
```  
cd frontend  
npm install  
```  
  
2. Run the React app:  
```  
npm start  
```

# 3. API Documentation

## 3.1 Authentication Endpoints

|  |  |  |  |
| --- | --- | --- | --- |
| Endpoint | Method | Description | Access |
| /api/auth/signup | POST | Register a new user | Public |
| /api/auth/login | POST | Log in a user (returns JWT) | Public |
| /api/auth/me | GET | Get logged-in user details | Private (JWT) |

## 3.2 Blog Post Endpoints

|  |  |  |  |
| --- | --- | --- | --- |
| Endpoint | Method | Description | Access |
| /api/posts | POST | Create a new blog post | Author/Admin |
| /api/posts | GET | Get all blog posts | Public |
| /api/posts/:id | PUT | Update a blog post | Author/Admin |
| /api/posts/:id | DELETE | Delete a blog post | Author/Admin |

## 3.3 Comment Endpoints

|  |  |  |  |
| --- | --- | --- | --- |
| Endpoint | Method | Description | Access |
| /api/posts/:id/comments | POST | Add a comment | Logged-in users |
| /api/comments/:id | DELETE | Delete a comment | Admin/Commenter |

# 4. Database Schema

## 4.1 User Model

{  
 \_id: ObjectId,  
 name: String,  
 email: String (unique),  
 password: String (hashed),  
 role: String (enum: ["admin", "author", "reader"])  
}

## 4.2 Post Model

{  
 \_id: ObjectId,  
 title: String,  
 content: String,  
 author: { userId: ObjectId, name: String },  
 tags: [String],  
 comments: [{  
 text: String,  
 user: { userId: ObjectId, name: String },  
 createdAt: Date  
 }]  
}

# 5. Screenshots

(Insert screenshots of your application here)  
1. Homepage – List of all blog posts  
2. Login Page – User authentication  
3. Dashboard – User’s posts (for authors/admins)  
4. Admin Panel – Manage users/posts

# 6. Postman Collection

- Import the provided Postman Collection (BloggingPlatform.postman\_collection.json) to test all APIs.  
- Includes sample requests for:  
 - User registration/login  
 - Creating/deleting posts  
 - Adding comments

# 7. Conclusion

## 7.1 Challenges Faced

- Implementing JWT authentication  
- Role-based access control logic

## 7.2 Future Improvements

- Adding user profiles  
- Implementing likes/dislikes on posts

GitHub Repository: [https://github.com/omkarsinghoks/MERN-Blog-Api-Backend]  
Submitted By: [Omkar Singh]