| | |
|---|---|
| Project Title | Web-Based Blogging Application |
| Technologies | MERN |
| Domain | Social Media |
| Project Level | Hard |

# Table

## Table of Contents

# 1. Problem Statement:

Design a web-based Blogging App using the MERN stack technology.

## 1.1. Overview of Blogging App and MERN Stack?

MERN stands for the Mongo DB, Express, React and Node JS. These four are one of the most popular technologies at this time which is used in the field of the web development.

A blogging is a web app which is used to write articles to which other users can read and comment their view on it. If a user has an account over the websites, then he can also contribute to the community by writing and publishing the blog to which other users can read and comment.

**Hash node** and **Medium** are some of the most popular blogging websites which also users the immense power of blogging in their hand.

### 1.1.1 Expected features of the Blogging Website

Here is a detailed list of the expected features which should be available in your Blogging Website-

a. The website should have a **landing page** where the user can see the **posts or blogs** which is created by the other users.
b. User should have a login and signup option so that the new user can create an account and the existing user can login through that, this means you need to have **proper authentication-based login system** for the user.
c. Without logging into the system, a user can only read the blogs of the other users but will not be able to comment or create his own blog articles.

d. If a user has created his account, then he should have the option **to write his blog articles**, for which he should have proper **workspace area (Text Editor)** having some basic functionalities which is used while writing a blog like **adding image, video, code** and other options.

e. You can also integrate the **MD (Mark Down)** facility to the webpage for writing the articles over there or can use a **simple editor**.

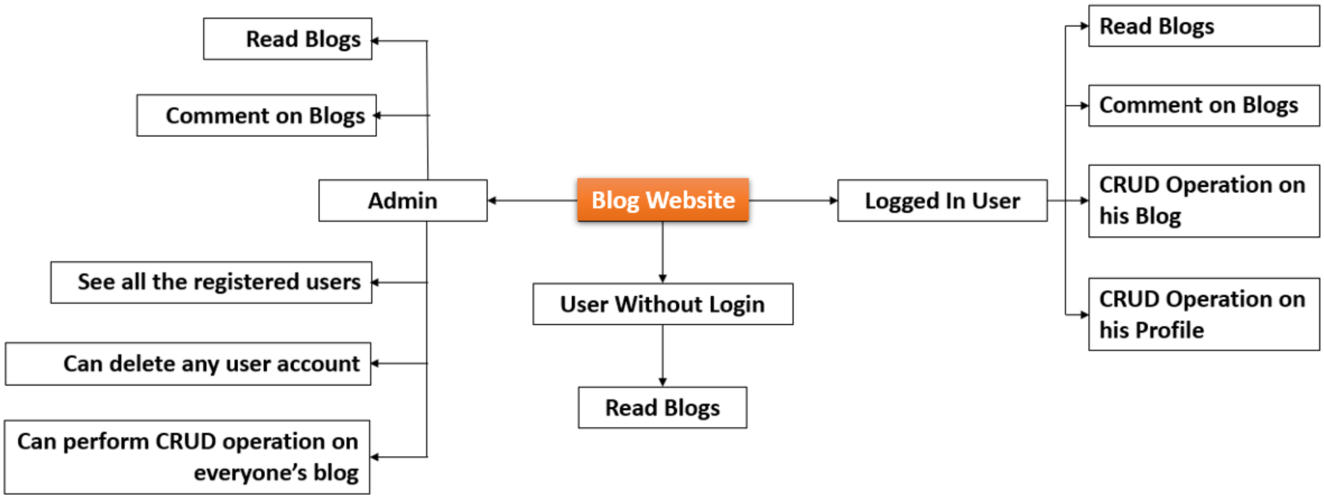
f. A proper dashboard is needed from where the user can see his profile where his basic details and all the articles should be visible on which he can perform the **CRUD Operation**.

g. A logged in user **can see, edit or even delete (CRUD Operation)** his articles from his user dashboard, which should be only accessible to the owner of the article.

h. A logged in user should also have an option to **log out** from his account, once he has finished with his work.

i. There should be a **search bar** through which the user can search for the article of his own interest.

j. There's need of a **tag option** to which user can use while **publishing the blog** through which the articles of the same type (on the same topic) can be filtered over using the search article by tag feature.



Read Blogs
Comment on Blogs
Admin
See all the registered users
Can delete any user account
Can perform CRUD operation on everyone's blog
Blog Website
User Without Login
Read Blogs
Logged In User
Read Blogs
Comment on Blogs
CRUD Operation on his Blog
CRUD Operation on his Profile

**Low Level user flow**

## 1.2. Project Objective

• **Blogging Platform:** A simple and easy to use blogging platform.

- **User-friendliness:** The project should be very easy to use enabling even a novice person to use it.

## 1.3. Scope of The Project

1. Blogging Web App will be solution for the user to create his own blog to share his knowledge and skills with others to create a good community.

2. This project will support articles to have text, image, video, code in it.

3. Users can join and share their skills here.

## 1.4. Functional and Non-Functional Requirements: -

### 1.4.1. Functional Requirements

1. **User Registration:** The user must be able to register for the application through an Email, Username, and Password. On Opening the application, users must be able to register themselves or they can directly log in if they have an account already. If the user skips this step, the user should be able to read blog. The user's email will be the unique identifier of his/her account on the Blogging Application.

2. **Adding New Blogs:** The website should provide the feature to add new blogs, edit or delete the existing blog.

3. **Comment:** The logged in user can comment on the blog post of another user.

5. **Dashboard:** The user should be able to see his personal details as well as his blogs in his dashboard.

### 1.4.2. Non-Functional Requirements

1. **Privacy:** Only logged in user can perform CRUD operation on their profile data and blog post, no one else can access those details.

2. **Robustness:** In case the user's system crashes, a backup of their blog post must be stored on remote database servers to enable recoverability.

3. **Performance:** The application must be lightweight and must loads the blogs instantly.

## 1.5. Use Case Table

| | | |
|---|---|---|
| Authentication System | Register, Login, Logout | User and Admin |
| Search Form | Search Blog | User |
| User Dashboard | Display User's details | User |
| Admin | Display All User's List with their blogs | Admin |

Table 1. Use Case

## 2. Project Evaluation Metrics:

### 2.1. Code:

- You are supposed to write code in a modular fashion
- Safe: It can be used without causing harm.
- Testable: It can be tested at the code level.
- Maintainable: It can be maintained, even as your codebase grows.
- Portable: It works the same in every environment (operating system).
- You have to maintain your code on GitHub.
- You have to keep your GitHub repo public so that anyone can check your code.
- Proper readme file you have to maintain for any project development.
- You should include the basic workflow and execution of the entire project in the readme file on GitHub.
- Follow the coding standards.

### 2.2. Database:

MongoDB is a source-available cross-platform document-oriented database program. Classified as a NoSQL database program, MongoDB uses JSON-like documents with optional schemas.

### 2.3. API Details or User Interface:

You have to expose your complete solution as an API or try to create a user interface for your model testing. Anything will be fine for us.

### 2.4. Deployment:

Implementation of reverse proxy, load balancing, and security group is mandatory for deployed applications.

### 2.5. Solutions Design:

You have to submit complete solution design strategies in High-level Document (HLD), Low-level Document (LLD), and Wireframe documents**.**

### 2.6. System Architecture:

You have to submit a system architecture design in your wireframe document and architecture document.

### 2.7. Optimization of solutions:

Try to optimize your solution on code level, architecture level, and mention all of these things in your final submission.

Mention your test cases for your project.

## 3. Submission requirements:

### 3.1. High-level Document:

You have to create a high-level document design for your project. You can reference the HLD form below the link.

Sample link: HLD Document Link

### 3.2. Low-level document:

You have to create a Low-level document design for your project; you can refer to the LLD from the link below.

Sample link: LLD Document Link

### 3.3. Architecture:

You have to create an Architecture document design for your project; you can refer to the Architecture from the link below.

Sample link: Architecture sample link

### 3.4. Wireframe:

You have to create a Wireframe document design for your project; refer to the Wireframe from the link below.

Demo link: Wireframe Document Link

### 3.5. Project code:

You have to submit your code to the GitHub repo in your dashboard when the final submission of your project.

Demo link: Project code sample link

### 3.6. Detail project report:

You have to create a detailed project report and submit that document as per the given sample.

Demo link: DPR sample link

### 3.7. Project demo video:

You have to record a project demo video for at least 5 Minutes and submit that link as per the given demo.

### 3.8. The project LinkedIn a post:

You have to post your project details on LinkedIn and submit that post link in your dashboard in your respective field.